BERGISCHE
UNIVERSITÄT
WUPPERTAL

Reihenfolge- und Maschinenbelegungsplanung für eine Klasse von Flexible Flow Shop Problemen

Dissertation

zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften
(Dr.-Ing.)

in der

Fakultät für Maschinenbau und Sicherheitstechnik

am

Fachgebiet Produktsicherheit und Qualität

der

Bergischen Universität Wuppertal

vorgelegt von

David Stalinski
aus Steinfurt

Tag der Einreichung: 7. Oktober 2020
Tag der mündlichen Prüfung: 20. Juli 2021

1. Gutachter: Prof. Dr.-Ing. Manuel Löwer (Bergische Universität Wuppertal)
2. Gutachter: Prof. Dr.-Ing. Dieter Scholz (Fachhochschule Münster)
3. Gutachter: Prof. Dr.-Ing. Peter Gust (Bergische Universität Wuppertal)

Wuppertal, den 25. April 2022

Die Dissertation kann wie folgt zitiert werden:

urn:nbn:de:hbz:468-20220602-094049-9
[http://nbn-resolving.de/urn/resolver.pl?urn=urn%3Anbn%3Ade%3Ahbz%3A468-20220602-094049-9]

DOI: 10.25926/sfhh-dr23
[https://doi.org/10.25926/sfhh-dr23]

Eidesstattliche Erklärung

Hiermit erkläre ich, David Stalinski, dass ich die vorliegende Dissertation mit dem Titel „**Reihenfolge- und Maschinenbelegungsplanung für eine Klasse von Flexible Flow Shop Problemen**“ selbstständig verfasst habe und nur die angegebenen Hilfsmittel benutzt sowie alle wörtlich oder inhaltlich übernommenen Stellen als solche gekennzeichnet habe.

Ich erkläre weiterhin, dass diese Dissertation an keiner anderen Universität vorgelegt wurde und dass weder ein erfolgloses Promotionsvorhaben noch die Aberkennung eines bereits erworbenen Doktorgrades in meiner Vergangenheit zurückliegen.

Ort, Datum	**Unterschrift**
Wuppertal, 25. April 2022	David Stalinski

Kurzfassung

Die logistische Leistung eines produzierenden Unternehmens beschreibt dessen Fähigkeit, kurze Lieferzeiten zu realisieren und zugesagte Liefertermine gegenüber seinen Kunden einzuhalten. Die Termintreue als logistisches Primärziel wird in der operativen Produktionsplanung und -steuerung u.a. durch die Ablaufplanung – also durch die Maschinenbelegung sowie durch die Reihenfolgebildung – beeinflusst.

Bei der Erstellung von Produktionsplänen liegt in der Regel ein hohes Potenzial in der Optimierung der Abläufe sowie in der Vermeidung von Verschwendungen. In vielen Unternehmen werden diese Planungsaufgaben historisch bedingt noch manuell durch entsprechende Planer durchgeführt. Diese stützen sich dabei auf ihre Erfahrungen und entscheiden demnach nach bestem Wissen und Gewissen. Im Zeitalter der Automatisierung und Digitalisierung wird die Aufgabe der Ablaufplanung zunehmend der Informationstechnik und damit dem Computer zugesprochen.

Für die Entwicklung der entsprechenden Entscheidungs- und Entscheidungsunterstützungssysteme sind in der Regel komplexe mathematische Optimierungsprobleme zu lösen. Die Anzahl der Kombinationsmöglichkeiten und damit die benötigte Rechenzeit für eine vollständige Enumeration wächst exponentiell mit der Problemgröße (kombinatorische Explosion). Probleme dieser Art haben daher gemein, dass selbst vermeintlich kleine Aufgabenstellungen sich als praktisch nicht exakt lösbar herausstellen.

Im Rahmen der vorliegenden Arbeit wird ein Verfahren für die simultane Reihenfolge- und Maschinenbelegungsplanung für eine Klasse von Flexible Flow Shop Problemen entwickelt. Das Verfahren soll in Bezug auf das Laufzeit- und Lösungsverhalten den Anforderungen für den Einsatz in Entscheidungsunterstützungssystemen realer Produktionen genügen.

Flexible Flow Shop Probleme bezeichnen in der Theorie der Ablaufplanung mehrstufige Produktionsprozesse, bei denen jede Stufe aus mindestens einer Maschine besteht und jeder Fertigungsauftrag mindestens in einer der Produktionsstufen bearbeitet werden muss. Flexible Flow Shop Probleme gibt es in nahezu unzähligen Varianten, welche sich wesentlich durch die Ausprägung der Maschinenumgebung sowie die zu berücksichtigenden Restriktionen unterscheiden. Die in dieser Arbeit behandelten Probleme werden durch die im folgenden Absatz zusammengefassten Restriktionen charakterisiert.

Ein Teil der Fertigungsaufträge muss einzelne Stufen überspringen und kann technologisch bedingt nicht an allen Maschinen bearbeitet werden. Es sind reihenfolgeabhängige Rüstzeiten zu berücksichtigen, wobei sich die Fertigungsaufträge in eine oder mehrere Auftragsfamilien pro Stufe zusammenfassen lassen. Manche Fertigungsaufträge müssen innerhalb einer Stufe zusammenhängend und am Stück auf derselben Maschine bearbeitet

werden. Weiterhin sollen begrenzte Rohmaterialien und Lagerkapazitäten berücksichtigt werden, so dass die Planungsergebnisse praktisch umsetzbar sind.

Die Entwicklung des Verfahrens beruht auf mehreren Ansätzen. Durch die zeitliche Diskretisierung des Planungshorizonts in gleichmäßige Planungsperioden wird das Problem bzw. dessen Suchraum zunächst in eine definierte Struktur transformiert. Dadurch wird die Komplexität auf ein definiertes Niveau reduziert und die Manipulation des Modells durch herkömmliche kombinatorische Optimierungsverfahren wird ermöglicht. Die Ableitung von konkreten Ergebnissen in Form von Produktionsplänen für eine bestimmte Kombination wird algorithmisch durch eine ereignisdiskrete Simulationsmethode in einem mathematischen Verfahren hinterlegt.

Das entwickelte Verfahren wird einer umfangreichen numerischen Untersuchung unterzogen, um es in Bezug auf die benötigte Rechenzeit sowie die erzielbare Lösungsqualität zu bewerten. Dafür werden zwei standardisierte Optimierungsalgorithmen – ein Ameisenalgorithmus sowie ein Genetischer Algorithmus – implementiert und mit dem Modell gekoppelt. Für beide Optimierungsalgorithmen werden die entsprechenden Verfahrensparameter für unterschiedliche Problemgrößen in statistisch abgesicherten Untersuchungen systematisch variiert und damit optimiert. Die Leistungsfähigkeit der jeweiligen Optimierungsalgorithmen kann damit für unterschiedliche Problemgrößen verglichen werden, so dass eine entsprechende Auswahl erfolgen kann. Das Modell kann dadurch mit optimierten Parametern auf unterschiedliche Probleminstanzen und -größen angewendet werden, um die Einflüsse weiterer Parameter des Verfahrens zu untersuchen.

Anhand einer Fallstudie in Form eines konkreten Industriebeispiels wird das Verfahren anschließend auf Datensätze angewandt, welche auf realen Daten eines produzierenden Unternehmens aus der holzverarbeitenden Industrie basieren. Die Produktion ist hochgradig automatisiert und bietet eine qualitativ hochwertige Datengrundlage, welche u.a. auf den Einsatz der RFID-Technik für sekundengenaue Buchungs- und Steuerungsprozesse zurückzuführen ist. Für die rechnerbasierte Unterstützung der operativen Produktionsplanung und -steuerung wurde das Verfahren dort in Form eines Entscheidungssystems implementiert. Es greift dabei auf gesammelte Daten aus einem Zeitraum von mehreren Jahren zurück, um z.B. statistisch abgesicherte Informationen über zu erwartende Produktionszeiten zu generieren. Das System unterstützt die Produktionsleitung bei der Planung und Steuerung der Produktion und hat sich seit der Implementierung erfolgreich etabliert.

Abstract

The logistical performance of a producing company describes its ability to realize short delivery times and meet the promissed delivery dates towards its customers. A great adherence to schedule as a logistic objective is affected within the operative production planning and control by specifying machine allocations and processing sequences.

Thus, the process of composing production schedules normally offers high potentials in optimizing the production processes and avoiding waste of time. In many companies these planning processes are still done manually for historical reasons. The schedulers often trust in their experience and take decisions based on the best of their knowledge. In the digital age the task of scheduling falls more and more into the remit of information technology and should therefore be done by computers.

In order to develop appropriate decision and decision support systems complex mathematical problems in the form of combinatorial optimization problems need to be solved. The amount of combinatorial possibilities and thus the needed amount of computation time for a full enumeration grows exponential with the problem size (combinatorial explosion). This type of problems is characterised by the fact that even small problem instances are practically not solvable in an exactly way.

This thesis deals with the development of a method for the simultaneous calculation of production sequences and machine allocations of a flexible flow shop problem. The method has to comply the requirements of computation times and solution quality for being implemented into decision support systems in real processing companies.

In the theory of scheduling flexible flow shop problems designate multi-stage production processes in which each stage consists of at least one machine and each production order must be processed in at least one of the production stages. There are flexible flow shop problems in almost countless variants which differ significantly in the form of the constraints to be taken into account. Solution strategies and approaches cut accordingly for use for special problem types. The problems dealt with in this work are characterized by the restrictions and constraints summarized in the following paragraph.

Some of the production orders have to skip some of the stages and cannot be processed on all machines for technological reasons. There are sequent dependent set up times to be taken into account whereby the production orders are classified into one or more families per stage. Some manufacturing orders have be processed simultaniously and in one piece on the same machine. Furthermore the available raw materials and storage capacities must be taken into account during the planning process, so that the planning results are practically feasible.

The development of the method is based on several approaches. The problem respectively its search space is initially transformed by dividing the planning horizon by a time grid. This reduces the complexity to a defined level and the manipulation of the model by conventional combinatorial optimization procedures is made possible. The generation of concrete solutions in the form of production plans for a certain combination is algorithmically implemented into the mathematical model as an event discrete simulation.

The developed model is subjected to an extensive numerical examination in order to evaluate it in terms of the required computing time and the achievable solution quality. For this, the two standardized optimization algorithms – the ant colony algorithm and the genetic algorithm – are implemented and linked to the model. For both algorithms the corresponding process parameters are examined for different problem sizes in statistically verified studies. Thus, the model can be applied to different problem instances and sizes with optimized Parameters.

Based on a case study in the form of a concrete industrial example the method is then applied to data records based on real data from a manufacturing company in the woodworking industry. The production is highly automated and uses the RFID technology for precise booking and control processes. Thus, it already offers a high quality data basis. For a computer-based support of the operational production planning and control the developed method was implemented in the form of a decision system. It uses collected data of several years back to generate e.g statistically verified information about expected production times of the manufacturing orders. The system supports the production management in the planning and control of the production and is successfully settled since its implementation.

Inhaltsverzeichnis

Eidesstattliche Erklärung **III**

Kurzfassung **V**

Abstract **VII**

Inhaltsverzeichnis **IX**

Abbildungsverzeichnis **XIII**

Tabellenverzeichnis **XVII**

Algorithmenverzeichnis **XIX**

Symbolverzeichnis **XXI**

1 Einleitung **1**
1.1 Motivation 1
1.2 Zielsetzung und Vorgehensweise 2

2 Das Flexible Flow Shop Problem **7**
2.1 Notation von Problemen der Ablaufplanung 7
2.1.1 Maschinenumgebungen 9
2.1.2 Randbedingungen 11
2.1.3 Zielgrößen 14
2.2 Definition der untersuchten Problemklasse 15
2.2.1 Reihenfolge- und Maschinenbelegungsproblem 15
2.2.2 Randbedingungen und Restriktionen 16
2.2.3 Zielsetzung 18
2.3 Komplexität 19
2.3.1 Komplexitätstheorie 19
2.3.2 Abschätzung der Komplexität 20
2.4 Stand der Forschung 21
2.4.1 Gemischt-ganzzahlige Optimierung 22
2.4.2 Suchverfahren 22
2.4.3 Ereignisdiskrete Simulation 25
2.4.4 Lösungsansätze für Flexible Flow Shop Probleme 28
2.4.5 Ablaufplanung in der Praxis 34

2.4.6 Zusammenfassende Übersicht relevanter Forschungsarbeiten . . 37

3 Handlungsbedarf und angewendete Methodik 41

4 Entwicklung eines ereignisdiskreten Planungsverfahrens 47
4.1 Formulierung eines gemischt-ganzzahligen Optimierungsmodells 47
4.1.1 Nomenklatur 47
4.1.2 Allgemeine Definitionen 50
4.1.3 Formulierung des Basismodells 51
4.1.4 Erweiterung des Basismodells 52
4.1.5 Formulierung der Zielgrößen 57
4.2 Definition von diskreten Planungsperioden 58
4.2.1 Nomenklatur 59
4.2.2 Definition der Entscheidungsvariablen 59
4.2.3 Definition des Such- und des Lösungsraums 61
4.3 Formulierung eines ereignisdiskreten Lösungsgenerators 64
4.3.1 Nomenklatur 64
4.3.2 Verfahrensablauf 65
4.3.3 Einbindung der Planungsregeln 74

5 Verwendete Optimierungsverfahren 77
5.1 Genetischer Algorithmus 78
5.1.1 Inspiration durch die natürliche Evolution 78
5.1.2 Ablauf des Verfahrens 79
5.1.3 Genetische Operatoren 81
5.1.4 Kopplung mit dem Planungsverfahren 85
5.2 Ameisenalgorithmus 87
5.2.1 Inspiration durch das Verhalten von Ameisen 88
5.2.2 Ablauf des Basisverfahrens 89
5.2.3 Varianten des Verfahrens 91
5.2.4 Kopplung mit dem Planungsverfahren 92
5.3 Zusammenfassende Diskussion 94

6 Numerische Untersuchungen 97
6.1 Generierung von randomisierten Testdatensätzen 98
6.2 Auswahl eines Optimierungsverfahrens 101
6.2.1 Festlegung der optimalen Parameterausprägungen 101
6.2.2 Vergleichen der Ergebnisse 104
6.3 Untersuchungen an den randomisierten Testdatensätzen 107
6.3.1 Auswirkungen der zeitlichen Diskretisierung 107
6.3.2 Auswirkungen unterschiedlicher Planungsregeln 109
6.3.3 Analyse des Laufzeitverhaltens 112
6.3.4 Beurteilung der Planungsergebnisse 114

6.4 Fallstudie Holzverarbeitung als konkretes Industriebeispiel 117
6.4.1 Beschreibung des Produktionsprozesses 117
6.4.2 Anwendung des Planungsverfahrens 121
6.4.3 Beurteilung der Ergebnisse 124
6.4.4 Einbindung des Planungsverfahrens in ein Entscheidungssystem 129
6.5 Fazit und kritische Würdigung der Ergebnisse 130
6.5.1 Auswertung der erzielten Ergebnisse 131
6.5.2 Abgrenzung des wissenschaftlichen Erkenntnisgewinns 134
6.5.3 Abschätzung der Auslastung und Eingrenzung der Problemgröße 136

7 Zusammenfassung und Ausblick **143**

Literaturverzeichnis **149**

Veröffentlichungen **167**

Lebenslauf **169**

A Anhang **171**
A.1 Parameterstudien für den Genetischen Algorithmus 171
A.1.1 Ergebnisse der Parameterstudien für S–224 171
A.1.2 Ergebnisse der Parameterstudien für M–424 176
A.1.3 Ergebnisse der Parameterstudien für L–824 181
A.2 Parameterstudien für den Ameisenalgorithmus 184
A.2.1 Ergebnisse der Parameterstudien für S–224 184
A.2.2 Ergebnisse der Parameterstudien für M–424 188
A.2.3 Ergebnisse der Parameterstudien für L–824 191
A.3 Abschätzung der Auslastung des Verfahrens 195

Abbildungsverzeichnis

2.1 Schematische Darstellung von Flexible Flow Shop Problemen 7
2.2 Modellierung von Materialflussmöglichkeiten durch Stapelbearbeitung . . 18
2.3 Klassifikation von Suchverfahren 23
2.4 Beispiel eines lokalen und des globalen Optimums 25
2.5 Kopplung einer Simulation mit einem Optimierer 26
2.6 Einordnung der ereignisdiskreten Simulationsmethoden 27
2.7 Vergleich zwischen zeit- und ereignisdiskreter Simulation 28

3.1 Gewählter Lösungsansatz und resultierender Handlungsbedarf 42
3.2 Definition von Planungsperioden anhand eines Beispiels 43
3.3 Angewendete Methodik (Kopplung von Simulation und Optimierung) . . 44
3.4 Überlagerung der Planungsperioden mit Ereigniszeitpunkten 45
3.5 Entkopplung von Suchraum und Lösungsraum 46

4.1 Methode für die Darstellung von Ablaufplänen 49
4.2 Diskretisierung des Planungshorizonts in gleichmäßige Planungsperioden 60
4.3 Aufgeteilter Planungshorizont für zwei Stufen mit jeweils zwei Maschinen 62
4.4 Entkopplung von Suchraum und Lösungsraum 63
4.5 Kopplung zwischen Metaheuristik und Lösungsgenerator 65
4.6 Ablaufplanung mit dem Planungshorizont H und $\theta = 4$ Planungsperioden 66
4.7 Ereigniszeitpunkte von drei Maschinen während der Planung 67
4.8 Ablaufplan mit Leerlauf an einer Maschine 70

5.1 Beispiel eines Chromosoms mit sechs Genen und möglichen Allelen . . . 80
5.2 Fitnessbasierte und rangbasierte Selektionswahrscheinlichkeiten 83
5.3 Selektion mittels Rouletteverfahren und Stochastic Universal Sampling . 84
5.4 Beispiel eines N-Point-Crossover-Operators mit zwei Punkten 84
5.5 Beispiel eines Uniform-Crossover-Operators 85
5.6 Beispiel eines Mutationsoperators 85
5.7 Kodierungsschema des Genetischen Algorithmus 86
5.8 Beispiel für ein dem Kodierungsschema entsprechenden Chromosom . . . 87
5.9 Ameisen legen auf ihrer Futtersuche Pheromonspuren auf den Wegen ab . 88
5.10 Sukzessive Ausbildung einer Pheromonspur auf dem kürzesten Weg . . . 89
5.11 Pheromonkonzentrationen in einem Rundreiseproblem 90

6.1 Erläuterung eines Boxplots anhand eines Beispiels 98
6.2 Vergleich: Genetischer Algorithmus und Ameisenalgorithmus (S–224) . . 105
6.3 Vergleich: Genetischer Algorithmus und Ameisenalgorithmus (M–424) . . 106

6.4 Vergleich: Genetischer Algorithmus und Ameisenalgorithmus (L–824) . . 106
6.5 Vergleich unterschiedlicher Planungsperioden für S–224 108
6.6 Vergleich unterschiedlicher Planungsperioden für M–424 108
6.7 Vergleich unterschiedlicher Planungsperioden für L–824 109
6.8 Vergleich der unterschiedlichen Planungsregeln für S–224 110
6.9 Vergleich der unterschiedlichen Planungsregeln für M–424 111
6.10 Vergleich der unterschiedlichen Planungsregeln für L–824 111
6.11 Laufzeitverhalten des Genetischen Algorithmus (zeitbasiert) 113
6.12 Laufzeitverhalten des Genetischen Algorithmus (iterationsbasiert) 114
6.13 Bester gefundener Ablaufplan für L–824 (EDD-Regel und $\theta = 10$) 116
6.14 Schema des Produktionssystems der Fallstudie 118
6.15 Vollautomatische Sägeanlage mit Peripherie 119
6.16 Vollautomatische Pressenanlage mit Auslaufband 120
6.17 Vollautomatisches Lagersystem mit motorisierten Rollenbahnen und RFID 120
6.18 Vergleich unterschiedlicher Planungsperioden anhand realer Daten 125
6.19 Vergleich der unterschiedlichen Planungsregeln anhand realer Daten . . . 126
6.20 Ablaufplan auf Basis realer Daten (S/OPN-Regel, $H = 48\,\mathrm{h}$ und $\theta = 14$) 127
6.21 Ausschnitt aus dem Ablaufplan auf Basis realer Daten 128
6.22 Laufzeitverhalten des Verfahrens auf Basis realer Daten (S/OPN, $\theta = 14$) 128
6.23 Implementierung des Verfahrens als Entscheidungssystem 129
6.24 Erzielte Ergebnisse mit dem Planungsverfahren 131
6.25 Laufzeitverhalten des Verfahrens auf Basis unterschiedlicher Daten 133
6.26 Auslastung des Verfahrens auf Basis unterschiedlicher Daten 137

A.1 Einfluss der Populationsgröße (Genetischer Algorithmus, S–224) 171
A.2 Einfluss der Tournamentselektion (Genetischer Algorithmus, S–224) . . . 172
A.3 Einfluss der Kreuzungswahrscheinlichkeit (Genetischer Algorithmus, S–224) 172
A.4 Einfluss des N-Point Crossovers (Genetischer Algorithmus, S–224) 173
A.5 Einfluss des Uniform Crossovers (Genetischer Algorithmus, S–224) 173
A.6 Einfluss der Mutationswahrscheinlichkeit (Genetischer Algorithmus, S–224) 174
A.7 Einfluss des Elitismus (Genetischer Algorithmus, S–224) 174
A.8 Einfluss der Rouletteselektion (Genetischer Algorithmus, S–224) 175
A.9 Einfluss der SUS-Selektion (Genetischer Algorithmus, S–224) 175
A.10 Feintuning des Genetischen Algorithmus (S–224) 176
A.11 Einfluss der Populationsgröße (Genetischer Algorithmus, M–424) 176
A.12 Einfluss der Tournamentselektion (Genetischer Algorithmus, M–424) . . . 177
A.13 Einfluss der Kreuzungswahrscheinlichkeit (Genetischer Algorithmus,M–424) 177
A.14 Einfluss des N-Point Crossovers (Genetischer Algorithmus, M–424) 178
A.15 Einfluss der Mutationswahrscheinlichkeit (Genetischer Algorithmus,M–424) 178
A.16 Einfluss des Elitismus (Genetischer Algorithmus, M–424) 179
A.17 Einfluss des Uniform Crossovers (Genetischer Algorithmus, M–424) . . . 179
A.18 Einfluss der Rouletteselektion (Genetischer Algorithmus, M–424) 180
A.19 Einfluss der SUS-Selektion (Genetischer Algorithmus, M–424) 180
A.20 Feintuning des Genetischen Algorithmus (M–424) 181

A.21 Einfluss der Populationsgröße (Genetischer Algorithmus, L–824) 181
A.22 Einfluss der Tournamentselektion (Genetischer Algorithmus, L–824) . . . 182
A.23 Einfluss des N-Point Crossovers (Genetischer Algorithmus, L–824) 182
A.24 Einfluss der Mutationswahrscheinlichkeit (Genetischer Algorithmus, L–824) 183
A.25 Feintuning des Genetischen Algorithmus (L–824) 183
A.26 Einfluss der Populationsgröße (Ameisenalgorithmus, S–224) 184
A.27 Einfluss des Pheromonadditionskoeffizienten (Ameisenalgorithmus, S–224) 185
A.28 Einfluss des Pheromonzerfallskoeffizienten (Ameisenalgorithmus, S–224) . 185
A.29 Einfluss der Minimalpheromonkonzentration (Ameisenalgorithmus, S–224) 186
A.30 Einfluss der Maximalpheromonkonzentration (Ameisenalgorithmus, S–224) 186
A.31 Einfluss der Initialpheromonkonzentration (Ameisenalgorithmus, S–224) . 187
A.32 Feintuning des Ameisenalgorithmus (S–224) 187
A.33 Einfluss der Populationsgröße (Ameisenalgorithmus, M–424) 188
A.34 Einfluss des Pheromonadditionskoeffizienten (Ameisenalgorithmus, M–424) 188
A.35 Einfluss des Pheromonzerfallskoeffizienten (Ameisenalgorithmus, M–424) 189
A.36 Einfluss der Minimalpheromonkonzentration (Ameisenalgorithmus, M–424) 189
A.37 Einfluss der Maximalpheromonkonzentration (Ameisenalgorithmus, M–424) 190
A.38 Einfluss der Initialpheromonkonzentration (Ameisenalgorithmus, M–424) 190
A.39 Feintuning des Ameisenalgorithmus (M–424) 191
A.40 Einfluss der Populationsgröße (Ameisenalgorithmus, L–824) 191
A.41 Einfluss des Pheromonadditionskoeffizienten (Ameisenalgorithmus, L–824) 192
A.42 Einfluss des Pheromonzerfallskoeffizienten (Ameisenalgorithmus, L–824) . 192
A.43 Einfluss der Minimalpheromonkonzentration (Ameisenalgorithmus, L–824) 193
A.44 Einfluss der Maximalpheromonkonzentration (Ameisenalgorithmus, L–824) 193
A.45 Einfluss der Initialpheromonkonzentration (Ameisenalgorithmus, L–824) . 194
A.46 Feintuning des Ameisenalgorithmus (L–824) 194
A.47 Auslastung des Verfahrens auf Basis unterschiedlicher Daten 195

Tabellenverzeichnis

2.1 Darstellung des Forschungsdefizits anhand bisheriger Forschungsergebnisse 39

5.1 Parametriermöglichkeiten des Genetischen Algorithmus 87
5.2 Parametriermöglichkeiten des Ameisenalgorithmus 94

6.1 Systematische Erzeugung von randomisierten Testdatensätzen 99
6.2 Konkrete Ausprägungen der Testdatensätze 100
6.3 Parametervariation des Genetischen Algorithmus für S–224 101
6.4 Parametervariation des Genetischen Algorithmus für M–424 102
6.5 Parametervariation des Genetischen Algorithmus für L–824 102
6.6 Parametervariation des Ameisenalgorithmus für S–224 103
6.7 Parametervariation des Ameisenalgorithmus für M–424 103
6.8 Parametervariation des Ameisenalgorithmus für L–824 103
6.9 Optimierte Parametrierung des Genetischen Algorithmus 104
6.10 Optimierte Parametrierung des Ameisenalgorithmus 104
6.11 Gegenüberstellung der erzielten Forschungsergebnisse 135
6.12 Komplexitäten und Rechenzeiten der untersuchten Probleme 138

Algorithmenverzeichnis

4.1 Hauptprozedur des Planungsverfahrens 65
4.2 Berechnung eines Zeitschritts an einer Maschine 68
4.3 Anwendung der Planungslogik an einer Maschine 69
4.4 Prüfung eines Auftragsstapels 71
4.5 Einplanung eines Fertigungsauftrags 72
4.6 Einplanung eines Auftragsstapels 73

5.1 Ablauf des Genetischen Basisalgorithmus 81
5.2 Ablauf des Ameisenalgorithmus 91

Symbolverzeichnis

Ameisenalgorithmus

$\Delta\rho_{k,i,\tau,f}$ . . . . . Pheromonänderung

ϵ . . . . . Pheromonverdampfungskoeffizient

ρ_{init} . . . . . Initiale Pheromonkonzentration

$\rho_{\max}$ . . . . . Maximale Pheromonkonzentration

$\rho_{\min}$ . . . . . Minimale Pheromonkonzentration

ρ_{+} . . . . . Pheromonadditionskoeffizient

$\rho_{k,i,\tau,f}$ . . . . . Pheromonmatrix

N . . . . . Populationsgröße

$p_{k,i,\tau,f}$ . . Auswahlwahrscheinlichkeit einer Auftragsfamilie für eine Planungsperiode

Genetischer Algorithmus

C . . . . . Kreuzungsschema

$E_{\max}$ . . . . . Rangbasierter Selektionsdruck

N . . . . . Populationsgröße

n_{C} . . . . . Anzahl der Kreuzungspunkte

n_{E} . . . . . Anzahl der Eliten

n_{T} . . . . . Tournamentgröße

p_{C} . . . . . Kreuzungswahrscheinlichkeit

p_{M} . . . . . Mutationswahrscheinlichkeit

p_{U} . . . . . Uniform Rate (Kreuzungswahrscheinlichkeit)

s_1 . . . . . Selektionsalgorithmus

s_2 . . . . . Auswahlalgorithmus

Optimierungsmodell und Lösungsverfahren

$\delta_{k,i,\tau}$ . . . . . Entscheidungsmatrix

ΔT . . . . . Länge einer Planungsperiode

$\epsilon_{k,j,\tau}$ . . . . . Wird der Job J_j in der Stufe K_k in der Planungsperiode τ gefertigt?

$\eta_{k,j}$ . . . . . Pufferbedarf des Jobs J_j in der Produktionsstufe K_k

$\lambda_{k,i,j}$ . . . . . Kompatibilität von Maschine $M_{k,i}$ für die Bearbeitung von Job J_j

$\mathcal{B}_k$ . . . . . Menge der Auftragsstapel der Produktionsstufe K_k

$\mathcal{F}_{k,i}$ . . . . . Menge der Auftragsfamilien der Maschine $M_{k,i}$

$\mathcal{J}$ . . . . . Menge der Fertigungsaufträge

$\mathcal{K}$ . . . . . Menge der Produktionsstufen

$\mathcal{M}_k$ . . . . . Menge der Maschinen in Produktionsstufe K_k

$\mathcal{R}$ . . . . . Menge der Ressourcen

$\mathcal{T}$ . . . . . Menge der Planungsperioden

$\mu_{k,i,j,f}$ . . . . . Zugehörigkeit des Jobs J_j zu Auftragsfamilie $F_{k,i,f}$

$\nu_{k,j,b}$ . . . . . Zugehörigkeit des Jobs J_j zu Batch $B_{k,b}$ in Stufe K_k

ω_r . . . . . Verfügbarkeit der Ressource R_r zum Zeitpunkt $t = 0$

ϕ_{k,i,f_1,f_2} . . . . . Rüstzeit von F_{f_1} auf F_{f_2} auf Maschine $M_{k,i}$

$\pi_{k,j,r}$ . . . . . Verbrauch der Ressource R_r durch J_j in K_k

$\Psi_{k,i,j,f}$ . . . . . wurde Job J_j als Auftragsfamilie $F_{k,i,f}$ bearbeitet?

ρ_k . . . . . Pufferkapazität der Produktionsstufe K_k

$\sigma_{k,j}$ . . . . . Notwendigkeit von Stufe K_k für die Bearbeitung von Job J_j

τ . . . . . Index Entscheidungszeitpunkte

θ . . . . . Anzahl der Planungsperioden

$A_{k,j}$ . . . . . Ankunftszeit (arrival time) von Job J_j in Stufe K_k

b . . . . . Index der Auftragsstapel

$B_{k,b}$ . . . . . Auftragsstapel (batch) b der Produktionsstufe K_k

c . . . . . Anzahl der Produktionsstufen

$C_{k,j}$ . . . . . Endzeit (completion time) von Job J_j in Stufe K_k

d_j . . . . . Planliefertermin (due date) von Job J_j

E_j . . . . . Verfrühung (earliness) von Job J_j

f . . . . . Index der Auftragsfamilien

F_j . . . . . Durchlaufzeit oder Flusszeit (flow time) von Job J_j

$F_{k,i,f}$ . . . . . Auftragsfamilie (family) f der Maschine $M_{k,i}$

$g_{k,j}$ . . . . . Befindet sich der Job J_j im Puffer der Stufe k?

H . . . . . Zeithorizont der Planung

H_j . . . . . Schlupf (slack) von Job J_j

i . . . . . Index der Maschinen

$I_{k,i}$ . . . . . Leerlaufzeit (idle time) von Maschine $M_{k,i}$

j . . . . . Index der Fertigungsaufträge

J_j . . . . . Fertigungsauftrag (job) j

k . . . . . Index der Produktionsstufen

K_k . . . . . Produktionsstufe (stage) k

l . . . . . Anzahl der Ressourcen

L_j . . . . . Terminabweichung (lateness) von Job J_j

m_k . . . . . Anzahl der Maschinen in Produktionsstufe K_k

$M_{k,i}$ . . . . . Maschine (machine) i der Produktionsstufe K_k

n . . . . . Anzahl der Fertigungsaufträge

$n_{k,b}$ . . . . . Anzahl der Jobs in Fertigungsstapel $B_{k,b}$ in Stufe K_k

$o_{k,i}$ . . . . . Anzahl der Auftragsfamilien der Maschine $M_{k,i}$

$p_{k,i,j}$ . . . . . Prozesszeit (processing time) von Job J_j auf Maschine $M_{k,i}$

q_k . . . . . Anzahl der Fertigungsstapel in Produktionsstufe K_k

r . . . . . Index der Ressourcen

R_r	Ressource (resource) r
s_{k,i,j_1,j_2}	Rüstzeit (setup time) von Job J_{j_1} auf Job J_{j_2} auf Maschine $M_{k,i}$
$s_{k,i}$	Auftragsfamilie, auf welche die Maschine $M_{k,i}$ gerüstet ist
$S_{k,j}$	Startzeit (start time) von Job J_j in Stufe K_k
t	Allgemeine Zeitvariable
t_τ	Entscheidungszeitpunkt τ
t_{End}	Endzeit des Planungsverfahrens bzw. des Ablaufplans
t_{Leerlauf}	Zeitkonstante, die für die Einplanung von Leerlauf notwendig ist
t_{Start}	Startzeit des Planungsverfahrens bzw. des Ablaufplans
T_j	Verspätung (tardiness) von Job J_j
$t_{k,i}$	Aktueller Zeitschritt der Maschine $M_{k,i}$
w_j	Gewichtung (weight) von Job J_j
$X_{k,i,j}$	wurde der Job J_j auf der Maschine $M_{k,i}$ gefertigt?
$Y_{k,i,j}$	Absolute Reihenfolge von Job J_j auf Maschine $M_{k,i}$
$Y_{k,i}$	Absolute Reihenfolge auf der Maschine $M_{k,i}$
Z_{k,i,j_1,j_2}	wurde Job J_{j_1} auf $M_{k,i}$ unmittelbar vor Job J_{j_2} gefertigt?
$Z_{k,i}$	Index j des letzten Jobs J_j auf Maschine $M_{k,i}$

1 Einleitung

Die vorliegende Arbeit befasst sich mit der Entwicklung eines Verfahrens für die Ablaufplanung von Flexible Flow Shop Problemen sowie mit der Anwendung des Verfahrens in der operativen Produktionsplanung und -steuerung. Flexible Flow Shop Probleme bezeichnen allgemeine Probleme der Ablaufplanung, bei denen eine Produktion in mehreren Fertigungsstufen organisiert ist und sich in den einzelnen Stufen jeweils mehrere parallel angeordnete Maschinen befinden. Das Planungsziel besteht im Rahmen dieser Arbeit in der Bestimmung von optimierten Produktionsreihenfolgen und Maschinenzuordnungen für die entsprechenden Fertigungsaufträge unter Einbeziehung von multikriteriellen Zielsetzungen.

1.1 Motivation

Die logistische Leistung von produzierenden Unternehmen stellt neben den Kosten und der Qualität der hergestellten Produkte einen wesentlichen Wettbewerbsfaktor dar (Nyhuis et al., 2012). In realen Produktionsprozessen sollen daher im Rahmen der operativen Produktionsplanung und -steuerung die entsprechenden Aufgaben und Prozesse derart koordiniert werden, dass eine Erfüllung der logistischen Ziele möglichst vollständig gelingt (Schuh et al., 2012). In der Ablaufplanung als Teil der operativen Produktionsplanung und -steuerung werden dafür Entscheidungen getroffen, welche u.a. die Auswahl von Maschinen für die Bearbeitung von Produktionsaufträgen sowie entsprechende Produktionsreihenfolgen für die jeweiligen Fertigungsaufträge auf den Maschinen betreffen (siehe z.B. Huisman et al., 2014, Parlier et al., 2020).

In der Beeinflussung eines Produktionsprozesses mit der Wahl unterschiedlicher Bearbeitungsreihenfolgen und Maschinenbelegungen sind zum Teil erhebliche Potenziale vorhanden. Diese logistischen Potenziale liegen im Wesentlichen in der Vermeidung von Verschwendungen, indem beispielsweise das Leerlaufen von Maschinen und das Auftreten von Rüstzeiten reduziert werden (Lödding, 2016). Durch sachgerechte Entscheidungen sollen die Potenziale bestmöglich genutzt und eine möglichst termingerechte Fertigstellung aller Kundenaufträge ermöglicht werden.

Die zunehmende Automatisierung von Produktionsprozessen führt zu immer komplexeren technischen Lösungen. Ausgehend von Robotern und automatischen Handhabungssystemen rückt die innerbetriebliche Logistik zunehmend in den Fokus der Automatisierung, so dass der technische Materialtransport beispielsweise durch motorisierte Rollenbahnen und Förderbänder abgewickelt wird. Weiterhin kommen vermehrt Technologien zur innerbetrieblichen Lokalisierung von Waren wie Indoor GPS Systeme, RFID (Radio Frequency Identification Device), WLAN, Bluetooth etc. zum Einsatz (Bracht et al., 2011).

Buchungs-, Transport- und Rüstungsprozesse werden dadurch zunehmend vollautomatisch durchgeführt.

Diese Aspekte der innerbetrieblichen Automatisierung haben u.a. eine entsprechend hohe Dichte an Betriebs-, Bewegungs-, und Bestandsdaten zur Folge. Diese laufen in der Regel mit den Stamm- und Auftragsdaten in dem ERP-System des Unternehmens zusammen und bilden die Basis für die Entscheidungen der operativen Produktionsplanung und -steuerung. Die Koordinierung moderner Produktionsprozesse wird entsprechend komplexer, so dass der Mensch allein mit diesen Aufgaben zunehmend überfordert ist. Dennoch dominiert in der unternehmerischen Praxis in vielen Fällen nach wie vor eine Planung und Steuerung der Produktion durch den Menschen (siehe z.B. März et al., 2011b).

Die Ablaufplanung als Teilgebiet des Operations Research beschäftigt sich mit diesem Thema auf mathematisch naturwissenschaftlicher Ebene. Der Versuch, die Potenziale einer realen Produktion auszuschöpfen und dadurch die gegebenen Ressourcen stets bestmöglich zu nutzen, führt in der Praxis zu komplexen mathematischen Optimierungsproblemen. Selbst augenscheinlich kleine Planungsaufgaben stellen sich aus mathematischer Sicht als äußerst schwerwiegende Probleme dar. Computerbasierte Systeme, die im operativen Bereich Unterstützung bieten sollen, basieren häufig auf mathematischen Optimierungsmodellen und sind bezüglich ihrer Ausführungen in der Regel nicht trivial, so dass ein System nicht ohne weiteres von einem Produktionsprozess auf einen anderen übertragbar ist (Gondek, 2011, S. 1).

1.2 Zielsetzung und Vorgehensweise

Das Ziel dieser Arbeit ist die Entwicklung eines Verfahrens für die Lösung einer Klasse von Flexible Flow Shop Problemen, welches den Anforderungen für den Einsatz in der operativen Produktionsplanung und -steuerung von zahlreichen realen Produktionsbetrieben genügt. Bei den behandelten Problemen handelt es sich um Flexible Flow Shop Probleme mit

- beliebig vielen Produktionsstufen,
- beliebig vielen Maschinen in jeder Produktionsstufe und
- unterschiedlichen Maschinen in jeder Produktionsstufe.

Die behandelte Problemklasse wird in Kapitel 2 wesentlich durch die Berücksichtigung der folgenden Randbedingungen und Restriktionen charakterisiert:

- Nicht jeder Job muss in jeder Stufe bearbeitet werden und nicht jeder Job kann an jeder Maschine bearbeitet werden.
- Es sind reihenfolgeabhängige Rüstzeiten zu berücksichtigen.

- Die Fertigungsaufträge lassen sich in den einzelnen Stufen jeweils zu maschinenspezifischen Auftragsfamilien bündeln. Zwischen den Aufträgen einer Familie treten entsprechend keine Rüstzeiten auf (wie beispielsweise bei Werkstücken, welche in einer Lackierstraße mit der gleichen Farbe lackiert werden sollen). Desweiteren können Fertigungsaufträge an einer Maschine zu mehreren Auftragsfamilien gehören, so dass für die Bearbeitung eines Auftrags bei der Planung eine entsprechende Familie festgelegt werden muss (z.B. die Bearbeitung mit einem bestimmten Werkzeug oder Hilfsmittel).

- Die Pufferkapazitäten sowie die Rohmaterialien sind begrenzt.

- Die Fertigungsaufträge können in den einzelnen Produktionsstufen jeweils zu einem Auftragsstapel (Auftragsbündel) gehören. Die Fertigungsaufträge eines Auftragsstapels können in der entsprechenden Produktionsstufe nur zusammenhängend und am Stück (ohne Unterbrechung) produziert werden.

Die Ausarbeitung des Handlungsbedarfs sowie der angewendeten Methodik erfolgt in Kapitel 3. Der methodische Fokus des Lösungsverfahrens liegt dabei auf

- der simultanen Reihenfolge- und Maschinenbelegungsberechnung für

- die multikriterielle Optimierung von nahezu beliebigen Zielgrößendefinitionen mithilfe

- der Kopplung eines ereignisdiskreten Lösungsverfahrens mit beliebigen, geeigneten Optimierungsalgorithmen.

Die Anforderungen an das Lösungsverfahren lassen sich im Wesentlichen zurückführen auf die Forderung praktischer Einsetzbarkeit in operativen Entscheidungssystemen. So beziehen sich die geforderten Eigenschaften auf

- ein kontrollierbares Laufzeitverhalten,

- ein nachvollziehbares und steuerbares Lösungsverhalten sowie auf

- die Skalierbarkeit der Genauigkeit des Verfahrens.

Der Einsatz in einem Entscheidungssystem zur Unterstützung der operativen Produktionsplanung und -steuerung verlangt die immer wiederkehrende (rollierende) Planung auf Basis stets aktualisierter Betriebsdaten. Der Lösungsprozess muss daher in Abhängigkeit der Problemgröße innerhalb einer vorgegebenen Zeitspanne ein praktisch brauchbares Ergebnis liefern.

Die in der Produktion jeweils vorliegende Situation unterliegt ständigen kurzfristigen Veränderungen wie beispielsweise Materialanlieferungen, Fertigmeldungen, Bestandsbewegungen oder die Ankunft neuer Bestellungen. Als Planungshorizont wird daher ein kurzfristiger Zeitraum von $24 - 72\,\mathrm{h}$ angestrebt, welcher in einem Takt von beispielsweise

30 − 120 min rollierend neu geplant werden muss. Das Planungsergebnis eines Optimierungslaufs muss daher entsprechend schnell (beispielsweise nach 5 − 60 min) zur Verfügung stehen. Das Leistungsvermögen eines aktuellen, konventionellen PC-Systems (z.B. auf Basis einer Intel i3- oder i5-Hardware, siehe beispielsweise `http://www.intel.de`) soll dabei für die Durchführung des Verfahrens ausreichen.

Für die einzelnen Fertigungsaufträge sollen in einer verschachtelten Planung simultan sowohl die Maschinenbelegungen als auch die Produktionsreihenfolgen an den jeweiligen Maschinen optimiert werden. Dafür wird in Kapitel 4 ein Lösungsgenerator entwickelt. Es soll dabei immer von einer reinen Auftragsfertigung ausgegangen werden, so dass die entsprechenden Daten der Fertigungsaufträge (Termine, Stücklisten, Arbeitspläne, zu erwartende Bearbeitungszeiten etc.) als gegeben vorausgesetzt werden. Die Optimierung von Losgrößen, wie es beispielsweise in der Lagerfertigung der Fall ist (siehe beispielsweise Lödding, 2016), ist nicht Teil des Verfahrens.

Mit Bezug auf die „No-free-Lunch-Theoreme“ (Wolpert et al., 1997) ist für Probleme ab einer bestimmten Größe innerhalb einer begrenzten Zeit nicht mit einer mathematisch exakten Lösung zu rechnen. Das Generieren von mathematisch exakten Lösungen ist bei der Größe der angestrebten Problemdatensätze mit realistischem Zeitaufwand daher nicht möglich und wird auch nicht angestrebt. Das Ziel muss folglich die Generierung von optimierten, brauchbaren Lösungen innerhalb der vorgegebenen Zeit sein, da diese in der Praxis für die Planung und Steuerung einer Produktion in der Regel ausreichen.

Die Planung soll nach (z.B. auch für einen Produktionsleiter) nachvollziehbaren Kriterien und Regeln erfolgen. Planungsregeln und Paradigmen, welche sich in einem bestehenden Produktionsprozess bewährt haben, sollen in das Verfahren integriert werden können. Es muss daher eine Kombination aus einem Optimierungsverfahren mit den entsprechenden Planungsregeln entwickelt werden.

Die Komplexität – die Größe des Lösungsraums – der vorliegenden Problemklasse lässt für die Optimierung eine vollständige Enumeration des Lösungsraums nicht zu. Auch die Anwendung von deterministischen Suchverfahren (z.B. Simplex, Branch-and-Bound etc.) wird aus Gründen der Laufzeit und des Lösungsverhaltens nicht präferiert. Aus diesem Grund sollen für die Optimierung der Probleme in Kapitel 5 unterschiedliche Metaheuristiken als Suchalgorithmen angewendet werden.

Neben der Beeinflussbarkeit und Nachvollziehbarkeit durch Planungsregeln soll das Verfahren durch die Definition von Zielgrößen steuerbar sein. Der Suchprozess soll durch die Gewichtung der einzelnen Teilziele (z.B. Rückstand, Auslastung, Rüstzeiten etc.) als Bestandteile einer multikriteriellen logistischen Zielsetzung entsprechend adaptierbar bleiben.

Je nach vorliegender Hardware soll das Verfahren an die zur Verfügung stehende Rechenleistung angepasst werden können. Einerseits wird damit implizit die Unterstützung für die parallele Rechnung auf mehreren CPU-Kernen gefordert. Andererseits ist eine rasterbasierte Variation der „Genauigkeit“ bzw. der „Auflösung“ (wie beispielsweise das Gitter bzw. das Netz bei der numerischen Simulationen von Strömungen oder der FEM) zweckmäßig.

Der Nachweis über die Funktionsfähigkeit und Anwendbarkeit des Verfahrens soll in Kapitel 6 auf zwei Ebenen geführt werden.

1. Zunächst wird das Verfahren in Abschnitt 6.3 auf randomisierte Testdaten unterschiedlicher Problemgrößen angewendet. In umfangreichen Parameterstudien werden dabei die Parameter der Metaheuristiken sowie des eigentlichen Planungsverfahrens statistisch abgesichert optimiert. Auf diese Weise wird zunächst für unterschiedliche Problemgrößen festgestellt, welche Metaheuristik mit welchen Parameterausprägungen jeweils die besten Resultate liefert. Darauf aufbauend werden die Einflüsse weiterer Parameter des Verfahrens auf die erzielbaren Ergebnisse untersucht und optimiert.

2. Anschließend wird das Verfahren in Abschnitt 6.4 mit den optimierten Parametern der Metaheuristiken auf reale Daten eines konkreten Produktionsbetriebs angewendet. Die Verfahrensparameter des Planungsverfahrens werden dabei auf die realen Daten angepasst und optimiert. Durch die Anwendung des Planungsverfahrens mit den optimierten Verfahrensparametern in der operativen Produktionsplanung und -steuerung des Betriebs wird die praktische Einsatz- und Anwendbarkeit nachgewiesen.

2 Das Flexible Flow Shop Problem

Unter dem Begriff Flexible Flow Shop werden Planungsprobleme zusammengefasst, bei denen die zu produzierenden Produkte in mehreren Produktionsstufen bearbeitet werden müssen. In jeder Produktionsstufe stehen dafür eine oder mehrere Maschinen zur Verfügung, so dass neben der reinen Reihenfolgeplanung zusätzlich eine Maschinenauswahl für die Bearbeitung der Fertigungsaufträge stattfinden muss. Dies ist in Abbildung 2.1 (vgl. Gondek, 2011, Quadt et al., 2007) schematisch dargestellt.

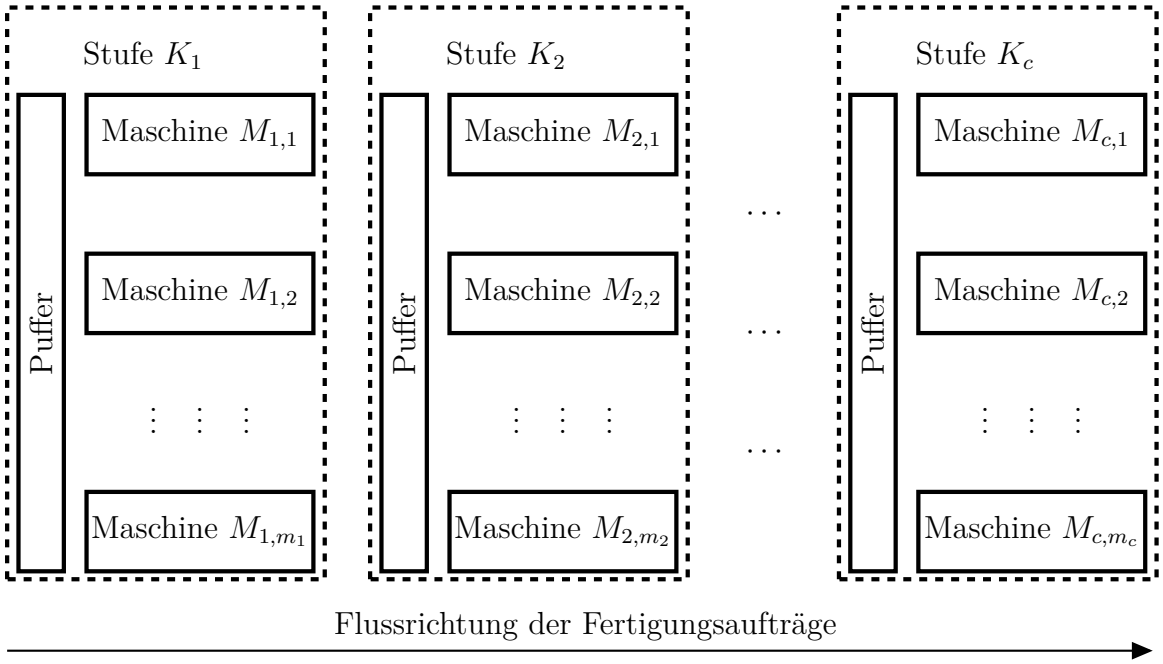


Abbildung 2.1: Schematische Darstellung von Flexible Flow Shop Problemen (vgl. Gondek, 2011, Quadt et al., 2007)

Produktionsbetriebe, die derartig organisiert sind, lassen sich in vielen realen Szenarien finden. Beispiele liefern Ruiz et al. (2010) u.a. mit der industriellen Produktion von Elektronikbauteilen, Papier und Textilien. Weiterhin nennen Ruiz et al. die Herstellung von Beton und fotografischem Filmmaterial.

2.1 Notation von Problemen der Ablaufplanung

Flexible Flow Shop Probleme gehören zu den Problemen der Ablaufplanung (engl.: **scheduling**, siehe Jaehn et al., 2019). In der Systematik der Ablaufplanung werden die

Planungsprobleme üblicherweise mithilfe der Dreifeldnotation (Jaehn et al., 2019, S. 17) klassifiziert. Die dabei verwendete Schreibweise (Formel 2.1, auch $\alpha|\beta|\gamma$-Schreibweise) geht auf die Arbeit von Graham et al. (1979) zurück.

$$\alpha\Big|\beta\Big|\gamma \tag{2.1}$$

Durch die drei Felder α, β und γ werden unabhängig voneinander die wesentlichen und charakteristischen Eigenschaften des entsprechenden Planungsproblems ausgedrückt (Graham et al., 1979).

- $\boldsymbol{\alpha}$ definiert dafür die vorliegende Maschinenumgebung. Es wird zwischen den Problemen mit nur einer Maschine (Einmaschinenprobleme), Problemen mit parallelen Maschinen (Parallelmaschinenprobleme) sowie Problemen mit seriellen und komplexeren Anordnungen mehrerer Maschinen (Shop Probleme) differenziert.
- $\boldsymbol{\beta}$ legt die zu berücksichtigenden Randbedingungen, Restriktionen und Ablaufeigenschaften fest. Diese können sowohl die Maschinenumgebung als auch die Eigenschaften der Fertigungsaufträge einschränken und spezifizieren.
- $\boldsymbol{\gamma}$ drückt die zu minimierende Zielfunktion aus. Die dafür zugrunde gelegten Optimierungsziele sind in der Regel zeitbasiert.

Sofern durch die Formulierung einer speziellen Problemklasse nichts anderes spezifiziert wird, gelten für alle Probleme zunächst die folgenden Grundvoraussetzungen.

- Alle notwendigen Daten sind deterministisch und stehen bei Beginn der Planung zur Verfügung (Baker et al., 2019, S. 12).
- Jede Maschine kann immer nur einen Fertigungsauftrag gleichzeitig bearbeiten und jeder Auftrag kann immer nur höchstens an einer Maschine gleichzeitig bearbeitet werden (Blazewicz et al., 2019, S. 61).
- Die Bearbeitung eines Fertigungsauftrags an einer Maschine kann nicht unterbrochen werden (Baker et al., 2019, S. 12).
- Es existieren keine Rüstzeiten, bzw. Rüstzeiten sind reihenfolgeunabhängig und bereits in der Prozesszeit enthalten (Esquirol et al., 2008, S. 20).
- Vor und hinter jeder Maschine steht beliebig viel Speicherplatz zur Verfügung, falls Fertigungsaufträge auf die weitere Bearbeitung an der nächsten Maschine warten müssen (Brucker et al., 2012, S. 240).

Durch die Berücksichtigung entsprechender Randbedingungen und Restriktionen (β-Feld) können diese Voraussetzungen teilweise ihre Gültigkeit verlieren.

Die im weiteren Verlauf dieses Kapitels verwendeten Definitionen, Begriffe, Symbole sowie deren Bedeutungen entstammen den Werken Baker et al. (2019), Blazewicz et al. (2019), Brucker (2007), Brucker et al. (2012), Emmons et al. (2013), Esquirol et al. (2008), Jaehn et al. (2019), Lopez (2008), Pinedo (2016) und Leung (2004).

2.1.1 Maschinenumgebungen

In der klassischen Dreifeldnotation nach Graham et al. kann das α-Feld eine der folgenden Ausprägungen annehmen (Pinedo, 2016, S. 14–15).

$$\alpha \in \{1, Pm, Qm, Rm, Fm, Jm, Om, FFc, FJc\} \tag{2.2}$$

a) Produktion mit einer Maschine

$\boldsymbol{\alpha = 1}$ legt fest, dass es sich um eine Produktionsumgebung mit nur **einer** Maschine handelt. Bei der Planung stellt die Optimierung der Bearbeitungsreihenfolge der Fertigungsaufträge auf dieser Maschine die einzige Stellgröße dar.

b) Produktion mit parallelen Maschinen

$\boldsymbol{\alpha = Pm}$ definiert eine Produktion mit m **identischen** (engl.: **identical**) und parallelen Maschinen. Falls nicht anders spezifiziert darf für jeden Fertigungsauftrag eine beliebige der zur Verfügung stehenden Maschinen ausgewählt werden. Die anzunehmende Bearbeitungszeit der Aufträge ist auf allen Maschinen identisch. Für jeden Job muss entsprechend die Auswahl einer Maschine und für jede Maschine muss jeweils die Bearbeitungsreihenfolge optimiert werden.

$\boldsymbol{\alpha = Qm}$ stellt ebenfalls eine Produktionsumgebung mit m parallelen Maschinen dar. In diesem Fall arbeiten die Maschinen mit **unterschiedlichen** Geschwindigkeiten, so dass für die Fertigungsaufträge unterschiedliche Produktionszeiten auf den einzelnen Maschinen anzunehmen sind. Die einzelnen Produktionszeiten lassen sich aus Auftragsvolumen und Maschinengeschwindigkeit immer systematisch und linear berechnen (engl.: **uniform**).

$\boldsymbol{\alpha = Rm}$ verallgemeinert die Produktion zu m parallelen und voneinander **unabhängigen** (engl.: **unrelated**) Maschinen, wobei die einzelnen Produktionszeiten sowohl von der jeweiligen Maschine als auch von den jeweiligen Fertigungsaufträgen abhängig sind. Für jede Kombination von Auftrag und Maschine muss entsprechend eine eigene Bearbeitungszeit angegeben werden.

c) Shop Probleme

$\boldsymbol{\alpha = Fm}$ definiert die Produktionsumgebung als **Flow Shop**. Dabei befinden sich m Maschinen in Bezug auf die Flussrichtung der Fertigungsaufträge hintereinander. Jeder Fertigungsauftrag muss in diesem Fall auf jeder Maschine bearbeitet werden. Die Maschinenreihenfolge ist dabei für jeden Fertigungsauftrag identisch und wird durch die Produktion vorgegeben.

$\boldsymbol{\alpha = Jm}$ verallgemeinert den Flow Shop dahingehend, dass jeder Fertigungsauftrag ebenfalls auf jeder der m Maschinen bearbeitet werden muss. Die Maschinenfolgen

sind im **Job Shop** allerdings nicht mehr durch die Produktion sondern individuell für jeden Fertigungsauftrag vorgegeben, so dass diese voneinander abweichen können.

$\boldsymbol{\alpha = Om}$ stellt eine weitere Verallgemeinerung des Job Shops dar. Im **Open Shop** darf die Bearbeitung der Jobs auf den m Maschinen in jeder beliebigen Reihenfolge erfolgen. Es muss allerdings weiterhin sichergestellt werden, dass jeder Fertigungsauftrag auf jeder Maschine bearbeitet wird.

$\boldsymbol{\alpha = FFc}$ erweitert den Flow Shop zum **Flexible Flow Shop**. Die Produktionsumgebung wird dabei von m Maschinen auf c Produktionsstufen mit jeweils mehreren Maschinen verallgemeinert. Jeder Fertigungsauftrag muss in diesem Fall in jeder Produktionsstufe an genau einer beliebigen Anlage bearbeitet werden. Wie beim Flow Shop ist die Stufenreihenfolge für jeden Auftrag identisch und durch die Produktionsumgebung vorgegeben.

$\boldsymbol{\alpha = FJc}$ erweitert die Produktion des Job Shops auf c Produktionsstufen. Im **Flexible Job Shop** ist entsprechend für jeden Job eine individuelle Stufenfolge vorgegeben, in welcher jeder Job jede Stufe passieren muss. Die Bearbeitung muss in jeder Stufe an genau einer beliebigen Maschine erfolgen.

Eine verallgemeinernde Erweiterung der $\alpha|\beta|\gamma$-Schreibweise nach Graham et al. geht auf die Arbeiten von Vignier et al. (1999) zurück. Diese Modifizierung (siehe Formel 2.3) erlaubt die wesentlich präzisere Angabe der behandelten Maschinenumgebung speziell bei Shop Problemen (siehe auch Ruiz et al., 2010).

$$\alpha = \alpha_1\alpha_2\left((\alpha_3\alpha_4)_{k=1}^{\alpha_2}\right) \tag{2.3}$$

$\boldsymbol{\alpha_1}$ gibt dabei die allgemeine Beschreibung des Produktionsprozesses an.

$$\alpha_1 \in \{\varnothing, F, FF, J, FJ, O\} \tag{2.4}$$

Es wird entweder ein (Flexible) Flow Shop (F, FF), ein (Flexible) Job Shop (J, FJ) oder ein Open Shop (O) angegeben. Wird das Feld leer gelassen ($\varnothing$), bezeichnet es kein Shop, sondern ein Einmaschinen- oder ein Parallelmaschinenproblem.

$\boldsymbol{\alpha_2}$ definiert die Größe der Produktionsumgebung. Bei einem Shop Problem wird damit die Anzahl der Stufen angegeben. Ansonsten wird dadurch die Anzahl der Maschinen angegeben.

$$\alpha_2 \in \mathbb{N} \tag{2.5}$$

$\boldsymbol{\alpha_3}$ beschreibt für jede vorhandene Stufe die Art der parallelen Maschinenausprägungen.

$$\alpha_3 \in \{\varnothing, P, Q, R\} \tag{2.6}$$

Die Maschinen einer Stufe können – wie bereits bei den einfachen Parallelmaschinenproblemen gezeigt – identisch (P), unterschiedlich schnell (Q) und komplett unterschiedlich (R) sein.

$\boldsymbol{\alpha_4}$ legt je Produktionsstufe die Anzahl der parallelen Maschinen fest.

$$\alpha_4 \in \mathbb{N} \tag{2.7}$$

In allgemeiner Form lässt sich ein beliebiges Problem der Ablaufplanung entsprechend Formel 2.8 durch die erweiterte Dreifeldnotation formulieren.

$$\alpha_1\alpha_2\left((\alpha_3\alpha_4)_{k=1}^{\alpha_2}\right)\Big|\beta\Big|\gamma \tag{2.8}$$

Diese Erweiterung der Dreifeldnotation ($\alpha|\beta|\gamma$-Schreibweise) erlaubt eine wesentlich exaktere Definition der entsprechenden Maschinenumgebungen im Vergleich zu der konventionellen Dreifeldnotation (Formel 2.1). Im weiteren Verlauf der Arbeit wird aufgrund ihrer Überlegenheit daher die erweiterte Dreifeldnotation (Formel 2.8) verwendet. Die Mächtigkeit dieser Erweiterung wird im Folgenden anhand von drei Beispielen gezeigt.

Beispiel 2.1 (Allgemeine Maschinenumgebungen eines Flexible Flow Shop Problems)
Die Maschinenumgebung eines allgemeinen Flexible Flow Shop Produktionsprozesses mit c Stufen und m_k unterschiedlichen Maschinen der Stufe k ist in Formel 2.9 definiert.

$$\alpha = FFc\left(Rm_k\right)_{k=1}^{c} \tag{2.9}$$

Es wird in diesem Fall lediglich der Maschinentyp der entsprechenden Produktionsstufen angegeben. Die Anzahl der Stufen sowie die Anzahl der Maschinen bleibt allgemein, so dass sich viele Anwendungen auf dieses Beispiel verallgemeinern lassen.

Beispiel 2.2 (Spezielle Maschinenumgebung eines Flexible Flow Shop Problems)
Wird die Maschinenumgebung aus Beispiel 2.1 weiter spezifiziert und die Anzahl der Produktionsstufen beispielsweise zu $c = 2$ gewählt sowie die Anzahl der Maschinen $m_1 = 4$ für die erste Stufe und $m_2 = 3$ für die zweite Stufe, lässt sich die Maschinenumgebung wie in Formel 2.10 formulieren.

$$\alpha = FF2\left(\left(R4\right)^{(1)}, \left(R3\right)^{(2)}\right) \tag{2.10}$$

Beispiel 2.3 (Spezielle Maschinenumgebung eines Flexible Flow Shop Problems)
Formel 2.11 formuliert eine Maschinenumgebung mit $c = 6$ Produktionsstufen.

$$\alpha = FF6\left(\left(R5\right)_{k=1}^{2}, \left(P3\right)_{k=3}^{5}, \left(Q4\right)^{(6)}\right) \tag{2.11}$$

Die Stufen 1 und 2 bestehen jeweils aus 5 unterschiedlichen Maschinen, in den Stufen 3 bis 5 befinden sich jeweils 3 identische Maschinen und die letzte Stufe 6 besteht aus 4 Maschinen mit unterschiedlichen Geschwindigkeiten.

2.1.2 Randbedingungen

Durch das β-Feld werden die im Planungsproblem zu berücksichtigenden Ablaufeigenschaften, Randbedingungen und Restriktionen spezifiziert. In Abhängigkeit des jeweiligen Planungsproblems darf das Feld leer bleiben oder entsprechend mehrere Werte gleichzeitig annehmen. Im Folgenden werden die üblicherweise untersuchten Ausprägungen zusammengefasst.

a) Ablaufbezogene Randbedingungen

$$\beta \in \{\text{prec}, \text{nwt}, \text{block/buffer}, \text{batch}, \text{pmtn}, \text{prmu}, \text{brkdwn}, \text{res}, \text{rcrc}, M_j\} \tag{2.12}$$

β = prec (precedence): Es existieren Vorrangbeziehungen zwischen den Fertigungsaufträgen. Der Bearbeitungsstart eines Jobs kann in diesem Fall durch das vorherige Bearbeitungsende von einem oder mehreren anderen Jobs abhängen. Taucht diese Bedingung nicht auf, können alle Fertigungsaufträge unabhängig voneinander gestartet werden.

β = nwt (no-wait): Diese Randbedingung kann bei Flow Shop Problemen auftreten. Dabei muss berücksichtigt werden, dass die Fertigungsaufträge zwischen ihren einzelnen Bearbeitungsschritten nicht warten dürfen. Wenn die Bearbeitung eines Jobs auf einer Maschine beendet ist, muss die nächste Maschine ohne Verzögerung mit der weiteren Bearbeitung des Jobs beginnen. Dies kann einerseits darauf zurückzuführen sein, dass zwischen den Maschinen oder Bearbeitungsstufen keine Puffer oder Speicher vorhanden sind. Andererseits kann aus technologischer Sicht das Warten zwischen den Maschinen ungünstige Folgen haben (z.B. Abkühlung in Schmiedeprozessen).

β = block/buffer (blocking/buffers): Diese Randbedingung betrifft ebenfalls Flow Shop Probleme. Zwischen den Maschinen bzw. Bearbeitungsstufen sind begrenzte Puffer- bzw. Speicherkapazitäten zu berücksichtigen. Ein Job kann eine Maschine (auch nach dessen Bearbeitungsende) in diesem Fall solange blockieren, bis im nächsten Puffer genügend Platz zur Speicherung vorhanden ist. Ist diese Randbedingung nicht zu berücksichtigen, darf zwischen den Maschinen beliebig viel Speicherkapazität angenommen werden.

β = batch (batch processing): Eine Maschine kann mehrere Fertigungsaufträge als Auftragsstapel gleichzeitig bearbeiten. Die Bearbeitungszeit von den einzelnen Jobs muss dabei nicht identisch sein. Die Gesamtbearbeitungsdauer des Stapels auf einer Maschine ergibt sich aus der längsten Bearbeitungszeit der einzelnen Jobs. Ohne diese Randbedingung kann jede Maschine immer nur maximal einen Job zur gleichen Zeit bearbeiten.

β = pmtn/prmp (preemption): Die Bearbeitung der Fertigungsaufträge darf unterbrochen werden. Die Jobs dürfen auf den einzelnen Maschinen beliebig oft gestartet, unterbrochen und anschließend weiterbearbeitet werden. Die insgesamt von der Maschine benötigte Bearbeitungszeit für diesen Job verlängert sich dabei nicht. Nach einer Unterbrechung darf entsprechend die bereits geleistete Bearbeitungszeit berücksichtigt werden. Ein Fertigungsauftrag muss nach dessen Start nicht zwangsläufig auf der entsprechenden Maschine bleiben, so dass eine verschachtelte Bearbeitung mehrerer Jobs auf einer Maschine ermöglicht wird. Tritt diese Randbedingung nicht auf, darf die Bearbeitung der Jobs an den Maschinen nicht unterbrochen werden.

β = **prmu (permutation):** In einem Flow Shop werden in diesem Fall sämtliche Puffer als strikte Warteschlangen behandelt. Die Reihenfolge in den Puffern kann dementsprechend nicht verändert werden und richtet sich nach dem FIFO-Prinzip. Es handelt sich folglich um eine reine Permutation eines Flow Shops, wobei die Bearbeitungsreihenfolge der Jobs einmalig festgelegt wird und sich die Jobs nicht mehr gegenseitig überholen können. Ohne diese Randbedingung können die Reihenfolgen in den Zwischenpuffern verändert werden und Fertigungsaufträge können sich gegenseitig überholen.

β = **brkdwn (machine breakdowns):** Es sind Ausfälle von Maschinen zu berücksichtigen. Es wird üblicherweise davon ausgegangen, dass es sich dabei um deterministische und geplante Ereignisse wie beispielsweise Wartungsarbeiten handelt.

β = **res (resources):** Für die Bearbeitung der Jobs werden neben den Maschinen noch weitere begrenzte Ressourcen benötigt. Diese können beispielsweise Personal, Werkzeuge, Materialien oder Lagerplatz berücksichtigen. Entsprechend ist zwischen diskreten (Stückgut, Personal etc.) und kontinuierlichen (Flüssigkeiten, Schüttgut etc.) Ressourcen zu differenzieren.

β = **rcrc (recirculation):** In einem Flow Shop oder in einem Job Shop kann in diesem Fall ein Fertigungsauftrag mehrfach hintereinander auf einer Maschine bearbeitet werden. Im Gegensatz dazu darf im Normalfall jeder Job an jeder Maschine nur einmal bearbeitet werden.

$\beta = M_j$ **(machine eligibility):** Durch diese Randbedingung wird berücksichtigt, dass in einer Maschinenumgebung mit parallelen Maschinen nicht jede Maschine in der Lage ist, jeden Auftrag zu bearbeiten. Dafür können beispielsweise technologische (eine Maschine kann die geforderte Toleranz nicht erfüllen) oder geometrische Aspekte (ein Bauteil ist zu groß für eine Maschine) zugrunde gelegt werden. Ohne diese Restriktion darf davon ausgegangen werden, dass jeder Job an jeder Maschine bearbeitet werden kann.

b) Zeitbezogene Randbedingungen

$$\beta \in \{s_{i,j_1,j_2}, r_j, p_j = p, d_j = d, \text{fmls}, \text{time} - \text{lags}, \text{transport}\} \tag{2.13}$$

$\beta = r_j$ **(release dates):** Ankunftszeiten und Freigabezeiten müssen berücksichtigt werden, so dass nicht für jeden Auftrag zu jedem Zeitpunkt mit der Produktion begonnen werden kann. Erst ab dem jeweiligen Freigabezeitpunkt r_j darf der entsprechende Job gestartet werden.

$\beta = d_j = d$ **(due dates):** Alle Fertigungsaufträge haben den gleichen Planliefertermin.

$\beta = p_j = p$ **(production times):** Alle Fertigungsaufträge haben die gleiche Produktionszeit.

$\beta = s_{i,j_1,j_2}$ **(sequence dependent setup times):** Es sind reihenfolgeabhängige Rüstzeiten zwischen den Jobs an den Maschinen zu berücksichtigen. Folgt an Maschine i Job j_2 direkt auf Job j_1, dann fällt die Zeit s_{i,j_1,j_2} zum Umrüsten der Maschine an. Die entsprechenden Rüstzeiten sind folglich abhängig von der jeweiligen Maschine, den Jobs sowie deren Reihenfolge. Ohne diese Randbedingung sind nur reihenfolgeunabhängige Rüstzeiten durch eine entsprechende Verlängerung der Bearbeitungszeit zu berücksichtigen.

$\beta =$ **fmls (job families):** Durch diese Restriktion werden die Fertigungsaufträge in Auftragsfamilien eingeteilt. Innerhalb einer Auftragsfamilie können zwar unterschiedliche Bearbeitungszeiten auftreten, zwischen den Jobs einer Familie treten allerdings keine Rüstzeiten auf. Bei der Lackierung von Bauteilen lassen sich beispielsweise Auftragsfamilien in Abhängigkeit des Lackes bilden. Zwischen Bauteilen mit der gleichen Farbe muss die Anlage entsprechend nicht gereinigt werden.

$\beta =$ **delays (delays):** Zwischen den Bearbeitungsschritten eines Fertigungsauftrags müssen Zeitverzögerungen berücksichtigt werden. Diese können z.B. auf Transportzeiten zurückzuführen sein.

2.1.3 Zielgrößen

Das Planungsziel wird im γ-Feld durch die Angabe genau einer Zielsetzung spezifiziert. Die angegebene Zielgröße ist dabei stets zu minimieren. Für die Angabe multikriterieller Zielsetzungen können Linearkombinationen einzelner Teilziele verwendetet werden. Im Folgenden werden auch für das γ-Feld die üblichen Ausprägungen eingeführt.

a) Ungewichtete Summen

$$\gamma \in \left\{ \sum C_j, \sum L_j, \sum T_j, \sum U_j \right\} \tag{2.14}$$

$\gamma = \sum C_j$ **(completion time):** Die Summe der Fertigstellungszeitpunkte der einzelnen Fertigungsaufträge wird minimiert.

$\gamma = \sum L_j$ **(lateness):** Die Summe der Verspätungen wird minimiert. Es ist zu beachten, dass dabei auch Verfrühungen (also negative Verspätungen) einberechnet werden. Fertigungsaufträge mit positiver Verspätung werden in diesem Fall entsprechend durch Fertigungsaufträge mit negativer Verspätung kompensiert.

$\gamma = \sum T_j$ **(tardiness):** Die summierten Terminüberschreitungen der Jobs werden minimiert. Die Größe der Terminüberschreitung berücksichtigt dabei nur positive (also nur reale) Verspätungen.

$\gamma = \sum E_j$ **(earliness):** Die Summe der verfrühten Fertigstellungszeitpunkte wird reduziert. Es handelt sich dabei um den Betrag der negativen Verspätungen.

$\gamma = \sum \boldsymbol{U_j}$ **(unit penalty):** Die Anzahl der Terminüberschreitungen wird minimiert.

b) Gewichtete Summen

Analog zu der Minimierung der ungewichteten Summen lassen sich die Optimierungsziele durch auftragsbezogene Gewichte verallgemeinern. Die Zielgrößen auf Basis der einzelnen Fertigungsaufträge (siehe Formel 2.14) werden entsprechend mit der auftragsspezifischen Größe w_j gewichtet.

$$\gamma \in \left\{ \sum w_j C_j, \sum w_j L_j, \sum w_j T_j, \sum w_j U_j \right\} \tag{2.15}$$

Dadurch lassen sich in der Zielsetzung die Einflüsse der einzelnen Aufträge derart gewichten, dass beispielsweise die Verspätung eines Auftrags mit einem größeren Gewicht die Zielgröße schwerwiegender beeinflusst als ein Auftrag mit niedrigerem Gewicht. Auf diese Weise können z.B. verschiedene Prioritäten oder unterschiedliche Kostenfaktoren der Fertigungsaufträge einbezogen werden (Jaehn et al., 2019, Pinedo, 2016, S. 21, S. 19).

c) Maximalwerte

Ergänzend zu der Minimierung von durch Summenfunktionen ausgedrückte Zielgrößen ist die Minimierung eines Maximalwertes ein häufiger Untersuchungsgegenstand.

$$\gamma \in \{C_{\max}, L_{\max}\} \tag{2.16}$$

$\gamma = \boldsymbol{C_{\max}}$**(makespan):** Von allen eingeplanten Fertigungsaufträgen wird derjenige mit dem spätesten Fertigstellungszeitpunkt betrachtet. Der späteste Fertigstellungszeitpunkt entspricht der Zykluszeit. Diese soll dabei minimiert werden.

$\gamma = \boldsymbol{L_{\max}}$ **(maximum lateness):** Analog zu der Minimierung der Zykluszeit wird die maximale Terminabweichung minimiert.

2.2 Definition der untersuchten Problemklasse

Die in dieser Arbeit behandelte und in Abschnitt 1.2 bereits eingeführte Klasse von Flexible Flow Shop Problemen lässt sich mit der in Abschnitt 2.1 eingeführten erweiterten Dreifeldnotation (Formel 2.8) wie folgt formulieren:

$$FFc\,(Rm_k)_{k=1}^{c} \Big| M_j, s_{k,i,j_1,j_2}, \text{fmls}, \text{batch}, \text{block}, \text{res} \Big| w_1'\gamma_1 + w_2'\gamma_2 + w_3'\gamma_3 + w_4'\gamma_4 \tag{2.17}$$

2.2.1 Reihenfolge- und Maschinenbelegungsproblem

Die Angabe der Maschinenumgebung des Reihenfolge- und Maschinenbelegungsproblems ist in Formel 2.17 durch den folgenden Teilausdruck gegeben:

$$\alpha = FFc\,(Rm_k)_{k=1}^{c}$$

In der Maschinenumgebung des Planungsproblems werden beliebig viele Produktionsstufen mit jeweils beliebig vielen und unterschiedlichen Maschinen berücksichtigt. Jeder Fertigungsauftrag muss in der gleichen (technologischen) Produktionsreihenfolge durch die Stufen bearbeitet werden. Für jede Maschine in jeder Bearbeitungsstufe sind die entsprechenden Produktionsreihenfolgen zu berechnen. Gleichzeitig muss jeder Job in den entsprechenden Stufen einer jeweiligen Maschine zugeordnet werden. Durch die allgemeine Formulierung mit unterschiedlichen Maschinen ist für die Bearbeitung eines Fertigungsauftrags auf jeder Maschine eine eigene Bearbeitungszeit anzunehmen.

Für jede Bearbeitungsstufe ist ein Bereitstellungspuffer zu berücksichtigen (siehe Abbildung 2.1). In diesen Puffern können bereits bei Planungsbeginn Fertigungsaufträge gespeichert sein. Die Jobs, die während der Planung die jeweils vorgelagerten Bearbeitungsstufen passieren, werden den entsprechenden Speichern hinzugefügt. Diese müssen entsprechend im weiteren Planungsverlauf für die jeweilige Stufe berücksichtigt werden.

2.2.2 Randbedingungen und Restriktionen

Die Randbedingungen und Restriktionen der Problemklasse aus Formel 2.17 sind die folgenden:

$$\beta = M_j, s_{k,i,j_1,j_2}, \text{fmls}, \text{batch}, \text{block}, \text{res}$$

a) Prozessmatrix und technologische Kompatiblitäten

$$\beta_1 = M_j$$

Es wird davon ausgegangen, dass technologisch bedingt nicht jede Maschine für die Bearbeitung von allen Fertigungsaufträgen zulässig ist. Auch die Kompatibilität zwischen den Jobs und den Maschinen soll folglich in einer entsprechenden Matrix festgelegt sein. Weiterhin wird davon ausgegangen, dass nicht jeder Job in jeder Stufe bearbeitet werden muss. Über eine Matrix soll für jeden Fertigungsauftrag festgelegt sein, in welcher Produktionsstufe dieser bearbeitet werden muss.

b) Reihenfolgeabhängige Rüstzeiten mit maschinenspezifischen Auftragsfamilien

$$\beta_{2,3} = s_{k,i,j_1,j_2}, \text{fmls}$$

An jeder Maschine sind grundsätzlich reihenfolgeabhängige Rüstzeiten zu berücksichtigen. Die Zeit, die an einer Maschine für das Umrüsten für die Bearbeitung eines Fertigungsauftrags notwendig ist, ist entsprechend abhängig von dem direkt zuvor bearbeiteten Fertigungsauftrag. Die dafür notwendigen Rüstzeiten sind allerdings nicht allein durch die beiden am Rüstvorgang beteiligten Aufträge ableitbar.

Die Fertigungsaufträge werden in maschinenspezifische Auftragsfamilien eingeteilt. Jede Auftragsfamilie einer Maschine stellt dabei eine Konfiguration (einen Rüstzustand) dieser Maschine dar, so dass die Familien für jede Maschine unabhängig von den Familien der anderen Maschinen definiert sind. Ein Job kann dabei an den einzelnen Maschinen

jeweils mit mehreren Familien kompatibel sein und entsprechend mit unterschiedlichen Maschinenkonfigurationen bearbeitet werden.

Es lässt sich daher nicht ohne weiteres festlegen, welche Rüstzeit zwischen zwei Jobs zu erwarten ist, da diese zusätzlich von der gewählten Maschinenkonfiguration für die jeweilige Bearbeitung abhängig ist. Die Maschinen lassen sich dementsprechend nicht auf Fertigungsaufträge sondern auf Konfigurationen (Auftragsfamilien) anrüsten. Beispielsweise könnte für ein Werkstück festgelegt sein, dass es mit unterschiedlichen Lacken grundiert werden darf. Das Umrüsten einer entsprechenden Lackierstraße hängt folglich sowohl von dem zuletzt lackierten Werkstück als auch von dem aktuell auf der Maschine gerüsteten Lack ab.

Bei der Einplanung eines Fertigungsauftrags an einer Maschine muss durch das Planungsverfahren entsprechend zusätzlich festgelegt werden, mit welcher Konfiguration die Bearbeitung durchgeführt wird. In der Planung entsteht dadurch ein zusätzlicher Freiheitsgrad, was zu einer Vergrößerung des Lösungsraums führt. Die Kompatiblitäten zwischen den Fertigungsaufträgen und den Auftragsfamilien sowie die resultierenden Rüstzeiten zwischen den maschinenspezifischen Auftragsfamilien sollen dem Verfahren über entsprechende Matrizen vorgegeben werden.

c) Materialflusskonvergenz und -divergenz durch stufenspezifische Stapelbearbeitung

$$\beta_4 = \text{batch}$$

Bei den üblichen Problemen der Ablaufplanung wird bei einem Job meist davon ausgegangen, dass es sich nur um ein einzelnes Werkstück handelt. Dieses bewegt sich analog zu dem Job von Stufe zu Stufe physisch durch die Produktion. Bei den im Rahmen dieser Arbeit behandelten Produktionssystemen sollen Szenarien mit divergierenden und konvergierenden Materialflüssen berücksichtigt werden. Dies berücksichtigt den Fall, dass zwei oder mehrere Werkstücke durch ein Fügeverfahren zu einem weiteren Werkstück vereint werden (Konvergenz) oder dass ein Werkstück durch ein Trennverfahren in zwei oder mehrere Werkstücke (Jobs) aufgeteilt wird (Divergenz).

Für die Abbildung dieser Möglichkeiten in den Flexible Flow Shop Problemen wird an dieser Stelle die stufenspezifische Stapelbearbeitung als Randbedingung berücksichtigt. Für Produktionsstufen, in denen die Jobs mit einem Trenn- oder einem Fügeverfahren bearbeitet werden, werden die jeweils beteiligten Fertigungsaufträge zu einem zusammenhängenden Auftragsstapel gebündelt. In Abbildung 2.2 (vgl. Suerie, 2005, S. 12) ist dies für zwei Beispiele dargestellt.

Die beiden Jobs J_1 und J_2 sind in beiden Beispielen nach dem Trennen bzw. vor dem Fügen unabhängig voneinander planbar. Vor dem Trennen bzw. nach dem Fügen wird die entsprechende Abhängigkeit voneinander durch das Bündeln in einem gemeinsamen Auftragsstapel (Batch) B_1 berücksichtigt.

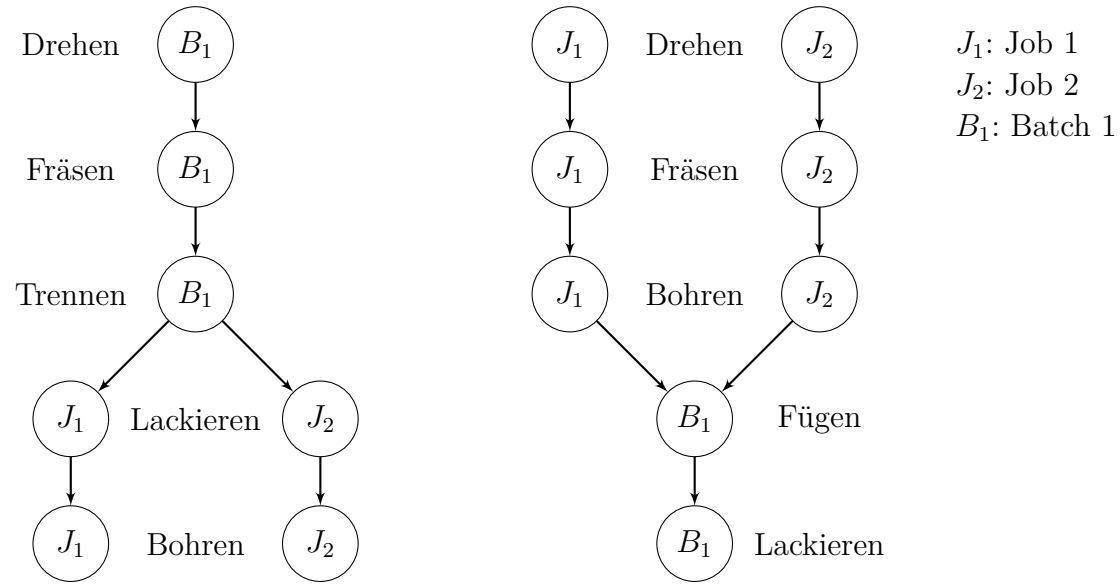


Abbildung 2.2: Modellierung von Materialflussmöglichkeiten durch Stapelbearbeitung

Die im Rahmen dieser Arbeit behandelten Probleme bleiben durch den in Abbildung 2.2 dargestellten Mechanismus allgemein formulierbar, so dass beide Möglichkeiten (Konvergenz und Divergenz) berücksichtigt werden können.

d) Begrenzte Ressourcen und begrenzte Kapazitäten der Logistikpuffer

$$\beta_5 = \text{res}, \text{block}$$

Der Lagerplatz sowohl für Rohmaterialien (Ressourcen) als auch für die Speicherung von Fertigungsaufträgen zwischen den Produktionsstufen (Logistikpuffer) ist in der Realität begrenzt. Um durch ein Planungsverfahren für die Praxis brauchbare Ergebnisse berechnen zu können, müssen diese Einschränkungen entsprechend berücksichtigt werden. Über eine Matrix soll daher für jede Ressource der zu Beginn der Planung verfügbare Bestand festgelegt werden. Über eine weitere Matrix sollen die Bedarfe der einzelnen Fertigungsaufträge an den jeweiligen Ressourcen festgelegt werden. Für die zwischen den Fertigungsstufen befindlichen Puffer (siehe auch Abbildung 2.1) sind die entsprechenden Pufferkapazitäten anzugeben.

2.2.3 Zielsetzung

Die in Formel 2.17 definierte Problemklasse basiert auf der folgenden Zielsetzung:

$$\gamma = w'_1\gamma_1 + w'_2\gamma_2 + w'_3\gamma_3 + w'_4\gamma_4$$

Die Zielsetzung ist als eine Linearkombination bestehend aus vier gewichteten Teilzielen formuliert. Damit soll das Planungsziel durch verschiedene Gewichtungen variabel bleiben und je nach Bedarf den entsprechenden Situationen angepasst werden können. Um eine dimensionslose Größe γ zu erhalten, sollen die Gewichte in der Einheit $\frac{1}{\mathrm{h}}$ angegeben werden.

$w'_1\gamma_1$: Minimierung der gewichteten Summe der Maschinenstillstandszeiten

$w'_2\gamma_2$: Minimierung der gewichteten Summe der Durchlaufzeiten der Fertigungsaufträge

$w'_3\gamma_3$: Minimierung der gewichteten Summe der resultierenden Rüstzeiten

$w'_4\gamma_4$: Minimierung der gewichteten Summe der Verspätungen der Fertigungsaufträge

2.3 Komplexität

Für das vorliegende Planungsproblem soll im Folgenden ein Maß für die zu erwartende Komplexität angegeben werden.

2.3.1 Komplexitätstheorie

Die Komplexität eines mathematischen Optimierungsproblems lässt sich durch die zu erwartende Laufzeit eines entsprechenden Lösungsalgorithmus ausdrücken. Für diesen Zweck hat sich das Konzept der Turingmaschine nach Alan Turing durchgesetzt (Turing, 1937).

Die Turingmaschine ist die theoretische Nachbildung eines Computers, der ein einziges Programm gleichzeitig abspielen kann und dem unendlich viel Arbeitsspeicher zur Verfügung steht. Für die Laufzeitberechnung von Algorithmen werden entsprechend die Rechenschritte der zugehörigen Turingmaschine bestimmt (Jaehn et al., 2019, S. 30).

In Abhängigkeit der Eingabelänge n – der Größe – eines Problems ergibt sich die Laufzeit $T(n)$ der zugehörigen Turingmaschine. Eine exakte Bestimmung der Laufzeit steht dabei nicht im Vordergrund. Vielmehr ist das Wachstumsverhalten der Laufzeit in Abhängigkeit steigender Problemgrößen n von Interesse (Dempe et al., 2006, S. 345). Diese wird üblicherweise in der Notation nach Edmund Landau ($\mathcal{O}$- und Ω-Schreibweise) angegeben (siehe beispielsweise Taraz, 2012, S. 35). Die Laufzeit einer Turingmaschine lässt sich dafür klassifizieren in (siehe Jaehn et al., 2019, S. 33):

Lineare Laufzeit ($T(n) \in \mathcal{O}(n)$): Es gibt eine natürliche Zahl $a \in \mathcal{N}$, sodass $T(n) \leq a \cdot n \; \forall n \in \mathrm{N}$ gilt.

Polynomielle Laufzeit ($T(n) \in \mathcal{O}\left(n^b\right)$): Es gibt zwei natürliche Zahlen a,$b \in \mathrm{N}$, sodass $T(n) \leq a \cdot n^b \; \forall n \in \mathrm{N}$ gilt.

Exponentielle Laufzeit ($T(n) \in \Omega(a^n)$): Es gibt eine natürliche Zahl $a \in \mathcal{N}, a > 1$, sodass $T(n) \geq a^n \; \forall n \in \mathrm{N}$ gilt.

Basierend auf den Laufzeitklassen werden die beiden Komplexitätsklassen $\mathcal{P}$ und $\mathcal{NP}$ gebildet. Probleme, die in der Klasse $\mathcal{P}$ liegen, lassen sich effizient in polynomieller Laufzeit mithilfe von Rechnerprogrammen optimal lösen. Für Probleme, die in der Klasse $\mathcal{NP}$ liegen (und gleichzeitig nicht in $\mathcal{P}$, siehe dazu Jaehn et al., 2019, S. 37), ist für die Suche nach einer optimalen Lösung hingegen – mindestens – mit einer exponentiellen Laufzeit zu rechnen. Diese Probleme werden als **$\mathcal{NP}$-schwer** bezeichnet und gehören damit zu den komplexesten (schwerwiegendsten) Problemen der mathematischen Optimierung (Kallrath, 2013, S. 116).

2.3.2 Abschätzung der Komplexität

Bereits die Optimierung der Produktionsreihenfolge bei einem Problem mit nur einer Maschine ist unter der Berücksichtigung von reihenfolgeabhängigen Rüstzeiten $\mathcal{NP}$-schwer (Jaehn et al., 2019, S. 48). Für die Komplexität κ des Reihenfolgeproblems an einer Maschine wird entsprechend die Anzahl der Kombinationsmöglichkeiten – die Permutationen – einer vollständigen Enumeration herangezogen. Diese beläuft sich für ein Problem mit nur einer Maschine und n Jobs auf $\kappa = n!$ (Jaehn et al., 2019, S. 119).

Das Flexible Flow Shop Problem der vorliegenden Arbeit ist entsprechend ebenfalls $\mathcal{NP}$-schwer. Aufgrund der gegenseitigen Wechselwirkungen und Interdependenzen zwischen den kombinierten Reihenfolge- und Maschinenzuordnungsproblemen in den einzelnen Stufen ist selbst ein Maß für die Anzahl der Lösungsmöglichkeiten nicht in geschlossener Form darstellbar. Auch in der Fachliteratur finden sich nur wenige bis gar keine Hinweise auf mögliche Werte bzw. Ansätze. Im Folgenden wird daher eine rekursive Berechnungsvorschrift hergeleitet, mit welcher eine Abschätzung der entsprechenden Größenordnung möglich ist.

Bei der reinen Permutation von n Jobs auf einer Maschine gilt für die Komplexität κ:

$$\kappa_{(1)}^{(n)} = n! \tag{2.18}$$

Das Problem wird um eine zweite – parallele – Maschine erweitert. Jeder Job j, der an einer der beiden Maschinen bearbeitet wird, kann nicht mehr an der anderen bearbeitet werden. Die Komplexität für die möglichen Maschinenbelegungen und Bearbeitungsreihenfolgen lässt sich gemäß Formel 2.19 berechnen.

$$\kappa_{(2)}^{(n)} = \sum_{j=0}^{n} \left((n-j)! \cdot j! \right) \tag{2.19}$$

Allgemein lässt sich die Komplexität für die Belegung und Reihenfolgebildung von m parallelen Maschinen mit n Jobs durch die rekursive Berechnungsvorschrift aus Formel 2.20 ausdrücken:

$$\kappa_{(m)}^{(n)} = \sum_{j=0}^{n} \left((n-j)! \cdot \kappa_{(m-1)}^{(j)} \right) \tag{2.20}$$

Durch die wiederholte Anwendung der Rechenvorschritt aus Formel 2.20 lässt sich für jede Stufe eines Flexible Flow Shop Problems die Komplexität des entsprechenden Parallelmaschinenproblems berechnen. Durch Multiplikation kann mithilfe der Formel 2.21 die Komplexität eines Flexible Flow Shop Problems mit c Produktionsstufen und m_k Maschinen in der k-ten Stufe abgeschätzt werden.

$$\kappa_{(FFc)}^{(n)} = \prod_{k=1}^{c} \left(\kappa_{(m_k)}^{(n)} \right) = \prod_{k=1}^{c} \left(\sum_{j=0}^{n} \left((n-j)! \cdot \kappa_{(m_k-1)}^{(j)} \right) \right) \tag{2.21}$$

Dabei muss vereinfachend angenommen werden, dass jeder Job in jeder Stufe bearbeitet werden muss und jeder Job auf jeder Maschine bearbeitet werden kann. Weiterhin muss an dieser Stelle vernachlässigt werden, dass Jobs unter Umständen als zusammenhängender Auftragsstapel verarbeitet werden müssen und mit unterschiedlichen Konfigurationen bearbeitet werden können. Eine Angabe der Komplexität ist bei Berücksichtigung dieser Einschränkungen nicht in allgemeiner Form möglich.

Beispiel 2.4 (Zahlenbeispiele für Komplexitäten unterschiedlicher Problemgrößen)
Die Formeln 2.22 bis 2.25 zeigen das drastische Wachstum der Komplexität von Flexible Flow Shop Problemen.

$$\kappa_{(FFc)}^{(50)} \left(FF2 \left((R2)_{k=1}^{2} \right) \right) = 3{,}86 \cdot 10^{129} \tag{2.22}$$

$$\kappa_{(FFc)}^{(50)} \left(FF2 \left((R4)_{k=1}^{2} \right) \right) = 1{,}68 \cdot 10^{130} \tag{2.23}$$

$$\kappa_{(FFc)}^{(50)} \left(FF4 \left((R4)_{k=1}^{4} \right) \right) = 2{,}81 \cdot 10^{260} \tag{2.24}$$

$$\kappa_{(FFc)}^{(100)} \left(FF4 \left((R4)_{k=1}^{4} \right) \right) = 2{,}19 \cdot 10^{634} \tag{2.25}$$

In Abhängigkeit der Anzahl der Produktionsstufen (Formeln 2.22 und 2.23: zwei Stufen, Formeln 2.24 und 2.25: vier Stufen) und der Gesamtzahl der Maschinen (Formel 2.22: vier Maschinen ($2 \cdot 2 = 4$), Formel 2.23: acht Maschinen ($2 \cdot 4 = 8$), Formeln 2.24 und 2.25: 16 Maschinen ($4 \cdot 4 = 16$)) sowie mit steigender Anzahl der Produktionsaufträge (Formeln 2.22 bis 2.24: 50 Aufträge, Formel 2.25: 100 Aufträge) ist das exponentielle Wachstum der Komplexität zu beobachten.

2.4 Stand der Forschung

Die Forschung auf dem Gebiet der Ablaufplanung basiert wesentlich auf den Werkzeugen des Operations Research. Die im Rahmen dieser Arbeit verwendeten Methoden des Operations Research umfassen

- die Formulierung eines gemischt-ganzzahligen Optimierungsmodells,
- die Anwendung metaheuristischer Suchverfahren sowie
- die Entwicklung eines Lösungsverfahrens auf Basis eines ereignisdiskreten Simulators.

Diese Methoden werden im Folgenden daher zunächst eingeführt, um darauf aufbauend den Forschungsstand von Flexible Flow Shop Problemen darzustellen.

2.4.1 Gemischt-ganzzahlige Optimierung

Flexible Flow Shop Probleme lassen sich in Form von gemischt-ganzzahligen Optimierungsmodellen formulieren (siehe z.B. Naderi et al., 2014). Gemischt-ganzzahlig bedeutet in diesem Zusammenhang, dass sowohl Variablen mit reellen Werten (z.B. Produktionszeiten oder Termine) als auch Variablen mit ganzzahligen Werten (z.B. Variablen für die Angabe von Bearbeitungsreihenfolgen, oder für die Angabe, ob ein Job auf einer Maschine gefertigt wird) Teil des Optimierungsmodells sind. In Abhängigkeit der jeweiligen Problemstruktur handelt es sich dabei entweder um lineare bzw. um nichtlineare Optimierungsprobleme.

Lineare Optimierung: Sowohl die Zielfunktion als auch alle Nebenbedingungen hängen linear von den Eingangsgrößen ab (Werners, 2013, S. 45).

Nichtlineare Optimierung: Entweder die Zielfunktion oder mindestens eine der Nebenbedingungen hängen nicht linear (z.B. quadratisch) von den Eingangsgrößen ab (Domschke et al., 2005, S. 175).

Formel 2.26 liefert einen allgemeinen Formalismus, um gemischt-ganzzahlige Optimierungsprobleme zu formulieren (in Anlehnung an Kallrath, 2013, S. 14).

$$\min \left\{ f\left(\overline{x},\overline{y}\right) \middle| \begin{pmatrix} \overline{h}\left(\overline{x},\overline{y}\right) = 0 \\ \overline{g}\left(\overline{x},\overline{y}\right) \geq 0 \end{pmatrix}, \begin{pmatrix} \overline{h} : X \times Y \to \mathbb{R}^{n_e} \\ \overline{g} : X \times Y \to \mathbb{R}^{n_i} \end{pmatrix}, \begin{pmatrix} \overline{x} \in X \subseteq \mathbb{R}^{n_c} \\ \overline{y} \in Y \subseteq \mathbb{Z}^{n_d} \end{pmatrix} \right\} \tag{2.26}$$

Es handelt sich dabei um die Minimierung der Funktion $f\left(\overline{x},\overline{y}\right)$, welche von den beiden Vektoren $\overline{x}$ und $\overline{y}$ abhängt. Durch die Funktionen $\overline{g}\left(\overline{x},\overline{y}\right)$ und $\overline{h}\left(\overline{x},\overline{y}\right)$ werden die Randbedingungen und Restriktionen des Problems ausgedrückt. Bei den Komponenten des Vektors $\overline{x}$ handelt es sich um reellwertige Größen, während es sich bei den Komponenten des Vektors $\overline{y}$ um ganzzahlige Größen handelt. Gesucht sind Werte für die entsprechenden Vektorkomponenten, so dass einerseits der Funktionswert $f\left(\overline{x},\overline{y}\right)$ minimiert wird und weiterhin die Nebenbedingungen $\overline{g}\left(\overline{x},\overline{y}\right) \geq 0$ und $\overline{h}\left(\overline{x},\overline{y}\right) = 0$ nicht verletzt werden. Die Einhaltung der Nebenbedingungen ist dabei komponentenweise zu lesen (Kallrath, 2013, S. 14).

Sind in dem Problem erheblich mehr ganzzahlige als reelwertige Variablen vorhanden, handelt es sich nach Kallrath (2013, S. 2) um ein kombinatorisches Optimierungsproblem.

2.4.2 Suchverfahren

Die kombinatorische Optimierung ist ein Suchprozess, in dem optimale Lösungen für ein entsprechendes kombinatorisches Optimierungsproblem gesucht werden. Du et al. (2016) teilen die zur Verfügung stehenden Methoden dafür in drei Mechanismen ein, so dass zwischen analytischen, heuristischen und enumerativen Techniken differenziert wird (siehe Abbildung 2.3). Da kombinatorische bzw. gemischt-ganzzahlige Optimierungsprobleme im Allgemeinen weder stetig noch differenzierbar sind, kommen analytische Methoden in der Regel nicht zum Einsatz und werden im Rahmen dieser Arbeit daher nicht weiter vertieft.

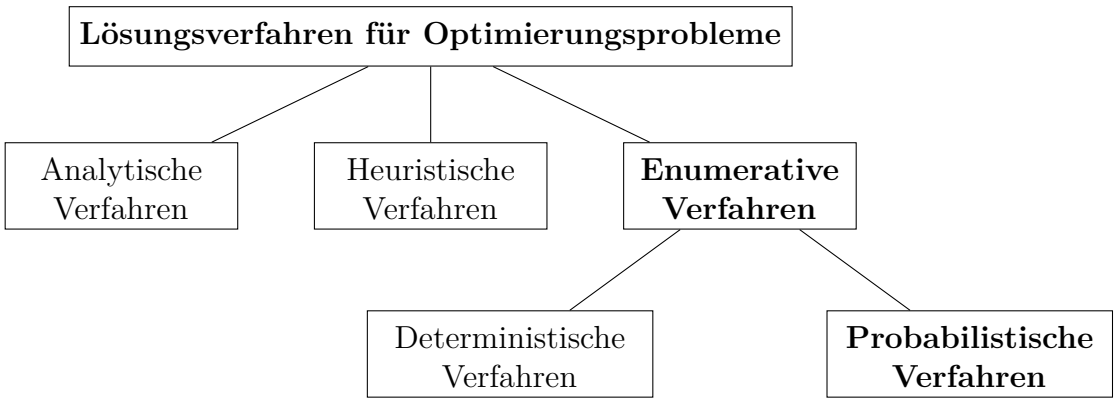


Abbildung 2.3: Klassifikation von Suchverfahren

Heuristische Verfahren sind Verfahren, welche durch Informationen über die jeweiligen Optimierungsprobleme zulässige Lösungen konstruieren. Heuristiken sind dementsprechend problemabhängige Lösungsmethoden und liefern in der Regel relativ gute Lösungen innerhalb einer relativ kurzen Zeit. In den meisten Fällen kann keine Garantie oder ein Maß für die Optimalität der erzeugten Lösungen angegeben werden. Allerdings gibt es für bestimmte Probleme bei der Berücksichtigung spezieller Randbedingungen Heuristiken, welche bekanntermaßen optimale Ergebnisse für spezielle Zielsetzungen liefern. (Siehe beispielsweise Du et al., 2016, S. 9)

Enumerative Suchverfahren durchforsten uninformiert und häufig randomisiert bzw. stochastisch den Suchraum. Sie werden in der Regel bei Problemen angewendet, welche strukturbedingt keine Informationen für die Lenkung des Verfahrens (z.B. Gradienten) bieten können. Der Suchraum wird durch die entsprechenden Verfahren systematisch und häufig probabilistisch durchsucht, wobei die Suchrichtung bei vielen Verfahren durch die Rückkopplung (positive feedback) mit den bereits gefundenen Ergebnissen beeinflusst wird (Dorigo et al., 1991). Die Verfahren sind in den meisten Fällen problemunabhängig formuliert und daher als universale Lösungsmethode anwendbar und implementierbar. (Siehe beispielsweise Burke et al., 2014b, Du et al., 2016)

Die enumerativen Suchverfahren spielen in dieser Arbeit eine besondere Rolle. Sie lassen sich entsprechend Abbildung 2.3 weitergehend zwischen deterministischen und probabilistischen Verfahren differenzieren.

a) Deterministische Suchstrategien

Deterministische Suchstrategien bezeichnen in diesem Zusammenhang enumerative Verfahren mit deterministischem Laufzeitverhalten ohne den Einfluss von Zufallszahlen und Stochastik. Wichtige deterministische Suchstrategien sind:

- Lokale Suche

- Branch and Bound
- Tabusuche

Da der Fokus dieser Arbeit in Bezug auf die verwendeten Suchstrategien aus Gründen der zu erwartenden Rechenzeiten nicht auf deterministischen Verfahren liegt, wird an dieser Stelle für weitergehende Informationen beispielsweise auf die Werke von Du et al. (2016), Zgurovsky et al. (2019) und Burke et al. (2014b) verwiesen.

b) Probabilistische Metaheuristiken

Metaheuristiken durchsuchen systematisch Teile des Suchraums, indem probabilistisch und randomisiert unterschiedliche Lösungen erzeugt und bewertet werden. Sie benötigen dafür kein spezifisches Wissen über das entsprechende Optimierungsproblem, da die Verfahren auf einer höheren, problemunabhängigen Ebene operieren (Du et al., 2016, S. 9). Es wird lediglich ein Mechanismus zur Bewertung der unterschiedlichen Lösungen benötigt (z.B. eine mathematische Funktion, Simulation etc., Burke et al., 2014a, S. 6).

Die meisten Metaheuristiken verwenden dabei sowohl Elemente, die für eine Konvergenz (Exploitation), als auch Elemente, die für eine Divergenz (Exploration) sorgen (Sastry et al., 2014, S. 106). Durch das Zusammenspiel der beiden Eigenschaften soll einerseits eine möglichst breitbandige Suche erreicht werden, die den Suchraum an möglichst weit auseinander liegenden Stellen durchsucht. Andererseits wird dadurch eine Intensität der Suche in vielversprechenden Gebieten des Suchraums erreicht, so dass eine Konvergenz des Verfahrens zu einem (lokalen oder globalen) Optimum erreicht werden kann.

Metaheuristische Suchverfahren arbeiten häufig systematisch probabilistisch und randomisiert bzw. stochastisch, so dass der selbe Suchalgorithmus (im Gegensatz zu rein deterministischen Verfahren und Heuristiken) bei demselben Problem in der Regel unterschiedliche Ergebnisse liefert. Es handelt sich bei den gefundenen Ergebnissen im Regelfall nicht um das globale Optimum sondern um lokale Extremstellen im Suchraum (siehe Abbildung 2.4).

Zu den Vorteilen der Verfahren gehört die hohe Flexibilität und Problemunabhängigkeit sowie die Fähigkeit, innerhalb einer relativ kurzen Zeit ein relativ gutes Ergebnis zu liefern. Nachteilig sind die teilweise umfangreichen Parametriermöglichkeiten, so dass das Lösungsverhalten stark von der korrekten Parametrierung des Verfahrens abhängen kann, welche wiederum eine starke Abhängigkeit von dem zu lösenden Optimierungsproblem aufweisen kann (Kramer, 2017, S. 21).

Viele der am häufigsten angewendeten Verfahren sind durch Beobachtungen oder Vorgänge aus der Natur inspiriert. Die populärsten und am häufigsten angewendeten Metaheuristiken sind die folgenden (Du et al., 2016):

- Ameisenalgorithmen
- Bienenalgorithmen

- Evolutionäre Algorithmen
- Genetische Algorithmen
- Partikelschwarmoptimierung
- Simuliertes Abkühlen

Für weitergehende Informationen und weitere Verfahren wird an dieser Stelle beispielsweise auf Blum et al. (2016), Du et al. (2016) sowie auf Burke et al. (2014b) verwiesen.

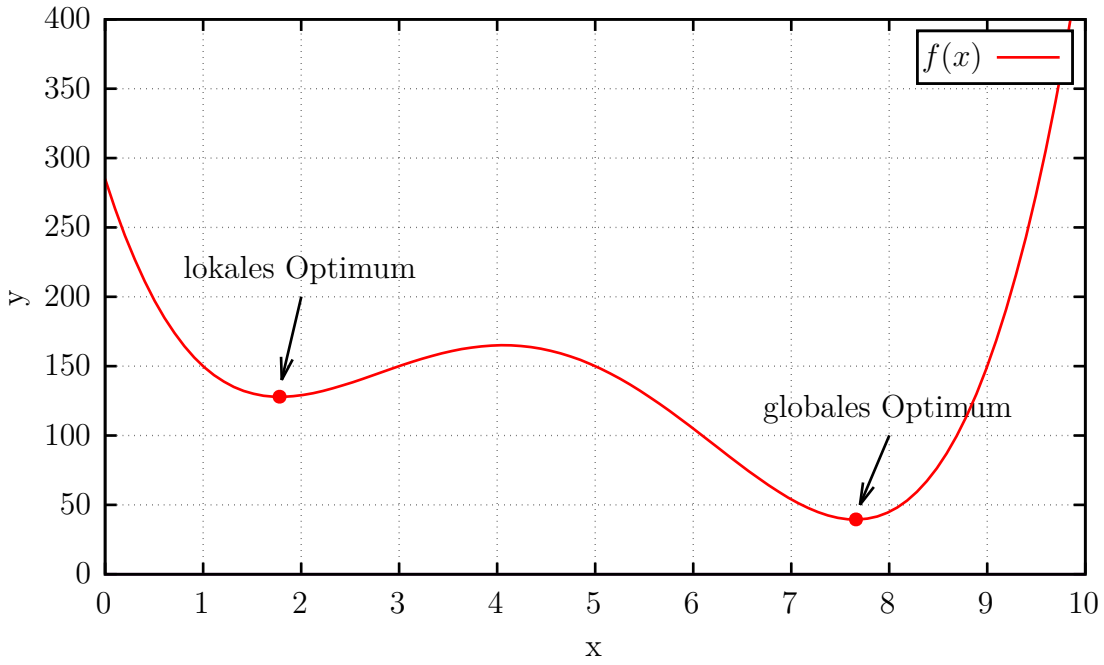


Abbildung 2.4: Beispiel eines lokalen und des globalen Optimums

2.4.3 Ereignisdiskrete Simulation

Die Simulation gehört im Zusammenhang mit der gemischt-ganzzahligen und der kombinatorischen Optimierung zu den für die Praxis wichtigsten Werkzeugen des Operations Research (Domschke et al., 2005, S. 223). Sie stellt Möglichkeiten zur Verfügung, Experimente an einem Modell durchzuführen, welches in den untersuchungsrelevanten Aspekten weitgehend der Realität eines dynamischen Systems entspricht (Gutenschwager et al., 2017, S. 22).

Während mathematische Optimierungsmodelle die entsprechenden Sachverhalte komplett und geschlossen darstellen, werden in der Simulation die Wirkmechanismen einzelner Teile und deren Wechselwirkungen untereinander abgebildet. Sind mathematische Optimierungsmodelle nicht oder nur mit unvertretbarem Aufwand zu entwickeln oder analytisch

zu lösen, sollte die Simulation als Werkzeug eingesetzt werden (Domschke et al., 2005, S. 223). Dies trifft auf die in dieser Arbeit behandelten Flexible Flow Shop Probleme zu.

Bei der Formulierung eines mathematischen Modells wird für das zugrunde liegende Problem ein Lösungsraum aufgebaut. Es sind Werte für die einzelnen Variablen gesucht, so dass das Modell einerseits nicht verletzt wird und andererseits die entsprechende Zielgröße möglichst optimal (in der Regel minimal) ist. Dafür stehen zwar problemunabhängige Solver zur mathematischen Lösung und Optimierung als universelles Werkzeug zur Verfügung. Flexible Flow Shop Probleme auf Basis realer Daten und Systeme sind in der Regel allerdings durch eine derart hohe Komplexität gekennzeichnet, dass diese Methode aufgrund der hohen Rechenzeiten in den meisten Anwendungsfällen nicht in Frage kommt. Ein Lösungsansatz für dieses Problem besteht in der Kopplung zwischen Simulation und Optimierung (siehe Abbildung 2.5).

Bei der Kopplung kann die Simulation mehrere Zwecke erfüllen. In den meisten Fällen dient sie als reine Bewertungsfunktion für den Optimierungsprozess (März et al., 2011a, S. 43). Dabei werden in dem zyklischen Ablauf der kombinatorischen Optimierung systematisch unterschiedliche Lösungsvarianten gebildet. Diese werden der Simulation als Eingangsgrößen übergeben. Die Simulation dient dabei als Ersatz für eine geschlossene mathematische Funktion (beispielsweise der Form $f(x_1, x_2, \dots) = \dots$). Anschließend wird das Ergebnis der Simulation bewertet und dem Optimierer wieder zurückgekoppelt. Als Optimierer sind grundsätzlich die deterministischen oder probabilistischen Suchverfahren einsetzbar. Heuristische Ansätze lassen sich in der Simulation beispielsweise in Form von Belegungs- und Ablaufregeln implementieren (siehe z.B. März et al., 2011a).

Für die Abbildung von logistischen Systemen (wie die Flexible Flow Shop Probleme) durch Simulationsmodelle empfiehlt Eley (2012, S. 8) den Einsatz der ereignisdiskreten Simulation. Abbildung 2.6 zeigt die Einordnung der in dieser Arbeit verwendeten ereignisdiskreten Simulationsmethode in die Systematik der unterschiedlichen Simulationsmethoden (in Anlehnung an Rose et al., 2011, S. 13–14).

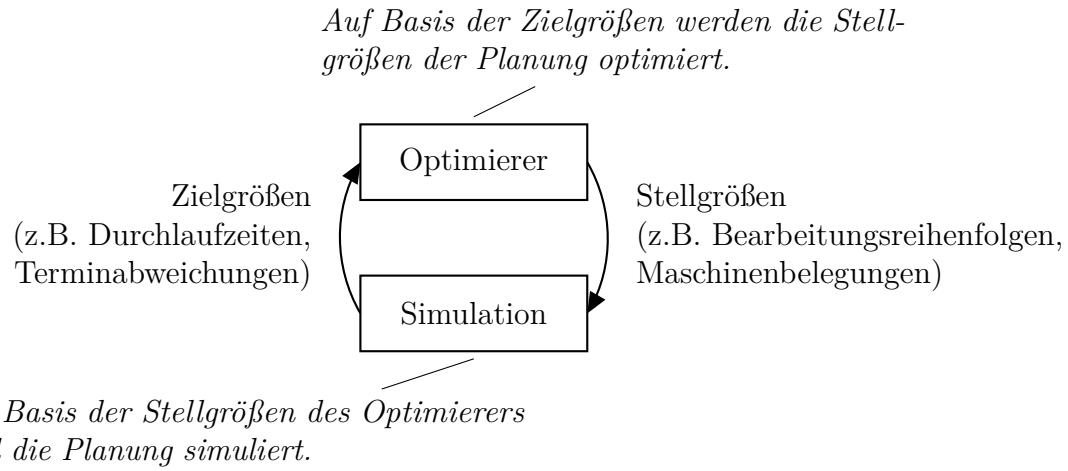


Abbildung 2.5: Kopplung einer Simulation mit einem Optimierer

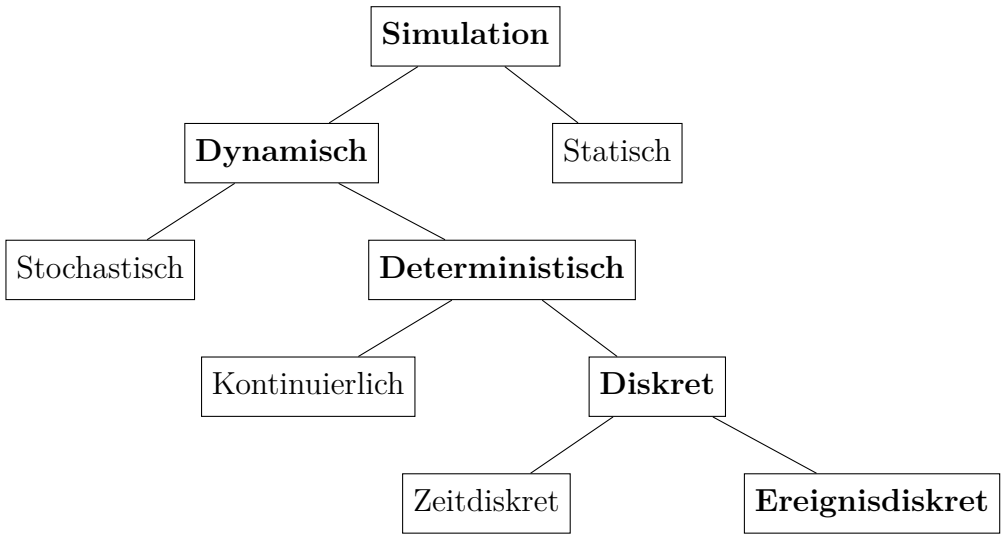


Abbildung 2.6: Einordnung der ereignisdiskreten Simulationsmethoden

Es wird das zeitliche Verhalten (**dynamisch**) eines Systems untersucht, wobei die Annahme getroffen wird, dass sich die Systemzustände nur an **diskreten** Zeitpunkten ändern können (Law et al., 1997). Stochastische Einflüsse auf das System treten dabei nicht auf, so dass sämtliche Abläufe in dem Simulationsmodell **deterministisch** sind. Im Gegensatz zu der zeitdiskreten Simulation wird bei der **ereignisdiskreten** Simulation das System dafür in variablen und nicht in festen Zeitintervallen untersucht (Law et al., 1997).

Die Systemzustände eines ereignisdiskreten Modells ändern sich nicht in festen Zeitintervallen, sondern immer nur bei dem Eintritt von verschiedenen Ereignissen. Der zeitliche Abstand zwischen den Ereignissen ist grundsätzlich unterschiedlich und hängt von den Variablen des Systems ab (Rose et al., 2011, S. 14). Auf logistische Systeme wie das Flexible Flow Shop Problem trifft dies generell zu. Beispielsweise ändert sich der Systemzustand, wenn eine Maschine die Bearbeitung eines Fertigungsauftrags beginnt oder beendet. Alle Zeitpunkte zwischen diesen beiden Ereignissen haben keinen Einfluss auf das betrachtete System und werden in der ereignisdiskreten Simulation entsprechend übersprungen (siehe z.B. Gutenschwager et al., 2017, S. 54).

Abbildung 2.7 zeigt den Unterschied zwischen der zeitdiskreten und der ereignisdiskreten Simulationsmethode. Das zeitliche Verhalten des simulierten Systems wird in beiden Fällen durch die Zielfunktion $f(t)$ repräsentiert und wird über einen Zeitraum von 25 s untersucht. An dem Modell muss bei der zeitdiskreten Variante bei einer Schrittweite von $\Delta t = 1\,\mathrm{s}$ an 26 Zeitpunkten gerechnet werden. Änderungen an der Zielfunktion treten dabei allerdings nur zu 9 Zeitpunkten auf.

Bei der ereignisdiskreten Variante werden entsprechend nur die benötigten 9 Zeitpunkte betrachtet, bei denen an dem Modell Änderungen durch die entsprechenden Ereignisse eintreten. Die Einsparung des Rechenaufwands gegenüber dem zeitdiskreten Modell wird dadurch deutlich reduziert.

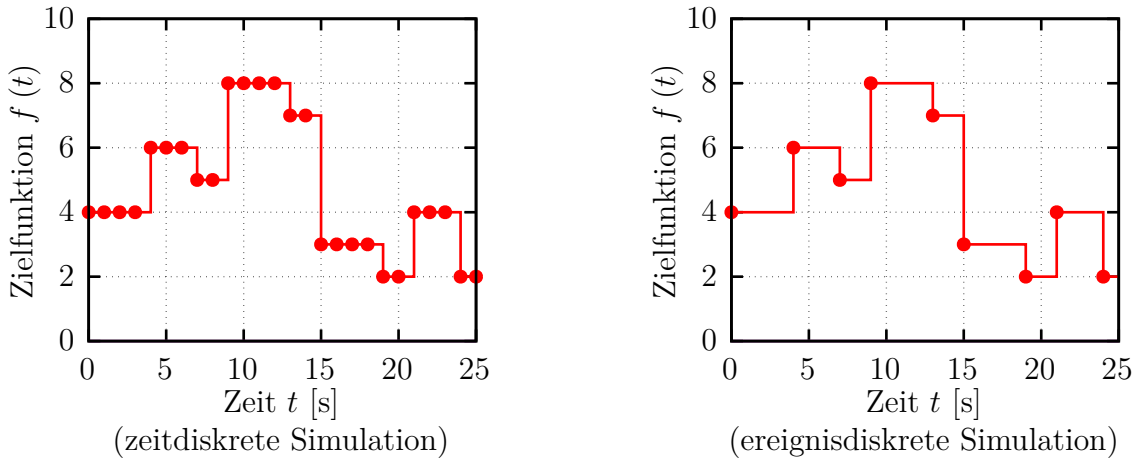


Abbildung 2.7: Vergleich zwischen zeit- und ereignisdiskreter Simulation

2.4.4 Lösungsansätze für Flexible Flow Shop Probleme

Flexible Flow Shop Probleme wurden laut Gondek (2011) erstmals von Salvador (1973) diskutiert und sind seitdem ein Schwerpunkt von Forschung und Entwicklung auf dem Gebiet der Ablaufplanung. Aus der Fülle der bereits geleisteten Forschungsarbeiten sind diejenigen herauszufiltern, die eine Relevanz zu der Fragestellung dieser Arbeit aufweisen. Es muss sich dabei folglich um Veröffentlichungen handeln, in welchen Lösungsansätze für Flexible Flow Shop Probleme mit beliebig vielen Stufen und beliebig vielen unterschiedlichen Maschinen pro Stufe entwickelt werden. Es sind Ansätze von Bedeutung, welche die Optimierung multikriterieller Zielgrößen im Fokus haben und den Einsatz in der operativen Produktionsplanung und -steuerung in realen Betrieben ermöglichen.

Ferner sind die berücksichtigten Randbedingungen und Restriktionen von hoher Bedeutung. So stellt die Berücksichtigung reihenfolgeabhängiger Rüstzeiten mit maschinenspezifischen Auftragsfamilien ein wichtiges Kriterium dar. Darüber hinaus sind begrenzte Pufferkapazitäten und begrenzte Materialressourcen ausschlaggebend sowie die stufenspezifische Stapelbearbeitung und Inkompatiblitäten zwischen Fertigungsaufträgen und Maschinen. Forschungsarbeiten, die jede dieser Kriterien berücksichtigen – wie es im Rahmen dieser Dissertation der Fall ist – konnten nicht identifiziert werden, daher wird der Fokus auf die entsprechenden Teilübereinstimmungen gelegt.

a) Literaturstudie und Recherche

Die Erhebung des aktuellen Stands der Forschung von Flexible Flow Shop Problemen basiert u.a. auf den ausführlichen und umfangreichen Literaturstudien von Dhiflaoui et al. (2018), Lee et al. (2019b), Potts et al. (2017b), Ribas et al. (2010), Ruiz et al. (2010) sowie von Potts et al. (2000). Weitere Ergebnisse liefern die Literaturstudien von Haouari et al. (2008), Linn et al. (1999), Potts et al. (2017a), Wang (2005) und Liaee et al. (1997).

Die Forschungsschwerpunkte der jeweiligen Arbeiten werden entsprechend der verwendeten Lösungsansätze in die folgenden Klassen eingeteilt:

- exakte Ansätze
- heuristische Ansätze
- metaheuristische Ansätze

Anschließend werden die jeweils untersuchten Problemstellungen in Bezug auf multikriterielle Zielsetzungen sowie auf die berücksichtigten Randbedingungen und Restriktionen klassifiziert.

b) Exakte Ansätze

In bestimmten Fällen sind für Probleme der Ablaufplanung effiziente Verfahren für das Finden optimaler Lösungen bekannt. Die meisten Untersuchungen mit exakten Methoden behandeln simplifizierte Versionen des Flexible Flow Shop Problems. Häufig handelt es sich dabei um Probleminstanzen mit nur zwei Stufen, wobei sich in der ersten Stufe nur eine Maschine und in der zweiten Stufe mehrere Maschinen befinden. Isenberg (2016, S. 38) stellt beispielsweise fest, dass Flow Shop Probleme mit geringer struktureller Komplexität bei zwei Maschinen nur mit bis zu 50 Fertigungsaufträgen durch lineare Optimierung und Branch and Bound Verfahren lösbar sind (siehe auch Pinedo, 2008, S. 172). Es werden in der Regel keine multikriteriellen Ziele untersucht und die berücksichtigten Randbedingungen und Restriktionen sind extrem simpel gehalten (Vairaktarakis, 2004, S. 83).

Beispiele für die Formulierung unterschiedlicher gemischt-ganzzahliger Optimierungsmodelle sowie deren Lösung mit mathematischen Solvern finden sich in Méndez et al. (2001), Naderi et al. (2014) und Hui et al. (2000). Konkrete Beispiele für Anwendung von Branch and Bound Verfahren liefern beispielsweise die Arbeiten von Allaoui et al. (2006), Kis et al. (2005) und Fattahi et al. (2014).

Die exakten Lösungsansätze eigenen sich in der Regel nur für sehr kleine Probleminstanzen, also Aufgaben mit nur wenigen Fertigungsaufträgen und einer stark begrenzten Anzahl von Produktionsstufen und Maschinen. An dieser Stelle werden die exakten Ansätze entsprechend nicht weiter vertieft, da sie aufgrund der Zielsetzung dieser Arbeit als Lösungsstrategie nicht geeignet sind.

c) Heuristische Ansätze

Bei den heuristischen Ansätzen handelt es sich um Verfahren, die zulässige Lösungen unter Zuhilfenahme von Planungsregeln und Sortieralgorithmen konstruieren. Folglich handelt es sich dabei immer um Algorithmen, die jeweils für den Einsatz einer speziellen Zielsetzung formuliert werden und immer auf ein spezielles Problem zugeschnitten sind. Im Gegensatz zu den metaheuristischen Suchverfahren (siehe Unterabschnitt 2.4.2 b))

handelt es sich bei den konventionellen Heuristiken in der Regel um vergleichsweise primitive Verfahren (Du et al., 2016, S. 9).

Mithilfe von Planungsheuristiken werden entsprechende Lösungen konstruiert, ohne dabei die Optimalitätsbedingung zu gewähren. Konstruktive Heuristiken, welche eine entsprechende Lösung direkt konstruieren, lassen sich von verbessernden Heuristiken differenzieren, welche ausgehend von einer Initiallösung einen iterativen Verbesserungsprozess verfolgen. Die am häufigsten verwendete Art von Heuristiken sind Einlastungsregeln (dispatching rules), bei denen die Fertigungsaufträge anhand von Prioritäten sortiert und eingelastet werden (Lee et al., 2019b).

Als historisch wichtige Beispiele sind an dieser Stelle zunächst die Regeln von Johnson (Johnson, 1954) und Moore (Moore, 1968) aufzuführen. Die Regel von Johnson liefert optimale Ergebnisse für zwei- und dreistufige Flow Shop Probleme (mit nur einer Maschine je Stufe) ohne die Berücksichtigung reihenfolgeabhängiger Rüstzeiten oder sonstigen Restriktionen. Die Regel von Moore minimiert die Anzahl der verspäteten Fertigungsaufträge für Einmaschinenprobleme mit beliebig vielen Fertigungsaufträgen. Diese Regeln richten sich in ihrem Ursprung zwar nicht an das Flexible Flow Shop Problem, sie bilden allerdings nach wie vor häufig die Basis für neue heuristische Verfahren und Ansätze.

Die im Zusammenhang mit dieser Arbeit wichtigsten heuristischen Methoden lassen sich durch die folgenden Einlastungsregeln repräsentieren (Gondek, 2011, Holthaus et al., 1997, Rajendran et al., 1999):

EDD (earliest due date): Die Fertigungsaufträge werden in aufsteigender Reihenfolge ihrer Planfertigstellungstermine sortiert, so dass die Fertigungsaufträge mit den dringensten Terminen als nächstes eingeplant werden.

S/OPN (slack per remaining operation): Die Schlupfzeitregel ermittelt für die Fertigungsaufträge den verbleibenden Zeitpuffer zwischen Planfertigstellungstermin und verbleibender, noch benötigter Produktionszeit (Schlupf). Der Fertigungsauftrag mit dem geringsten Restschlupf wird als nächstes eingeplant.

SPT (shortest process time): Die Fertigungsaufträge werden gemäß ihrer noch benötigten Produktionszeit aufsteigend sortiert, so dass der Fertigungsauftrag mit der kürzesten Produktionszeit als nächstes eingeplant wird.

LPT (longest process time): Die Fertigungsaufträge werden gemäß ihrer noch benötigten Produktionszeit absteigend sortiert, so dass Fertigungsaufträge mit der längsten Produktionszeit als nächstes eingeplant werden.

FIFO (first in first out): Die Fertigungsaufträge werden in der Reihenfolge ihrer Ankunft wie in einer reinen Warteschlange sortiert und priorisiert. Fertigungsaufträge, die zuerst in der Warteschlange eintreffen (und sich damit vorne in der Warteschlange befinden), werden zuerst gefertigt.

LIFO (last in first out): Die Fertigungsaufträge werden in der umgekehrten Reihenfolge ihrer Ankunft wie in einem reinen Stapelsystem sortiert und priorisiert.

Fertigungsaufträge, die dem Stapel zuletzt hinzugefügt werden (und damit oben auf dem Stapel liegen), werden zuerst gefertigt.

Weitere häufig angewendete heuristische Planungsheuristiken sind (siehe auch Gondek, 2011, Holthaus et al., 1997, Rajendran et al., 1999):

NEH-Heuristik: Der Name dieser Heuristik ist auf die Erfinder (Nawaz, Enscore und Ham, 1983) zurückzuführen. Für jeden Produktionsauftrag wird die zu erwartende Gesamtproduktionszeit berechnet und die Aufträge werden absteigend sortiert. Zunächst wird der erste Fertigungsauftrag aus dieser Liste in den Ablaufplan eingefügt. Jeder weitere Auftrag aus der Liste wird nun nacheinander in jede mögliche Position in dem Ablaufplan eingeplant und die entsprechende Zielgröße berechnet. Die Aufträge werden jeweils an die Position geplant, welche die bestmögliche Zielgröße ermöglicht. Die Positionen der bereits verplanten Fertigungsaufträge bleiben dabei in jedem Schritt unverändert. Es handelt sich entsprechend um ein Verfahren, welches eine Teilenumeration durchführt. Anwendungsbeispiele liefern beispielsweise Liu et al. (2017).

Shifting Bottleneck Heuristik: Es wird zunächst die Produktionsstufe identifiziert, die einen Engpass (Flaschenhals) darstellt. Die Verplanung der Fertigungsaufträge erfolgt zuerst an dieser Stufe. Dies wird solange wiederholt, bis die Planung abgeschlossen ist. In jedem Schritt wird jeweils nur eine Planung für die jeweilige Engpassstufe durchgeführt, was einer Reduzierung des Gesamtproblems und damit dem Lösen von Teilproblemen entspricht. Nach welchen Regeln die Planungen der einzelnen Stufen vorgenommen werden, kann variieren. Anwendungen dieses Verfahrens sind in Xu et al. (2003) und Acero-Dominguez et al. (2004) beschrieben.

Für weiterführende Ergebnisse zu der Anwendung und Entwicklung von heuristischen Lösungsansätzen wird beispielsweise auf die Arbeiten von Kochhar et al. (1987), Kurz et al. (2003, 2004) und Kim et al. (2009) verwiesen.

d) Metaheuristische Ansätze

Bei der Entwicklung von metaheuristischen Suchverfahren für die Lösung von Flexible Flow Shop Problemen ist die Art der Anwendung durch unterschiedliche Definitionen der Schnittstelle des Suchverfahrens – der Problemrepräsentation – von hoher Bedeutung. Grundsätzlich lassen sich daher Ansätze mit direkten Problemrepräsentationen von Ansätzen mit indirekten Problemrepräsentationen unterscheiden (Werner, 2011).

Direkte Repräsentationen bieten die Möglichkeit, die gesuchten Größen direkt durch das Suchverfahren manipulieren zu lassen. Die Schnittstelle zwischen dem Suchverfahren und dem Planungsproblem beinhaltet folglich die Bearbeitungsreihenfolgen und Auftragszuordnungen für die Maschinen in einer geeigneten Form. Ein Ablaufplan kann in diesem Fall direkt in einem Code durch ein zugehöriges Schema

kodiert werden. Die Metaheuristik kann auf diese Weise für jede Stufe, jeden Fertigungsauftrag und jede Maschine die entsprechenden Maschinenzuordnungen und Produktionsreihenfolgen bestimmen bzw. verändern.

Häufig wird durch die Schnittstelle nur ein Teil des Problems in direkter Form repräsentiert, so dass die Metaheuristik beispielsweise nur die Produktionsreihenfolgen für die erste Produktionsstufe bestimmt. Die übrige Planung der restlichen Maschinen erfolgt dann z.B. über heuristische Verfahren (Lee et al., 2019b).

Indirekte Repräsentationen bieten die Möglichkeit, Such- bzw. Optimierungsalgorithmen auf einer abstrakteren Ebene anzuwenden. Die Metaheuristik sucht in diesem Fall nicht direkt in dem Lösungsraum des Planungsproblems. Durch ein Kodierungsschema werden abstrakte Informationen wie beispielsweise der Einsatz von bestimmten Planungsregeln oder deren Gewichtungen zueinander repräsentiert.

Die gesuchten Größen und der zugehörige Ablaufplan werden in einem Planungsverfahren über konventionelle Planungsregeln und Heuristiken bestimmt (Werner, 2011, S. 8). Boll führt für diesen Fall den Begriff des „Lösungsgenerators“ ein (Boll, 2003, S. 51).

Die Genetischen Algorithmen gehören zu den bislang populärsten Metaheuristiken auf dem Gebiet der Ablaufplanung. Anwendungen von Genetischen Algorithmen mit unterschiedlichen Kodierungen liefern beispielsweise Kacem (2013) und Costa et al. (2014). Anwendungsbeispiele mit indirekten Problemrepräsentationen geben u.a. Lee et al. (2019a). Lee et al. kombinieren dabei für ein Beispiel aus der Halbleiterindustrie verschiedene Planungsheuristiken mit einem Genetischen Algorithmus, um eine multikriterielle Zielsetzung verfolgen zu können. Die Autoren nutzen die Metaheuristik dabei, um jeweils die besten Belegungsregeln für die jeweiligen Zielsetzungen und deren Gewichtungen zu ermitteln.

Weitere Beispiele für die Anwendung von Genetischen Algorithmen auf Flexible Flow Shop Probleme finden sich in den Arbeiten von Ruiz et al. (2006) (reihenfolgeabhängige Rüstzeiten), Quadt et al. (2005) (Ablaufplanung mit Losgrößenplanung), Wu et al. (2003) und Chaaben et al. (2013) (Verfügbarkeitsrestriktionen). Sie decken jeweils nur einen Teil der in dieser Arbeit geforderten Zielsetzung ab.

Anwendungen mit Ameisenalgorithmen finden sich beispielsweise in den Arbeiten von Alaykýran et al. (2007), Chernigovskiy et al. (2017), Tavares et al. (2011), Ying et al. (2006) und Colorni et al. (1994). Gong et al. (2019) zeigen die Anwendung eines Bienenalgorithmus auf ein Job Shop Problem, Naderi et al. (2014) und Naderi et al. (2009) wenden in ihren Forschungsarbeiten u.a. eine Partikelschwarmoptimierung sowie das Simulierte Abkühlen auf Flexible Flow Shop Probleme an. Eine weitere Anwendung der Partikelschwarmoptimierung auf Flexible Flow Shop Probleme findet sich in Tseng et al. (2008).

Beispiele für die Optimierung mithilfe der Tabusuche liefern Wang et al. (2009), Wardono et al. (2004) und Chen et al. (2007). Engin et al. (2004), Niu et al. (2012) und Alisantoso

et al. (2003) wenden die Nachbildung künstlicher Immunsysteme als Suchverfahren für die Planung von Flexible Flow Shop Problemen an. Anwendungsbeispiele für die Implementierung von Nachbarschaftssuchen finden sich u.a. in Ribas et al. (2015) und Almeder et al. (2013).

Weitere Beispiele für die Anwendung verschiedener Suchverfahren können u.a. in den Werken Blazewicz et al. (2019), Cevik Onar et al. (2016), Deroussi (2016), Jarboui et al. (2013) und Janiak et al. (2007) nachgeschlagen werden.

e) Multikriterielle Zielsetzungen

Laut den Studien von Lee et al., Ruiz et al. und Minella et al. (2008) beschäftigt sich nur die Minderheit der Forschungsarbeiten mit der multikriteriellen Optimierung von Flexible Flow Shop Problemen.

Ebrahimi et al. (2014) zielen beispielsweise auf die gleichzeitige Minimierung der Zykluszeit sowie der gewichteten Verfrühung ab. Jungwattanakit et al. (2009) vergleichen verschiedene Heuristiken für die Minimierung der Zykluszeit bei gleichzeitiger Minimierung der Anzahl verspäteter Fertigungsaufträge.

Joo et al. (2013) verfolgen mit der Maximierung der Produktionsqualität sowie der Minimierung der mittleren Verspätung Optimierungsziele in direktem Bezug zur Produktionseffizienz und Kundenzufriedenheit. Naderi et al. (2009) wenden das Simulierte Abkühlen an, um in einem Flexible Flow Shop Problem gleichzeitig die Summe der Fertigstellungszeitpunkte sowie die Summe der Verspätungen zu minimieren. In der Arbeit von Janiak et al. (2007) werden verschiedene Ansätze mit Metaheuristiken (Tabusuche und Simuliertes Abkühlen) und konstruktiven Heuristiken diskutiert, um in einem Flexible Flow Shop Problem eine zusammengesetzte Zielgröße bestehend aus der gewichteten Verfrühung, der gewichteten Verspätung sowie der gewichteten Wartezeit der Fertigungsaufträge zu minimieren.

f) Beliebig viele Stufen mit unterschiedlichen parallelen Maschinen

Lee et al. und Ruiz et al. kommen in ihren Studien zu dem Ergebnis, dass Untersuchungen an allgemeinen Klassen von Flexible Flow Shop Problemen, wie es in dieser Arbeit der Fall ist, selten sind. Entsprechend fokussiert sich ein großer Teil der Forschungsarbeiten auf die Entwicklung von Lösungsansätzen für Flexible Flow Shop Probleme mit einer vorgegebenen (begrenzten) Zahl von Stufen und Maschinen und geht dabei nicht von unterschiedlichen Maschinen in den Stufen aus.

Ansätze für Flexible Flow Shop Probleme mit unterschiedlichen Maschinen liefern beispielsweise Behnamian et al. (2011), Chen et al. (2009, 2007), Jenabi et al. (2007), Ruiz et al. (2008), Yaurima et al. (2009) und Sawik (2006).

g) Berücksichtigung spezieller Randbedingungen und Restriktionen

Reihenfolgeabhängige Rüstzeiten und stufenspezifische Rüstfamilien: Die Berücksichtigung von reihenfolgeabhängigen Rüstzeiten führt zu einer drastischen

Erhöhung der Komplexität des Problems (siehe beispielsweise Pinedo, 2008, S. 84). Wenn sich die Fertigungsaufträge in Abhängigkeit der entsprechenden Rüstzeiten zu Auftragsfamilien zusammenfassen lassen, treten an den Maschinen Rüstzeiten nur zwischen dem Wechsel der Auftragsfamilien auf (Pinedo, 2008, S. 16). Liaee et al. (1997) zeigen in ihrer Studie auf, dass die Komplexität in diesem Fall schon bei Ein- und Parallelmaschinenproblemen auf ein mathematisch nicht zu beherrschendes Niveau steigt.

Beispiele für die Optimierung von Flexible Flow Shop Problemen mit reihenfolgeabhängigen Rüstzeiten sowie der Berücksichtigung von Auftragsfamilien finden sich in den Arbeiten von Ebrahimi et al. (2014), Keshavarz et al. (2013), Luo et al. (2015) und Logendran et al. (2006). Javadian et al. (2013), Kianfar et al. (2012) und Tang et al. (2006) zeigen beispielsweise Forschungsaktivitäten an Flexible Flow Shop Problemen mit reihenfolgeabhängigen Rüstzeiten ohne die Berücksichtigung von Auftragsfamilien.

Begrenzte Zwischenpuffer und Ressourcen: Beispiele für Untersuchungen an Flexible Flow Shop Problemen mit begrenzten Zwischenpuffern liefern die Arbeiten von Ernst et al. (2019), Gondek (2011), Han et al. (2019, 2018), Lin et al. (2020), Shi et al. (2019) und Akrami et al. (2006). Beispiele für die Berücksichtigung von begrenzten Ressourcen finden sich in Cheng et al. (2012), Dhiflaoui et al. (2018), Laribi et al. (2016), Lin et al. (2013) und Leu et al. (2002).

Stufenspezifische Stapelbearbeitung: Bei Problemen der Ablaufplanung wird zunächst immer davon ausgegangen, dass jede Maschine immer nur einen Fertigungsauftrag zur gleichen Zeit bearbeiten kann. Einen Sonderfall stellen daher Flexible Flow Shop Probleme mit Stapelverarbeitung (batch processing) dar. In diesem Fall sind in dem Problem Maschinen vorhanden, die mehrere Aufträge gleichzeitig verarbeiten können oder müssen.

Tan et al. (2018) und Zhang et al. (2016) entwickeln beispielsweise Ansätze für zweistufe Probleme mit Stapelverarbeitung. Zhang et al. berücksichtigen dabei weiterhin reihenfolgeabhängige Rüstzeiten mit Auftragsfamilien und begrenzte Pufferkapazitäten.

Costa et al. (2014) und Quadt et al. (2005) entwickeln Ansätze mit Genetischen Algorithmen für die Lösung von Flexible Flow Shop Problemen mit Stapelverarbeitung. In der Arbeit von Quadt et al. steht dabei neben der Optimierung der Rüstzeiten auch die Optimierung der Stapelgrößen im Vordergrund. Weitere Forschungsergebnisse werden in den Studien von Potts et al. (2017b), Zeng et al. (2018) und Potts et al. (2000) zusammengefasst.

2.4.5 Ablaufplanung in der Praxis

Die in den vorherigen Abschnitten diskutierten Forschungsansätze und Methoden verfolgen unterschiedliche Motivationen. Praktische Anwendungen liegen dabei in vielen

Fällen aber nicht vor. In produzierenden Unternehmen treten die mathematischen Probleme der Ablaufplanung in der Realität zwar auf, entsprechende Methoden zur Planung und Steuerung werden in der Regel allerdings nicht ausreichend angewendet (siehe z.B. Günther et al., 2012, S. 146).

In der unternehmerischen Praxis wird daher häufig keine realistische Ablaufplanung durchgeführt. Viel mehr werden in diesen Fällen Termine ohne Berücksichtigung der entsprechenden Beziehungen und Wechselwirkungen durch das ERP-System berechnet. Die operative Planung und deren Umsetzung erfolgt dann durch den Menschen, teilweise mit der Unterstützung von Tabellenkalkulationsprogrammen.

Die Umsetzung von Forschungsergebnissen in Form von realen Systemen zur Entscheidungsunterstützung ist daher von entsprechend hoher Bedeutung. Produktionsbegleitend werden in den entsprechenden Systemen die Bearbeitungsreihenfolgen und Maschinenbelegungen teils mit rollierendem Horizont (überlappende Planung) in ausreichender Geschwindigkeit und Qualität optimiert.

a) Allgemeine Ansätze

Gong et al. (2017) führen beispielsweise eine Fallstudie an einem belgischen Plastikflaschenhersteller durch. Es handelt sich dabei um ein Einmaschinenproblem, welches unter Berücksichtigung von energetischen Aspekten optimal geplant werden soll. Sie formulieren dafür ein entsprechendes Optimierungsmodell und nutzen für die Lösung einen Genetischen Algorithmus. Sie schlussfolgern, dass in dem Unternehmen (Energie-) Kosten durch die entsprechenden Planungen reduziert werden können.

Isenberg (2017) und Klement et al. (2017) beschäftigen sich mit der simultanen Losgrößen- und Reihenfolgeplanung im Zusammenhang mit Parallelmaschinenproblemen. Berücksichtigt werden dabei u.a. reihenfolgeabhängige Rüstzeiten und unterschiedliche Maschinen. Isenberg entwickelt ein mathematisches Optimierungsmodell und rechnet Modellinstanzen mit realen Daten aus der spanenden Fertigung. Klement et al. nutzen für die Lösung ihres Planungsproblems metaheuristische Ansätze. Sie führen eine Fallstudie in der Herstellung von Acrylfasern für die Textilindustrie durch. Es handelt sich dabei um eine Produktion mit zehn unterschiedlichen Maschinen.

Ramezanian et al. (2017) führen eine Losgrößen- und Reihenfolgeplanung für ein Flexible Flow Shop Problem durch. Es handelt sich dabei um eine Fliesenmanufaktur mit vier Produktionsstufen und jeweils vier Maschinen pro Stufe, so dass insgesamt 16 Maschinen geplant werden müssen. Sie berücksichtigen in ihrem Problem ebenfalls reihenfolgeabhängige Rüstzeiten und nutzen für die Lösung ihres Optimierungsmodells die Partikelschwarmoptimierung. Sie zeigen, dass dadurch in angemessener Zeit ausreichend gute Lösungen generiert werden können.

Gondek (2011) entwickelt eine Planungsheuristik für die Ablaufplanung in einem vollautomatischen Prozesslabor für die Qualitätskontrolle in einem Stahlwerk. Sie modelliert diesen Prozess in ihrer Dissertation als ein Flexible Flow Shop Problem mit sieben

Produktionsstufen und insgesamt 32 Maschinen. Sie behandelt dabei neben dem Reihenfolgeproblem in erster Linie auch ein Routing Problem, bei dem für den Transport der Stahlproben verschiedene Roboter zu Verfügung stehen.

Zhong et al. (2014) entwickeln heuristische Ansätze für ein Flexible Flow Shop Problem mit zwei Produktionsstufen und insgesamt fünf Maschinen. Der Forschungsschwerpunkt liegt bei den Autoren auf Echtzeitfähigkeit im Zusammenhang mit der RFID-Technik. Weitere Forschungsergebnisse zu RFID-basierten Systemen liefern beispielsweise die Arbeiten von Engelhardt (2015), Zhong et al. (2013a,b) und Hanisch et al. (2008).

Waldherr behandelt das Planungsproblem eines deutschen Herstellers für Küchenmöbel. Es handelt sich dabei um ein Flow Shop Problem mit synchroner Werkstückbewegung. Er führt dafür zunächst eine Komplexitätsuntersuchung durch, behandelt exakte Ansätze und entwickelt heuristische Methoden für die Lösung des Problems (Waldherr, 2015, Waldherr et al., 2014, 2015).

b) Simulationsbasierte Ansätze

Simulationsbasierte Lösungsansätze und insbesondere die Kopplung ereignisdiskreter Simulationsmethoden mit metaheuristischen Optimierungsalgorithmen finden zunehmend Anwendung bei der Umsetzung von Entscheidungssystemen für die operative Produktionsplanung und -steuerung. Beispiele sind die simulationsgestützte Entscheidungsunterstützung für das Produktionsmanagement einer Verzinkerei (Richter et al., 2017) und die simulationsbasierte Optimierung der Reihenfolgeplanung in der Automobilindustrie (Heger et al., 2015).

Nahhas et al. (2015) nutzen die Simulation für die Optimierung eines zweistufigen Flexible Flow Shop Problems aus der Leiterplattenbestückung. Es handelt sich bei dem Problem um vier Maschinen in der ersten und um fünf Maschinen in der zweiten Produktionsstufe. Sie nutzen das Simulierte Abkühlen als Optimierungsmethode. Weitere Beispiele für die Kopplung von Simulation mit einem Optimierungsverfahren sind die simulationsbasierte Optimierung von Farbgebungsanlagen (Lemessi et al., 2010), die simulationsgestützte Optimierung der Produktionspläne eines Herstellers von Halbleiterprodukten (Klemmt et al., 2011) sowie die simulationsbasierte Reihenfolgeoptimierung in der Automobilindustrie (Krug et al., 2011).

Weitere Forschungsarbeiten mit simulationsbasierten Ansätzen finden sich u.a. bei Zhang et al. (2015) (Flexible Job Shop Probleme mit veränderlichen Maschinen), bei Weigert et al. (2008) (Flexible Job Shop Probleme mit alternativen Montageprozessen) sowie bei Schmidt et al. (2017) (Berücksichtigung begrenzter Ressourcen). Sie können jeweils nur teilweise zu der im Rahmen dieser Arbeit geforderten Zielsetzung beitragen, da sowohl die jeweils berücksichtigten Randbedingungen als auch die verfolgten Zielsetzungen wesentlich von der Aufgabenstellung dieser Arbeit abweichen.

2.4.6 Zusammenfassende Übersicht relevanter Forschungsarbeiten

Tabelle 2.1 fasst mit 38 Publikationen die für diese Arbeit wesentlichen Forschungsaktivitäten an Flexible Flow Shop Problemen zusammen. Für jede Publikation wird durch die Spalten c, m_k und α_3 in Anlehnung an die erweiterte Dreifeldnotation (siehe Formel 2.8) angegeben, welche Maschinenumgebungen jeweils behandelt werden. Bei den Ausprägungen $c = c$ und $m_k = m_k$ handelt es sich jeweils um die Behandlung beliebig viele Produktionsstufen ($c = c$) bzw. um beliebig viele Maschinen je Stufe ($m_k = m_k$). Durch die Angabe von α_3 erfolgt die Klassifizierung der jeweils betrachteten parallelen Maschinenausprägungen (siehe Formel 2.6). In Bezug auf Unterabschnitt 2.2.1 sind für diese Arbeit Publikationen mit $c = c$, $m_k = m_k$ und $\alpha_3 = R$ (unterschiedliche Maschinen) relevant.

Durch die β-Spalten werden die untersuchten Forschungsarbeiten in Bezug auf die im Zusammenhang dieser Arbeit relevanten Randbedingungen und Restriktionen klassifiziert (siehe Unterabschnitt 2.2.2). Wird die Berücksichtigung einer entsprechenden Randbedingung durch die jeweilige Forschungsarbeit abgedeckt, wird dies in der Tabelle durch eine „×“-Markierung gekennzeichnet. Andernfalls erfolgt die Kennzeichnung durch eine „−“-Markierung.

Die in den einzelnen Forschungsarbeiten verfolgten Zielsetzungen werden in der γ-Spalte gekennzeichnet. Für die vorliegende Arbeit sind im Wesentlichen Forschungsansätze von Bedeutung, welche sich mit der Optimierung hinsichtlich beliebiger, multikriterieller Zielsetzungen beschäftigen (siehe Unterabschnitt 2.2.3). Publikationen, welche auf der multikriteriellen Optimierung beruhen, sind daher in der Tabelle mit dem Ausdruck „multi“ gekennzeichnet.

Anschließend erfolgt für jede untersuchte Publikation eine grobe Klassifizierung des jeweils verfolgten Lösungsansatzes. Handelt es sich beispielsweise um Studien, in denen Planungsregeln und Heuristiken untersucht werden, wird dies mit dem Ausdruck „Heuristik“ angedeutet. Bei Untersuchungen auf Basis von Metaheuristiken werden die jeweils verwendeten Optimierungsmethoden durch ein Kürzel gekennzeichnet.

Schließlich wird jede Forschungsarbeit hinsichtlich der jeweils verfolgten Motivation durch die Spalte „Praxisbezug“ gekennzeichnet. Handelt es sich beispielsweise um eine Studie mit einem rein mathematischen bzw. theoretischen Hintergrund – ohne Bezug zu realen Daten oder einem realen Produktionsprozess – wird dies durch ein „−“ gekennzeichnet. Forschungsaktivitäten mit einem praktischen Bezug – wie beispielsweise die Entwicklung von Entscheidungssystemen und die Durchführung von Studien auf Basis realer Daten – werden mit einem „×“ markiert.

Tabelle 2.1 stellt den aktuellen Forschungsstand in Bezug auf die in dieser Arbeit behandelte Aufgabenstellung dar. Insgesamt werden folgende Forschungsdefizite deutlich:

- Bislang werden hauptsächlich Probleme mit jeweils relativ wenigen Randbedingungen gleichzeitig untersucht. So behandeln lediglich ca. 11 % der 38 in Tabelle 2.1

aufgeführten Forschungsarbeiten mehr als zwei Randbedingungen gleichzeitig. Hervorzuheben sind an dieser Stelle lediglich die Arbeiten von Yaurima et al. (2009) und von Zhang et al. (2016), welche sich zumindest auf formaler Ebene mit vier (Zhang et al.) bzw. fünf (Yaurima et al.) von den sechs für diese Arbeit relevanten Randbedingungen beschäftigen. Allerdings behandeln Yaurima et al., Zhang et al. in ihren Forschungsarbeiten nur die Zielsetzung $C_{\max}$. Zhang et al. beschränken sich darüber hinaus auf Problemfälle mit nur $c = 2$ Produktionsstufen.

- Bislang werden hauptsächlich Probleme mit vergleichsweise einfachen Zielsetzungen behandelt. Bei den in Tabelle 2.1 aufgeführten Publikationen ist insgesamt eine deutliche Tendenz zu einfachen – nicht multikriteriellen – Zielsetzungen zu erkennen. Bei ca.58 % der Publikationen wird die Zielgröße $C_{\max}$ (Minimierung der Zykluszeit) verfolgt. Nur ca. 26 % der Arbeiten beschäftigen sich mit der Optimierung hinsichtlich multikriterieller Zielsetzungen wie es im Rahmen dieser Arbeit der Fall ist.

Tabelle 2.1: Darstellung des Forschungsdefizits anhand bisheriger Forschungsergebnisse

Publikation	c	m_k	α_3	$\beta = M_j$	$\beta = s_{i,j_1,j_2}$	$\beta =$ fmls	$\beta =$ batch	$\beta =$ block	$\beta =$ res	γ	Lösungsansatz	Praxisbezug
Akrami et al. 2006	c	m_k	R	–	–	–	–	×	–	multi	GA, TS	–
Alaykýran et al. 2007	c	m_k	P	–	–	–	–	–	–	$C_{\max}$	AA	–
Alisantoso et al. 2003	c	m_k	P	–	–	–	–	–	–	$C_{\max}$	KIS	×
Behnamian et al. 2011	c	m_k	R	–	×	–	–	–	–	multi	GA	–
Chaaben et al. 2013	c	m_k	P	–	–	–	–	–	–	$C_{\max}$	GA	×
Chen et al. 2009	c	m_k	R	–	–	–	–	–	–	T_j	Heuristik	–
Chen et al. 2007	3	m_k	R	–	×	–	–	×	–	$C_{\max}$	TS	×
Costa et al. 2014	c	m_k	R	×	–	×	×	–	–	$C_{\max}$	GA	–
Ebrahimi et al. 2014	c	m_k	P	–	×	×	–	–	–	multi	GA	–
Engin et al. 2004	c	m_k	P	–	–	–	–	–	–	$C_{\max}$	KIS	–
Ernst et al. 2019	2	m_k	P	–	–	–	–	×	–	$C_{\max}$	Heuristik	–
Gondek 2011	c	m_k	P	–	–	–	–	×	–	w_jC_j	Heuristik	×
Han et al. 2019	c	m_k	P	–	–	–	–	×	–	$C_{\max}$	SA	×
Janiak et al. 2007	c	m_k	P	–	–	–	–	×	–	multi	TS, SA	–
Javadian et al. 2013	c	m_k	P	–	×	–	–	–	–	$C_{\max}$	GA	–
Jungwattanakit et al. 2009	c	m_k	Q	–	×	–	–	–	–	multi	SA, TS, GA	–
Keshavarz et al. 2013	c	m_k	P	–	×	×	–	–	–	$C_{\max}$	GA	–
Kianfar et al. 2012	c	m_k	P	–	×	–	–	–	–	T_j	GA	–
Klement et al. 2017	1	10	R	×	×	×	–	–	–	multi	LS	×
Laribi et al. 2016	c	1	P	–	–	–	–	–	×	$C_{\max}$	Heuristik	–
Lin et al. 2020	c	m_k	P	–	–	–	–	×	–	multi	GA	–
Logendran et al. 2006	2	2	P	–	–	–	–	–	–	$C_{\max}$	TS	–
Luo et al. 2015	c	m_k	P	–	–	×	–	–	–	$C_{\max}$	GA	×
Naderi et al. 2009	c	m_k	P	–	×	–	–	–	–	multi	SA	–
Nahhas et al. 2015	2	4,5	P	–	×	×	–	–	–	$C_{\max}$	SA, SIM	×
Niu et al. 2012	c	m_k	P	–	–	–	–	–	–	$C_{\max}$	KIS	–
Ramezanian et al. 2017	c	m_k	R	–	×	–	–	–	–	multi	PSO	×
Ruiz et al. 2006	c	m_k	R	×	×	–	–	–	–	$C_{\max}$	GA	×
Sawik 2006	c	m_k	R	–	–	–	–	–	–	multi	Heuristik	×
Shi et al. 2019	c	m_k	P	–	×	–	–	×	–	$C_{\max}$	WOA	×
Tan et al. 2018	2	m_k	P	–	–	×	×	–	–	$T_{\max}$	NS	×
Tang et al. 2006	c	m_k	P	–	–	–	–	×	–	w_jC_j	DP	×
Wang et al. 2009	c	m_k	P	–	–	–	–	×	–	w_jC_j	TS, SS	–
Wardono et al. 2004	c	m_k	P	–	–	–	–	×	–	$C_{\max}$	TS	–
Wu et al. 2003	c	m_k	R	×	–	–	–	–	–	$C_{\max}$	GA	×
Yaurima et al. 2009	c	m_k	R	×	×	×	×	×	–	$C_{\max}$	GA	×
Ying et al. 2006	c	m_k	P	–	–	–	–	–	–	$C_{\max}$	AA	–
Zhang et al. 2016	2	1	P	–	–	×	×	×	×	$C_{\max}$	Heuristik	–

AA: Ameisenalgorithmus, GA: Genetischer Algorithmus, KIS: Künstliche Imunsysteme
LS: Lokale Suche, PSO: Partikelschwarmoptimierung
SA: Simuliertes Abkühlen, SS: Streusuche, TS: Tabusuche, SIM: Simulation
multi: multikriterielle Zielsetzung

3 Handlungsbedarf und angewendete Methodik

Die gewählte Lösungsstrategie für die Lösung der in Abschnitt 2.2 definierte Klasse von Flexible Flow Shop Problemen beruht auf einer Vorgehensweise, die sich in der Praxis als Planungsgrundlage bereits bewährt hat. Basierend auf den maschinenspezifischen Auftragsfamilien (siehe Unterabschnitt 2.2.2 b)) wird zwischen wesentlichen und unwesentlichen Entscheidungen differenziert. Entscheidungen über das Umrüsten der einzelnen Maschinen auf Basis der einzelnen Auftragsfamilien werden dabei als wesentlich klassifiziert, da sie unmittelbare Auswirkungen auf die Zielgröße verursachen (siehe Unterabschnitt 2.2.3). Entscheidungen über die einzelnen Reihenfolgen der Produktionsaufträge innerhalb der Auftragsfamilien werden als unwesentlich klassifiziert, da zwischen den einzelnen Aufträgen keine reihenfolgeabhängigen Rüstzeiten auftreten und die Auswirkungen auf die Zielgröße entsprechend gering ausfallen.

In Anlehnung an Keshavarz et al. (2013), Logendran et al. (2006) ist die Ablaufplanung auf Basis von Auftragsfamilien eine bereits bewährte Methodik und findet beispielsweise auch in der Losgrößenoptimierung Anwendung (eine umfangreiche Übersicht bietet Isenberg, 2017). Im Gegensatz zu den bestehenden Forschungsarbeiten sollen im Rahmen dieser Arbeit allerdings metaheuristische Optimierungsmethoden basierend auf den Auftragsfamilien mit konventionellen heuristischen Methoden gekoppelt werden (siehe dazu auch Stalinski et al., 2018a,b, 2019b). Die wesentlichen Entscheidungen über das Umrüsten der Maschinen auf die jeweiligen Auftragsfamilien sollen dabei über die metaheuristischen Suchverfahren optimiert werden. Die konkreten Bearbeitungsreihenfolgen und Maschinenzuordnungen der einzelnen Fertigungsaufträge sollen hingegen über einfache Heuristiken bestimmt werden.

Aufbauend auf den in Unterabschnitt 2.4.4 aufgezeigten Lösungsansätzen für Flexible Flow Shop Probleme verdeutlicht Abbildung 3.1 den Hintergrund der gewählten Lösungsstrategie sowie den resultierenden Handlungsbedarf dieser Arbeit. So soll die vorliegende Klasse von Flexible Flow Shop Problemen (siehe Abschnitt 2.2) durch die Anwendung von geeigneten metaheuristischen Suchverfahren auf Basis einer indirekten Problemrepräsentation gelöst werden.

Lösungsansätze, die auf der reinen Anwendung von Heuristiken beruhen, sind aufgrund der beliebigen, multikriteriellen Zielsetzung der behandelten Problemklasse (Unterabschnitt 2.2.3) nicht zielführend. Dies begründet den Einsatz von Metaheuristiken (universelle Suchverfahren). Für die angestrebten Problemgrößen sind bei dem Einsatz von Suchverfahren mit einer direkten Problemrepräsentation extrem große Suchräume zu erwarten. Daher wird eine indirekte Form der Problemrepräsentation gewählt.

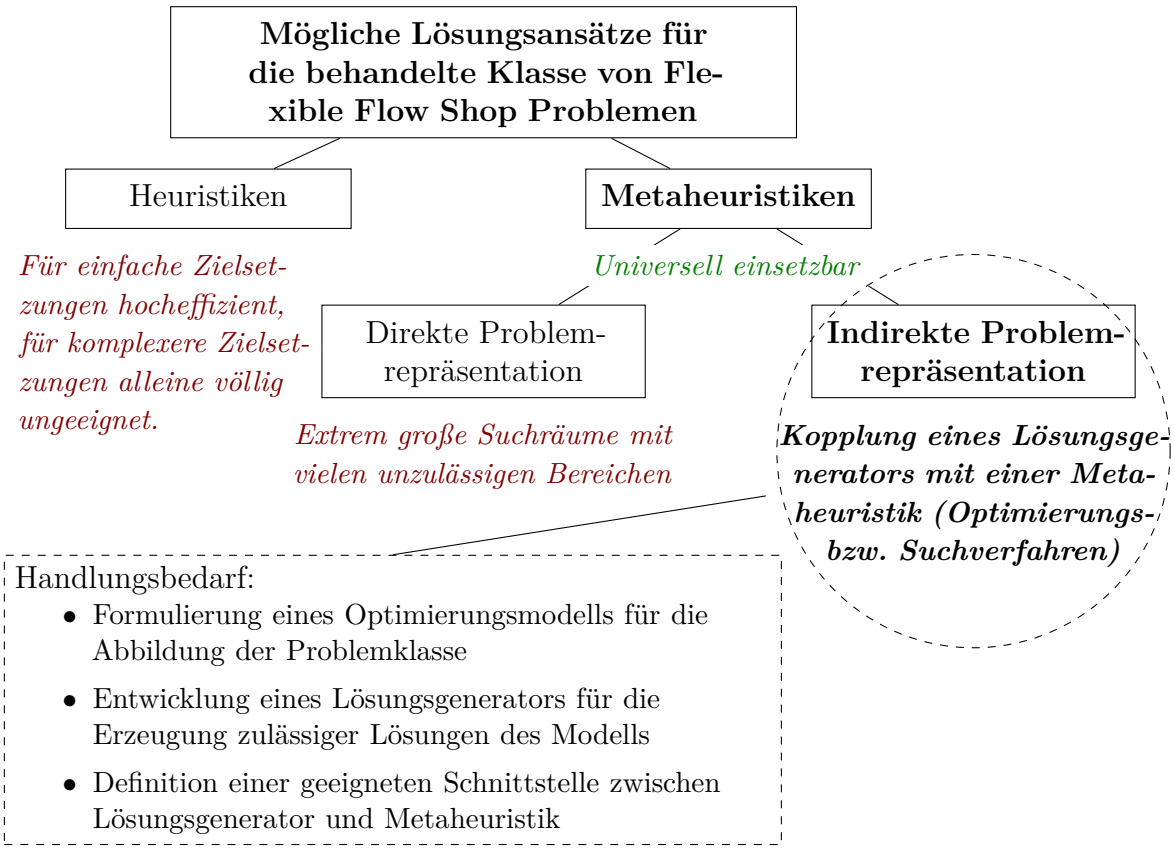


Abbildung 3.1: Gewählter Lösungsansatz und resultierender Handlungsbedarf

Dafür soll zunächst ein mathematisches, gemischt-ganzzahliges Optimierungsmodell formuliert werden, welches die behandelte Problemklasse (Abschnitt 2.2) vollständig erfasst. Die verwendeten Suchstrategien beruhen auf einer systematisch- probabilistischen Erzeugung und Bewertung von unterschiedlichen Lösungen des Optimierungsmodells. Bei jeder Bewertung einer möglichen Lösung muss das mathematische Optimierungsmodell entsprechend erfüllt sein.

Aufgrund der behandelten Randbedingungen und Restriktionen ist eine geschlossene und effiziente Lösung für die angestrebten Problemgrößen der vorliegenden Problemklasse nach derzeitigem Kenntnisstand nicht möglich. Aus diesem Grund soll das Suchverfahren über eine Schnittstelle (die indirekte Problemrepräsentation) mit einem entsprechenden Planungsverfahren in Form eines Lösungsgenerators gekoppelt werden.

Das zu entwickelnde Planungsverfahren soll auf Basis einer ereignisdiskreten Simulationsmethode (Unterabschnitt 2.4.3) zulässige Lösungen erzeugen, die das Optimierungsmodell nicht verletzen. Dafür wird ein Lösungsgenerator entwickelt, welcher mit entsprechenden Planungsregeln bzw. Heuristiken (Unterabschnitt 2.4.4 c)) für die Generierung zulässiger Ablaufpläne ausgestattet wird.

Um eine Kopplung zwischen dem Lösungsgenerator und einem Suchverfahren herstellen zu können, muss sowohl das Optimierungsmodell als auch der Lösungsgenerator um eine

geeignete Schnittstelle erweitert werden. Die gewählte Art der Kopplung – und damit der Schnittstelle – basiert dabei auf einer indirekten, der Problemklasse angepassten Problemrepräsentation (Unterabschnitt 2.4.4 d)) und ist daher ein wesentlicher Bestandteil des gesamten Lösungsansatzes.

Abbildung 3.2 zeigt das angewendete Prinzip für die Realisierung der indirekten Problemrepräsentation anhand eines Beispiels. Die Methode basiert auf der Einplanung von verschiedenen Auftragsfamilien an den einzelnen Maschinen für diskrete Planungsperioden. Der angestrebte Planungshorizont wird dafür zunächst in diskrete, äquidistante Planungsperioden aufgeteilt. In dem dargestellten Beispiel (Abbildung 3.2) wird der Planungshorizont von 24 h dafür in vier gleichmäßig aufgeteilte Planungsperioden mit einer Länge von jeweils 6 h diskretisiert.

Auf Basis der Randbedingungen der behandelten Problemklasse (Abschnitt 2.2) lassen sich die Fertigungsaufträge an den einzelnen Maschinen jeweils zu unterschiedlichen Auftragsfamilien zuordnen (Unterabschnitt 2.2.2 b)). So soll über das Suchverfahren für jede Maschine lediglich die Einplanung der entsprechenden Auftragsfamilien in den einzelnen Planungsperioden bestimmt und damit optimiert werden (Abbildung 3.2). Die konkrete Einplanung der Fertigungsaufträge (also das Bestimmen, welcher Fertigungsauftrag in welcher Reihenfolge an welcher Maschine gefertigt wird) soll dann durch den Lösungsgenerator durchgeführt werden. Der Lösungsgenerator soll dabei die Fertigungsaufträge an jeder Maschine für jede Planungsperiode in Abhängigkeit der für die Planungsperiode gewählten Auftragsfamilie über heuristische Methoden (siehe Unterabschnitt 2.4.4 c)) einplanen.

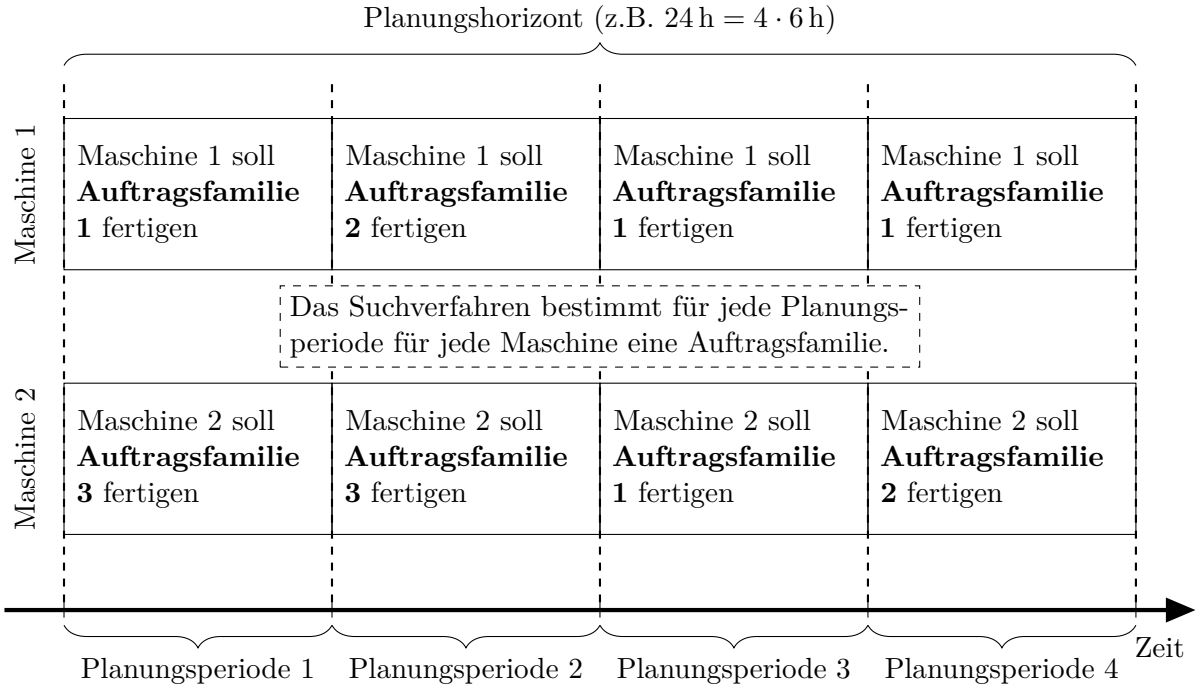


Abbildung 3.2: Definition von Planungsperioden anhand eines Beispiels

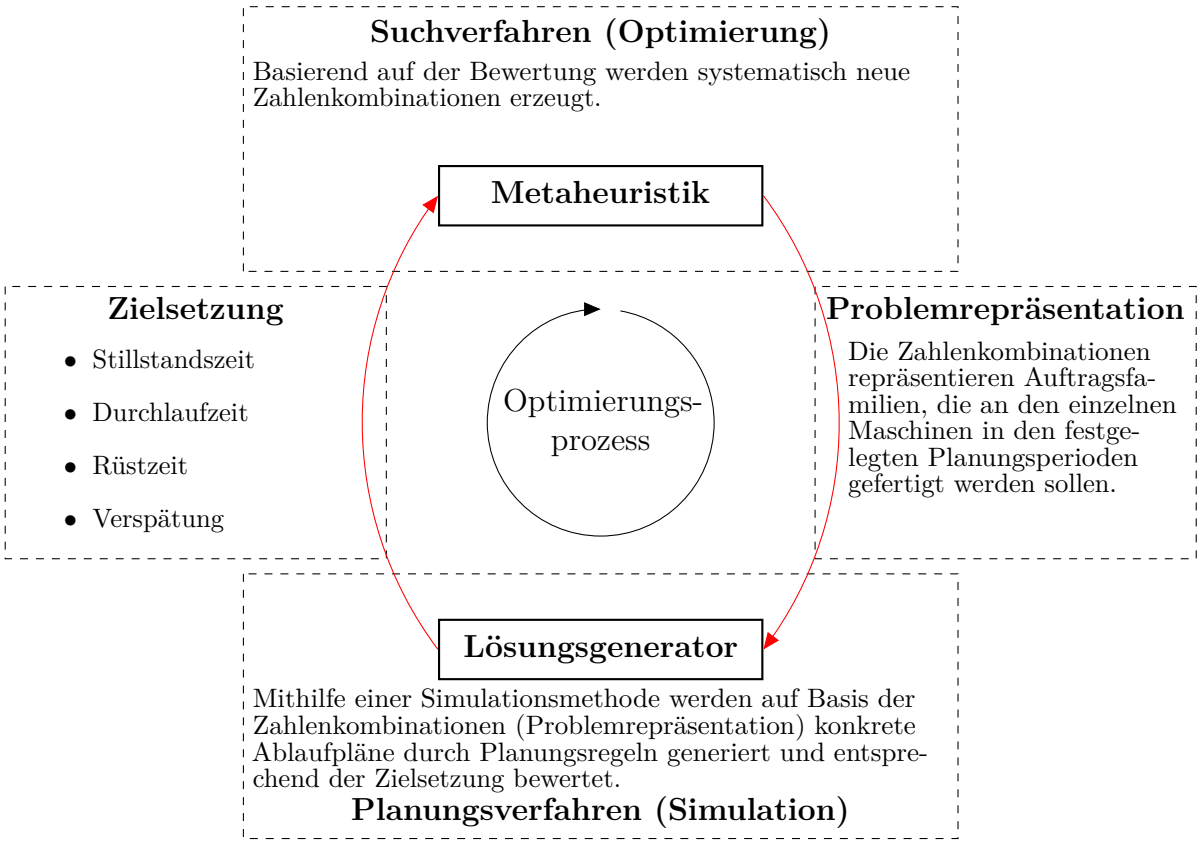


Abbildung 3.3: Angewendete Methodik (Kopplung von Simulation und Optimierung)

Die Schnittstelle zwischen dem Lösungsgenerator und dem Suchverfahren basiert entsprechend auf der Repräsentation von Auftragsfamilien, welche für die jeweiligen Planungsperioden an den einzelnen Maschinen eingeplant werden sollen. In Abbildung 3.3 ist der Optimierungsprozess auf Basis der Kopplung zwischen dem Lösungsgenerator und dem Suchverfahren als methodischer Leitfaden für diese Arbeit dargestellt.

Über das Suchverfahren (eine Metaheuristik) werden dabei zunächst unterschiedliche Zahlenkombinationen generiert. Jede Zahlenkombinationen entspricht dabei für die jeweiligen Maschinen einer Zuordnung von Auftragsfamilien zu den einzelnen Planungsperioden (Problemrepräsentation, siehe auch Abbildung 3.2).

Durch den Lösungsgenerator werden über eine ereignisdiskrete Simulationsmethode jeweils zulässige Lösungen auf Basis der von dem Suchverfahren erstellten Zahlenkombinationen generiert. Dabei werden die Zahlenkombinationen der Problemrepräsentation entsprechend als Auftragsfamilien interpretiert, die zu den einzelnen Planungsperioden an den jeweiligen Maschinen eingeplant werden. Die während der Simulation berechneten Ereigniszeitpunkte (vgl. Abbildung 2.7) sind dabei unabhängig von den Planungsperioden. In Abbildung 3.4 ist die Unterscheidung zwischen den Planungsperioden und den Ereigniszeitpunkten hervorgehoben.

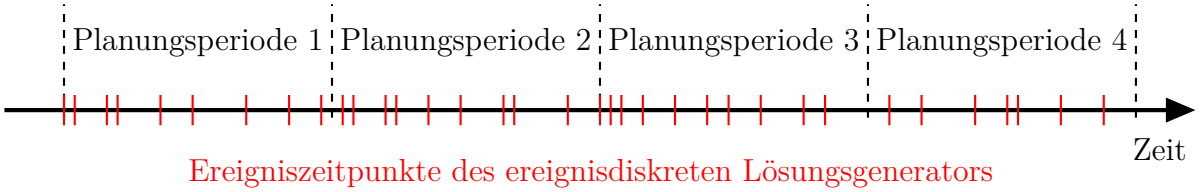


Abbildung 3.4: Überlagerung der Planungsperioden mit Ereigniszeitpunkten

Über die Definition der multikriteriellen Zielsetzung werden die erzeugten Ergebnisse schließlich bewertet und dem Suchverfahren rückgekoppelt. Dieser Suchprozess wird solange wiederholt, bis er durch das Erreichen eines Abbruchkriteriums (z.B. das Überschreiten einer bestimmten Zeitspanne) abgebrochen wird.

Auf diese Weise sollen die Anforderungen an das Planungsverfahren und damit die Zielsetzung dieser Arbeit (Abschnitt 1.2) möglichst effizient erfüllt werden. Die gewählte Methode erlaubt die Anwendung optimierender Suchverfahren bei der gleichzeitigen Anwendung von heuristischen Planungsregeln für die Berechnung von Ablaufplänen. Durch den Einsatz von metaheuristischen Suchverfahren wird die Anforderung an die multikriterielle Optimierung von beliebigen Zielgrößendefinitionen erfüllt.

Für jede Produktionsstufe sollen in dem Lösungsgenerator unterschiedliche heuristische Planungsregeln gewählt werden können (siehe Unterabschnitt 2.4.4 c)). Die Anforderung an ein nachvollziehbares und steuerbares Lösungsverhalten wird dadurch entsprechend erfüllt.

Die geforderte Kontrolle über das Laufzeitverhalten sowie die Skalierbarkeit der Genauigkeit des Planungsverfahrens erschließen sich aus der Ausgestaltung der entsprechenden Verfahrensparameter. Die Auswahl der Planungsperioden wirkt sich direkt auf die zu erwartende Laufzeit und erreichbare Lösungsgüte des Verfahrens aus. Anschaulich kann dies mit der Wahl eines entsprechenden Rechengitters (wie beispielsweise bei der FEM) verglichen werden.

In Bezug auf das Beispiel aus Abbildung 3.2 führt eine Verringerung der Anzahl an Planungsperioden (von vier auf z.B. zwei) zu einem wesentlich kleineren Suchraum. Der Optimierungsprozess wird dann entsprechend bereits nach einer kürzeren Zeit ein gutes Ergebnis finden. Die damit durch den Lösungsgenerator erzielbare Qualität der Ablaufpläne wird allerdings entsprechend geringer ausfallen.

Wird die Anzahl der Planungsperioden erhöht (von vier auf z.B. acht), so ist die Suche in dem vergrößerten Suchraum wiederum wesentlich komplexer. Die Wahrscheinlichkeit, in dem Suchraum eine optimale Zahlenkombination zu finden, ist entsprechend geringer, so dass dem Suchprozess mehr Zeit für die Optimierung eingeräumt werden müsste. Die durch den Lösungsgenerator erzielbare Qualität wird allerdings entsprechend höher ausfallen. Es muss daher ein Kompromiss zwischen in Kauf zu nehmender Rechenzeit und erzielbarer Lösungsqualität auf Basis der zur Verfügung stehenden Hardware-Ressourcen eingegangen werden.

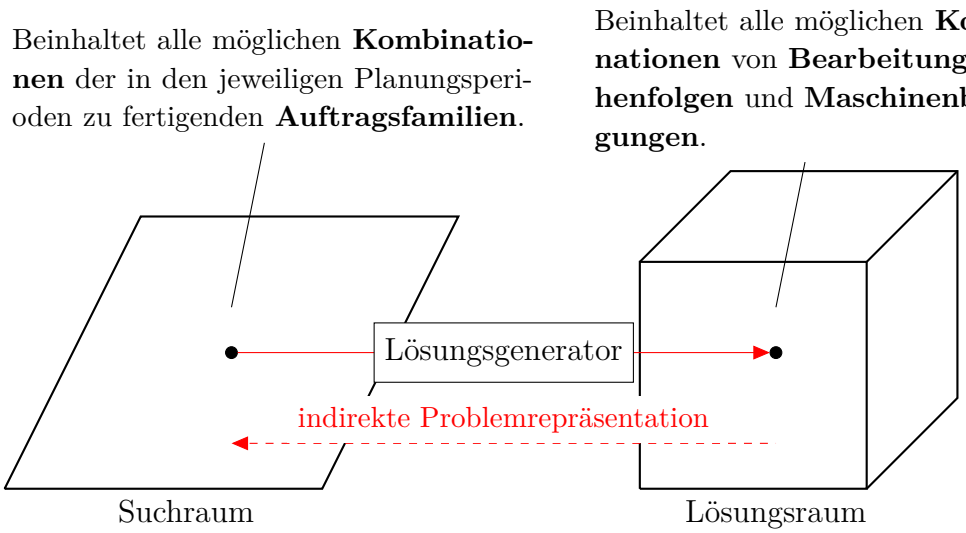


Abbildung 3.5: Entkopplung von Suchraum und Lösungsraum

Basierend auf dem Vorhandensein der maschinenspezifischen Auftragsfamilien wird bei der kombinatorischen Optimierung durch das metaheuristische Suchverfahren der Fokus auf die wesentlichen Stellgrößen gelegt. Der Rechenaufwand für die Suche wird dadurch auf die im Rahmen der behandelten Problemklasse relevanten Größen (die Auftragsfamilien) reduziert. Diese Form der Ablaufplanung (die „Familienstrategie“) wird in der operativen Produktionsplanung und -steuerung realer Produktionssysteme durch den Menschen bereits manuell eingesetzt.

Der Fokus dieser Strategie liegt auf dem Treffen wesentlicher Entscheidungen. Dabei wird eine Entscheidung über das Umrüsten einer Maschine wesentlich höher gewichtet als Entscheidungen über die Bearbeitungsreihenfolgen einzelner Fertigungsaufträge zwischen denen keine Rüstzeiten auftreten. Der Rechenaufwand für die Optimierung dieser nahezu irrelevanten Einzelreihenfolgen mit Metaheuristiken wird entsprechend ersetzt durch den Einsatz von einfachen, heuristischen Planungsregeln mit entsprechend geringem Rechenaufwand.

Auf diese Weise wird die Suche nach guten Ablaufplänen durch die indirekte Problemrepräsentation von dem eigentlichen Lösungsraum der Problemklasse entkoppelt (Abbildung 3.5). Während der Lösungsraum alle möglichen Lösungen eines konkreten Problems enthält, enthält der Suchraum nur die möglichen Einplanungen von Auftragsfamilien in den Planungsperioden. Die Suche durch das metaheuristische Suchverfahren findet dadurch entsprechend in einem wesentlich kleineren Suchraum (in Abbildung 3.5 angedeutet durch die Reduktion einer räumlichen Dimension) statt. Durch den Lösungsgenerator werden die von dem Suchverfahren erstellten Zahlenkombination aus dem Suchraum entsprechend in zulässige Lösungen im Lösungsraum transformiert.

4 Entwicklung eines ereignisdiskreten Planungsverfahrens

Dieses Kapitel widmet sich der Entwicklung des ereignisdiskreten Planungsverfahrens. Dazu wird in Abschnitt 4.1 zunächst ein gemischt-ganzzahliges Optimierungsmodell des Planungsproblems formuliert. Dieses wird ausgehend von einem Basismodell sukzessive um die jeweiligen Randbedingungen und Restriktionen sowie um die entsprechende multikriterielle Zielsetzung erweitert.

In Abschnitt 4.2 wird das Modell nochmals erweitert. Durch die Diskretisierung des Planungshorizonts in gleichmäßig aufgeteilte Planungsperioden werden dabei die Entscheidungsmatrizen auf Basis der maschinenbezogenen Auftragsfamilien für die Planungsperioden gebildet.

Schließlich wird in Abschnitt 4.3 der ereignisdiskrete Lösungsgenerator formuliert. Dieser ist in der Lage, auf Basis der entsprechenden Daten zulässige Lösungen für das mathematische Modell zu erzeugen.

4.1 Formulierung eines gemischt-ganzzahligen Optimierungsmodells

4.1.1 Nomenklatur

a) Indizes

k . . . . . Index der Produktionsstufen
i . . . . . Index der Maschinen
j . . . . . Index der Fertigungsaufträge
f . . . . . Index der Auftragsfamilien
b . . . . . Index der Auftragsstapel
r . . . . . Index der Ressourcen

b) Objekte

K_k . . . . . Produktionsstufe (stage) k
$M_{k,i}$ . . . . . Maschine (machine) i der Produktionsstufe K_k
J_j . . . . . Fertigungsauftrag (job) j

$F_{k,i,f}$ . . . . . Auftragsfamilie (family) f der Maschine $M_{k,i}$
$B_{k,b}$ . . . . . Auftragsstapel (batch) b der Produktionsstufe K_k
R_r . . . . . Ressource (resource) r

c) Anzahlen

c . . . . . Anzahl der Produktionsstufen
m_k . . . . . Anzahl der Maschinen in Produktionsstufe K_k
n . . . . . Anzahl der Fertigungsaufträge
$o_{k,i}$ . . . . . Anzahl der Auftragsfamilien der Maschine $M_{k,i}$
q_k . . . . . Anzahl der Fertigungsstapel in Produktionsstufe K_k
$n_{k,b}$ . . . . . Anzahl der Jobs in Fertigungsstapel $B_{k,b}$ in Stufe K_k
l . . . . . Anzahl der Ressourcen (Rohmaterialien)

d) Mengen

$\mathcal{K}$ . . . . . Menge der Produktionsstufen
$\mathcal{M}_k$ . . . . . Menge der Maschinen in Produktionsstufe K_k
$\mathcal{J}$ . . . . . Menge der Fertigungsaufträge
$\mathcal{F}_{k,i}$ . . . . . Menge der Auftragsfamilien der Maschine $M_{k,i}$
$\mathcal{B}_k$ . . . . . Menge der Auftragsstapel der Produktionsstufe K_k
$\mathcal{R}$ . . . . . Menge der Ressourcen (Rohmaterialien)

e) Vorgabegrößen

d_j . . . . . Planliefertermin (due date) von Job J_j
$p_{k,i,j}$ . . . . . Prozesszeit (processing time) von Job J_j auf Maschine $M_{k,i}$
w_j . . . . . Gewichtung (weight) von Job J_j
$\sigma_{k,j}$ . . . . . Notwendigkeit von Stufe K_k für die Bearbeitung von Job J_j
$\lambda_{k,i,j}$ . . . . . Kompatibilität von Maschine $M_{k,i}$ für die Bearbeitung von Job J_j
$\nu_{k,j,b}$ . . . . . Zugehörigkeit des Jobs J_j zu Batch $B_{k,b}$ in Stufe K_k
$\mu_{k,i,j,f}$ . . . . . Zugehörigkeit des Jobs J_j zu Auftragsfamilie $F_{k,i,f}$
$\pi_{k,j,r}$ . . . . . Verbrauch der Ressource R_r durch J_j in K_k
ω_r . . . . . Verfügbarkeit der Ressource R_r zum Zeitpunkt $t = 0$
ϕ_{k,i,f_1,f_2} . . . . . Rüstzeit von F_{f_1} auf F_{f_2} auf Maschine $M_{k,i}$
ρ_k . . . . . Pufferkapazität der Produktionsstufe K_k
$\eta_{k,j}$ . . . . . Pufferbedarf des Jobs J_j in der Produktionsstufe K_k

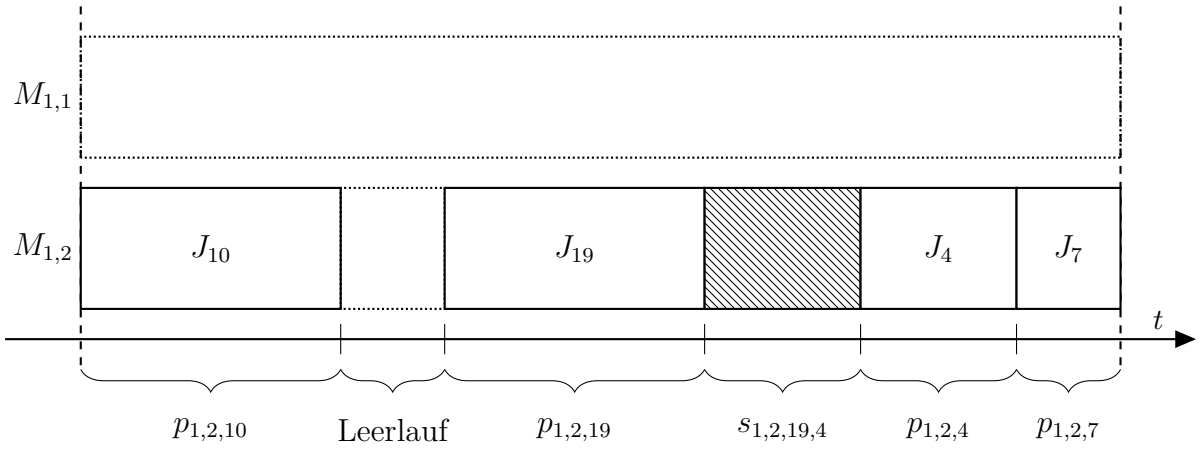
f) Ergebnisgrößen

$S_{k,j}$ . . . . . Startzeit (start time) von Job J_j in Stufe K_k
$C_{k,j}$ . . . . . Endzeit (completion time) von Job J_j in Stufe K_k
F_j . . . . . Durchlaufzeit oder Flusszeit (flow time) von Job J_j
L_j . . . . . Terminabweichung (lateness) von Job J_j
E_j . . . . . Verfrühung (earliness) von Job J_j

T_j . . . . . Verspätung (tardiness) von Job J_j
H_j . . . . . Schlupf (slack) von Job J_j
$I_{k,i}$ . . . . . Leerlaufzeit (idle time) von Maschine $M_{k,i}$
$X_{k,i,j}$ . . . . . wurde der Job J_j auf der Maschine $M_{k,i}$ gefertigt?
Z_{k,i,j_1,j_2} . . . . . wurde Job J_{j_1} auf $M_{k,i}$ unmittelbar vor Job J_{j_2} gefertigt?
$\Psi_{k,i,j,f}$ . . . . . wurde Job J_j als Auftragsfamilie $F_{k,i,f}$ bearbeitet?
s_{k,i,j_1,j_2} . . . . . Rüstzeit (setup time) von Job J_{j_1} auf Job J_{j_2} auf Maschine $M_{k,i}$

g) Darstellung von Ablaufplänen

Die im weiteren Verlauf dieser Arbeit verwendete Methode für die Darstellung von Ablaufplänen ist in Abbildung 4.1 dargestellt. In Anlehnung an die von Gantt eingeführte Konvention (siehe z.B. Pinedo, 2016) werden die jeweiligen Produktionsstufen und Maschinen auf der y-Achse übereinander aufgetragen. Die Jobs werden entsprechend ihrer Produktionszeit auf der jeweiligen Maschine horizontal über die Zeitachse t aufgetragen. Rüst- und Leerlaufzeiten werden (falls erforderlich) gemäß Abbildung 4.1 gekennzeichnet.



$M_{1,1}$: Maschine ($i = 1$) der Produktionsstufe K_1
$M_{1,2}$: Maschine ($i = 2$) der Produktionsstufe K_1
$p_{1,2,10}$: Zeit, um Job J_{10} auf Maschine $M_{1,2}$ zu bearbeiten (Produktionszeit)
$p_{1,2,19}$: Zeit, um Job J_{19} auf Maschine $M_{1,2}$ zu bearbeiten (Produktionszeit)
$s_{1,2,19,4}$: Zeit, um Maschine $M_{1,2}$ von Job J_{19} auf Job J_4 zu rüsten (Rüstzeit)
$p_{1,2,4}$: Zeit, um Job J_4 auf Maschine $M_{1,2}$ zu bearbeiten (Produktionszeit)
$p_{1,2,7}$: Zeit, um Job J_7 auf Maschine $M_{1,2}$ zu bearbeiten (Produktionszeit)
Rüstzeiten: schraffiert

Abbildung 4.1: Methode für die Darstellung von Ablaufplänen

4.1.2 Allgemeine Definitionen

Formeln 4.1 bis 4.6 definieren die im weiteren Verlauf der Arbeit verwendeten Mengen (vgl. Unterunterabschnitt d)).

$$\mathcal{K} = \{K_1, \ldots, K_k, \ldots, K_c\} \tag{4.1}$$
$$\mathcal{M}_k = \{M_{k,1}, \ldots, M_{k,i}, \ldots, M_{k,m_k}\} \tag{4.2}$$
$$\mathcal{J} = \{J_1, \ldots, J_j, \ldots, J_n\} \tag{4.3}$$
$$\mathcal{F}_{k,i} = \left\{F_{k,i,1}, \ldots, F_{k,i,f}, \ldots, F_{k,i,o_{k,i}}\right\} \tag{4.4}$$
$$\mathcal{B}_k = \{B_{k,1}, \ldots, B_{k,b}, \ldots, B_{k,q_k}\} \tag{4.5}$$
$$\mathcal{R} = \{R_1, \ldots, R_r, \ldots, R_l\} \tag{4.6}$$

Um den Platzbedarf bei der Formulierung des Modells zu senken und dadurch die Lesbarkeit deutlich zu verbessern, sollen im weiteren Verlauf der Arbeit die in den Formeln 4.7 bis 4.12 festgelegten Regeln für eine abgekürzte Schreibweise der Mengenenumeration gelten.

$$\forall k \;\hat{=}\; \forall\, K_k \in \mathcal{K} \tag{4.7}$$
$$\forall i \;\hat{=}\; \forall\, M_i \in \mathcal{M}_k \tag{4.8}$$
$$\forall j \;\hat{=}\; \forall\, J_j \in \mathcal{J} \tag{4.9}$$
$$\forall f \;\hat{=}\; \forall\, F_{k,i,f} \in \mathcal{F}_{k,i} \tag{4.10}$$
$$\forall b \;\hat{=}\; \forall\, B_{k,b} \in \mathcal{B}_k \tag{4.11}$$
$$\forall r \;\hat{=}\; \forall\, R_r \in \mathcal{R} \tag{4.12}$$

Die Flusszeit (**flow time**) F_j eines Fertigungsauftrags ist definiert als die Zeitspanne zwischen der Start- und der Endzeit (Formel 4.13). Die Terminabweichung (**lateness**) L_j eines Jobs wird als Differenz zwischen der Fertigstellungszeit und des Planliefertermins berechnet (Formel 4.14) (Pinedo, 2016, S. 18–19).

$$F_j = C_j - S_j \tag{4.13}$$
$$L_j = C_j - d_j \tag{4.14}$$

Die Verspätung (**tardiness**) T_j eines Jobs wird gemäß Formel 4.15 berechnet. Dabei werden entsprechend nur positive Terminabweichungen berücksichtigt. Mit Formel 4.16 wird die Verfrühung (die verfrühte Fertigstellung, **earliness**) E_j eines Jobs berechnet. Es werden entsprechend nur die negativen Terminabweichungen berücksichtigt (Jaehn et al., 2019, S. 21–23).

$$T_j = \max(0, L_j) \tag{4.15}$$
$$E_j = \max(0, -L_j) \tag{4.16}$$

4.1.3 Formulierung des Basismodells

Aufgrund des gesamten Umfang des Modells wird als Basis zunächst ein Modell allein für die Maschinenumgebung formuliert.

$$\min \{w_1'\gamma_1 + w_2'\gamma_2 + w_3'\gamma_3 + w_4'\gamma_4\} \tag{4.17}$$

so dass

$$\sum_{i=1}^{m_k} X_{k,i,j} \leq 1 \qquad \forall k,j \tag{4.18}$$

$$Z_{k,i,j,j} = 0 \qquad \forall k,i,j \tag{4.19}$$

$$\sum_{j_2=1}^{n} (Z_{k,i,j,j_2}) \leq X_{k,i,j} \qquad \forall k,i,j \tag{4.20}$$

$$\sum_{j_2=1}^{n} (Z_{k,i,j_2,j}) \leq X_{k,i,j} \qquad \forall k,i,j \tag{4.21}$$

$$\sum_{k_1=1}^{k-1} \sum_{i=1}^{m_{k_1}} X_{k_1,i,j} \geq (k-1) \cdot \sum_{i=1}^{m_k} X_{k,i,j} \qquad \forall k,j \tag{4.22}$$

$$C_{k,j} \geq S_{k,j} + \sum_{i=1}^{m_k} (X_{k,i,j} \cdot p_{k,i,j}) \qquad \forall k,j \tag{4.23}$$

$$S_{k,j} \geq C_{k-1,j} \qquad \forall k,j \tag{4.24}$$

$$S_{k,j} \geq C_{k,j_2} \cdot \sum_{i=1}^{m_k} (X_{k,i,j} \cdot X_{k,i,j_2} \cdot Z_{k,i,j_2,j}) \qquad \forall k,j,j_2 \tag{4.25}$$

$$S_j \leq \min \left(\sum_{i=1}^{m_k} (S_{k,j} \cdot X_{k,i,j}) \right) \qquad \forall k,j \tag{4.26}$$

$$C_j \geq \max \left(\sum_{i=1}^{m_k} (C_{k,j} \cdot X_{k,i,j}) \right) \qquad \forall k,j \tag{4.27}$$

$$X_{k,i,j} \in \{0,1\} \qquad \forall k,i,j \tag{4.28}$$

$$Z_{k,i,j_1,j_2} \in \{0,1\} \qquad \forall k,i,j_1,j_2 \tag{4.29}$$

$$p_{k,i,j} \geq 0 \qquad \forall k,i,j \tag{4.30}$$

$$S_{k,j} \geq 0 \qquad \forall k,j \tag{4.31}$$

$$C_{k,j} \geq 0 \qquad \forall k,j \tag{4.32}$$

Die Einbindung der Randbedingungen und Restriktionen sowie die explizite Formulierung der Teilzielgrößen erfolgt in Unterabschnitt 4.1.4 und 4.1.5. Das Basismodell entspricht der Problemklasse aus Formel 4.33.

$$FFc\,(Rm_k)_{k=1}^{c}\;\Big|\Big|\;w_1'\gamma_1 + w_2'\gamma_2 + w_3'\gamma_3 + w_4'\gamma_4 \tag{4.33}$$

Formel 4.17 gibt an, dass es sich bei dem Modell um die Minimierung der gewichteten Zielgrößen ($w_1'\gamma_1 + w_2'\gamma_2 + w_3'\gamma_3 + w_4'\gamma_4$) handelt. Durch Formel 4.18 wird sichergestellt, dass jeder Job J_j in jeder Stufe K_k höchstens an einer Maschine $M_{k,i}$ gefertigt wird. Jeder Fertigungsauftrag darf dabei an jeder Maschine höchstens einen Vorgänger bzw. Nachfolger haben (Formeln 4.20 und 4.21) und darf an keiner Maschine sein eigener Vorgänger oder Nachfolger sein (Formel 4.19).

Formel 4.22 stellt sicher, dass jeder Job in jeder Stufe erst bearbeitet werden kann, wenn die Bearbeitung des Jobs in der vorherigen Stufe bereits beendet ist. Die Startzeit der Bearbeitung eines Jobs muss in jeder Stufe mindestens der Endzeit in der vorherigen Stufe entsprechen (Formel 4.24). Die Endzeit eines Jobs muss für jede Stufe mindestens um die benötigte Bearbeitungszeit größer sein als die entsprechende Startzeit in der Stufe (Formel 4.23).

Durch Formel 4.25 wird sichergestellt, dass die Startzeit eines Jobs in einer Stufe größer sein muss als die Endzeit des direkten Vorgänger-Jobs an der Maschine. Formeln 4.26 und 4.27 sorgen dafür, dass die Start- und Endzeiten der Jobs in jeder Produktionsstufe mindestens der absoluten Startzeit bzw. höchstens der absoluten Endzeit des jeweiligen Jobs entsprechen müssen.

Formeln 4.28 und 4.29 stellen sicher, dass die Entscheidungsmatrizen $X_{k,i,j}$ und Z_{k,i,j_1,j_2} für die Angaben, ob ein Job auf einer Maschine einer Stufe gefertigt wurde (Formel 4.28) bzw. ob ein Job der direkte Vorgänger eines anderen Jobs ist (Formel 4.29), nur die Werte 0 und 1 annehmen können. Formeln 4.30 bis 4.32 stellen sicher, dass es sich bei den Angaben der Prozesszeiten $p_{k,i,j}$ sowie bei den Start- und Endzeitpunkten ($S_{k,i,j}$ und $C_{k,i,j}$) nur um positive Zeitangaben handeln kann.

4.1.4 Erweiterung des Basismodells

Das Basismodell aus Unterabschnitt 4.1.3 wird in den folgenden Abschnitten sukzessive um die jeweiligen Randbedingungen erweitert.

a) Prozessmatrix

Durch die Formeln 4.34 bis 4.37 wird das Basismodell dahingehend ergänzt, dass die Fertigungsaufträge nicht in jeder Produktionsstufe bearbeitet werden müssen.

$$\sum_{i=1}^{m_k} X_{k,i,j} \leq \sigma_{k,j} \qquad \forall k,j \qquad (4.34)$$

$$\sum_{k_1=1}^{k-1} \sum_{i=1}^{m_{k_1}} X_{k_1,i,j} \geq \sum_{k_1=1}^{k-1} \sigma_{k_1,j} \cdot \sum_{i=1}^{m_k} X_{k,i,j} \qquad \forall k,j \qquad (4.35)$$

$$\sum_{k=1}^{c} \sigma_{k,j} \geq 1 \qquad \forall j \qquad (4.36)$$

$$\sigma_{k,j} \in \{0,1\} \qquad \forall k,j \qquad (4.37)$$

Formel 4.34 ergänzt Formel 4.18 und stellt sicher, dass jeder Job J_j in jeder Stufe nur auf einer Maschine bearbeitet wird, wenn die Bearbeitung laut der Matrix $\sigma_{k,j}$ notwendig ist. Durch Formel 4.35 wird Formel 4.22 derart ergänzt, dass dass jeder Job J_j in jeder Stufe erst bearbeitet werden kann, wenn der Job in allen durch die Matrix $\sigma_{k,j}$ festgelegten Produktionsstufen bereits bearbeitet wird.

Durch Formel 4.36 wird sichergestellt, dass jeder Job mindestens in einer Stufe bearbeitet werden muss. Formel 4.37 stellt sicher, dass die Matrix $\sigma_{k,j}$ nur mit den Werten 0 und 1 besetzt sein kann.

b) Kompatibilitäten

Durch die Formeln 4.38 bis 4.40 wird berücksichtigt, dass nicht jede Maschine für die Bearbeitung jedes Jobs zulässig ist ($\beta = M_j$).

$$\sum_{i=1}^{m_k} (X_{k,i,j} \cdot \lambda_{k,i,j}) \leq \sigma_{k,j} \qquad \forall k,j \qquad (4.38)$$

$$\sum_{i=1}^{m_k} \lambda_{k,i,j} \geq \sigma_{k,j} \qquad \forall k,j \qquad (4.39)$$

$$\lambda_{k,i,j} \in \{0,1\} \qquad \forall k,i,j \qquad (4.40)$$

Formel 4.38 ergänzt Formel 4.34 und stellt sicher, dass jeder Job in jeder Stufe nur auf einer Maschine bearbeitet wird, falls diese kompatibel ist. Durch Formel 4.39 wird sichergestellt, dass es für jeden Job in jeder Stufe mindestens eine kompatible Maschine gibt, falls die Stufe laut der Matrix $\sigma_{k,j}$ für die Bearbeitung des Jobs notwendig ist. Formel 4.40 legt fest, dass die Matrix $\lambda_{k,i,j}$ nur die Werte 0 und 1 annehmen kann.

c) Reihenfolgeabhängige Rüstzeiten

Formeln 4.41 und 4.42 erweitern das Modell um die Berücksichtigung reihenfolgeabhängiger Rüstzeiten ($\beta = s_{k,i,j_1,j_2}$).

$$S_{k,j} \geq (C_{k,j_2} + s_{k,i,j_2,j}) \cdot \sum_{i=1}^{m_k} (X_{k,i,j} \cdot X_{k,i,j_2} \cdot Z_{k,i,j_2,j}) \qquad \forall k, j, j_2 \qquad (4.41)$$

$$s_{k,i,j_1,j_2} \geq 0 \qquad \forall k, i, j_1, j_2 \qquad (4.42)$$

Formel 4.41 stellt sicher, dass die Startzeit $S_{k,j}$ eines Jobs J_j in einer Stufe mindestens um die reihenfolgeabhängige Rüstzeit $s_{k,i,j_2,j}$ größer sein muss als die Fertigstellungszeit C_{k,j_2} des an der Maschine vorherig bearbeiteten Auftrags. Durch Formel 4.42 werden negative Rüstzeiten ausgeschlossen.

d) Auftragsfamilien

Die Erweiterung des Modells um maschinenspezifische Auftragsfamilien (β = fmls) erfolgt durch die Formeln 4.43 bis 4.50.

$$\sum_{f=1}^{o_{k,i}} \mu_{k,i,j,f} \geq \sigma_{k,j} \cdot \lambda_{k,i,j} \qquad \forall k, i, j \qquad (4.43)$$

$$s_{k,i,j_1,j_2} = \sum_{f_1=1}^{o_{k,i}} \sum_{f_2}^{o_{k,i}} (\phi_{k,i,f_1,f_2} \cdot \Psi_{k,i,j_1,f_1} \cdot \Psi_{k,i,j_2,f_2}) \qquad \forall k, i, j_1, j_2 \qquad (4.44)$$

$$\sum_{f=1}^{o_{k,i}} (\Psi_{k,i,j,f} \cdot \mu_{k,i,j,f}) = X_{k,i,j} \qquad \forall k, i, j \qquad (4.45)$$

$$X_{k,i,j} = \sum_{f=1}^{o_{k,i}} \Psi_{k,i,j,f} \qquad \forall k, i, j \qquad (4.46)$$

$$\phi_{k,i,f,f} = 0 \qquad \forall k, i, f \qquad (4.47)$$

$$\mu_{k,i,j,f} \in \{0, 1\} \qquad \forall k, i, j, f \qquad (4.48)$$

$$\Psi_{k,i,j,f} \in \{0, 1\} \qquad \forall k, i, j, f \qquad (4.49)$$

$$\phi_{k,i,f_1,f_2} \geq 0 \qquad \forall k, i, f_1, f_2 \qquad (4.50)$$

Formel 4.43 stellt dafür durch die Matrix $\mu_{k,i,j,f}$ sicher, dass jeder Job J_j an jeder Maschine mindestens einer Familie $F_{k,i,f}$ zugehörig ist, falls dieser in dieser Stufe bearbeitet werden muss und mit der Maschine kompatibel ist. Durch Formel 4.44 lässt sich die

reihenfolgeabhängige Rüstzeit zweier Jobs durch die reihenfolgeabhängige Rüstzeit der entsprechenden Auftragsfamilien berechnen.

Wird ein Job auf einer Maschine eingeplant, so wird durch Formel 4.45 dafür gesorgt, dass eine entsprechende Familie festgelegt wird, welcher der Job angehörig ist. Formel 4.46 stellt dabei sicher, dass für jeden Job nur genau eine Familie festgelegt wird, falls dieser bearbeitet wird.

Innerhalb einer Auftragsfamilie treten keine Rüstzeiten auf, dies wird durch Formel 4.47 sichergestellt. Formeln 4.48 und 4.49 stellen sicher, dass die Einträge der Matrizen $\mu_{k,i,j,f}$ und $\Psi_{k,i,j,f}$ nur mit den Werten 0 und 1 besetzt werden können. Formel 4.50 schließt negative Rüstzeiten aus.

e) Auftragsstapel

Das Modell wird durch die Formeln 4.51 bis 4.56 um die Berücksichtigung der gleichzeitigen Bearbeitung mehrerer Aufträge als Auftragsstapel ($\beta = \text{batch}$) ergänzt.

$$n_{k,b} = \sum_{j=1}^{n} \nu_{k,j,b} \qquad \forall k, b \qquad (4.51)$$

$$\sum_{j=1}^{n} (X_{k,i,j} \cdot \nu_{k,j,b}) \in \{0, n_{k,b}\} \qquad \forall k, i, b \qquad (4.52)$$

$$\sum_{j_1=1}^{n} \sum_{j_2=1}^{n} (Z_{k,i,j_1,j_2} \cdot \nu_{k,j_1,b} \cdot \nu_{k,j_2,b}) \in \{0, n_{k,b} - 1\} \qquad \forall k, i, b \qquad (4.53)$$

$$\sum_{j_1=1}^{n} \sum_{j_2=1}^{n} (s_{k,i,j_1,j_2} \cdot \nu_{k,j_1,b} \cdot \nu_{k,j_2,b}) = 0 \qquad \forall k, b \qquad (4.54)$$

$$\sum_{b=1}^{q_k} \nu_{k,j,b} \leq 1 \qquad \forall k, j \qquad (4.55)$$

$$\nu_{k,j,b} \in \{0, 1\} \qquad \forall k, j, b \qquad (4.56)$$

Durch die Matrix $\nu_{k,j,b}$ wird entsprechend für jeden Job in jeder Produktionsstufe die Zugehörigkeit zu einem Auftragsstapel festgelegt. Formel 4.51 stellt dabei für jede Stufe sicher, dass die Größe eines Stapels $B_{k,b}$ mit der Anzahl $n_{k,b}$ der ihm zugeordneten Jobs übereinstimmt.

Durch Formel 4.52 wird dafür gesorgt, dass alle Fertigungsaufträge eines Batches auf derselben Maschine gefertigt werden. Die Fertigungsaufträge eines Batches müssen zusammenhängend und ohne Unterbrechung an einer entsprechenden Maschine gefertigt werden (Formel 4.53). Formel 4.54 stellt dafür sicher, dass zwischen den Jobs eines Batches keine Rüstzeiten auftreten können.

Jeder Job kann in jeder Produktionsstufe nur maximal einem Auftragsstapel zugeordnet sein (Formel 4.55). Formel 4.56 stellt sicher, dass die Matrix $\nu_{k,j,b}$ nur mit den Werten 0 und 1 besetzt sein kann.

f) Ressourcen

Durch die Formeln 4.57 bis 4.59 wird das Modell um die Berücksichtigung begrenzter Ressourcen ($\beta = \text{res}$) in Form von Rohmaterialien ergänzt.

$$\sum_{k=1}^{c} \sum_{i=1}^{m_k} \sum_{j=1}^{n} \left(X_{k,i,j} \cdot \pi_{k,j,r} \right) \leq \omega_r \qquad \forall r \qquad (4.57)$$

$$\omega_r \geq 0 \qquad \forall r \qquad (4.58)$$

$$\pi_{k,j,r} \geq 0 \qquad \forall k, j, r \qquad (4.59)$$

Für jeden Job J_j wird entsprechend durch Formel 4.57 sichergestellt, dass dieser nur verplant werden kann, wenn sein Bedarf $\pi_{k,j,r}$ an der Ressource R_r durch den Bestand ω_r gedeckt ist. Formeln 4.58 und 4.59 schließen dafür negative Bestände und negative Bedarfe aus.

g) Pufferkapazitäten

Die Berücksichtigung der begrenzten Pufferkapazitäten ($\beta = \text{block}$) erfolgt durch die Erweiterung des Modells durch die Formeln 4.60 bis 4.66. Es muss für jeden Zeitpunkt t innerhalb eines Ablaufplanes sichergestellt werden, dass die Belastung der Logistikpuffer durch die Fertigungsaufträge die zur Verfügung stehenden Kapazitäten nicht überschreitet.

$$S'_{k,j}(t) = \begin{cases} 1 & \text{für } S_{k,j} < t \\ 0 & \text{für } S_{k,j} \geq t \end{cases} \qquad \forall k, j, t \qquad (4.60)$$

$$C'_{k,j}(t) = \begin{cases} 1 & \text{für } C_{k,j} < t \\ 0 & \text{für } C_{k,j} \geq t \end{cases} \qquad \forall k, j, t \qquad (4.61)$$

Durch die Formeln 4.60 und 4.61 werden dafür die binären Variablen $S'_{k,j}(t)$ und $C'_{k,j}(t)$. Diese bestimmen für einen Zeitpunkt t, ob ein Job J_j in der Produktionsstufe K_k bereits gestartet bzw. beendet wurde.

$$\zeta_{k,j}(t) = \sigma_{k,j} \cdot \left(1 - S'_{k,j}(t)\right) \cdot \prod_{k_1=1}^{k} \left(C'_{k_1,j}(t) \cdot \sigma_{k_1,j} + (1 - \sigma_{k_1,j})\right) \qquad \forall k, j, t \qquad (4.62)$$

$$\sum_{j=1}^{n} \left(\zeta_{k,j}(t) \cdot \eta_{k,j}\right) \leq \rho_k \qquad \forall k, t \qquad (4.63)$$

$$\rho_k > 0 \qquad \forall k \qquad (4.64)$$

$$\eta_{k,j} \geq 0 \qquad \forall k, j \qquad (4.65)$$

$$\zeta_{k,j}(t) \in \{0, 1\} \qquad \forall k, j, t \qquad (4.66)$$

Mithilfe von Formel 4.62 wird die binäre Variable $\zeta_{k,j}(t)$ berechnet. Diese bestimmt für einen beliebigen Zeitpunkt t, ob sich ein Job J_j im Puffer der Stufe K_k befindet. Mit Formel 4.63 wird entsprechend sichergestellt, dass zu jedem Zeitpunkt t die Pufferkapazitäten ρ_k der einzelnen Produktionsstufen nicht überschritten werden.

Durch die Formeln 4.64 und 4.65 werden negative Pufferkapazitäten und negative Pufferbedarfe der Jobs ausgeschlossen. Formel 4.66 stellt sicher, dass die Matrix $\zeta_{k,j}(t)$ nur mit 0 und 1 belegt werden kann.

4.1.5 Formulierung der Zielgrößen

Die für die behandelte Problemklasse (Formel 2.17) definierte Zielsetzung lautet:

$$\gamma = w'_1\gamma_1 + w'_2\gamma_2 + w'_3\gamma_3 + w'_4\gamma_4$$

Die konkrete Ausformulierung der dabei verwendeten Zielgrößen (γ_1, γ_2, γ_3 und γ_4) erfolgt durch die Formeln 4.67 bis 4.70.

a) Minimierung der Maschinenstillstandszeiten

Durch die Minimierung der Maschinenstillstandszeiten erfolgt implizit die Maximierung der Auslastung aller Maschinen. Für die Minimierung der Stillstandszeiten werden die summierten Bearbeitungszeiten von den an jeder Maschine bearbeiteten Fertigungsaufträge maximiert. In Formel 4.67 werden daher für alle Maschinen $M_{k,i}$ aller Produktionsstufen K_k die Bearbeitungszeiten $p_{k,i,j}$ aller Jobs J_j, die auf der entsprechenden Maschine eingeplant sind ($X_{k,i,j} = 1$) aufsummiert. Damit die Auslastung maximiert wird, muss das Vorzeichen der Summe geändert werden (Minimierung der Funktion γ_1).

$$\gamma_1 = \sum_{k=1}^{c} \sum_{i=1}^{m_k} \sum_{j=1}^{n} \left(-X_{k,i,j} \cdot p_{k,i,j}\right) \qquad (4.67)$$

b) Minimierung der Durchlaufzeiten

Die Berechnung der Durchlaufzeit F_j eines Jobs ist in Formel 4.13 gegeben. Die Minimierung der gewichteten Durchlaufzeit erfolgt durch die entsprechende Summierung (Formel 4.68).

$$\gamma_2 = \sum_{j=1}^{n} (w_j \cdot F_j) \tag{4.68}$$

c) Minimierung der Rüstzeiten

Für die Minimierung der Rüstzeiten werden in Formel 4.69 sämtliche Rüstzeiten s_{k,i,j_1,j_2} aufsummiert, die an jeder Maschine $M_{k,i}$ jeder Stufe K_k zwischen den Jobs J_{j_1} und J_{j_2} auftreten. Dafür dürfen nur die Jobs berücksichtigt werden, die an der Maschine direkt hintereinander gefertigt werden ($X_{k,i,j_1} = 1$, $X_{k,i,j_2} = 1$, $Z_{k,i,j_1,j_2} = 1$).

$$\gamma_3 = \sum_{k=1}^{c} \sum_{i=1}^{m_k} \sum_{j_1=1}^{n} \left(X_{k,i,j_1} \cdot \sum_{j_2=1}^{n} (X_{k,i,j_2} \cdot Z_{k,i,j_1,j_2} \cdot s_{k,i,j_1,j_2}) \right) \tag{4.69}$$

d) Minimierung der Verspätungen

Die Berechnung der Verspätung T_j eines Jobs ist in Formel 4.15 gegeben. Die Minimierung der gewichteten Verspätungen erfolgt durch die entsprechende Summierung (Formel 4.70).

$$\gamma_4 = \sum_{j=1}^{n} (w_j \cdot T_j) \tag{4.70}$$

4.2 Definition von diskreten Planungsperioden

In der behandelten Problemklasse lassen sich die Fertigungsaufträge auf Basis der reihenfolgeabhängigen Rüstzeiten sowie der Maschinenkonfigurationen zu maschinenspezifischen Auftragsfamilien zusammenfassen (siehe Unterabschnitt 2.2.2 b)). Fertigungsaufträge können dabei in Bezug auf eine Maschine mehreren Familien zugeordnet sein. Dementsprechend ist für die Jobs durch die Ablaufplanung festzulegen, mit welcher Auftragsfamilie sie an den entsprechenden Maschinen gefertigt werden sollen.

In der Praxis stellt sich in diesem Szenario entsprechend nicht mehr nur die Frage, welche Jobs an welcher Maschine in welcher Reihenfolge gefertigt werden sollen. Es sollte viel mehr für jede Maschine festgelegt werden, welche Auftragsfamilie sie in einem definierten Zeitabschnitt fertigen soll. Jobs, die dieser Auftragsfamilie angehören und die Bedingungen für eine Einplanung erfüllen (Materialverfügbarkeit, logistische Verfügbarkeit etc.), werden an der Maschine entsprechend ihrer Dringlichkeit eingeplant.

4.2.1 Nomenklatur

H . . . . . Zeithorizont der Planung
t . . . . . Allgemeine Zeitvariable
τ . . . . . Index Entscheidungszeitpunkte
t_τ . . . . . Entscheidungszeitpunkt τ
θ . . . . . Anzahl der Planungsperioden
$\mathcal{T}$ . . . . . Menge der Planungsperioden
ΔT . . . . . Länge einer Planungsperiode
$\delta_{k,i,\tau}$ . . . . . Entscheidungsmatrix
$\epsilon_{k,j,\tau}$ . . . . . Wird der Job J_j in der Stufe K_k in der Planungsperiode τ gefertigt?

4.2.2 Definition der Entscheidungsvariablen

Für jeden Job J_j, der im Laufe der Durchführung des Planungsverfahrens an einer Maschine $M_{k,i}$ in der Stufe K_k eingeplant wird, müssen

- die Belegung der Maschine $X_{k,i,j}$,
- die Reihenfolge auf der Maschine Z_{k,i,j,j_2},
- die Startzeit der Bearbeitung $S_{k,i,j}$,
- die Endzeit der Bearbeitung $C_{k,i,j}$ sowie
- die Festlegung auf eine Auftragsfamilie $\Psi_{k,i,j,f}$

derart festgelegt werden, dass das zugrunde liegende Modell nicht verletzt wird.

Durch die zeitliche Diskretisierung des Planungshorizonts soll die Berechnung dieser Entscheidungsmatrizen durch einen Lösungsgenerator erfolgen. Über die Entscheidungsmatrix $\delta_{k,i,\tau}$ werden für die jeweiligen Maschinen $M_{k,i}$ die Auftragsfamilien $F_{k,i,f}$ vorgegeben, die in den diskreten Zeitperioden t_τ gefertigt werden sollen (Formel 4.71).

$$\delta_{k,i,\tau} \in \{1, \ldots, o_{k,i}\} \qquad \forall k, i, \tau \tag{4.71}$$

Die Zeitperioden sind dazu in der Menge $\mathcal{T}$ zusammengefasst (Formel 4.72). Durch das metaheuristische Optimierungsverfahren wird entsprechend nur noch die Entscheidungsmatrix $\delta_{k,i,\tau}$ (teil-) enumeriert.

$$\mathcal{T} = \{t_1, \ldots, t_\tau, \ldots, t_\theta\} \tag{4.72}$$

Durch die Größe θ wird die Anzahl der Zeitperioden festgelegt. Der Planungshorizont H wird entsprechend Formel 4.73 in θ gleichmäßige Zeitabschnitte der Länge ΔT eingeteilt, so dass Formel 4.74 gültig ist.

$$\Delta T = \frac{H}{\theta} \tag{4.73}$$

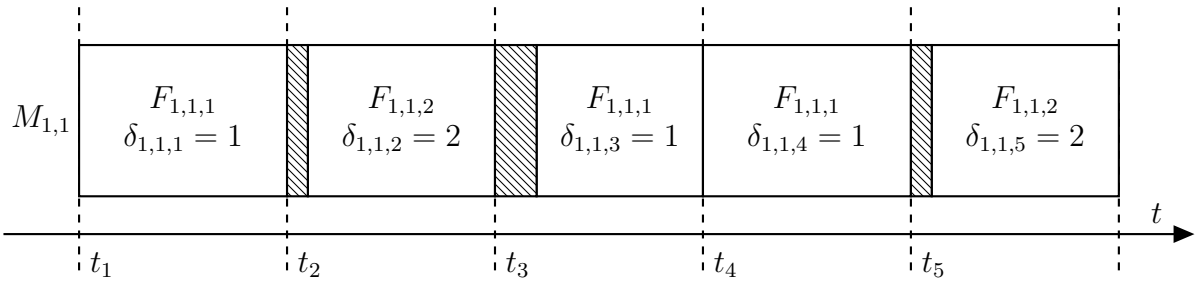


Abbildung 4.2: Diskretisierung des Planungshorizonts in gleichmäßige Planungsperioden

$$H = \sum_{\tau=1}^{\theta} \Delta T \tag{4.74}$$

Die Entscheidungsmatrix $\epsilon_{k,j,\tau}$ gibt an, ob ein Job J_j in der Produktionsstufe K_k in der Planungsperiode t_τ eingeplant ist (Formel 4.75).

$$\epsilon_{k,j,\tau} \in \{0,1\} \qquad \forall k,j,\tau \tag{4.75}$$

Die Angabe durch die Entscheidungsmatrix $X_{k,i,j}$, ob ein Job J_j in einem Ablaufplan auf einer Maschine $M_{k,i}$ gefertigt wird, muss mit den Entscheidungsmatrizen $\Psi_{k,i,j,f}$ und $\epsilon_{k,j,\tau}$ übereinstimmen. Wird ein Job auf einer entsprechenden Maschine gefertigt ($X_{k,i,j} = 1$), so muss dieser in einer entsprechenden Zeitperiode t_τ eingeplant werden ($\epsilon_{k,j,\tau} = 1$). Weiterhin muss für den Job eine entsprechende Auftragsfamilie festgelegt werden, welche wiederum mit der für die Periode t_τ festgelegten Familie übereinstimmen muss. Diese wird durch die Entscheidungsmatrix $\delta_{k,i,\tau}$ festgelegt. Die Validierung der Belegungsvariable $X_{k,i,j}$ wird dem Modell entsprechend durch Formel 4.76 hinzugefügt.

$$X_{k,i,j} = \sum_{\tau=1}^{\theta} \left(\Psi_{k,i,j,\delta_{k,i,\tau}} \cdot \epsilon_{k,j,\tau} \right) \qquad \forall k,i,j \tag{4.76}$$

Durch Formel 4.77 wird darüber hinaus sichergestellt, dass für einen Job J_j in der Produktionsstufe K_k eine Einplanung in einer Zeitperiode t_τ nur stattfinden kann, wenn die Produktionsstufe laut der Matrix $\sigma_{k,j}$ erforderlich ist. Weiterhin sorgt Formel 4.77 dafür, dass ein Job in einer Stufe nur höchstens innerhalb genau einer Zeitperiode verplant werden kann.

$$\sigma_{k,j} \geq \sum_{\tau=1}^{\theta} \epsilon_{k,j,\tau} \qquad \forall k,j \tag{4.77}$$

Die Diskretisierung des Planungshorizonts ist in Abbildung 4.2 anhand eines Beispiels dargestellt. Es handelt sich um ein Einmaschinenproblem mit der Maschine $M_{1,1}$. Die Anzahl der Entscheidungszeitpunkte wurde zu $\theta = 5$ gewählt. Die Werte der Entscheidungsmatrix $\delta_{k,i,\tau}$ sollen entsprechend von einem Optimierungsverfahren generiert und ausprobiert werden und sind für dieses Beispiel in Formel 4.78 gegeben.

$$\delta_{k,i,\tau} = \begin{pmatrix} 1 & 2 & 1 & 1 & 2 \end{pmatrix} \tag{4.78}$$

Die Maschine $M_{1,1}$ soll entsprechend für die Perioden t_1, t_3 und t_4 auf die Auftragsfamilie $F_{1,1,1}$ gerüstet werden. In den Perioden t_2 und t_5 wird die Maschine auf die Auftragsfamilie $F_{1,1,2}$ gerüstet. Der Zeitaufwand für das Umrüsten der Maschine ($\phi_{1,1,1,2}$ und $\phi_{1,1,2,1}$) wird durch die schraffierten Zeitabschnitte angedeutet.

Für jeden Job J_j, der beispielsweise in der Zeitperiode t_5 an der Maschine $M_{1,1}$ eingeplant wird, muss folglich gelten:

- Die Stufe K_1 ist für die Bearbeitung notwendig ($\sigma_{1,j} = 1$).
- Die Maschine $M_{1,1}$ ist mit dem Job kompatibel ($\lambda_{1,1,j} = 1$).
- Der Job gehört zu der Auftragsfamilie $F_{1,1,2}$ ($\mu_{1,1,j,2} = 1$).

Durch das Einplanen des Jobs auf der Maschine müssen ohne Verletzung des Modells die folgenden Variablen gesetzt werden:

- die Belegungsvariable: $X_{1,1,j} = 1$
- die Wahl der Auftragsfamilie für den Job: $\Psi_{1,1,j,2} = 1$
- die Zuordnung der Zeitperiode: $\epsilon_{1,j,5} = 1$

Ein weiteres Beispiel ist in Abbildung 4.3 mit zwei Produktionsstufen ($c = 2$) und jeweils zwei Maschinen ($m_1 = 2$, $m_2 = 2$) gegeben. Die Anzahl der Entscheidungszeitpunkte wird wieder zu $\theta = 5$ gewählt. Die zu der Abbildung 4.3 gehörende Entscheidungsmatrix ist in Formel 4.79 angegeben.

$$\delta_{k,i,\tau} = \begin{pmatrix} 1 & 2 & 1 & 1 & 2 \\ 2 & 2 & 1 & 2 & 2 \\ 3 & 2 & 3 & 1 & 1 \\ 1 & 1 & 2 & 2 & 2 \end{pmatrix} \tag{4.79}$$

4.2.3 Definition des Such- und des Lösungsraums

Ein konkretes Problem wird charakterisiert durch eine konsistente Angabe der problemspezifischen Daten. Dies sind die Anzahlen (siehe Unterabschnitt 4.1.1 c)) und die Vorgabegrößen (siehe Unterabschnitt 4.1.1 e)). Sind die Angaben konsistent, so dass das Modell nicht verletzt wird, ist das Vorhandensein zulässiger Lösungen in Form von zulässigen Ablaufplänen zu erwarten. In dem durch die Ergebnisgrößen

- $X_{k,i,j}$ (Belegung),
- Z_{k,i,j_1,j_2} (Reihenfolge),
- $\Psi_{k,i,j,f}$ (Festlegung Auftragsfamilie),
- $S_{k,j}$ (Startzeit) und

- $C_{k,j}$ (Endzeit)

aufgespannten Raum sind entsprechend alle zulässigen Lösungen des Modells enthalten. Auch alle Kombinationen, durch welche das Modell verletzt wird, sind in diesem Lösungsraum vorhanden.

Durch die zeitliche Diskretisierung des Modells und durch die Definition der Entscheidungsmatrix $\delta_{k,i,\tau}$ wird ein weiterer, wesentlich kleinerer Raum aufgespannt. Durch die Suche nach optimierten Werten für $\delta_{k,i,\tau}$ wird eine von dem Lösungsraum entkoppelte Suche in einem separaten Suchraum ermöglicht (siehe Abbildung 4.4).

Die Matrizen $S_{k,j}$ (Startzeit) und $C_{k,j}$ (Endzeit) sind für zulässige Lösungen mit reellwertigen Größen zu besetzen. Gleichzeitig sind die Matrizen $X_{k,i,j}$ (Belegung), Z_{k,i,j_1,j_2} (Reihenfolge) und $\Psi_{k,i,j,f}$ (Festlegung Auftragsfamilie) mit diskreten (binären) Werten zu besetzen. Der von diesen Matrizen gemeinsam aufgespannte Raum ist folglich ein gemischt-ganzzahliger Lösungsraum (Abbildung 4.4). Da die Entscheidungsmatrix $\delta_{k,i,\tau}$ nur mit bestimmten, positiven und ganzzahligen Werten besetzt werden kann, handelt es sich bei dem durch $\delta_{k,i,\tau}$ aufgespannten Suchraum um einen rein kombinatorischen Raum (Abbildung 4.4).

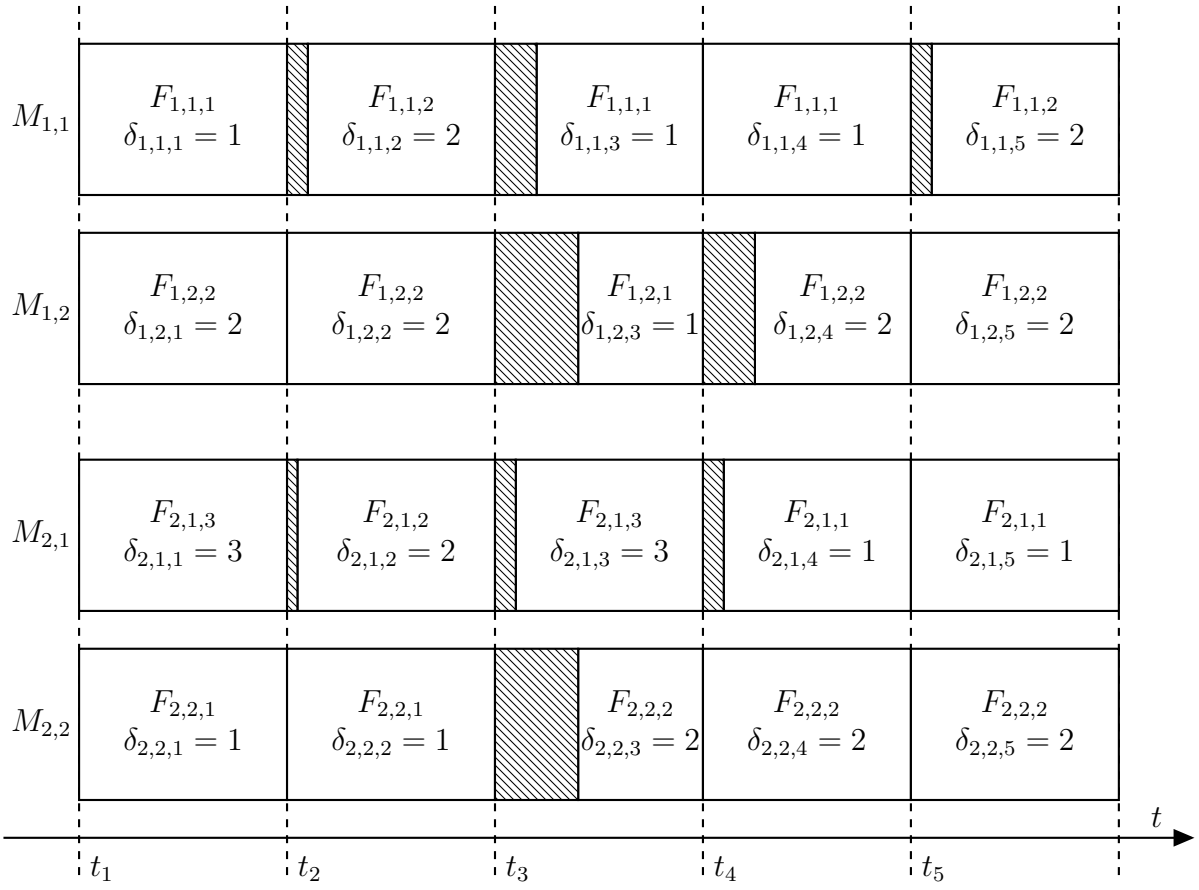


Abbildung 4.3: Aufgeteilter Planungshorizont für zwei Stufen mit jeweils zwei Maschinen

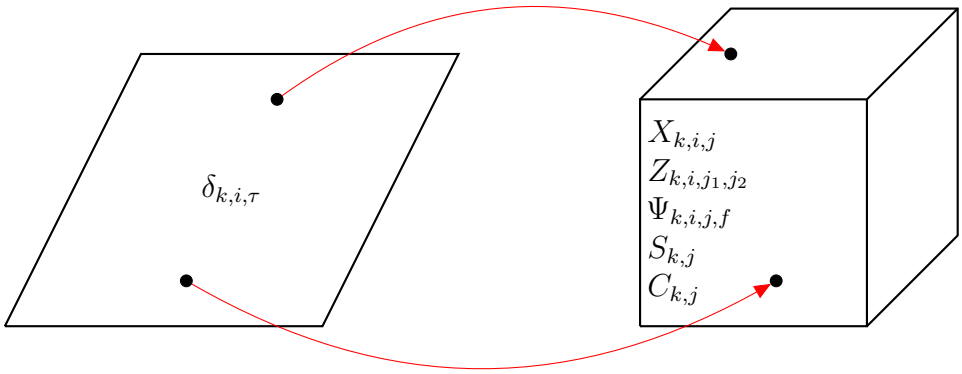


Abbildung 4.4: Entkopplung von Suchraum und Lösungsraum

Jedem Punkt in dem Suchraum ist durch die Transformation über den Lösungsgenerator ein entsprechender Punkt in dem Lösungsraum zugeordnet. Dadurch kann selbst bei einer theoretisch vollständigen Enumeration des Suchraums nur noch ein Teil des Lösungsraums abgetastet werden.

Durch $\delta_{k,i,\tau}$ wird an jeder Maschine $M_{k,i}$ jeder Produktionsstufe K_k für jede Planungsperiode t_τ eine Auftragsfamilie $F_{k,i,f}$ festgelegt. Die Größe des dadurch aufgespannten Suchraums (und damit die Komplexität κ) ergibt sich aus der Anzahl aller möglichen Kombinationsmöglichkeiten von $\delta_{k,i,\tau}$. Dementsprechend sind für die Berechnung der Komplexität κ die Anzahl der Produktionsstufen c, die Angabe der Maschinenzahl m_k jeder Stufe K_k, die Anzahl der Auftragsfamilien $o_{k,i}$ jeder Maschine $M_{k,i}$ sowie die gewählte Anzahl der Planungsperioden θ relevant (Formel 4.80).

$$\kappa = \prod_{k=1}^{c} \prod_{i=1}^{m_k} (o_{k,i})^{\theta} \tag{4.80}$$

Die Komplexität κ des Suchproblems kann explizit durch die Wahl von θ beeinflusst werden und ist unabhängig von der Anzahl der Fertigungsaufträge (vgl. Formel 2.21). Kleinere Werte von θ bedeuten, dass nur an wenigen Zeitpunkten Entscheidungen getroffen werden können, die für längere Zeitabschnitte gelten. Größere Werte von θ bedeuten hingegen, dass an mehreren Zeitpunkten Entscheidungen getroffen werden können, die für kürzere Zeitabschnitte gelten.

Analog zu der Auswahl eines Rechengitters (bzw. bei der FEM) muss durch eine den Anforderungen entsprechende Auswahl von θ die Komplexität des Suchraums mit der erzielbaren Lösungsqualität durch eine angemessene „zeitliche Auflösung“ abgewogen werden. Im Vergleich zu der in Formel 2.21 abgeschätzten Komplexität für die Enumeration des Lösungsraums zeigt Formel 4.80 für den durch das Planungsverfahren definierten Suchraum eine deutliche Reduktion der Komplexität.

Beispiel 4.1 (Zahlenbeispiele für Komplexitäten unterschiedlicher Probleme)
Die Formeln 4.81 bis 4.86 zeigen die resultierenden Komplexitätswerte für unterschiedliche Flexible Flow Shop Probleme.

$$\kappa\left(FF2\left(R2_{k=1}^{2}\right), o_{k,i}=4, \theta=4\right) = 4{,}3 \cdot 10^{9} \tag{4.81}$$
$$\kappa\left(FF2\left(R2_{k=1}^{2}\right), o_{k,i}=4, \theta=8\right) = 1{,}8 \cdot 10^{19} \tag{4.82}$$
$$\kappa\left(FF2\left(R2_{k=1}^{2}\right), o_{k,i}=6, \theta=4\right) = 2{,}8 \cdot 10^{12} \tag{4.83}$$
$$\kappa\left(FF2\left(R2_{k=1}^{2}\right), o_{k,i}=6, \theta=8\right) = 8{,}0 \cdot 10^{24} \tag{4.84}$$
$$\kappa\left(FF2\left(R4_{k=1}^{2}\right), o_{k,i}=4, \theta=4\right) = 1{,}8 \cdot 10^{19} \tag{4.85}$$
$$\kappa\left(FF2\left(R4_{k=1}^{2}\right), o_{k,i}=4, \theta=8\right) = 3{,}4 \cdot 10^{38} \tag{4.86}$$

Allen Problemen des Beispiels liegt eine Anzahl von $c = 2$ Produktionsstufen zugrunde. Bei den Formeln 4.81 bis 4.84 handelt es sich um jeweils $m_k = 2$ Maschinen je Stufe, bei den Formeln 4.85 und 4.86 handelt es sich um jeweils $m_k = 4$ Maschinen je Stufe. Die Formeln 4.81, 4.82, 4.85 und 4.86 berechnen die Komplexität auf Basis von $o_{k,i} = 4$ Auftragsfamilien je Maschine, die Formeln 4.83 und 4.84 berechnen die Komplexität auf Basis von $o_{k,i} = 6$ Auftragsfamilien je Maschine. Die Anzahl der Planungsperioden wird mit $\theta = 4$ (Formeln 4.81, 4.83 und 4.85) und mit $\theta = 8$ (Formeln 4.82, 4.84 und 4.86) variiert.

4.3 Formulierung eines ereignisdiskreten Lösungsgenerators

Zulässige Ablaufpläne werden indirekt durch entsprechende Ausprägungen der Entscheidungsmatrix $\delta_{k,i,\tau}$ repräsentiert. Das Muster, nach welchem ein Ablaufplan von einer konkreten Ausprägung von $\delta_{k,i,\tau}$ erzeugt wird, ist der Hauptbestandteil des Lösungsgenerators (Abbildung 4.5). Durch den Ablauf einer ereignisdiskreten Simulation wird über den Generator auf Basis der Entscheidungsmatrix $\delta_{k,i,\tau}$ ein entsprechender Ablaufplan ($S_{k,j}$, $C_{k,j}$, $X_{k,i,j}$, Z_{k,i,j_1,j_2}, $\Psi_{k,i,j,f}$) erzeugt.

4.3.1 Nomenklatur

t_{Start} . . . . . Startzeit des Planungsverfahrens bzw. des Ablaufplans
t_{End} . . . . . Endzeit des Planungsverfahrens bzw. des Ablaufplans
$t_{k,i}$ . . . . . Aktueller Zeitschritt der Maschine $M_{k,i}$
$s_{k,i}$ . . . . . Auftragsfamilie, auf welche die Maschine $M_{k,i}$ gerüstet ist
$g_{k,j}$ . . . . . Befindet sich der Job J_j im Puffer der Stufe k?
t_{Leerlauf} . . . . . Zeitkonstante, die für die Einplanung von Leerlauf notwendig ist
$A_{k,j}$ . . . . . Ankunftszeit (arrival time) von Job J_j in Stufe K_k
$Y_{k,i,j}$ . . . . . Absolute Reihenfolge von Job J_j auf Maschine $M_{k,i}$
$Y_{k,i}$ . . . . . Absolute Reihenfolge auf der Maschine $M_{k,i}$
$Z_{k,i}$ . . . . . Index j des letzten Jobs J_j auf Maschine $M_{k,i}$

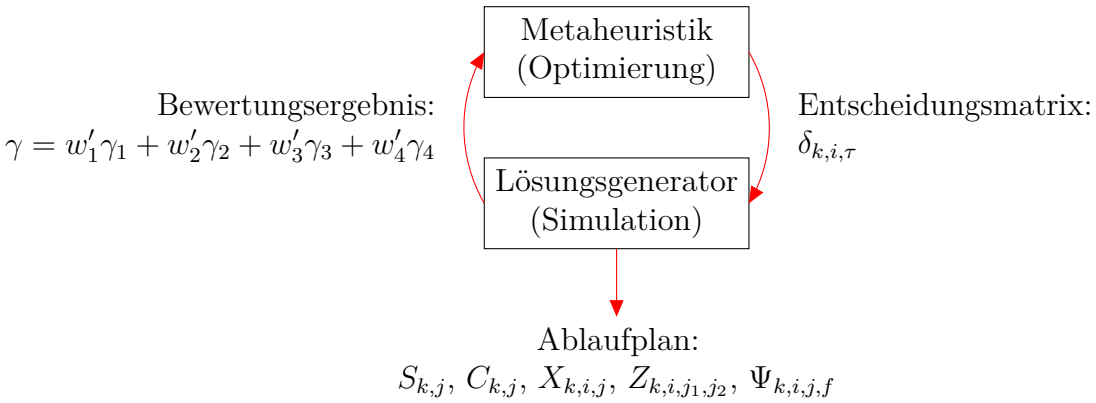


Abbildung 4.5: Kopplung zwischen Metaheuristik und Lösungsgenerator

4.3.2 Verfahrensablauf

Der Ablauf des gesamten ereignisdiskreten Planungsverfahrens ist in Algorithmus 4.1 gegeben. Das Vorhandensein von konsistenten Problemdaten wird an dieser Stelle für die Ausführbarkeit des Verfahrens vorausgesetzt.

Algorithmus 4.1 Hauptprozedur des Planungsverfahrens

```
1: prozedur AblaufplanBerechnen(δ_{k,i,τ})
2:     t := t_Start
3:     solange t < t_End führe durch
4:         M_{k,i} := MaschineErmitteln
5:         ZeitschrittRechnen(M_{k,i})
6:     ende solange
7: ende prozedur
```

Über die Prozedur AblaufplanBerechnen wird auf Basis der problemspezifischen Daten ein Ablaufplan für ein konkretes Flexible Flow Shop Problem berechnet. Der Prozedur wird dafür die Matrix $\delta_{k,i,\tau}$ durch das entsprechende Optimierungsverfahren übergeben.

Der Ablaufplan wird für das Zeitintervall zwischen den Zeitpunkten t_{Start} und t_{End} generiert (Abbildung 4.6). Der Startzeitpunkt des Ablaufplans muss zusammen mit den problemspezifischen Daten vorgegeben werden. Für das zeitliche Ende des Ablaufplans gilt dann Formel 4.87 (siehe Abbildung 4.6).

$$t_{\text{End}} = t_{\text{Start}} + H \tag{4.87}$$

(4.88)

Der exakte Startzeitpunkt t_τ einer Planungsperiode τ (bzw. ein konkreter Entscheidungszeitpunkt) lässt sich gemäß Formel 4.89 berechnen. Die umgekehrte Rechnung – also

die Angabe der Planungsperiode τ zu einem gegebenen Zeitpunkt t – ist Formel 4.90 zu entnehmen.

$$t_\tau = t_{\text{Start}} + \Delta T \cdot (\tau - 1) \tag{4.89}$$

$$\tau = \frac{t_\tau - t_{\text{Start}}}{\Delta T} + 1 \tag{4.90}$$

Zu Beginn der Prozedur wird die Zeitvariable t des Verfahrens auf den Startzeitpunkt t_{Start} initialisiert (Algorithmus 4.1, Zeile 2). Solange sich die Planung innerhalb des Zeitfensters des Ablaufplans befindet ($t < t_{\text{End}}$, Zeile 3), wird in einer Schleife an den einzelnen Maschinen $M_{k,i}$ für jeden Zeitschritt t ein Planungsschritt durchgeführt. Über die Prozedur MaschineErmitteln (Zeile 4) wird in jedem Schleifendurchlauf genau die Maschine $M_{k,i}$ ermittelt, an welcher das nächste Ereignis auftritt.

Durch die Prozedur ZeitschrittRechnen (Zeile 5) wird an genau dieser Maschine der aktuelle Zeitschritt gerechnet. Die Inkrementierung der Zeitvariablen t findet dabei in der Prozedur ZeitschrittRechnen statt. Die definierte Terminierung der Schleife in der Hauptprozedur wird dadurch sichergestellt.

Die Ermittlung der Maschine, welche als nächstes gerechnet werden muss, erfolgt über die maschinenspezifischen Ereigniszeitpunkte $t_{k,i}$. Abbildung 4.7 veranschaulicht dies an einem Beispiel mit den drei Maschinen $M_{1,1}$, $M_{1,2}$ und $M_{2,1}$. Für jede Maschine $M_{k,i}$ wird dafür während der Planung in der Variablen $t_{k,i}$ der Zeitpunkt gespeichert, an dem an der Maschine das nächste Ereignis eintreten wird. An der Maschine mit dem nächsten Ereigniszeitpunkt muss als nächstes mit der Planung fortgefahren werden. In dem Beispiel aus Abbildung 4.7 ist dies die Maschine $M_{1,2}$ mit dem Zeitpunkt $t_{1,2}$.

Die Berechnung des nächsten Zeitschritts an der entsprechenden Maschine $M_{k,i}$ ist in Algorithmus 4.2 gegeben. Zunächst wird die globale Zeitvariable t auf den aktuellen Ereigniszeitpunkt $t_{k,i}$ der Maschine $M_{k,i}$ gesetzt (Algorithmus 4.2, Zeile 2). Dadurch wird in der Hauptprozedur die Terminierung der Schleife sichergestellt (siehe Algorithmus 4.1, Zeile 3).

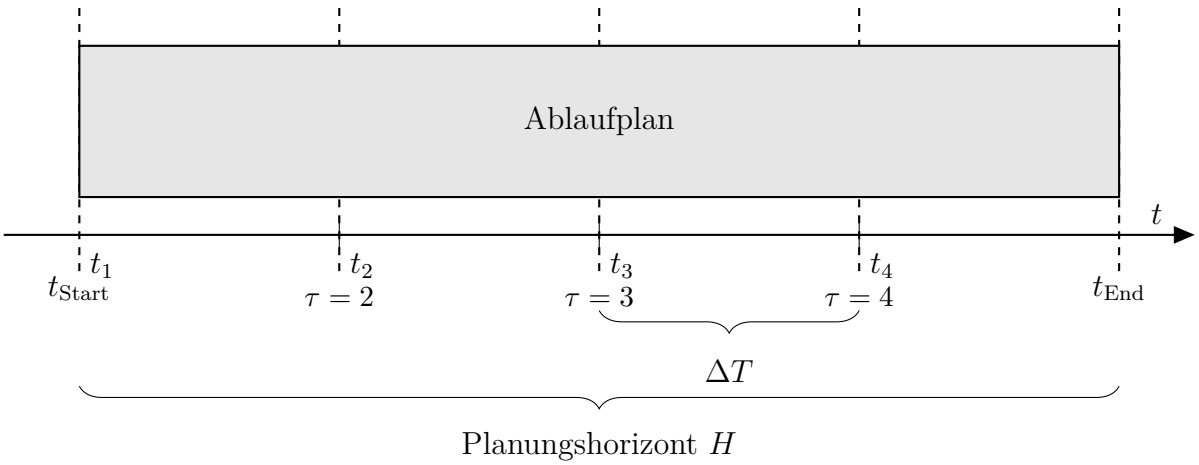


Abbildung 4.6: Ablaufplanung mit dem Planungshorizont H und $\theta = 4$ Planungsperioden

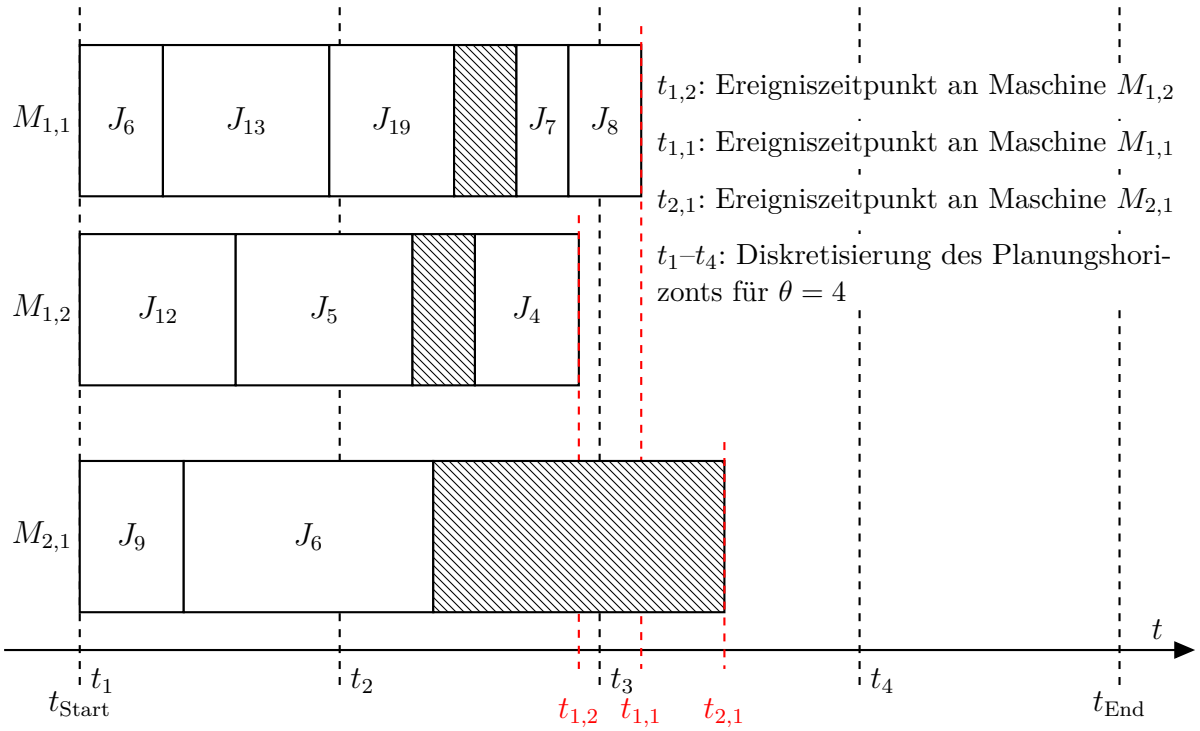


Abbildung 4.7: Ereigniszeitpunkte von drei Maschinen während der Planung

Als nächstes muss die Auftragsfamilie f bestimmt werden, welche an der Maschine eingeplant werden soll. Dafür wird entsprechend zunächst die Zeitperiode t_τ bestimmt, in welcher sich die Planung an der Maschine aktuell befindet (Zeile 3). Über die Entscheidungsmatrix $\delta_{k,i,\tau}$ wird schließlich die entsprechende Auftragsfamilie f ermittelt (Zeile 4).

Die an der Maschine im letzten Zeitschritt eingeplante Auftragsfamilie ist in der Variablen $s_{k,i}$ gespeichert (Zeile 5). Über die Matrix ϕ_{k,i,f_1,f_2} wird die entsprechende Rüstzeit ϕ ermittelt, die durch das Umplanen von der Auftragsfamilie $F_{k,i,s}$ auf die Auftragsfamilie $F_{k,i,f}$ zu erwarten ist (Zeile 6). Durch das Umrüsten der Maschine auf die Auftragsfamilie $F_{k,i,f}$ muss die Variable $s_{k,i}$ entsprechend auf die neue Auftragsfamilie aktualisiert werden (Zeile 7).

Der nächste Ereigniszeitpunkt $t_{k,i}$ der Maschine $M_{k,i}$ berechnet sich aus dem aktuellen Zeitschritt, der um die resultierende Rüstzeit erhöht wird (Zeile 8). Die eigentliche Einplanung des nächsten Fertigungsauftrags erfolgt schließlich durch den Aufruf der Prozedur PlanungslogikAnwenden (Zeile 9).

Der Ablauf der Prozedur PlanungslogikAnwenden ist Algorithmus 4.3 zu entnehmen. Es soll ein Liste L erstellt werden, die nur solche Jobs enthält, welche alle notwendigen Bedingungen erfüllen, um an der Maschine $M_{k,i}$ eingeplant werden zu können (Algorithmus 4.3, Zeile 2).

Algorithmus 4.2 Berechnung eines Zeitschritts an einer Maschine

1: **prozedur** ZeitschrittRechnen($M_{k,i}$)
2: $t := t_{k,i}$
3: $\tau := \dfrac{t - t_{\text{Start}}}{\Delta T} + 1$
4: $f := \delta_{k,i,\tau}$
5: $s := s_{k,i}$
6: $\phi := \phi_{k,i,s,f}$
7: $s_{k,i} := f$
8: $t_{k,i} := t_{k,i} + \phi$
9: PlanungslogikAnwenden($M_{k,i}$)
10: **ende prozedur**

Jeder Job J_j wird dafür in einer Schleife auf die entsprechenden Voraussetzungen geprüft (Zeile 3). Das Statement **prüfe** ist dabei als eine Kurzform einer **wenn**–**dann**–Abfrage zu betrachten. Wenn das zu prüfende Argument wahr ist, wird mit der Schleife fortgefahren. Ist das Argument allerdings falsch, so wird die Schleife an der aktuellen Stelle unterbrochen und mit dem nächsten Schleifendurchlauf fortgefahren. Trifft mindestens eine der notwendigen Bedingungen nicht auf einen Job J_j zu, so findet eine Prüfung der weiteren Bedingungen nicht mehr statt. Stattdessen wird mit der Prüfung des Jobs J_{j+1} fortgefahren.

Jeder Job J_j muss für die Bearbeitung in der Produktionsstufe K_k vorgesehen sein (Zeile 4) und die Maschine $M_{k,i}$ muss für die Bearbeitung des Jobs kompatibel sein (Zeile 5). Desweiteren muss der Job J_j mit der Auftragsfamilie f kompatibel sein (Zeile 6).

Der Job J_j muss sich in dem Puffer der Stufe K_k befinden (Zeile 7). Weiterhin müssen die Bestände aller Materialien die jeweiligen Bedarfe des Jobs decken (Zeilen 8 und 9).

Nur wenn in der nachgelagerten Produktionsstufe genügend Lagerkapazität zur Verfügung steht, kann der Job J_j eingeplant werden. Die nachgelagerten Produktionsstufen $K_{k'}$ werden entsprechend in aufsteigender Reihenfolge (Zeile 11) geprüft, ob sie für die Bearbeitung des Jobs J_j notwendig sind. Sobald die erste (und damit die relevante) Produktionsstufe ermittelt ist (Zeile 12), wird überprüft, ob die Pufferkapazität $\rho_{k'}$ der Stufe $K_{k'}$ den Lagerbedarf $\eta_{k',j}$ des Auftrags J_j entsprechend deckt (Zeile 13). Dafür müssen sämtliche Jobs $J_{j'}$, die sich zu diesem Planungszeitpunkt in dem Puffer der Stufe $K_{k'}$ befinden ($g_{k',j'} = 1$), miteinbezogen werden (Zeile 13).

Kann der Puffer der Stufe diesen Job nicht mehr aufnehmen, wird die übergeordnete Schleife der Prozedur PlanungslogikAnwenden entsprechend an dieser Stelle unterbrochen, so dass die Prüfung des nächsten Jobs erfolgen kann. Ist keine weitere Produktionsstufe für die Bearbeitung des Jobs notwendig, wird die Abarbeitung der Schleife entsprechend fortgeführt.

Es folgt die Überprüfung, ob der Job J_j in der Produktionsstufe K_k zu einem Auftragsstapel gehört. Dafür wird für jeden Batch $B_{k,b}$ geprüft (Zeile 16), ob der Job Teil des Batches ist (Zeile 17). Falls dies für einen Batch der Fall ist ($\nu_{k,j,b} = 1$, Zeile 17), wird durch die

Prozedur BatchPrüfen (Algorithmus 4.4) der gesamte Auftragsstapel überprüft. Nur wenn der gesamte Batch $B_{k,b}$ alle notwendigen Bedingungen für die Einplanung aller ihm zugehörigen Jobs erfüllt, kann der Batch eingeplant werden. Die einzelne Einplanung von Jobs, die zu einem Auftragsstapel gehören, ist ohne die Einplanung des gesamten Batches (und damit aller Jobs des Batches) nicht erlaubt.

Algorithmus 4.3 Anwendung der Planungslogik an einer Maschine

1: **prozedur** PlanungslogikAnwenden($M_{k,i}$,f,τ)
2: Erstelle eine Liste L mit Jobs, die die folgenden Bedingungen erfüllen.
3: **für** $j := 1$ **bis** $j = n$ **führe durch** ▷ *Prüfe die Bedingungen für jeden Job J_j.*
4: **prüfe** $\sigma_{k,j} = 1$ ▷ *Die Stufe der Maschine muss für den Job notwendig sein.*
5: **prüfe** $\lambda_{k,i,j} = 1$ ▷ *Der Job muss kompatibel mit dieser Maschine sein.*
6: **prüfe** $\mu_{k,i,j,f} = 1$ ▷ *Der Job muss kompatibel mit der Familie sein.*
7: **prüfe** $g_{k,i} = 1$ ▷ *Der Job muss sich in dem Puffer dieser Stufe befinden.*
8: **für** $r := 1$ **bis** $r = l$ **führe durch**
9: **prüfe** $\omega_r \geq \pi_{k,j,r}$ ▷ *Das benötigte Material muss verfügbar sein.*
10: **ende für**
11: **für** $k' := k + 1$ **bis** $k' = c$ **führe durch** ▷ *In der nächsten Stufe muss Platz sein.*
12: **wenn** $\sigma_{k',j} = 1$ **dann**
13: **prüfe** $\rho_{k'} \geq \eta_{k',j} + \sum_{j'=1}^{n} (g_{k',j'} \cdot \eta_{k',j'})$
14: **ende wenn**
15: **ende für**
16: **für** $b := 1$ **bis** $b = q_k$ **führe durch** ▷ *Gehört der Job zu einem Stapel?*
17: **wenn** $\nu_{k,j,b} = 1$ **dann**
18: **prüfe** BatchPrüfen($M_{k,i}, B_b$) ▷ *Bedingungen müssen für Batch gelten.*
19: **ende wenn**
20: **ende für**
21: Füge den Job J_j der Liste L hinzu
22: **ende für**
23: **wenn** Liste L nicht leer **dann**
24: Liste L Sortieren
25: $J_j :=$ Erster Job aus Liste L
26: JobEinplanen($M_{k,i}, J_j$)
27: **sonst**
28: $t_{k,i} := t_{k,i} + t_{\text{Leerlauf}}$ ▷ *Leerlauf einplanen.*
29: **ende wenn**
30: **ende prozedur**

Erfüllt ein Job J_j sämtliche Bedingungen, so wird er der Liste L hinzugefügt (Zeile 21). Sobald die Schleife für die Überprüfung der Jobs terminiert, muss zunächst geprüft werden, ob die Liste L leer ist (Zeile 23). Wenn die Liste L nicht leer ist, sind die in

der Liste gespeicherten Jobs durch eine geeignet Heuristik zu sortieren (Zeile 24). Die Erläuterung der verwendeten Regeln folgt in Unterabschnitt 4.3.3.

Aus der absteigend sortierten Liste wird der erste (der beste) Job J_j ausgewählt (Zeile 25) und an der Maschine $M_{k,i}$ mit der Auftragsfamilie f in der Planungsperiode τ eingeplant. Dafür wird die Prozedur JobEinplanen (Algorithmus 4.5) aufgerufen (Zeile 26).

Befindet sich kein Job in der Liste L (Zeile 27), kann an der Maschine $M_{k,i}$ in dem aktuellen Zeitschritt t kein Fertigungsauftrag eingeplant werden. Es erfolgt daher keine zeitliche Veränderung des nächsten Ereignisses an der Maschine $M_{k,i}$ durch die Fertigstellung eines Jobs. Der nächste Ereigniszeitpunkt des Verfahrens bleibt folglich der Ereigniszeitpunkt der Maschine $M_{k,i}$. Damit die Hauptprozedur des Planungsverfahrens in diesem Fall nicht in einer Endlosschleife verharrt (Algorithmus 4.1, Zeile 4), muss an der Maschine $M_{k,i}$ eine Leerlaufzeit t_{Leerlauf} eingeplant werden (Zeile 28). Dadurch wird die Fortschreibung der Zeitvariablen t auch für die Fälle gesichert, in denen kein Fertigungsauftrag eingeplant wird.

In Abbildung 4.8 ist dies an einem Beispiel mit den Maschinen $M_{1,1}$ und $M_{2,1}$ verdeutlicht. Die Maschine $M_{2,1}$ produziert (nach einer Rüstzeit) zunächst die beiden Jobs J_3 und J_{13}, die zuvor in der vorgelagerten Produktionsstufe an $M_{1,1}$ bearbeitet wurden. Nach dem Zeitpunkt t_2 wird die Maschine $M_{2,1}$ umgerüstet. Es sind nun keine weiteren Fertigungsaufträge für die Einplanung an $M_{2,1}$ verfügbar. Erst nachdem der Job J_6 an der Maschine $M_{1,1}$ bearbeitet wurde, können an $M_{2,1}$ wieder weitere Jobs eingeplant werden.

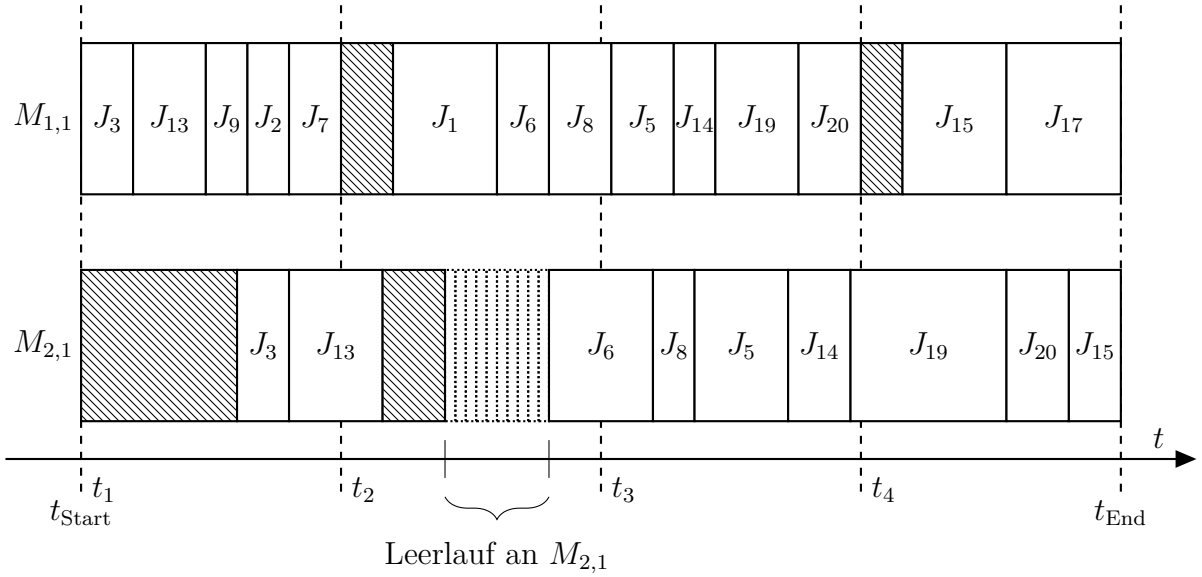


Abbildung 4.8: Ablaufplan mit Leerlauf an einer Maschine

Durch das Planungsverfahren werden zu diesem Zweck an $M_{2,1}$ solange Leerlaufzeiten eingeplant, bis wieder Jobs zur Verfügung stehen. Ohne die Einplanung der Leerlaufzeiten würde das Planungsverfahren in jedem Iterationsschritt erneut die Planungslogik für

$M_{2,1}$ aufrufen, da immer nach dem zeitlich nächsten Ereignis gesucht wird. Das nächste Ereignis an $M_{1,1}$ (das Ende des Jobs J_1) wäre allerdings immer später als das nächste Ereignis an $M_{2,1}$ (das Ende der zweiten Rüstvorgangs). Dies würde zu einer Endlosschleife des Verfahrens führen.

Algorithmus 4.4 zeigt die Prozedur BatchPrüfen, welche in der Prozedur PlanungslogikAnwenden aufgerufen wird (Algorithmus 4.3, Zeile 18). Gehört ein Fertigungsauftrag in der Stufe K_k zu einem Batch $B_{k,b}$, so kann der Job an der Maschine $M_{k,i}$ nur zusammen mit dem gesamten Batch eingeplant werden. Nur wenn für den gesamten Auftragsstapel die entsprechenden Bedingungen erfüllt sind, können die Jobs des Stapels eingeplant werden.

Algorithmus 4.4 Prüfung eines Auftragsstapels

1: **prozedur** BatchPrüfen($M_{k,i}, B_b$)
2: **prüfe** $\sum_{j}^{n} (\nu_{k,j,b} \cdot g_{k,i}) = n_b$ ▷ *Die Jobs müssen sich im Puffer dieser Stufe befinden.*
3: **für** $r := 1$ **bis** $r = l$ **führe durch**
4: **prüfe** $\omega_r \geq \sum_{j}^{n} (\nu_{k,j,b} \cdot \pi_{k,j,r})$ ▷ *Das benötigte Material muss verfügbar sein.*
5: **ende für**
6: **für** $k' := k + 1$ **bis** $k' = c$ **führe durch** ▷ *In der nächsten Stufe muss Platz sein.*
7: **prüfe** $\rho_{k'} \geq \sum_{j=1}^{n} (\nu_{k,j,b} \cdot \sigma_{k',j} \cdot \eta_{k',j}) + \sum_{j'=1}^{n} (g_{k',j'} \cdot \eta_{k',j'})$
8: **ende für**
9: **ende prozedur**

Zunächst wird für jeden Job J_j des Stapels geprüft, ob er sich in dem Puffer der Stufe K_k befindet (Algorithmus 4.4, Zeile 2). Es folgt die Prüfung der Materialverfügbarkeit. (Zeile 3) Die Summe der einzelnen Materialbedarfe muss dabei durch die entsprechenden Materialbestände gedeckt werden (Zeile 4).

Abschließend wird geprüft, ob für jeden einzelnen Job des Batches genügend Speicherplatz im Puffer der nächsten Stufe vorhanden ist (Zeile 6). Für jede nachgelagerte Stufe $K_{k'}$ wird daher geprüft, ob die Aufnahme der weiteren Jobs des Stapels durch die Kapazität $\rho_{k'}$ gedeckt wird, falls die Stufe für die Bearbeitung der jeweiligen Jobs relevant ist (Zeile 7). Dafür werden die zu dem aktuellen Planungszeitpunkt $t_{k,i}$ bereits in dem Puffer gespeicherten Jobs $J_{j'}$ berücksichtigt ($g_{k',j'} \cdot \eta_{k',j'}$).

Die konkrete Einplanung eines Jobs J_j auf der Maschine $M_{k,i}$ mit der Auftragsfamilie f in der Planungsperiode τ wird durch die Prozedur JobEinplanen (Algorithmus 4.5) vorgenommen. Dafür wird zunächst geprüft, ob der Job J_j zu einem Auftragsstapel gehört (Algorithmus 4.5, Zeile 2). Ist dies der Fall, wird der entsprechende Auftragsstapel $B_{k,b}$ gesucht (Zeile 3) und an der Maschine über den Aufruf der Przozedur BatchEinplanen eingeplant (Zeile 5). Die aktuelle Prozedur JobEinplanen wird in diesem Fall durch den

Abbruch der Suchschleife verlassen (Zeile 6), da die Einplanung des Jobs J_j in diesem Fall bereits über die Prozedur BatchEinplanen stattfindet.

Gehört der Job J_j nicht zu einem Auftragsstapel (Zeile 9) wird dieser als einzelner Auftrag an der Maschine $M_{k,i}$ eingeplant. Dafür werden zunächst die Materialien abgebucht. Der Bestand ω_r jedes Materials R_r (Zeile 10) wird entsprechend des Bedarfs $\pi_{k,j,r}$ reduziert (Zeile 11).

Algorithmus 4.5 Einplanung eines Fertigungsauftrags

```
1: prozedur JobEinplanen(M_{k,i}, J_j, f, τ)
2:     wenn Σ_{b=1}^{q_k} ν_{k,j,b} = 1 dann          ▷ Gehört der Job zu einem Stapel?
3:         für b := 1 bis b = q_k führe durch          ▷ Stapel suchen.
4:             wenn ν_{k,j,b} = 1 dann
5:                 StapelEinplanen(M_{k,i}, B_{k,b})   ▷ Gesamten Stapel einplanen.
6:                 schleife abbrechen                  ▷ Schleife abbrechen und Prozedur verlassen.
7:             ende wenn
8:         ende für
9:     sonst                                           ▷ Nur diesen Job einplanen.
10:        für r := 1 bis r = l führe durch
11:            ω_r := ω_r − π_{k,j,r}                  ▷ Material abbuchen.
12:        ende für
13:        g_{k,j} := 0                                ▷ Job aus Puffer der Stufe entfernen.
14:        S_{k,j} := t_{k,i}                          ▷ Startzeit setzen.
15:        t_{k,i} := t_{k,i} + p_{k,i,j}              ▷ Zeit aktualisieren.
16:        C_{k,j} := t_{k,i}                          ▷ Endzeit setzen.
17:        ϵ_{k,j,τ} := 1                              ▷ Belegungsvariable für die Zeitperiode setzen
18:        X_{k,i,j} := 1                              ▷ Belegungsvariable für die Maschine setzen.
19:        Ψ_{k,i,j,f} := 1                            ▷ Belegungsvariable für die Familie setzen.
20:        Y_{k,i,j} := Y_{k,i}                        ▷ Reihenfolgevariable setzen.
21:        Y_{k,i} := Y_{k,i} + 1                      ▷ Absolute Reihenfolge der Maschine erhöhen.
22:        j_1 := Z_{k,i}                              ▷ Vorherigen Job merken.
23:        Z_{k,i,j_1,j} := 1                          ▷ Reihenfolgevariable setzen.
24:        Z_{k,i} := j                                ▷ Diesen Job als letzten an dieser Maschine merken.
25:        für k' := k + 1 bis k' = c führe durch      ▷ Job transportieren.
26:            wenn σ_{k',j} = 1 dann
27:                g_{k',j} := 1                       ▷ Position setzen.
28:                A_{k',j} := t_{k,i}                 ▷ Ankunftszeit setzen.
29:                schleife abbrechen
30:            ende wenn
31:        ende für
32:    ende wenn
33: ende prozedur
```

Die Bearbeitung des Jobs J_j in der Produktionsstufe K_k kann beginnen, so dass der Job aus dem Puffer der Stufe entfernt wird (Zeile 13). Die Startzeit $S_{k,j}$ der Bearbeitung des Jobs J_j in der Stufe K_k wird auf den aktuellen Zeitpunkt an der Maschine $M_{k,i}$ gesetzt (Zeile 14). Der Planungszeitpunkt an der Maschine muss anschließend um die Produktionsdauer $p_{k,i,j}$ des Jobs J_j an der Maschine $M_{k,i}$ erhöht werden (Zeile 15). Die Endzeit $C_{k,j}$ der Bearbeitung des Jobs J_j in der Stufe K_k wird analog der Startzeit auf den aktuellen Zeitpunkt an der Maschine $M_{k,i}$ gesetzt (Zeile 16).

Es folgt das Setzen der weiteren Entscheidungsmatrizen (Zeilen 17 bis 24). Die Belegungsvariable $\epsilon_{k,j,\tau}$ wird gesetzt, da der Job J_j für die Bearbeitung in der Stufe K_k in der Zeitperiode t_τ eingeplant wird (Zeile 17). Die Variable $X_{k,i,j}$ muss gesetzt werden, weil der Job J_j an der Maschine $M_{k,i}$ bearbeitet wird (Zeile 18). Schließlich wird die Variable $\Psi_{k,i,j,f}$ gesetzt, da die Maschine $M_{k,i}$ für die Bearbeitung des Jobs J_j auf die Auftragsfamilie $F_{k,i,f}$ gerüstet ist (Zeile 19).

Algorithmus 4.6 Einplanung eines Auftragsstapels

```
1:  prozedur BatchEinplanen(M_{k,i}, B_{k,b}, f, τ)
2:      für j := 1 bis j = n führe durch
3:          wenn ν_{k,j,b} = 1 dann
4:              für r := 1 bis r = l führe durch
5:                  ω_r := ω_r − π_{k,j,r}                ▷ Material abbuchen.
6:              ende für
7:              g_{k,j} := 0                              ▷ Job aus Puffer der Stufe entfernen.
8:              S_{k,j} := t_{k,i}                        ▷ Startzeit setzen.
9:              t_{k,i} := t_{k,i} + p_{k,i,j}            ▷ Zeit aktualisieren.
10:             C_{k,j} := t_{k,i}                        ▷ Endzeit setzen.
11:             ε_{k,j,τ} := 1                            ▷ Belegungsvariable für die Zeitperiode setzen
12:             X_{k,i,j} := 1                            ▷ Belegungsvariable für die Maschine setzen.
13:             Ψ_{k,i,j,f} := 1                          ▷ Belegungsvariable für die Familie setzen.
14:             Y_{k,i,j} := Y_{k,i}                      ▷ Reihenfolgevariable setzen.
15:             Y_{k,i} := Y_{k,i} + 1                    ▷ Absolute Reihenfolge der Maschine erhöhen.
16:             j_1 := Z_{k,i}                            ▷ Vorherigen Job merken.
17:             Z_{k,i,j_1,j} := 1                        ▷ Reihenfolgevariable setzen.
18:             Z_{k,i} := j                              ▷ Diesen Job als letzten an dieser Maschine merken.
19:             für k' := k + 1 bis k' = c führe durch    ▷ Job transportieren.
20:                 wenn σ_{k',j} = 1 dann
21:                     g_{k',j} := 1                     ▷ Position setzen.
22:                     A_{k',j} := t_{k,i}               ▷ Ankunftszeit setzen.
23:                     schleife abbrechen und mit nächstem Job fortfahren
24:                 ende wenn
25:             ende für
26:         ende wenn
27:     ende für
28: ende prozedur
```

Die absolute Reihenfolge $Y_{k,i,j}$, in welcher der Job J_j auf der Maschine $M_{k,i}$ eingeplant wird, wird durch die absolute Folge $Y_{k,i}$ für die Maschine gespeichert (Zeile 21). Diese muss anschließend um eins erhöht werden (Zeile 20). Für die Festlegung der relativen Reihenfolge durch die Variable $Z_{k,i,j_1,j}$ muss zunächst durch die Variable $Z_{k,i}$ der Job J_{j_1} ermittelt werden, welcher an der Maschine $M_{k,i}$ vor J_j bearbeitet wurde (Zeile 22). Die Reihenfolgevariable $Z_{k,i,j_1,j}$ kann dadurch gesetzt werden, so dass der Job J_{j_1} an der Maschine $M_{k,i}$ der direkte Vorgänger von J_j ist (Zeile 23). Anschließend muss in der Variablen $Z_{k,i}$ der Job J_j als an der Maschine $M_{k,i}$ zuletzt bearbeiteter Job gespeichert werden (Zeile 24).

Falls der Job J_j durch eine nachgelagerte Produktionsstufe $K_{k'}$ weiterbearbeitet werden muss (Zeilen 25 und 26), wird der Job in den Puffer der nächsten benötigten Bearbeitungsstufe transportiert (Zeile 27) und die Ankunftszeit wird gesetzt (Zeile 28). Sobald die erste notwendige Stufe gefunden ist, kann die Prozedur verlassen werden (Zeile 29).

Der Ablauf der Prozedur BatchEinplanen (Aufruf in Algorithmus 4.5, Zeile 5) ist in Algorithmus 4.6 dargestellt. Für die Einplanung des Batches $B_{k,b}$ an der Maschine $M_{k,i}$ mit der Auftragsfamilie f in der Planungsperiode τ wird zunächst in einer Schleife für jeden Job J_j geprüft, ob dieser zu dem Batch gehört (Algorithmus 4.6, Zeile 2). Falls dies der Fall ist (Zeile 3), wird der entsprechende Job eingeplant. Die für die konkrete Einplanung durchgeführten Schritte (Zeilen 4 bis 27) entsprechen den jeweiligen Schritten der Prozedur JobEinplanen (Algorithmus 4.5, Zeilen 10 bis 31).

4.3.3 Einbindung der Planungsregeln

Die Prozedur PlanungslogikAnwenden (Algorithmus 4.3) erstellt in jedem Iterationsschritt des Verfahrens für die entsprechende Maschine eine Liste L mit Jobs, die zu dem aktuellen Planungszeitpunkt $t_{k,i}$ eingeplant werden können. Die Liste wird anschließend nach einem entsprechenden Kriterium sortiert (Zeile 24). Favorisierte Jobs stehen dadurch am Anfang der Liste. Die in dem Planungsverfahren implementierten Heuristiken für die Sortierung beruhen im Wesentlichen auf den in Unterabschnitt 2.4.4 c) vorgestellten Regeln.

Für die Ausführung des entsprechenden Sortierverfahrens an der Maschine $M_{k,i}$ im Zeitschritt t soll für jeden Job J_j der Liste L eine Größe $\zeta(k,j,t)$ bestimmt werden. Die Liste der Jobs wird anschließend anhand dieser Größe in absteigender Reihenfolge sortiert, so dass Jobs J_j mit höheren Werten von $\zeta(k,j,t)$ an den Anfang der Liste und Jobs J_j mit niedrigeren Werten von $\zeta(k,j,t)$ an das Ende der Liste sortiert werden. Der Job J_j mit dem größten $\zeta(k,j,t)$-Wert wird schließlich von dem Planungsverfahren ausgewählt, da er sich nach dem Sortieren am Anfang der Liste befindet.

Als Grundlage für die Berechnung der Sortierreihenfolge muss dabei für einige der Planungsregeln die noch verbleibende Produktionszeit $p'(k,j)$ für die Fertigstellung des

Jobs in den noch folgenden Produktionsstufen ermittelt werden. Die Berechnung ist in Formel 4.91 gegeben.

$$p'(k,j) = \sum_{k'=k}^{c} \left(\sigma_{k',j} \cdot \frac{\sum_{i'=1}^{m_{k'}} (\lambda_{k',i',j} \cdot p_{k',i',j})}{\sum_{i'=1}^{m_{k'}} \lambda_{k',i',j}} \right) \tag{4.91}$$

Es muss dafür für alle weiteren Produktionsstufe $K_{k'}$, die nach der Produktionsstufe K_k für die Bearbeitung des Jobs J_j relevant sind ($\sigma_{k',j} = 1$), die zu erwartende Produktionszeit ermittelt werden. Dafür wird für jede mögliche Maschine $M_{k',i'}$, die mit dem Auftrag kompatibel ist ($\lambda_{k',i',j} = 1$), die Bearbeitungszeit $p_{k',i',j}$ ermittelt. Da für den Job J_j zu diesem Planungszeitpunkt noch nicht feststeht, welche Maschine $M_{k',i'}$ den Job bearbeiten wird, werden alle möglichen Bearbeitungszeiten der Maschinen durch aufsummieren und Teilen durch die Anzahl der Maschinen gemittelt.

Bei der Berechnung der $\zeta(k,j,t)$-Werte werden für jede Planungsregel die individuellen Gewichtungen w_j der Jobs berücksichtigt. Jobs mit einer höheren Gewichtung werden dadurch tendenziell bevorzugt ausgewählt. Die Festlegung der einzelnen Gewichtungen w_j kann beispielsweise durch einen Eingriff des Planers erfolgen. Dieser kann dadurch einzelne Jobs bevorzugen (priorisieren) $w_j > 1$ oder zurückstellen $w_j < 1$. Für eine neutrale Gewichtung ist $w_j = 1$ zu wählen.

Die implementierten Planungsregeln (Unterabschnitt 2.4.4 c)) sowie die entsprechenden Berechnungen der $\zeta(k,j,t)$-Werte sind die folgenden:

EDD (frühster Termin zuerst): Die Jobs sollen gemäß ihres geplanten Liefertermins sortiert werden. Jobs mit kleinerem Termin d_j sind entsprechend dringender als Jobs mit größerem Termin. $\zeta(k,j,t)$ ist entsprechend umgekehrt proportional zu dem jeweiligen Liefertermin d_j zu berechnen (Formel 4.92).

$$\zeta(k,j,t) \sim \frac{w_j}{d_j} \tag{4.92}$$

S/OPN (geringste Schlupfzeit zuerst): Die Jobs werden anhand der jeweils noch verbleibenden Schlupfzeit sortiert, so dass Jobs mit der geringsten Schlupfzeit in der Liste nach oben sortiert werden. Die Schlupfzeit eines Jobs ist ein Maß für die Dringlichkeit, diesen Job einzuplanen. Die Schlupfzeit orientiert sich wie die EDD-Regel an den jeweiligen Terminen d_j. Zusätzlich wird die zu erwartende Zeit $p'(k,j)$ einbezogen, die benötigt wird, bis der Job J_j in allen weiteren Produktionsstufen fertiggestellt ist (Formel 4.93). Die Jobs werden dadurch anhand der Zeitspanne sortiert, die noch als Reserve zur Verfügung steht, um noch eine pünktliche Lieferung zu garantieren.

$$\zeta(k,j,t) \sim \frac{w_j}{d_j - t - p'(k,j)} \tag{4.93}$$

SPT (kürzeste Operationszeit zuerst): Der Job mit der geringsten zu erwartenden Bearbeitungszeit in allen weiteren noch relevanten Produktionsstufen soll als nächstes eingeplant werden (Formel 4.94).

$$\zeta\left(k,j,t\right) \sim \frac{w_j}{p'\left(k,j\right)} \tag{4.94}$$

LPT (längste Operationszeit zuerst): Der Job mit der längsten zu erwartenden Bearbeitungszeit in allen weiteren noch relevanten Produktionsstufen soll als nächstes eingeplant werden (Formel 4.95).

$$\zeta\left(k,j,t\right) \sim w_j \cdot p'\left(k,j\right) \tag{4.95}$$

FIFO (früheste Ankunftszeit zuerst): Der als erstes im Puffer der Stufe K_k eingetroffene Job J_j soll als nächstes eingeplant werden (Formel 4.96). Durch diese Regel lassen sich Logistikpuffer als reine Warteschlangen abbilden, die (z.B. technologisch bedingt) keine Sortierung der Jobs zulassen.

$$\zeta\left(k,j,t\right) \sim \frac{w_j}{A_{k,j}} \tag{4.96}$$

LIFO (späteste Ankunftszeit zuerst): Der als letztes im Puffer der Stufe K_k eingetroffene Job J_j soll als nächstes eingeplant werden (Formel 4.97). Diese Regel lässt die Abbildung von Stapelsystemen zu, in welchen die Werkstücke der Jobs in einem Speicher aufeinandergestapelt werden. In dem Fall wird immer der oberste Job ausgewählt.

$$\zeta\left(k,j,t\right) \sim w_j \cdot A_{k,j} \tag{4.97}$$

5 Verwendete Optimierungsverfahren

Die Komplexität des in Kapitel 4 aufgespannten Suchraums ist wesentlich geringer als die Komplexität des gesamten Lösungsraums (siehe Unterabschnitt 4.2.3). Für eine vollständige Enumeration des Suchraums reicht die Reduktion der Komplexität allerdings nicht aus, da das verbleibende kombinatorische Optimierungsproblem noch immer $\mathcal{NP}$-schwer ist (siehe Abschnitt 2.3 und Unterabschnitt 4.2.3). In diesem Kapitel werden daher für die kombinatorische Optimierung zwei verschiedene Metaheuristiken als Such- bzw. Optimierungsverfahren entwickelt.

Bei den in Unterabschnitt 2.4.2 b) vorgestellten Metaheuristiken handelt es sich nach aktuellem Stand der Forschung um die populärsten Verfahren (siehe z.B. Burke et al., 2014b, Du et al., 2016). Jede der vorgestellten Metaheuristiken hat ihre eigenen Vor- und Nachteile. Keine Metaheuristik kann daher grundsätzlich ausgeschlossen oder bevorzugt werden.

Aufgrund der zahlreichen erfolgreichen Anwendungen und Forschungsergebnisse in den unterschiedlichsten Disziplinen (siehe z.B. Burke et al., 2014b, Dorigo et al., 2004, Du et al., 2016, Gerdes et al., 2004) fällt die Wahl an dieser Stelle auf die Entwicklung eines Genetischen Algorithmus und eines Ameisenalgorithmus. In Voruntersuchungen (siehe beispielsweise Stalinski et al., 2018b, 2019b) konnte für diese Verfahren (im Gegensatz zu beispielsweise dem Simulierten Abkühlen, welches sich im Rahmen von Voruntersuchungen nicht bewährte) ein für die Problemstellung geeignetes Lösungsverhalten bereits nachgewiesen werden. Für beide Varianten werden jeweils zunächst die Grundlagen und Begrifflichkeiten eingeführt, um die Verfahren im Anschluss jeweils mit dem ereignisdiskreten Lösungsgenerator zu koppeln.

Beide Optimierungsverfahren gehören zu den metaheuristischen Suchverfahren und arbeiten stochastisch. Beide Verfahren sind durch Beobachtungen und Theorien aus der Natur bzw. der Biologie inspiriert und entstammen daher im weitesten Sinne der Bionik. Eine weitere Gemeinsamkeit der beiden Verfahren besteht in der populationsbasierten Optimierung. So werden in einem Iterationsprozess anstelle einer einzigen Lösung immer jeweils mehrere Lösungen in einem Iterationsschritt erzeugt und bewertet. Die Parallelisierung der Verfahren (die simultane Nutzung mehrerer CPU-Kerne) kann entsprechend dadurch erreicht werden, dass jeweils einem CPU-Kern die Berechnung eines Teils der Population zugewiesen wird.

Neben diesen Gemeinsamkeiten besteht zwischen den beiden Verfahren ein wesentlicher Unterschied. Genetische Algorithmen arbeiten verändernd, so dass zu Beginn des

Verfahrens einmalig (zufällig) zulässige Lösungen erzeugt und im Laufe des Optimierungsprozesses verändert werden. Ameisenalgorithmen arbeiten erzeugend, so dass während des Optimierungsprozesses permanent neue Lösungen generiert werden.

Es handelt sich bei beiden Verfahren um grundsätzlich problemunabhängige Optimierungsstrategien. In der Formulierung der Entscheidungsmatrix $\delta_{k,i,\tau}$ besteht eine entsprechend definierte Schnittstelle für die Anwendung. Beide Verfahren bieten allerdings umfangreiche Möglichkeiten der Parametrierung, deren Optimum wiederum abhängig von den konkreten Problemen sein kann.

5.1 Genetischer Algorithmus

Genetische Algorithmen gehören zusammen mit den Evolutionsstrategien, der Evolutionären Programmierung und der Genetischen Programmierung zu den Evolutionären Algorithmen (Weicker, 2015, S. 127). Sie basieren auf den Prinzipien der Genetik und der natürlichen Auslese durch die Evolution (Sastry et al., 2014, S. 93).

Durch die natürliche Evolution werden Lebewesen innerhalb relativ kurzer Zeit an veränderte Umgebungen angepasst. Lösungen für komplexe Probleme werden dabei durch die Prüfung nur weniger Möglichkeiten erschaffen (Gerdes et al., 2004, S. 33).

Die Anfänge der Evolutionären Strategien gehen u.a. auf die (voneinander unabhängigen) Arbeiten von Ingo Rechenberg und Hans Bremermann in den 1960er Jahren zurück (Nissen, 1997, S. 9). Die Entwicklung und Anwendung von Genetischen Algorithmen werden den Arbeiten von Holland (1975) und Goldberg (1989) zugeschrieben (Nissen, 1997, S. 33).

5.1.1 Inspiration durch die natürliche Evolution

Die Evolutionstheorie beschreibt die natürliche Entwicklung und Weiterentwicklung von Lebewesen und Arten. Über lange Zeiten und Epochen unterliegen die Arten natürlichen Veränderungen. Dabei wird auf unbewusste Weise eine Art Optimierungsprozess vorgenommen.

In dem Erbgut eines Lebewesens sind alle notwendigen Informationen für die Ausprägung seiner Eigenschaften in Form eines „Bauplans“ gespeichert. Bei der Vermehrung werden durch die Paarung zweier Lebewesen die Gene der Eltern kombiniert und bei der Entstehung eines neuen Individuums vereinigt. Üblicherweise unterliegt das Erbgut des Kindes dabei zusätzlich natürlichen Störungen, so dass die Ausprägung einzelner Gene durch Genmutationen zufällig verändert wird.

Über mehrere Generationen hinweg greift dabei das Prinzip „survival of the fittest“ – also die natürliche Selektion durch das Überleben der stärksten Individuen. Lebewesen, die sich durch die entsprechende Ausprägung ihres Erbgutes besser an ihren Lebensraum angepasst haben, haben dementsprechend höhere Überlebenschancen. Die Wahrscheinlichkeit, dass

sich diese stärkeren Lebewesen paaren und ihr Erbgut dadurch in die nächste Generation weitergeben, ist entsprechend höher.

Auf diese Weise liegen dem Evolutionsprinzip die zwei Effekte zugrunde. Einerseits werden neuen Lebewesen durch die mutationsbedingten und dadurch zufälligen Änderungen des Erbgutes auf natürliche Weise Eigenschaften und deren Kombination zugeschrieben. Auf der anderen Seite werden die Erbgüter, welche sich durch die natürliche Auslese als stark herausgestellt haben, weitervererbt und kombiniert.

Die für den weiteren Verlauf dieser Arbeit relevanten Begrifflichkeiten sowie deren Bedeutungen sind Gerdes et al. (2004, S. 34) und Du et al. (2016, S. 39–43) entnommen.

Individuum: eine konkrete Lösung.

Chromosom: Ausprägung der Gene eines Individuums. Durch ein Chromosom werden die Ausprägungen der einzelnen Gene eines konkreten Individuums festgelegt.

Gen: Stelle eines Chromosoms. Ein Chromosom ist ein Strang bestehend aus einer definierten Anzahl von Genen. Ein Gen ist in der Regel jeweils für die Ausprägung einer bestimmten Eigenschaft verantwortlich.

Allel: Konkrete Ausprägung eines Gens. Ein Gen kann nur bestimmte Werte – die Allele – annehmen. Ein Allel kann in diesem Zusammenhang als die kleinstmögliche Informationseinheit angesehen werden.

Kodierung: Überführung eines konkreten Individuums (einer Lösung) in das zugehörige Chromosom.

Dekodierung: Überführung eines Chromosoms zu einem konkreten Individuum (einer Lösung).

Fitness: Die Fitness eines Individuums ist ein Maß für die Qualität der entsprechenden Lösung.

Population: Die in einem Iterationsschritt gerechnete Menge an Individuen.

Abbildung 5.1 zeigt an einem Beispiel den schematischen Aufbau von einem Chromosoms eines Genetischen Algorithmus. Das dargestellte Chromosom besteht aus den sechs Genen und kodiert den Wert $2-1-3-4-2-1$. Die ersten beiden Gene können die Werte 1 und 2 annehmen, das dritte und vierte Gen kann jeweils die ganzzahligen Werte 1 bis 6 annehmen und das fünfte und sechste Gen kann jeweils die Werte 1, 2, 3 und 4 annehmen.

5.1.2 Ablauf des Verfahrens

Der Ablauf des Genetischen Basisalgorithmus ist in Algorithmus 5.1 in Anlehnung an Kramer (2017, S. 12–17) dargestellt. Die Ausführung der einzelnen Schritte des Algorithmus entstammt Sastry et al. (2014, S. 94) und Kramer (2017, S. 12–17).

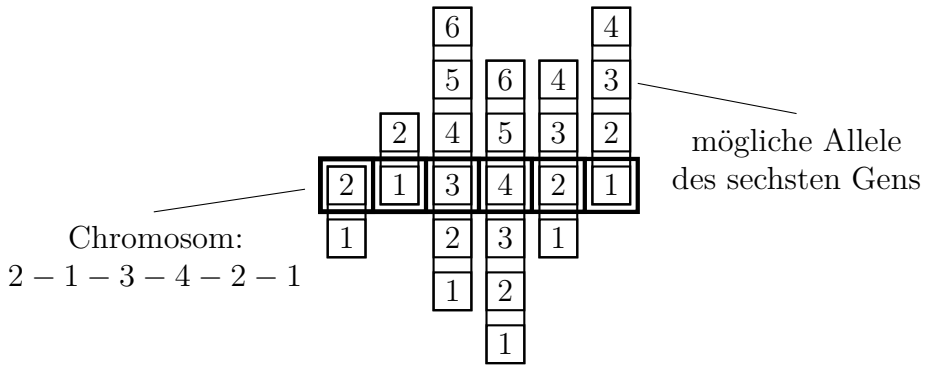


Abbildung 5.1: Beispiel eines Chromosoms mit sechs Genen und möglichen Allelen

Initialisierung: Bevor der iterative Evolutionsprozess durchgeführt wird, muss eine initiale Population erzeugt werden. Die Population sollte aus möglichst unterschiedlichen Individuen bestehen, so dass ein ausreichendes Maß an Diversität gegeben ist. Dadurch soll verhindert werden, dass der Algorithmus bereits am Anfang der Suche zu stark konvergiert und sichergestellt werden, dass die Suche möglichst breit gestartet wird (die Punkte im Suchraum möglichst weit auseinander liegen). Durch die zufallsbasierte Erzeugung der Startpopulation soll eine gleichmäße Verteilung im Suchraum erreicht werden (Kramer, 2017, S. 12). Anstelle der randomisierten Erzeugung kann die Population auch durch ein Eröffnungsverfahren erzeugt werden (Kramer, 2017, S. 12).

Evaluation: Jedes Individuum der Population wird entsprechend der zugrunde gelegten Fitnessfunktion bewertet. Dabei wird die Zeichenkette des entsprechenden Chromosoms zunächst dekodiert und die zugehörige Lösung erzeugt. Anschließend wird die Qualität der Lösung bewertet und dem Individuum als Fitnesswert zugeordnet.

Selektion: Aus der Population werden Individuen basierend auf ihrer Fitness ausgewählt. Dem Prinzip des Überleben der Stärksten entsprechend werden dabei Individuen mit einem höheren Fitnesswert bevorzugt. Die selektierten Individuen werden in einem Paarungspool gesammelt. Ein Individuum kann sich dabei auch mehrmals in dieser Sammlung befinden, wodurch sich die Wahrscheinlichkeit erhöht, dass es am Paarungsprozess teilnehmen kann.

Rekombination: Aus dem Paarungspool werden wiederholt randomisiert je zwei Individuen als Eltern ausgewählt. Auf Basis der Rekombinationswahrscheinlichkeit p_C werden diese Individuen zufallsbasiert entweder rekombiniert oder nicht. Im Falle der Rekombination werden die Chromosomen der beiden Eltern gekreuzt, so dass zwei Kinder entstehen, deren Chromosomen die Gene der beiden Eltern enthalten. Andernfalls entsprechen die Chromosomen der beiden Nachkommen exakt denen der beiden Eltern. Dieser Vorgang wird wiederholt, bis eine der Population entsprechende Anzahl von Nachkommen erzeugt ist.

Mutation: Die Chromosomen der Nachkommen werden zufällig verändert. Für jedes Gen der Nachkommen wird zufallsbasiert entsprechend der Mutationswahrscheinlichkeit entschieden, ob eine Mutation stattfindet. Ist dies der Fall, wird das Allel des entsprechenden Gens auf einen zufälligen Wert geändert.

Ersetzung: Die vorherige Generation wird entweder ganz oder teilweise durch die neue Generation der Nachkommen ersetzt.

Abbruchkriterium: Der Vorgang wird solange wiederholt bis ein entsprechendes Abbruchkriterium erfüllt ist. Dies kann das Erreichen einer zeitlichen Grenze oder einer bestimmten Anzahl von Iterationen sein. Auch die Definition eines Konvergenzkriteriums ist möglich, so dass das Verfahren abgebrochen wird, falls keine weitere Verbesserung der höchsten Fitness mehr festgestellt wird.

Algorithmus 5.1 Ablauf des Genetischen Basisalgorithmus

```
prozedur Genetischer Basisalgorithmus
    Initialisierung
    wiederhole
        Selektion
        Rekombination
        Mutation
        Ersetzung
        Evaluation
    bis Abbruchkriterium
ende prozedur
```

5.1.3 Genetische Operatoren

Dem Verfahren liegen die evolutionstheoretischen Konzepte der Selektion, der Rekombination und der Mutation zugrunde. Bei der Anwendung von Genetischen Algorithmen für die mathematische bzw. die kombinatorische Optimierung können diese Prinzipien jeweils unterschiedlich interpretiert und umgesetzt werden.

In der Literatur sind zahlreiche Ansätze und Variationen für die Umsetzung der einzelnen Prinzipien zu finden. Auf der einen Seite sind diese teilweise stark zugeschnitten auf spezielle Problemfälle. Andererseits handelt sich bei den Umsetzungen teilweise um Varianten, welche sich nur minimal voneinander unterscheiden.

Teilweise unterscheiden sich einzelne Implementierungen des Genetischen Algorithmus nur durch die Reihenfolge von Selektion, Mutation, Rekombination, Evaluation und Ersetzung. Aus diesem Grund werden im Folgenden nur die für diese Arbeit relevanten Operatoren behandelt.

a) Selektionsoperatoren

Die Selektion besteht bei den Genetischen Algorithmen aus den beiden separaten Teilschritten

1. Selektionsalgorithmus und
2. Auswahlalgorithmus. (Nissen, 1997, S. 64)

Zunächst wird in dem Selektionsalgorithmus für jedes Individuum I_i die Auswahlwahrscheinlichkeit $p_\mathrm{S}(i)$ berechnet. Unabhängig von der verwendeten Methode, die zur Berechnung der Auswahlwahrscheinlichkeiten gewählt wird, muss dabei stets Formel 5.1 erfüllt sein.

$$\sum_{i}^{N} p_S(i) = 1 \tag{5.1}$$

Darüber lässt sich der entsprechende Erwartungswert $E(i)$ berechnen (siehe Formel 5.2).

$$E(i) = N \cdot p_\mathrm{S}(i) \tag{5.2}$$

Dies ist ein Maß für die durch die Selektion zu erwartende Anzahl von Nachkommen dieses Individuums.

Fitnessproportionale Bewertung: Bei dieser Basisvariante ist die Selektionswahrscheinlichkeit für ein Individuum I_i direkt proportional zu der Fitness $F(i)$ des Individuums (Kramer, 2017, S. 17). Die Berechnung ist in Formel 5.3 aufgeführt (siehe Du et al., 2016, S. 44).

$$p_\mathrm{S}(i) = \frac{F(i)}{\sum_j^N F(j)} \tag{5.3}$$

Der proportionale Zusammenhang zwischen der Fitness und der Auswahlwahrscheinlichkeit eines Individuums ist zunächst der einfachste und naheliegenste Ansatz. Allerdings kann dieser Operator zu vorzeitiger Konvergenz des Suchverfahrens führen (Du et al., 2016, S. 44).

Rangbasierte Bewertung: Um einer zu schnellen Konvergenz gegen lokale Optima entgegenzuwirken, wird für jedes Individuum I_i zunächst der Rang $r(i)$ berechnet. Dafür werden die Lösungen ihrer Fitness entsprechend absteigend sortiert, so dass das Individuum mit der größten Fitness den Rang $r(i) = N$ und das Individuum mit der geringsten Fitness den Rang $r(i) = 1$ erhält. Die Selektionswahrscheinlichkeit für ein Individuum I_i ist in Formel 5.4 aufgeführt (siehe Gerdes et al., 2004, S. 83).

$$p_\mathrm{S}(i) = \frac{1}{N} \cdot \left(E_\mathrm{min} + (E_\mathrm{max} - E_\mathrm{min}) \cdot \frac{r(i) - 1}{N - 1} \right) \tag{5.4}$$

$$E_\mathrm{max} \in [1; 2] \tag{5.5}$$

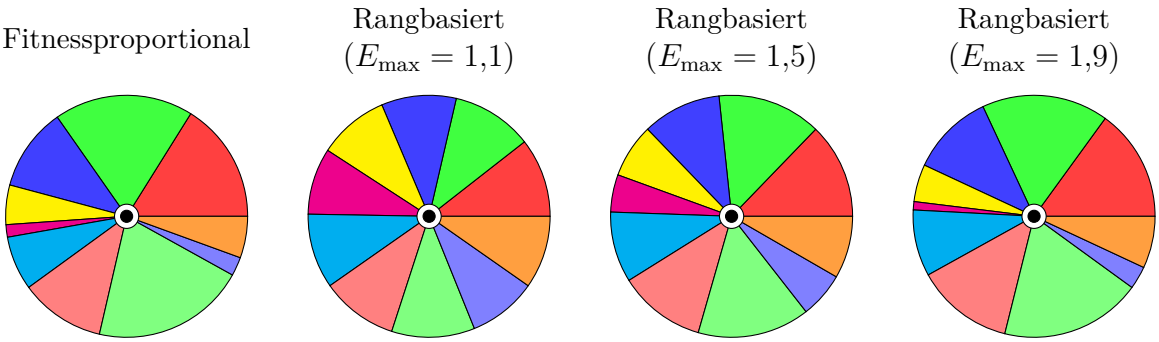


Abbildung 5.2: Fitnessbasierte und rangbasierte Selektionswahrscheinlichkeiten

$$E_{\min} = 2 - E_{\max} \tag{5.6}$$

Über den Parameter $E_{\max}$ wird der Selektionsdruck der rangbasierten Bewertung gesteuert. Je größer $E_{\max}$ gewählt wird, desto höher ist der Selektionsdruck.

In Abbildung 5.2 sind die resultierenden Wahrscheinlichkeitswerte für die beiden Verfahren anhand eines Beispiels aufgeführt. Es handelt sich dabei um zufällig erzeugte Fitnesswerte für zehn Individuen, die jeweils mit einer entsprechenden Farbe markiert sind. Proportional zu der jeweiligen Auswahlwahrscheinlichkeit auf Basis des jeweiligen Selektionsalgorithmus wird jedem Individuum ein Segment auf einem Rad zugewiesen. Beiden Verfahren liegen in diesem Beispiel die gleichen Fitnesswerte zugrunde. Für die rangbasierten Bewertung werden die Einflüsse der unterschiedlichen Werte von $E_{\max}$ auf die Auswahlwahrscheinlichkeiten dargestellt ($E_{\max} = 1{,}1$: geringerer Selektionsdruck, $E_{\max} = 1{,}9$: höherer Selektionsdruck).

Basierend auf den berechneten Auswahlwahrscheinlichkeiten wird mithilfe des Auswahlalgorithmus die stochastische Selektion durchführt. Dafür stehen die folgenden Varianten zur Verfügung:

Rouletteselektion: Als Analogie für dieses Verfahren dient häufig die Vorstellung eines Rouletterades bzw. eines Glücksrades mit einem feststehenden Zeiger (siehe Abbildung 5.3). Jedem Individuum wird entsprechend seiner Auswahlwahrscheinlichkeit ein Segment auf dem Rad zugewiesen (siehe Abbildung 5.2). Das Glücksrad wird nun N Mal hintereinander auf einen zufälligen Winkel verdreht, so dass der feststehende Zeiger auf ein beliebiges Segment des Rades zeigt. Das Individuum, welches zu dem entsprechenden Segment gehört, wird jeweils selektiert.

Stochastic Universal Sampling (SUS): Auch diesem Algorithmus liegt die Vorstellung eines Glücksrades zugrunde. Anstelle eines einzelnen, feststehenden Zeigers werden allerdings N Zeiger gleichmäßig um das Rad herum verteilt (siehe Abbildung 5.3). Das Rad wird nun nur einmal um einen zufälligen Winkel verdreht. Jedes Individuum wird anschließend entsprechend der Anzahl der Zeiger, die in dessen Segment zeigen, selektiert.

Tournamentselektion: Dieser Auswahlalgorithmus ist gleichzeitig ein Selektionsalgorithmus. Hierbei werden zwischen den Individuen Turniere (Wettkämpfe) abgehalten, bei denen der Gewinner des jeweiligen Wettkampfes selektiert wird. Dazu muss im Vorfeld durch den Parameter n_T die Turniergröße festgelegt werden. Es werden zufällig Individuen (unabhängig von ihrer Fitness) aus der Population für das Turnier ausgewählt, bis die Turniergröße erreicht ist. Das Individuum mit der größten Fitness gewinnt den Wettkampf und wird selektiert. Dieser Vorgang wird N mal wiederholt.

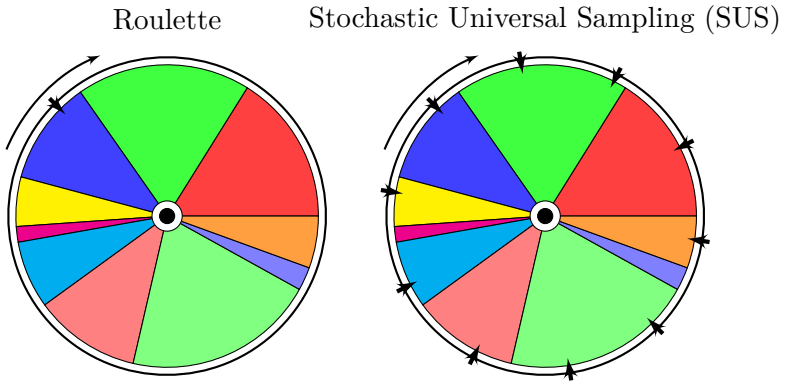


Abbildung 5.3: Selektion mittels Rouletteverfahren und Stochastic Universal Sampling

b) Rekombinationsoperatoren

Für die Kreuzung der beiden selektierten Elternchromosomen stehen die beiden Basisoperatoren N-Point-Crossover und Uniform-Crossover zur Verfügung.

N-Point-Crossover: Über den Parameter n_C wird dem Verfahren eine Anzahl von Kreuzungspunkten vorgegeben. Für jedes selektierte Elternpaar werden die Punkte zufällig auf den Chromosomen verteilt. An diesen Punkten werden die Chromosomen entsprechend geteilt und abschnittsweise vertauscht. In Abbildung 5.4 ist das Verfahren mit zwei Punkten dargestellt.

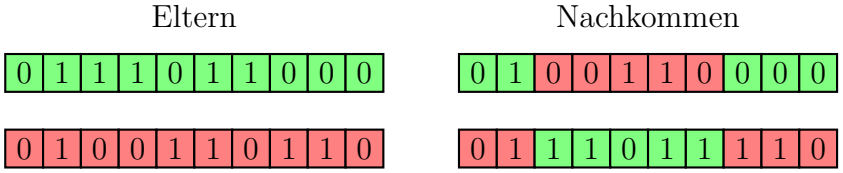


Abbildung 5.4: Beispiel eines N-Point-Crossover-Operators mit zwei Punkten

Uniform-Crossover: Diesem Verfahren wird über den Parameter p_U eine genbezogene Wahrscheinlichkeit für die Kreuzung vorgegeben. Die Gene der beiden selektierten

Elternchromosomen werden von vorne bis hinten durchlaufen. Für jedes Gen wird zufällig anhand der vorgegeben Wahrscheinlichkeit entschieden, ob die Allele der beiden Eltern vertauscht werden (siehe Abbildung 5.5).

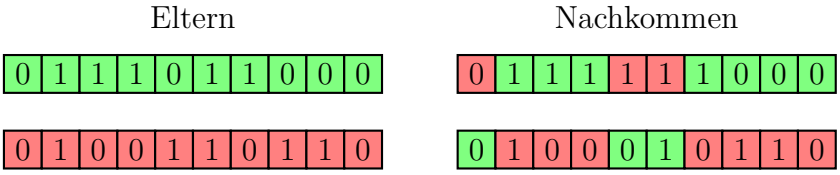


Abbildung 5.5: Beispiel eines Uniform-Crossover-Operators

c) Mutationsoperator

Die Standardform des Mutationsoperators ist die Inversion (bei binärer Lösungsrepräsentation) bzw. die zufällige Veränderung der einzelnen Gene auf zulässige Werte (siehe Abbildung 5.6). Über den Parameter p_{M} wird dem Operator dafür die Wahrscheinlichkeit vorgegeben, mit welcher die einzelnen Gene mutieren. Für jedes Gen eines Individuums wird dabei durch das Verfahren probabilistisch entschieden, ob eine Mutation stattfindet.

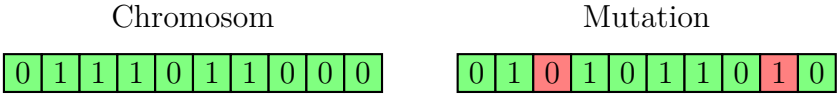


Abbildung 5.6: Beispiel eines Mutationsoperators

d) Ersetzungsschemen

Nach dem Anwenden der genetischen Operatoren auf die Population muss entschieden werden, wie die erzeugten Nachkommen der nächsten Generation die vorherige Generation ersetzen.

Vollständige Ersetzung: Die gesamte Population wird ausgelöscht und durch die neue Generation ersetzt.

Elitismus: Aus der vorherigen Population wird eine vorgegebene Anzahl n_{E} der besten Individuen herausgefiltert. Diese werden als Eliten in die neue Generation übernommen.

5.1.4 Kopplung mit dem Planungsverfahren

Die Anwendung des Genetischen Algorithmus auf die in dieser Arbeit behandelte Klasse von Flexible Flow Shop Problemen beruht wesentlich auf der Repräsentation der Entscheidungsmatrix $\delta_{k,i,\tau}$ in Form eines entsprechenden Schemas. Abbildung 5.7 zeigt den Aufbau des gewählten Kodierungsschemas.

Das entsprechende Chromosom besteht aus jeweils einem Abschnitt für jede Produktionsstufe K_k. Jeder dieser Abschnitte besteht wiederum aus je einem Unterabschnitt für jede Maschine $M_{k,i}$ der entsprechenden Produktionsstufe K_k. Für jeden Entscheidungszeitpunkt t_τ ist auf jedem Unterabschnitt ein Gen vorgesehen. Auf diese Weise wird die gesamte Matrix $\delta_{k,i,\tau}$ in Form eines Gen-Strangs abgebildet.

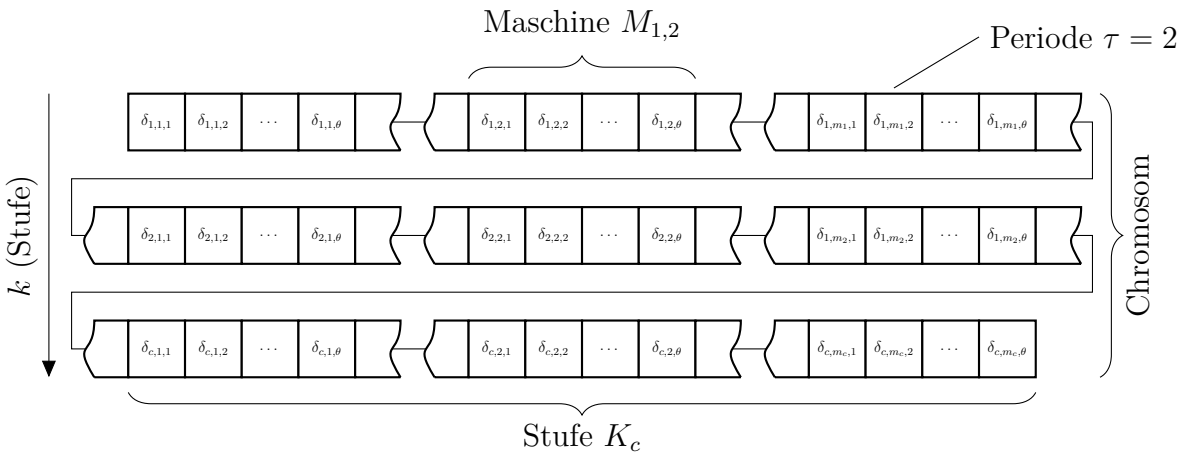


Abbildung 5.7: Kodierungsschema des Genetischen Algorithmus

Die Länge L des resultierenden Codes ist entsprechend abhängig von der Anzahl der Produktionsstufen c, der Maschinenanzahl m_k der einzelnen Stufen sowie von der gewählten Anzahl der Entscheidungszeitpunkte θ (Formel 5.7).

$$L\left(c, m_k, \theta\right) = \theta \cdot \sum_{k}^{c} m_k \tag{5.7}$$

Der Wert einer beliebigen Stelle x in dem Kodierungsschema entspricht dem zugehörigen Wert der Matrix $\delta_{k,i,\tau}$. Für einen beliebigen Koeffizienten der Matrix lässt sich die zugehörige Stelle x in dem Gen-Code gemäß Formel 5.8 berechnen.

$$x\left(\delta_{k,i,\tau}\right) = x\left(k, i, \tau\right) = \theta \cdot \left(i + \sum_{k'}^{k} m_{k'} \right) + \tau \tag{5.8}$$

In Tabelle 5.1 sind alle Parameter des Genetischen Algorithmus zusammengefasst.

Beispiel 5.1 (Zweistufiges Flexible Flow Shop Problem mit insgesamt vier Maschinen) Abbildung 5.8 zeigt für ein gültiges Chromosom, welches dem in dieser Arbeit entwickelten Kodierungsschema entspricht. Es handelt sich bei dem Beispiel um ein Flexible Flow Shop Problem mit zwei Produktionsstufen ($c = 2$) mit jeweils zwei Maschinen ($m_1 = m_2 = 2$). Die Anzahl der Entscheidungszeitpunkte ist in diesem Beispiel zu $\theta = 4$ gewählt.

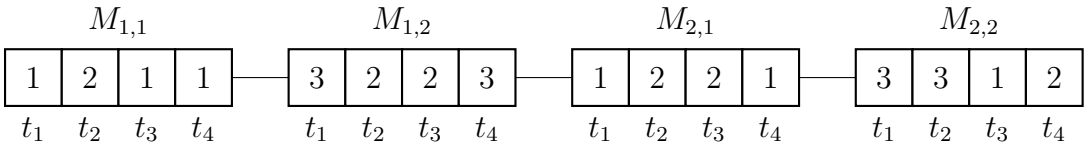


Abbildung 5.8: Beispiel für ein dem Kodierungsschema entsprechenden Chromosom

Die dem Chromosom entsprechende Matrix $\delta_{k,i,\tau}$ ist Formel 5.9 zu entnehmen.

$$\delta_{k,i,\tau} = \begin{pmatrix} 1 & 2 & 1 & 1 \\ 3 & 2 & 2 & 3 \\ 1 & 2 & 2 & 1 \\ 3 & 3 & 1 & 2 \end{pmatrix} \tag{5.9}$$

Tabelle 5.1: Parametriermöglichkeiten des Genetischen Algorithmus

Parameter	Formelzeichen	Bemerkung
Populationsgröße	N	
Selektionsalgorithmus	s_1	fitnessproportional rangproportional
Auswahlalgorithmus	s_2	Roulette Tournament Stochastic Universal Sampling
Tournamentgröße	n_T	nur bei Tournamentselektion
Selektionsdruck	E_max	nur bei rangbasierter Selektion
Kreuzungsschema	C	N-Point Crossover Uniform Crossover
Kreuzungswahrscheinlichkeit	p_C	
Kreuzungspunkte	n_C	nur bei N-Point Crossover
Kreuzungsrate	p_U	nur bei Uniform Crossover
Mutationswahrscheinlichkeit	p_M	
Elitismus	n_E	

5.2 Ameisenalgorithmus

Ameisenalgorithmen sind im Gegensatz zu den Genetischen Algorithmen noch relativ junge Verfahren. Die Basis dieser Algorithmen mit der Bezeichnung „Ant System“ stammt von Dorigo, Maniezzo und Colorni (1991, 1996) (siehe Merkle et al., 2014, S. 215). Die Weiterentwicklungen zu dem Verfahren „Ant Colony Optimization“ stammen im Wesentlichen von Dorigo und Gambardella (1997, 1996). Die folgenden Ausführungen

zur Funktionsweise von Ameisenalgorithmen stammen im Wesentlichen von Gerdes et al. (2004, S. 29), Kozak (2019) und Dorigo et al. (2004).

5.2.1 Inspiration durch das Verhalten von Ameisen

Ameisenalgorithmen basieren auf dem natürlichen Verhalten von Ameisen während der Futtersuche. Für die Futtersuche müssen die Ameisen einer Ameisenbevölkerung möglichst kurze Wege zwischen ihrem Nest und einer möglichen Nahrungsquelle finden. Obwohl die Insekten nahezu blind sind, schaffen sie es als Population, während der Futtersuche auch ihnen noch unbekannte Gebiete effizient zu erkunden.

Ohne äußere Einflüsse bewegen sich die Insekten mehr oder weniger nach dem Zufallsprinzip in ihrer Umgebung. Während der Futtersuche hinterlassen sie allerdings permanent Pheromone – also Duftstoffe – auf ihren zurückgelegten Wegen (siehe Abbildung 5.9).

Wege mit einer höheren Konzentration dieses hormonellen Lockstoffes werden von den Ameisen mit einer höheren Wahrscheinlichkeit für die eigene Futtersuche ausgewählt. Je mehr Ameisen auf denselben Teilstrecken laufen, desto höher ist entsprechend die Wahrscheinlichkeit, dass diese Wege – bedingt durch die erhöhte Pheromonkonzentration – von den weiteren Tieren ebenfalls ausgewählt werden.

Zu Beginn der Futtersuche läuft jede Ameise zunächst in zufällige Richtungen. Wird eine Futterquelle gefunden, läuft sie den gleichen Weg wieder zurück, den sie dorthin gefunden hat und hinterlässt darauf die Duftstoffe. Die Duftstoffe verfliegen nach einiger Zeit, so dass sich auf häufig genutzten Wegen langfristig eine höhere Pheromonkonzentration einstellt. Auf diese Weise werden auf kürzeren Teilstrecken innerhalb einer gegebenen Zeit höhere Pheromonkonzentrationen ausgebildet als auf längeren Wegstrecken. In Abbildung 5.10 ist die Pheromonkonzentration ρ der drei Wegstücke 1, 2 und 3 in rot gekennzeichnet.

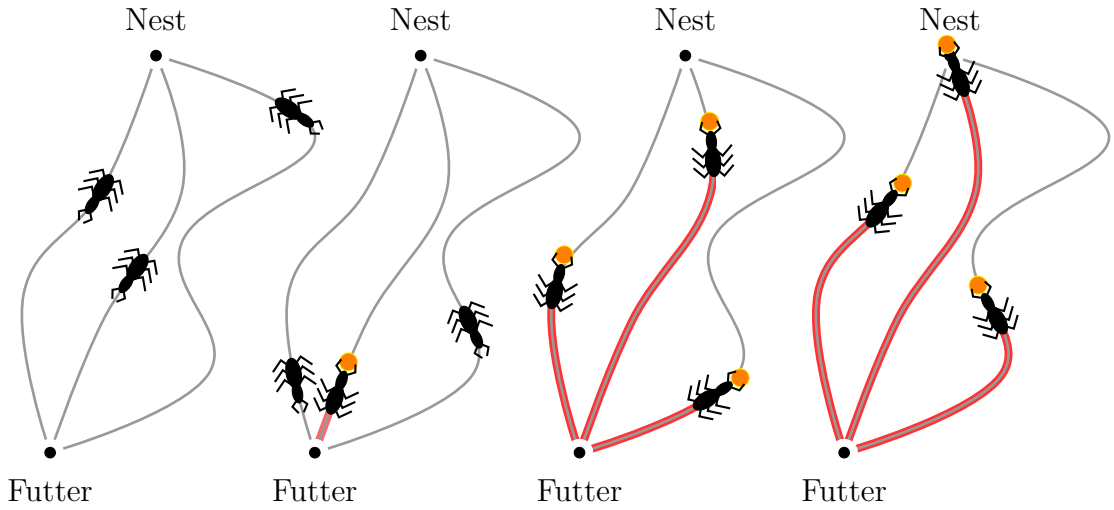


Abbildung 5.9: Ameisen legen auf ihrer Futtersuche Pheromonspuren auf den Wegen ab

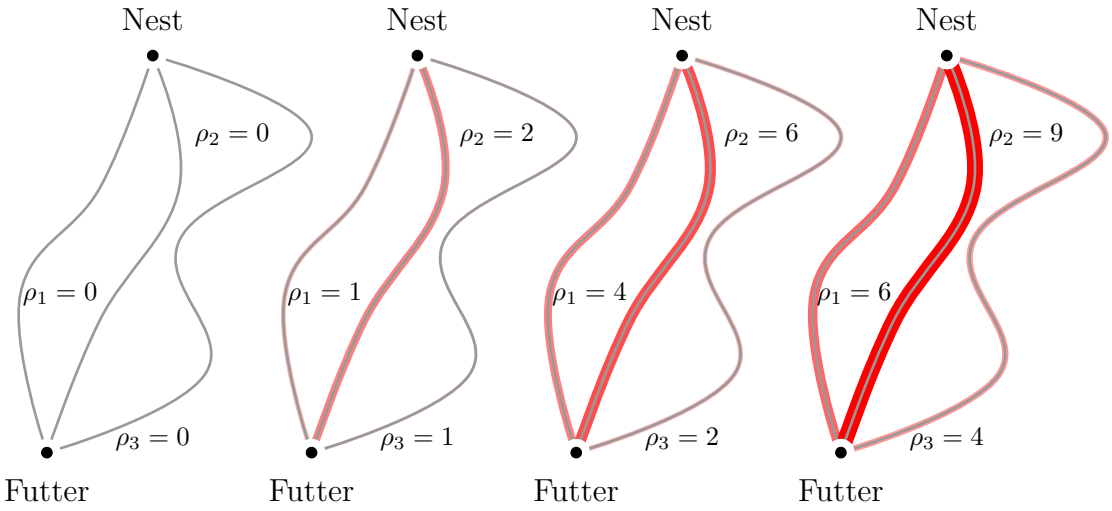


Abbildung 5.10: Sukzessive Ausbildung einer Pheromonspur auf dem kürzesten Weg

5.2.2 Ablauf des Basisverfahrens

Die ursprüngliche Anwendung von Ameisenalgorithmen bezieht sich auf das Finden möglichst kurzer Wege für das Problem des Handlungsreisenden (siehe beispielsweise Dorigo et al., 2004, S. 65). Dies entspricht weitestgehend dem Ursprung des Verfahrens, der entsprechend in der Suche von möglichst effizienten Wegen zwischen dem Nest der Ameisen und einer Futterquelle liegt. Durch die Anpassung der Verfahrensparameter und geeignete Problemformulierungen kann das Verfahren allerdings auf allgemeine kombinatorische Optimierungsprobleme angewendet werden.

Abbildung 5.11 zeigt ein entsprechendes Rundreiseproblem. Dargestellt sind die zehn Orte $j = 1$ bis $j = 10$. Es ist eine möglichst kurze, geschlossene Rundreise gesucht, bei welcher eine Ameise jeden Ort genau einmal besucht und am Ende der Reise wieder an ihrem Ursprungsort herauskommt.

Zwischen allen Orten besteht eine Verbindung. So kann eine Ameise an jedem Ort entscheiden, welchen Ort sie als nächstes besucht. Die Pheromonkonzentrationen auf den einzelnen Strecken sind wiederum in rot gekennzeichnet. Die kürzeste Route sticht entsprechend mit einer stärkeren Konzentration hervor.

Für eine Ameise, die sich während einer Rundreise in dem Ort j_1 befindet, wird die Wahrscheinlichkeit p_{j_1,j_2}, mit welcher sie als nächstes den Ort j_2 besucht, durch die Formel 5.10 berechnet (Dorigo et al., 2004, S. 70).

$$p_{j_1,j_2} = \begin{cases} \dfrac{\rho_{j_1,j_2}^{\alpha} \cdot \eta_{j_1,j_2}^{\beta}}{\sum\limits_{j_3 \in \mathcal{J}_{j_1}} \left(\rho_{j_1,j_3}^{\alpha} \cdot \eta_{j_1,j_3}^{\beta} \right)} & \text{für } j_2 \in \mathcal{J}_{j_1} \\ 0 & \text{sonst} \end{cases} \tag{5.10}$$

Dafür wird zunächst geprüft, ob der Ort j_2 in der Menge $\mathcal{J}_1$ vorhanden ist. Diese enthält nur die Städte, die von der entsprechenden Ameise während der Rundreise noch

nicht besucht wurden. Befindet sich die Stadt j_2 nicht in der Menge $\mathcal{J}_1$, so gilt für die Wahrscheinlichkeit $p_{j_1,j_2} = 0$.

Für den Fall, dass j_2 von der Ameise noch nicht besucht wurde, wird ein entsprechender Wahrscheinlichkeitswert auf Basis der Pheromonkonzentration $\rho_{j1,j2}$ gebildet. Eine lokale Suchstrategie wird über den Koeffizienten $\eta_{j1,j2}$ in die Berechnung eingebunden. Im Falle des Rundreiseproblems wird dafür in der Regel die Länge der Teilstrecke d_{j_1,j_2} zwischen den Orten j_1 und j_2 eingesetzt (Formel 5.11). Dadurch werden kürzere Strecken mit einer höheren Wahrscheinlichkeit ausgewählt.

$$\eta_{j_1,j_2} = d_{j_1,j_2} \tag{5.11}$$

Über die beiden Exponenten α und β wird das Verhältnis zwischen der globalen Suchstrategie über die Pheromone und der lokalen Suchstrategie angepasst. Höhere Werte für α verstärken die Suche auf Basis der Pheromonspuren, höhere Werte für β verstärken die Suche auf Basis der lokalen Strategie. Das Produkt der beiden Größen wird entsprechend zu einem Wahrscheinlichkeitswert zwischen 0 und 1 normiert.

Der Ablauf des gesamten Verfahrens ist mit Algorithmus 5.2 gegeben (Dorigo et al., 2005, in Anlehnung an). Der Algorithmus beginnt mit der Initialisierung der Pheromonmatrix ρ_{j_1,j_2}. Dafür werden die Pheromonwerte zwischen den einzelnen Orten zu Beginn des Verfahrens jeweils auf definierte Werte initialisiert.

Anschließend wird der Suchprozess gestartet, welcher solange durchgeführt wird, bis ein Abbruchkriterium erfüllt ist. Dies kann beispielsweise das Erreichen einer bestimmten Anzahl an Iterationsschritten, das Erreichen einer bestimmten Laufzeit oder das Erfüllen eines Konvergenzkriteriums sein.

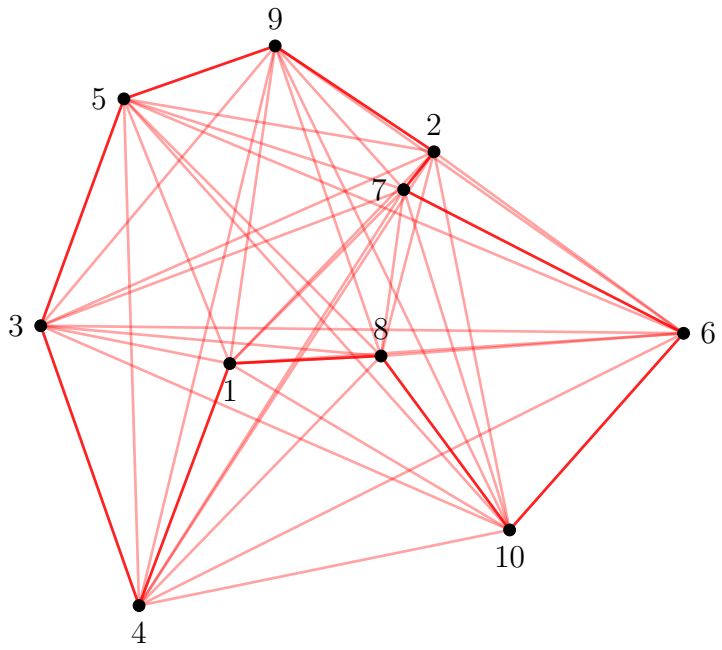


Abbildung 5.11: Pheromonkonzentrationen in einem Rundreiseproblem

Algorithmus 5.2 Ablauf des Ameisenalgorithmus

```
prozedur AMEISENALGORITHMUS
    Initialisierung
    wiederhole
        für jede Ameise k führe durch
            Lösung konstruieren
        ende für jede
        Pheromonmatrix aktualisieren
    bis Abbruchkriterium
ende prozedur
```

In jedem Iterationsschritt wird eine vorgegebene Anzahl m an künstlichen Ameisen generiert. Jede dieser Ameisen konstruiert anschließend eine Lösung auf Basis der in Formel 5.10 gegebenen Wahrscheinlichkeitsberechnung. Darauf folgt die Aktualisierung der Pheromonwerte für die einzelnen Teilstrecken durch die in Formel 5.12 gegebene Rechenvorschrift (Dorigo et al., 2004, S. 72).

$$\rho_{j_1,j_2} := (1-\epsilon) \cdot \rho_{j_1,j_2} + \sum_{k=1}^{m} \Delta\rho_{j_1,j_2}(k) \tag{5.12}$$

Für jede Teilstrecke s_{j_1,j_2} zwischen den Orten j_1 und j_2 wird die Pheromonkonzentration aktualisiert, indem zunächst das Verfliegen der Duftstoffe durch den Faktor $(1-\epsilon)$ angewendet wird. Anschließend legt jede Ameise k der m Ameisen ihre individuelle Pheromonspur $\Delta\rho_{j_1,j_2}(k)$ auf die entsprechende Strecke ab. Dafür wird für die Teilstrecke s_{j_1,j_2} geprüft, ob sie Teil der von der Ameise k generierten Lösung ist. Ist dies der Fall, so wird eine Pheromonmenge abgelegt, die proportional zu der Qualität der entsprechenden Lösung ist. In Formel 5.13 wird für das Rundreiseproblem entsprechend für jede Ameise k eine Pheromonmenge $\Delta\rho_{j_1,j_2}$ berechnet, die umgekehrt proportional zu der Länge L_k der entsprechenden Lösung T_k ist (Dorigo et al., 2004, S. 72).

$$\Delta\rho_{j_1,j_2}(k) = \begin{cases} \dfrac{1}{L_k} & \text{für } s_{j_1,j_2} \in T_k \\ 0 & \text{sonst} \end{cases} \tag{5.13}$$

5.2.3 Varianten des Verfahrens

Das in Unterabschnitt 5.2.2 vorgestellte Basisverfahren lässt zahlreiche Variationen und zusätzliche Definitionen zu. Es lassen sich dementsprechend viele Varianten ableiten und entwickeln, die dem Basisverfahren für bestimmte Aufgabenstellungen überlegen sein können. Die für die Anwendung des Verfahrens im Rahmen dieser Arbeit relevanten Varianten beziehen sich im Wesentlichen auf die folgenden Aspekte:

- Welche Ameise darf in einem Iterationsschritt eine Pheromonspur legen?
- Wie viel Pheromone darf eine Ameise legen?

- Sind die Pheromonkonzentrationen auf den Teilstrecken begrenzt?

Gutjahr (2011, S. 230) entwickelt für diesen Zweck ein Schema für die systematische Zusammenfassung und Klassifizierung der unterschiedlichen Varianten in einem Formelausdruck. In Anlehnung an Gutjahr lassen sich Ameisenalgorithmen durch den in Formel 5.14 gegebenen Ausdruck zusammenfassen.

$$\#\text{ants}\big|\text{reinf}\big|\text{reward}\big|\text{pherbound} \tag{5.14}$$

#ants: Entspricht der Populationsgröße N und gibt an, wie viele künstliche Ameisen in einem Iterationsschritt erzeugt werden.

reinf: Dieser Parameter bestimmt, welche Ameisen Pheromonspuren legen dürfen. Für das Basisverfahren gilt reinf = ac (all of current iteration), womit alle Ameisen einer Iteration Pheromonspuren legen. Für reinf = ib (r) (iteration-best with r ranked ants) dürfen nur die Ameisen einer Iteration Pheromonspuren legen, welche die r besten Lösungen generiert haben. Der Fall reinf = bf (best-so-far) gibt an, dass nur die bislang beste Ameise Pheromonspuren legen darf. Unabhängig von den Ameisen eines aktuellen Iterationsschritts darf immer nur die Ameise Pheromonspuren legen, welche die bislang beste Lösung erzeugt hat.

reward: Durch diesen Parameter wird die Pheromonmenge festgelegt, welche von einer Ameise auf den Teilstrecken abgelegt wird. Bei dem Basisverfahren gilt reward = fp (fitness-proportional), so dass die von einer Ameise abgelegte Pheromonmenge proportional zu der Qualität der entsprechenden Lösung ist. reward = co (constant) gibt an, dass die abgelegte Pheromonmenge immer konstant ist.

pherbound: Mit diesem Parameter können die Pheromonkonzentrationen auf den Streckenabschnitten nach oben und nach unten begrenzt werden. Mit pherbound = nb (no bound) findet – wie beim Basisverfahren – keine Begrenzung statt. Mit pherbound = lb (lower bound) wird die Pheromonmenge auf ein definiertes Minimum $\rho_{\min}$, mit pherbound = ub (upper bound) auf ein definiertes Maximum $\rho_{\max}$ begrenzt. Durch die Angabe von pherbound = ulb (upper and lower bound) wird die Pheromonkonzentration sowohl nach oben als auch nach unten begrenzt.

5.2.4 Kopplung mit dem Planungsverfahren

Der im Rahmen dieser Arbeit entwickelte Ameisenalgorithmus ist gemäß dem Schema aus Formel 5.14 in Formel 5.15 angegeben.

$$N\big|\text{bf}\big|\text{co}\big|\text{ulb} \tag{5.15}$$

Es handelt sich entsprechend um einen Ameisenalgorithmus, bei dem nur die global beste Ameise (bf) eine konstante Menge an Pheromonen (co) ablegen darf. Die Anzahl der Ameisen bleibt entsprechend durch den Faktor N variabel und die Pheromonkonzentration in der Pheromonmatrix ist auf $\rho_{\min}$ und $\rho_{\max}$ begrenzt (ulb).

Die Suche durch den Ameisenalgorithmus soll rein probabilistisch, d.h. ohne den Einfluss eines lokalen Suchparameters durchgeführt werden. Die Basisparameter α, β und η werden entsprechend den Formeln 5.16 bis 5.18 ausgewählt.

$$\alpha = 1 \tag{5.16}$$

$$\beta = 0 \tag{5.17}$$

$$\eta = 1 \tag{5.18}$$

Die Wahrscheinlichkeit $p_{k,i,\tau,f}$, dass eine Ameise für die Maschine $M_{k,i}$ für die Planungsperiode t_τ die Produktion der Auftragsfamilie $F_{k,i,f}$ auswählt, lässt sich dann gemäß Formel 5.19 berechnen.

$$p_{k,i,\tau,f} = \frac{\rho_{k,i,\tau,f}}{\sum_{f'}^{o_{k,i}} \rho_{k,i,\tau,f'}} \tag{5.19}$$

Die Auswahlwahrscheinlichkeiten der Auftragsfamilien einer Maschine $M_{k,i}$ müssen dabei für eine Zeitperiode t_τ in Summe stets eins ergeben (Formel 5.20).

$$\sum_{f=1}^{o_{k,i}} p_{k,i,\tau,f} = 1 \tag{5.20}$$

Nur die Ameise, die iterationsübergreifend die bislang beste Lösung erzeugt hat, legt eine Pheromonspur mit dem festen Wert ρ_+ ab (Formel 5.21).

$$\Delta\rho_{k,i,\tau,f} = \begin{cases} \rho_+ & \text{für } \delta_{k,i,\tau}^{\text{Lösung}} = f \\ 0 & \text{für } \delta_{k,i,\tau}^{\text{Lösung}} \neq f \end{cases} \tag{5.21}$$

Die Regel für die Aktualisierung der Pheromonmatrix $\rho_{k,i,\tau,f}$ ist in Formel 5.22 gegeben.

$$\rho_{k,i,\tau,f} := (1 - \epsilon) \cdot \rho_{k,i,\tau,f} + \Delta\rho_{k,i,\tau,f} \tag{5.22}$$

Die Pheromonkonzentration $\rho_{k,i,\tau,f}$ wird durch die Formeln 5.23 und 5.24 in jedem Iterationsschritt auf den Minimalwert ($\rho_{\min}$) bzw. Maximalwerte ($\rho_{\max}$) begrenzt.

$$\rho_{k,i,\tau,f} := \max\left(\rho_{\min}; \rho_{k,i,\tau,f}\right) \tag{5.23}$$

$$\rho_{k,i,\tau,f} := \min\left(\rho_{k,i,\tau,f}; \rho_{\max}\right) \tag{5.24}$$

In Tabelle 5.2 sind alle Parameter des Ameisenalgorithmus zusammengefasst.

Beispiel 5.2 (Zweistufiges Flexible Flow Shop Problem mit insgesamt vier Maschinen) Die Anwendung des Ameisenalgorithmus auf das Planungsverfahren wird anhand eines Beispiels verdeutlicht. Gegenstand des Beispiels ist in Anlehnung an das Beispiel 5.1 wieder ein Flexible Flow Shop Problem mit $c = 2$ Produktionsstufen mit jeweils $m_1 = m_2 = 2$ Maschinen in jeder Stufe. Die dabei angenommene Anzahl der jeweiligen Auftragsfamilien für die Maschinen ist in Formeln 5.25 und 5.26 gegeben.

$$o_{1,1} = o_{2,1} = 2 \tag{5.25}$$

$$o_{1,2} = o_{2,2} = 3 \tag{5.26}$$

Bei den Maschinen $M_{1,1}$ und $M_{2,1}$ sind also jeweils zwei Auftragsfamilien vorhanden (Formel 5.25). Bei den Maschinen $M_{1,2}$ und $M_{2,2}$ sind jeweils drei Auftragsfamilien vorhanden (Formel 5.26).

Für $\theta = 4$ Planungsperioden ist in Formel 5.27 die resultierende Pheromonmatrix des Ameisenalgorithmus dargestellt.

$$\rho_{k,i,\tau,f} = \begin{pmatrix} \rho_{1,1,1,1} & \rho_{1,1,2,1} & \rho_{1,1,3,1} & \rho_{1,1,4,1} \\ \rho_{1,1,1,2} & \rho_{1,1,2,2} & \rho_{1,1,3,2} & \rho_{1,1,4,2} \\ \rho_{1,2,1,1} & \rho_{1,2,2,1} & \rho_{1,2,3,1} & \rho_{1,2,4,1} \\ \rho_{1,2,1,2} & \rho_{1,2,2,2} & \rho_{1,2,3,2} & \rho_{1,2,4,2} \\ \rho_{1,2,1,3} & \rho_{1,2,2,3} & \rho_{1,2,3,3} & \rho_{1,2,4,3} \\ \rho_{2,1,1,1} & \rho_{2,1,2,1} & \rho_{2,1,3,1} & \rho_{2,1,4,1} \\ \rho_{2,1,1,2} & \rho_{2,1,2,2} & \rho_{2,1,3,2} & \rho_{2,1,4,2} \\ \rho_{2,2,1,1} & \rho_{2,2,2,1} & \rho_{2,2,3,1} & \rho_{2,2,4,1} \\ \rho_{2,2,1,2} & \rho_{2,2,2,2} & \rho_{2,2,3,2} & \rho_{2,2,4,2} \\ \rho_{2,2,1,3} & \rho_{2,2,2,3} & \rho_{2,2,3,3} & \rho_{2,2,4,3} \end{pmatrix} \tag{5.27}$$

So gibt beispielsweise $\rho_{1,2,4,3}$ die Pheromonkonzentration für die Auftragsfamilie $F_{1,2,3}$ ($f = 3$) der Maschine $M_{1,2}$ ($k = 1$, $i = 2$) in der Planungsperiode t_4 ($\tau = 4$) an. Je höher die Pheromonkonzentration $\rho_{1,2,4,3}$ ist, desto wahrscheinlicher ist die Einplanung der Auftragsfamilie $F_{1,2,3}$ in der Planungsperiode t_4 an der Maschine $M_{1,2}$.

Tabelle 5.2: Parametriermöglichkeiten des Ameisenalgorithmus

Parameter	Formelzeichen
Populationsgröße	N
Initialpheromonkonzentration	ρ_{init}
Minimalpheromonkonzentration	$\rho_{\min}$
Maximalpheromonkonzentration	$\rho_{\max}$
Pheromonadditionskoeffizient	ρ_{+}
Pheromonzerfallskoeffizient	ϵ

5.3 Zusammenfassende Diskussion

Bei den im Rahmen dieser Arbeit verwendeten Suchverfahren handelt es sich um einen Genetischen Algorithmus und einen Ameisenalgorithmus. Für beide Verfahren kann aufgrund ihrer positiven Forschungsergebnisse ein geeignetes Lösungsverhalten für die in dieser Arbeit behandelte Problemstellung angenommen werden.

Es handelt sich bei beiden Suchverfahren um Metaheuristiken, deren Mechanismen jeweils einem natürlichen Vorbild nachempfunden sind. So basiert der Genetische Algorithmus auf den Prinzipien der Genetik während des Evolutionsprozess, während der Ameisenalgorithmus auf dem natürlichen Verhalten von Ameisen bei der Futtersuche basiert.

Beide Metaheuristiken arbeiten stochastisch, probabilistisch und populationsbasiert. Innerhalb jeweils eines Iterationsschritts werden daher bei beiden Verfahren immer zufallsbasiert (stochastisch) mehrere Lösungen gleichzeitig (populationsbasiert) erzeugt und miteinander verglichen (probabilistisch).

Ein wesentlicher Unterschied in Bezug auf die Anwendung der beiden Verfahren – neben den eindeutig unterschiedlichen Wirkprinzipien – besteht in den Parametriermöglichkeiten der beiden Verfahren (siehe dazu auch die Tabellen 5.1 und 5.2). Für die Erzeugung von optimalen Ergebnissen müssen die zur Verfügung stehenden Parameter der beiden Verfahren jeweils auf das zu lösende Problem eingestellt werden. Dabei ist festzustellen, dass bei dem in Abschnitt 5.1 entwickelten Genetischen Algorithmus 11 Parameter und bei dem in Abschnitt 5.2 entwickelten Ameisenalgorithmus 6 Parameter optimiert werden müssen.

Desweiteren verfolgt der Ameisenalgorithmus im Gegensatz zum Genetischen Algorithmus einen vergleichsweise simplen Programmablauf (siehe dazu die Algorithmen 5.1 und 5.2). Während eines Iterationsschritts des Genetischen Algorithmus werden sequenziell die vier unterschiedlichen Genetischen Operatoren

1. Selektion,
2. Rekombination,
3. Mutation und
4. Ersetzung

angewendet, deren konkrete Ausgestaltung teilweise mit mehreren Rechenvorschriften verbunden ist. Bei dem Ameisenalgorithmus handelt es sich im Wesentlichen um die beiden Schritte

1. Lösung konstruieren und
2. Pheromonmatrix aktualisieren.

Sowohl aufgrund der geringeren Anzahl von Parametern als auch aufgrund der geringeren Komplexität des Lösungsverhaltens ist bei dem Ameisenalgorithmus davon auszugehen, dass dessen Anpassung auf die Problemstellung dieser Arbeit weniger komplex ist als die Anpassung des Genetischen Algorithmus. Andererseits lässt die höhere Anzahl von Parametern bei dem Genetischen Algorithmus die Vermutung zu, dass nach einer erfolgreichen Anpassung der Parameter möglicherweise bessere Ergebnisse zu erwarten sind.

6 Numerische Untersuchungen

Die numerischen Untersuchungen des Verfahrens erfolgen zunächst an randomisierten Testdaten. Für drei unterschiedliche Problemgrößen S (klein), M (mittel) und L (groß) wird dabei in einem ersten Schritt analysiert, welches der beiden Optimierungsverfahren (Genetischer Algorithmus und Ameisenalgorithmus) jeweils die beste Performance liefert. Um die jeweilige Leistung der beiden Optimierungsalgorithmen (Metaheuristiken) objektiv miteinander vergleichen zu können, werden die einzelnen Parameter der beiden Optimierungsverfahren über Parameterstudien zunächst optimiert.

Auf Basis der optimierten Verfahrensparameter erfolgen anschließend unter vergleichbaren Bedingungen weitere Untersuchungen an den drei Testdatensätzen S, M und L. Dabei werden zunächst die Einflüsse unterschiedlicher Anzahlen von Planungsperioden auf die Planungsergebnisse untersucht. Weiterhin werden die Einflüsse der unterschiedlichen Planungsregeln auf die Planungsergebnisse analysiert. Darauf aufbauend wird das Laufzeitverhalten des Verfahrens auf Basis der optimierten Parameter beurteilt. Abschließend werden beispielhaft die besten mithilfe des Verfahrens gefundenen Ergebnisse in Form von konkreten Ablaufplänen analysiert.

Das Planungsverfahren wird im Anschluss auf die Produktion eines realen Betriebs angewendet. Dies erfolgt durch die Implementierung in ein Entscheidungssystem für die operative Produktionsplanung und -steuerung. Das Verfahren wird dafür im Rahmen dieser Arbeit mit den optimierten Parametern der Optimierungsverfahren auf die realen Daten des Betriebs angewendet. Die Durchführung der numerischen Untersuchungen im Rahmen dieser Arbeit erfolgt an einem Rechencluster. Für die parallelisierte Ausführung des Verfahrens stehen dabei 40 CPU-Kerne zur Verfügung. Bei sämtlichen Studien dieser Arbeit wird als Abbruchkriterium der Suchverfahren das Erreichen einer Zeitschwelle von 4 min gewählt. Für vergleichbare Rechnungen, die auf einem gewöhnlichen PC-System (Intel i3, siehe Abschnitt 1.2) durchgeführt werden, entspricht dies einer Rechenzeit von ca. 20 min.

Für die statistische Absicherung der Ergebnisse wird eine Wiederholrate von jeweils 50 Optimierungsläufen gewählt. Dementsprechend wird im Rahmen der numerischen Untersuchungen jede Parameterveränderung 50-mal hintereinander gerechnet. Jede statistisch abgesicherte Rechnung nimmt folglich eine Rechenzeit von $50 \cdot 4\,\text{min} = 3\,\text{h}$ und 20 min in Anspruch (gewöhnliches PC-System: $50 \cdot 20\,\text{min} = 16\,\text{h}$ und 40 min).

Ohne Berücksichtigung der umfangreichen (dieser Arbeit vorgelagerten) Voruntersuchungen werden im Rahmen dieser Arbeit auf dem Rechencluster ca. 250 Studien durchgeführt. Bei einer Wiederholrate von 50 werden dementsprechend ca. $50 \cdot 250 = 12\,500$ Optimierungsläufe durchgeführt. Die dafür benötigte reine Rechenzeit auf dem Rechencluster liegt

entsprechend bei ca. $12\,500 \cdot 4\,\text{min} \approx 35\,\text{d}$ (Tage). Auf einem gewöhnlichen PC-System entspricht dies einer (ununterbrochenen) Rechenzeit von ca. 175 d (Tage) am Stück.

Die statistische Auswertung der untersuchten Datensätze erfolgt über die in Abbildung 6.1 dargestellte Methode. Die statistische Streuung der Ergebnisse (jeweils 50 Rechnungen) wird entsprechend über Boxplots visualisiert und analysiert. Die jeweils erreichte Zielgröße $\gamma\prime = \frac{\gamma}{\gamma_{\min}}$ wird dabei immer auf die beste gefundene Lösung der entsprechenden Untersuchung normiert. Dadurch wird eine objektive Vergleichbarkeit der Ergebnisse sichergestellt.

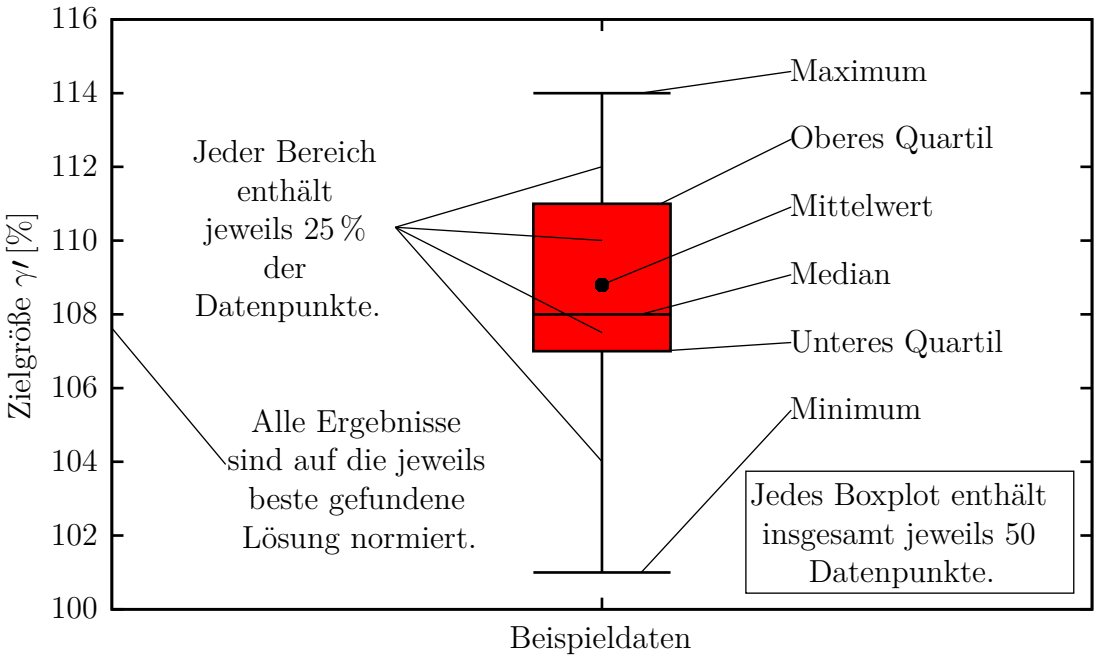


Abbildung 6.1: Erläuterung eines Boxplots anhand eines Beispiels

6.1 Generierung von randomisierten Testdatensätzen

Für die Untersuchung des Verfahrens anhand beliebiger und zufällig generierter Datensätze ist in Tabelle 6.1 ein Muster für die Erzeugung der entsprechenden Problemdaten gegeben. Es handelt sich dabei um die systematische Generierung von Datensätzen, die jeweils einer der drei Größenklassen **S**, **M** oder **L** zuzuordnen sind. Für die weiteren Untersuchungen im Rahmen dieser Arbeit werden die drei Testdatensätze **S–224**, **M–424** und **L–824** ausgewählt, die dem Muster der Tabelle 6.1 entsprechend erzeugt werden.

Die drei Probleme unterscheiden sich jeweils zunächst in der Anzahl der Produktionsstufen c. Die Anzahl der Maschinen $m_k = 2$ je Produktionsstufe sowie die jeweilige Anzahl an Auftragsfamilien $o_{k,i} = 4$ je Maschine ist bei den gewählten Problemen identisch. Die

resultierende Länge L des entsprechenden Kodierungsschemas (siehe Abbildung 5.7 und Formel 5.7) lässt sich dafür in vereinfachter Form gemäß Formel 6.1 berechnen.

$$L = c \cdot m_k \cdot \theta \tag{6.1}$$

Für die Komplexität κ der entsprechenden Probleme gilt dann der in Formel 6.2 vereinfachte Ausdruck (vgl. Formel 4.80).

$$\kappa = o^L \tag{6.2}$$

Die Anzahl der Planungsperioden wird bei allen Problemen zunächst zu $\theta = 8$ gewählt (Formel 6.3).

$$\theta = 8 \tag{6.3}$$

Tabelle 6.1: Systematische Erzeugung von randomisierten Testdatensätzen

Problem	Größe	Stufen c	Maschinen m_k	Familien $o_{k,i}$	Code-Länge L	Komplexität $\kappa\,(\theta = 8)$
S–144	S	1	4	4	4θ	$1{,}8 \cdot 10^{19}$
S–224	**S**	**2**	**2**	**4**	$\boldsymbol{4\theta}$	$\mathbf{1{,}8 \cdot 10^{19}}$
S–414	S	4	1	4	4θ	$1{,}8 \cdot 10^{19}$
M–244	M	2	4	4	8θ	$3{,}4 \cdot 10^{38}$
M–424	**M**	**4**	**2**	**4**	$\boldsymbol{8\theta}$	$\mathbf{3{,}4 \cdot 10^{38}}$
L–284	L	2	8	4	16θ	$1{,}2 \cdot 10^{77}$
L–444	L	4	4	4	16θ	$1{,}2 \cdot 10^{77}$
L–824	**L**	**8**	**2**	**4**	$\boldsymbol{16\theta}$	$\mathbf{1{,}2 \cdot 10^{77}}$

Die Gewichtungen der einzelnen Teilziele (siehe Unterabschnitt 4.1.5) ist in den Formeln 6.4 bis 6.7 gegeben. Die Teilziele sollen bei dem Optimierungsprozess entsprechend generell gleichmäßig berücksichtigt werden (Formeln 6.4, 6.5 und 6.7), wobei ein besonderer Fokus auf der Reduzierung der Rüstzeiten liegt (Formel 6.6).

$$w_1' = 1{,}0\,\frac{1}{\mathrm{h}} \qquad \text{(Stillstandszeiten)} \tag{6.4}$$

$$w_2' = 1{,}0\,\frac{1}{\mathrm{h}} \qquad \text{(Durchlaufzeiten)} \tag{6.5}$$

$$w_3' = 1{,}5\,\frac{1}{\mathrm{h}} \qquad \text{(Rüstzeiten)} \tag{6.6}$$

$$w_4' = 1{,}0\,\frac{1}{\mathrm{h}} \qquad \text{(Verspätungen)} \tag{6.7}$$

Um die Vergleichbarkeit der einzelnen Studien zu gewährleisten, werden sämtliche Rechnungen zunächst mit nur einer Planungsregel durchgeführt. Bei jedem Datensatz wird

dafür in jeder Produktionsstufe die Planungsregel **S/OPN (geringste Schlupfzeit zuerst)** eingesetzt.

Tabelle 6.2: Konkrete Ausprägungen der Testdatensätze

Parameter	Wert	Beschreibung
n (S–224)	200	Anzahl der Jobs
n (M–424)	320	Anzahl der Jobs
n (L–824)	480	Anzahl der Jobs
H [h]	48	Planungshorizont
θ	8	Anzahl der Planungsperioden
w_j	1	Gewichtung der Jobs
d_j [d]	$U\,[-5;5]$	Planliefertermine
$\sigma_{k,j}$	$U\,\{0;1\}$	Prozessmatrix
$\lambda_{k,i,j}$	$U\,\{0;1\}$	Kompatiblitätsmatrix
$\mu_{k,i,j,f}$	$U\,\{0;1\}$	Familienzugehörigkeit
c (S–224)	2	Anzahl der Produktionsstufen
c (M–424)	4	Anzahl der Produktionsstufen
c (L–824)	8	Anzahl der Produktionsstufen
m_k	2	Anzahl der Maschinen je Stufe
$o_{k,i}$	4	Anzahl der Auftragsfamilien
$p_{1,1,j}$ [min] und $p_{5,1,j}$ [min]	$U\,[20;40]$	Bearbeitungszeiten
$p_{1,2,j}$ [min] und $p_{5,2,j}$ [min]	$U\,[20;40]$	Bearbeitungszeiten
$p_{2,1,j}$ [min] und $p_{6,1,j}$ [min]	$U\,[60;150]$	Bearbeitungszeiten
$p_{2,2,j}$ [min] und $p_{6,2,j}$ [min]	$U\,[70;100]$	Bearbeitungszeiten
$p_{3,1,j}$ [min] und $p_{7,1,j}$ [min]	$U\,[50;80]$	Bearbeitungszeiten
$p_{3,2,j}$ [min] und $p_{7,2,j}$ [min]	$U\,[20;40]$	Bearbeitungszeiten
$p_{4,1,j}$ [min] und $p_{8,1,j}$ [min]	$U\,[30;60]$	Bearbeitungszeiten
$p_{4,2,j}$ [min] und $p_{8,2,j}$ [min]	$U\,[60;90]$	Bearbeitungszeiten
$\phi_{1,1,f_1,f_2}$ [min] und $\phi_{5,1,f_1,f_2}$ [min]	$U\,[5;60]$	Rüstzeiten
$\phi_{1,2,f_1,f_2}$ [min] und $\phi_{5,2,f_1,f_2}$ [min]	$U\,[5;60]$	Rüstzeiten
$\phi_{2,1,f_1,f_2}$ [min] und $\phi_{6,1,f_1,f_2}$ [min]	$U\,[100;190]$	Rüstzeiten
$\phi_{2,2,f_1,f_2}$ [min] und $\phi_{6,2,f_1,f_2}$ [min]	$U\,[170;220]$	Rüstzeiten
$\phi_{3,1,f_1,f_2}$ [min] und $\phi_{7,1,f_1,f_2}$ [min]	$U\,[60;160]$	Rüstzeiten
$\phi_{3,2,f_1,f_2}$ [min] und $\phi_{7,2,f_1,f_2}$ [min]	$U\,[40;120]$	Rüstzeiten
$\phi_{4,1,f_1,f_2}$ [min] und $\phi_{8,1,f_1,f_2}$ [min]	$U\,[30;40]$	Rüstzeiten
$\phi_{4,2,f_1,f_2}$ [min] und $\phi_{8,2,f_1,f_2}$ [min]	$U\,[30;40]$	Rüstzeiten

Die konkrete Generierung der zufallsbasierten Testdatensätze basiert auf der Erzeugung gleichverteilter Zufallszahlen durch die Funktion $U\,[\dots]$ bzw. $U\,\{0;1\}$. Die für die Erzeugung der jeweiligen Datensätze zugrunde gelegten Daten sind in der Tabelle 6.2 gegeben.

Der Planungshorizont wird bei den drei Problemen jeweils zu $H = 48\,\text{h}$ gewählt. Die Anzahl der Fertigungsaufträge beträgt jeweils $n = 200$ (S–224), $n = 320$ (M–424) und $n = 480$ (L–824).

6.2 Auswahl eines Optimierungsverfahrens

Für jeden der drei randomisierten Testdatensätze (S–224, M–424 und L–824) soll im Folgenden die Performance des Genetischen Algorithmus mit der Performance des Ameisenalgorithmus verglichen werden.

6.2.1 Festlegung der optimalen Parameterausprägungen

Beide Optimierungsalgorithmen (Genetischer Algorithmus und Ameisenalgorithmus) müssen für einen aussagekräftigen Vergleich jeweils mit auf das Planungsverfahren optimierten Parametern analysiert werden. Die optimale Parametrierung einer Metaheuristik stellt aufgrund der zahlreichen Einstellmöglichkeiten der Parameter sowie deren starken Wechselwirkungen selbst wiederum ein hartes Optimierungsproblem dar.

Mithilfe von Voruntersuchungen konnten für die entsprechenden Parameter – falls erforderlich – sinnvolle Größenordnungen bereits eingegrenzt werden. Über die systematische Variation der einzelnen Verfahrensparameter erfolgt im Rahmen der Parameterstudien dieser Arbeit die optimierende Anpassung der Parameter auf die randomisierten Planungsprobleme S–224, M–424 und L–824. Die Ausprägungen und Variationen der einzelnen Optimierungsparameter sowie deren Initialwerte sind dafür den Tabellen 6.3 bis 6.5 (Genetischer Algorithmus) sowie den Tabellen 6.6 bis 6.8 (Ameisenalgorithmus) zu entnehmen.

Tabelle 6.3: Parametervariation des Genetischen Algorithmus für S–224

Parameter	Initialwert	Variation
N (Populationsgröße)	400	{100; 200; 400; 800; 1600; 3200}
s_1 (Selektionsalgorithmus)	–	{rangbasiert; fitnessbasiert}
s_2 (Auswahlalgorithmus)	Tournament	{Tournament; Roulette; SUS}
n_T (Tournamentgröße)	40	{5; 10; 20; 40; 80}
E_max (Selektionsdruck)	–	{fitness; 1,1; 1,3; 1,5; 1,7; 1,9}
C (Kreuzungsschema)	N-Punkt	{N-Punkt; Uniform}
p_C (Kreuzungswahrscheinlichkeit)	0,8	{0,0; 0,2; 0,4; 0,6; 0,8; 1,0}
n_C (Kreuzungspunkte)	2	{1; 2; 4; 8; 16}
p_U (Kreuzungsrate)	–	{0,2; 0,4; 0,6; 0,8}
p_M (Mutationswahrscheinlichkeit)	0,1	{0,00; 0,01; 0,05; 0,10; 0,15; 0,20}
n_E (Elitismus)	0	{0, 1, 2, 3, 4, 5}

Bei der Variation eines Parameters werden die Ausprägungen der weiteren Parameter jeweils festgehalten. Ausgehend von dem jeweiligen Initialwert eines Parameters wird entsprechend immer nur ein einziger Parameter gleichzeitig verändert.

Tabelle 6.4: Parametervariation des Genetischen Algorithmus für M–424

Parameter	Initialwert	Variation
N (Populationsgröße)	800	$\{400; 800; 1600; 3200; 6400\}$
s_1 (Selektionsalgorithmus)	–	{rangbasiert; fitnessbasiert}
s_2 (Auswahlalgorithmus)	Tournament	{Tournament; Roulette; SUS}
n_T (Tournamentgröße)	10	$\{1; 2; 5; 10; 20\}$
E_max (Selektionsdruck)	–	{fitness; 1,3; 1,5; 1,7}
C (Kreuzungsschema)	N-Punkt	{N-Punkt; Uniform}
p_C (Kreuzungswahrscheinlichkeit)	0,6	$\{0{,}2; 0{,}4; 0{,}6; 0{,}8; 1{,}0\}$
n_C (Kreuzungspunkte)	4	$\{1; 2; 4; 6; 8\}$
p_U (Kreuzungsrate)	–	$\{0{,}2; 0{,}5; 0{,}8\}$
p_M (Mutationswahrscheinlichkeit)	0,05	$\{0{,}01; 0{,}03; 0{,}05; 0{,}07; 0{,}10\}$
n_E (Elitismus)	1	$\{0, 1, 2\}$

Tabelle 6.5: Parametervariation des Genetischen Algorithmus für L–824

Parameter	Initialwert	Variation
N (Populationsgröße)	1600	$\{400; 800; 1600; 3200; 6400\}$
s_2 (Auswahlalgorithmus)	Tournament	–
n_T (Tournamentgröße)	20	$\{10; 20; 40; 80; 160\}$
C (Kreuzungsschema)	N-Punkt	–
p_C (Kreuzungswahrscheinlichkeit)	1,0	$\{0{,}2; 0{,}4; 0{,}6; 0{,}8; 1{,}0\}$
n_C (Kreuzungspunkte)	6	$\{4; 6; 8; 10; 12\}$
p_M (Mutationswahrscheinlichkeit)	0,03	$\{0{,}01; 0{,}02; 0{,}03; 0{,}04; 0{,}05\}$
n_E (Elitismus)	0	–

Die konkreten Einflüsse der einzelnen Parameter auf das gesamte Planungsverfahren sind dieser Arbeit als Anhang angefügt. Die Ergebnisse der Parametrierung des Genetischen Algorithmus zeigen die Abbildungen A.1 bis A.9 (S–224), Abbildungen A.11 bis A.19 (M–424) und Abbildungen A.21 bis A.24 (L–824). Die Abbildungen A.10, A.20 und A.25 zeigen die Ergebnisse der in einem jeweils anschließend durchgeführten Feintuning gefundenen optimalen Parametereinstellungen des Genetischen Algorithmus.

Tabelle 6.6: Parametervariation des Ameisenalgorithmus für S–224

Parameter	Initialwert	Variation
N (Populationsgröße)	400	$\{200; 400; 800; 1600; 3200; 6400; 12800\}$
$\rho_{\min}$ (Minimalpheromonkonzentration)	0,05	$\{0{,}001; 0{,}005; 0{,}01; 0{,}05; 0{,}1\}$
$\rho_{\max}$ (Maximalpheromonkonzentration)	10	$\{1; 5; 10; 50; 100\}$
ρ_{init} (Initialpheromonkonzentration)	0,5	$\{0{,}0; 0{,}1; 0{,}5; 1; 5; 10\}$
ρ_+ (Pheromonadditionskoeffizient)	0,5	$\{0{,}3; 0{,}5; 0{,}7; 0{,}9; 1{,}1; 1{,}3; 1{,}5\}$
ϵ (Pheromonzerfallskoeffizient)	0,05	$\{0{,}01; 0{,}03; 0{,}05; 0{,}07; 0{,}09\}$

Tabelle 6.7: Parametervariation des Ameisenalgorithmus für M–424

Parameter	Initialwert	Variation
N (Populationsgröße)	3200	$\{200; 400; 800; 1600; 3200; 6400\}$
$\rho_{\min}$ (Minimalpheromonkonzentration)	0,1	$\{0{,}01; 0{,}05; 0{,}1; 0{,}5; 1{,}0\}$
$\rho_{\max}$ (Maximalpheromonkonzentration)	10	$\{5; 10; 50; 100; 500\}$
ρ_{init} (Initialpheromonkonzentration)	1	$\{0{,}0; 0{,}1; 0{,}5; 1; 5\}$
ρ_+ (Pheromonadditionskoeffizient)	0,5	$\{0{,}3; 0{,}5; 0{,}7; 0{,}9; 1{,}1\}$
ϵ (Pheromonzerfallskoeffizient)	0,05	$\{0{,}03; 0{,}05; 0{,}07; 0{,}09; 0{,}11\}$

Tabelle 6.8: Parametervariation des Ameisenalgorithmus für L–824

Parameter	Initialwert	Variation
N (Populationsgröße)	3200	$\{400; 800; 1600; 3200; 6400\}$
$\rho_{\min}$ (Minimalpheromonkonzentration)	0,1	$\{0{,}01; 0{,}02; 0{,}05; 0{,}1; 0{,}5\}$
$\rho_{\max}$ (Maximalpheromonkonzentration)	10	$\{5; 10; 50; 100; 500\}$
ρ_{init} (Initialpheromonkonzentration)	1	$\{0{,}0; 0{,}1; 0{,}5; 1; 5\}$
ρ_+ (Pheromonadditionskoeffizient)	0,5	$\{0{,}3; 0{,}5; 0{,}7; 0{,}9; 1{,}1\}$
ϵ (Pheromonzerfallskoeffizient)	0,05	$\{0{,}03; 0{,}05; 0{,}07; 0{,}09; 0{,}11\}$

Analog zeigen die Abbildungen A.26 bis A.31 (S–224), Abbildungen A.33 bis A.38 (M–424) und Abbildungen A.40 bis A.45 (S–224) die Ergebnisse der Parametrierung des Ameisenalgorithmus für die jeweiligen Probleme. In den Abbildungen A.32, A.39 und A.46 sind für den Ameisenalgorithmus die Ergebnisse des Feintuning der Parameter dargestellt.

Die Resultate der Parameterstudien sind in Tabelle 6.9 (Genetischer Algorithmus, S–224, M–424, L–824) und in Tabelle 6.10 (Ameisenalgorithmus, S–224, M–424, L–824) festgehalten. Für jedes der drei randomisierten Planungsprobleme zeigen die Tabellen die gefundene Parametrierung des Genetischen Algorithmus und des Ameisenalgorithmus.

Tabelle 6.9: Optimierte Parametrierung des Genetischen Algorithmus

Parameter	S–224	M–424	L–824
N (Populationsgröße)	3200	3200	3200
s_2 (Auswahlalgorithmus)	Tournament	Tournament	Tournament
n_T (Tournamentgröße)	20	10	20
p_C (Kreuzungswahrscheinlichkeit)	1,0	1,0	1,0
C (Kreuzungsschema)	N-Point	N-Point	N-Point
n_C (Kreuzungspunkte)	4	4	6
p_M (Mutationswahrscheinlichkeit)	0,05	0,05	0,03
n_E (Elitismus)	0	0	0

Tabelle 6.10: Optimierte Parametrierung des Ameisenalgorithmus

Parameter	S–224	M–424	L–824
N (Populationsgröße)	3200	1600	400
ρ_min (Minimale Pheromonkonzentration)	0,01	0,1	0,05
ρ_max (Maximale Pheromonkonzentration)	50	50	10
ρ_init (Initiale Pheromonkonzentration)	0,0	0,0	0,0
ρ_+ (Pheromonadditionskoeffizient)	1,1	0,5	0,9
ϵ (Pheromonverdampfungskoeffizient)	0,05	0,07	0,12

6.2.2 Vergleichen der Ergebnisse

Die mithilfe des Genetischen Algorithmus erzielbaren Ergebnisse werden mit den mithilfe des Ameisenalgorithmus erzielbaren Ergebnissen verglichen. Dazu zeigt Abbildung 6.2 den Vergleich für das Problem S–224, Abbildung 6.3 zeigt den Vergleich für das Problem M–424 und Abbildung 6.4 zeigt den Vergleich für das Problem L–824.

Insgesamt lässt sich festhalten, dass der Genetische Algorithmus im Rahmen der durchgeführten Untersuchungen bei wachsender Problemgröße wesentlich bessere Resultate erzielt. Allein bei dem kleinsten Problem (S–224) liefert der Ameisenalgorithmus ein im Durchschnitt um nur ca. 0,2 % besseres Ergebnis als der Genetische Algorithmus. Bei den weiteren Problemen (M–424 und L–824) erzielt der Genetische Algorithmus deutlich bessere Ergebnisse (durchschnittlich jeweils ca. 2 % besser).

Darüber hinaus lässt sich für die optimierte Parametrierung des Genetischen Algorithmus im Gegensatz zu der Parametrierung des Ameisenalgorithmus eine wesentlich geringere Abhängigkeit von den jeweiligen Planungsproblemen feststellen. Bezugnehmend auf Tabelle 6.9 unterscheiden sich die gefundenen Parametrierungen des Genetischen Algorithmus für die unterschiedlichen Planungsprobleme nur geringfügig. Dagegen unterscheidet sich die optimierte Parametrierung des Ameisenalgorithmus für die jeweiligen Probleme deutlich voneinander (Tabelle 6.10).

Der Genetische Algorithmus zeigt daher insgesamt ein wesentlich günstigeres Verhalten in Bezug auf eine optimierte Parametrierung. Es ist dementsprechend anzunehmen, dass der Einsatz des Genetischen Algorithmus für die Lösung weiterer, noch unbekannter Planungsprobleme mit der gefundenen Parametrierung bessere Ergebnisse als der Ameisenalgorithmus liefern wird.

Der Genetische Algorithmus weist gegenüber dem Ameisenalgorithmus allein aufgrund der Anzahl und der Kombinationsmöglichkeiten der Parameter eine wesentlich höhere Komplexität bezüglich der optimierten Parametrierung auf. Allerdings zeigt sich im Rahmen der durchgeführten Untersuchungen, dass für ein optimiertes Lösungsverhalten nur ein stark eingeschränkter Teil der Parameter in Frage kommt. So zeigen beispielsweise die Abbildungen A.5 und A.17, dass der Einsatz des Universal Crossover Operators als Rekombinationsoperator im Gegensatz zu dem N-Punkt-Crossover im Zusammenhang mit diesem Planungsverfahren grundsätzlich nicht zu bevorzugen ist.

Weiterhin zeigen die Abbildungen A.8, A.9, A.18 und A.19 einen ähnlichen Sachverhalt bezüglich der Auswahlalgorithmen Rouletteselektion und SUS. Diese sind unabhängig der verwendeten Selektionsalgorithmen (fitnessproportional oder rangbasiert) der Tournamentselektion deutlich unterlegen. Wird diese gewonnene Erkenntnis bei der Anwendung des Genetischen Algorithmus auf noch unbekannte Datensätze zusätzlich vorausgesetzt, wird die Komplexität der optimalen Parametrierung stark herabgesetzt.

Für die weiteren Untersuchungen im Rahmen dieser Arbeit wird entsprechend auf die Anwendung des Genetischen Algorithmus zurückgegriffen.

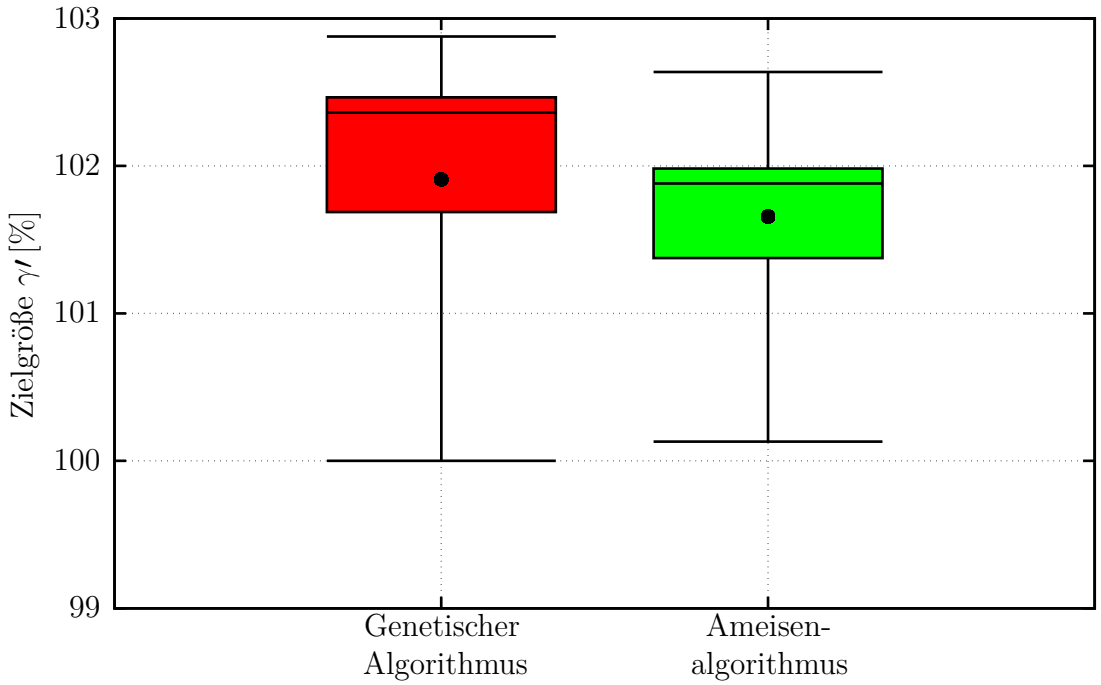


Abbildung 6.2: Vergleich: Genetischer Algorithmus und Ameisenalgorithmus (S–224)

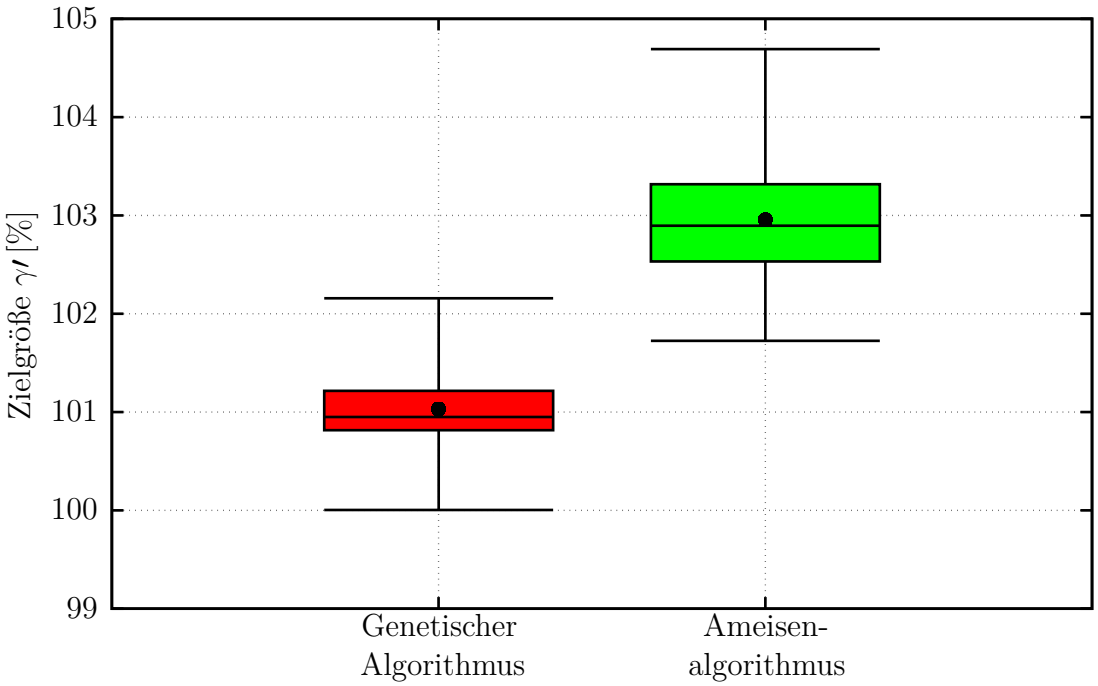


Abbildung 6.3: Vergleich: Genetischer Algorithmus und Ameisenalgorithmus (M–424)

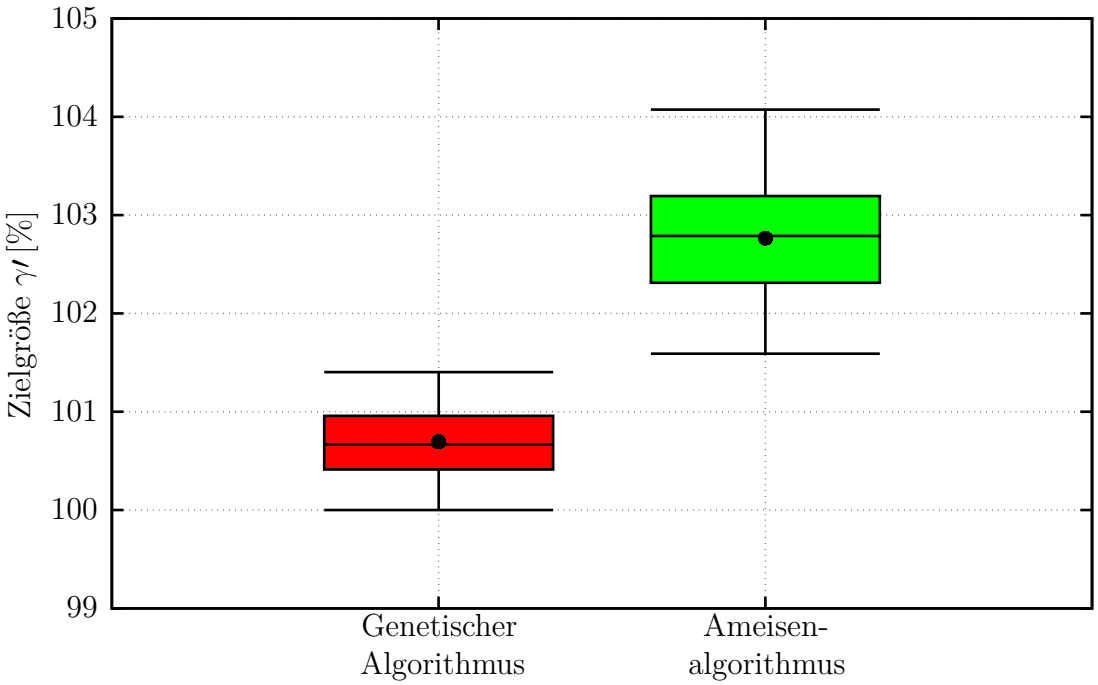


Abbildung 6.4: Vergleich: Genetischer Algorithmus und Ameisenalgorithmus (L–824)

6.3 Untersuchungen an den randomisierten Testdatensätzen

An den drei Testdatensätzen S–224, M–424 und L–824 werden im Folgenden weitere Untersuchungen durchgeführt. Als Optimierungsalgorithmus wird dabei jeweils der Genetische Algorithmus mit den entsprechenden Parametern verwendet.

6.3.1 Auswirkungen der zeitlichen Diskretisierung

Allen bisherigen Untersuchungen liegt ein Planungshorizont von $H = 48\,\mathrm{h}$ sowie eine Anzahl von $\theta = 8$ Planungsperioden zugrunde. Die Größe des durch den Optimierungsalgorithmus abtastbaren Suchraums (und damit die Komplexität des Suchproblems) ist direkt abhängig von der Größe θ (siehe Formel 4.80 und Abbildungen 4.4, 4.5, 5.7 und 5.8). Die (theoretisch) erzielbare Lösungsqualität wird entsprechend durch die Wahl von θ in Form eines Rechengitters beeinflusst (vgl. Kapitel 3).

Um unter den gegebenen Randbedingungen und Ressourcen (vier Minuten Rechenzeit auf dem Rechencluster) jeweils das bestmögliche Ergebnis erzielen zu können, soll in einer Untersuchung festgestellt werden, welcher Wert für θ jeweils die besten Resultate liefert. Für jeden der drei Testdatensätze werden die Werte $\theta \in \{4; 6; 8; 10; 12; 14; 16; 18; 20\}$ systematisch variiert. Sämtliche weitere Parameter bleiben auch bei dieser Studie unverändert.

Bezugnehmend auf Abbildung 3.2 und die in Kapitel 3 ausgeführte Erläuterung ist für jeden Datensatz die Existenz eines optimalen Wertes von θ zu erwarten. Die Auswirkungen der unterschiedlichen Werte für die zeitliche Diskretisierung (also die unterschiedlichen Anzahlen θ von Planungsperioden) zeigen Abbildung 6.5 (S–224), Abbildung 6.6 (M–424) und Abbildung 6.7 (L–824).

Die Studie zeigt, dass bei dem Datensatz S–224 mit einem Wert von $\theta = 16$ die besten Ergebnisse erzeugt werden. Sowohl das in dieser Studie bestmögliche Ergebnis ($\gamma\prime = 100\,\%$) als auch der Durchschnittswert ($\gamma\prime \approx 100{,}5\,\%$) ist für $\theta = 16$ im Vergleich mit den weiteren Werten von θ unübertroffen. Für die Datensätze M–424 und L–824 werden mit Werten von $\theta = 8$ und $\theta = 10$ die besten Ergebnisse erzielt.

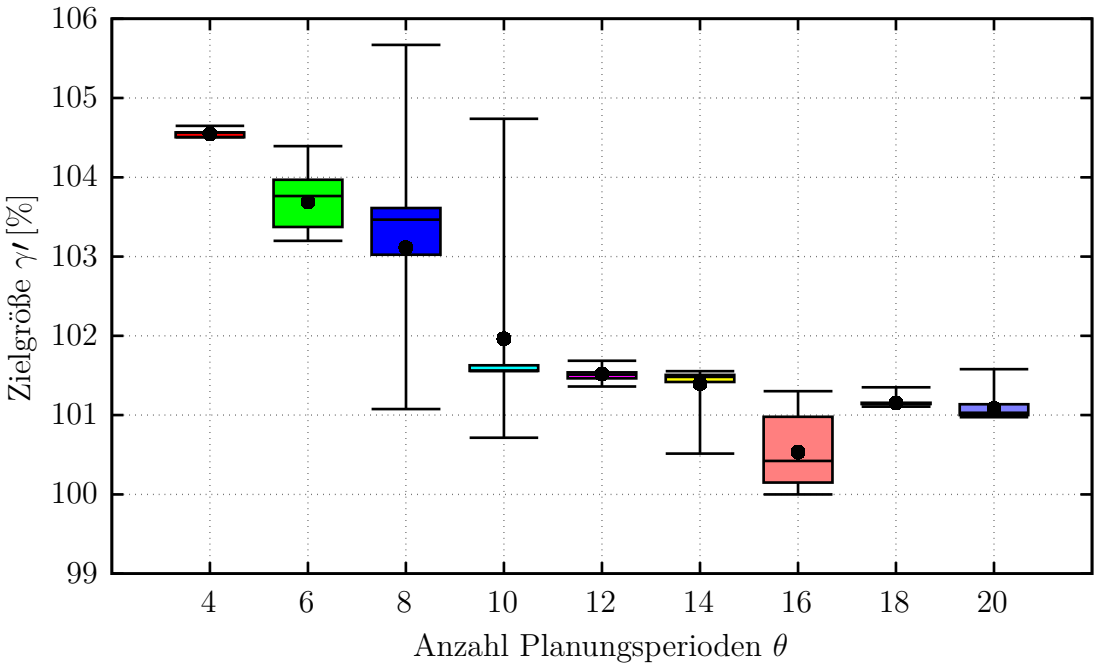


Abbildung 6.5: Vergleich unterschiedlicher Planungsperioden für S–224

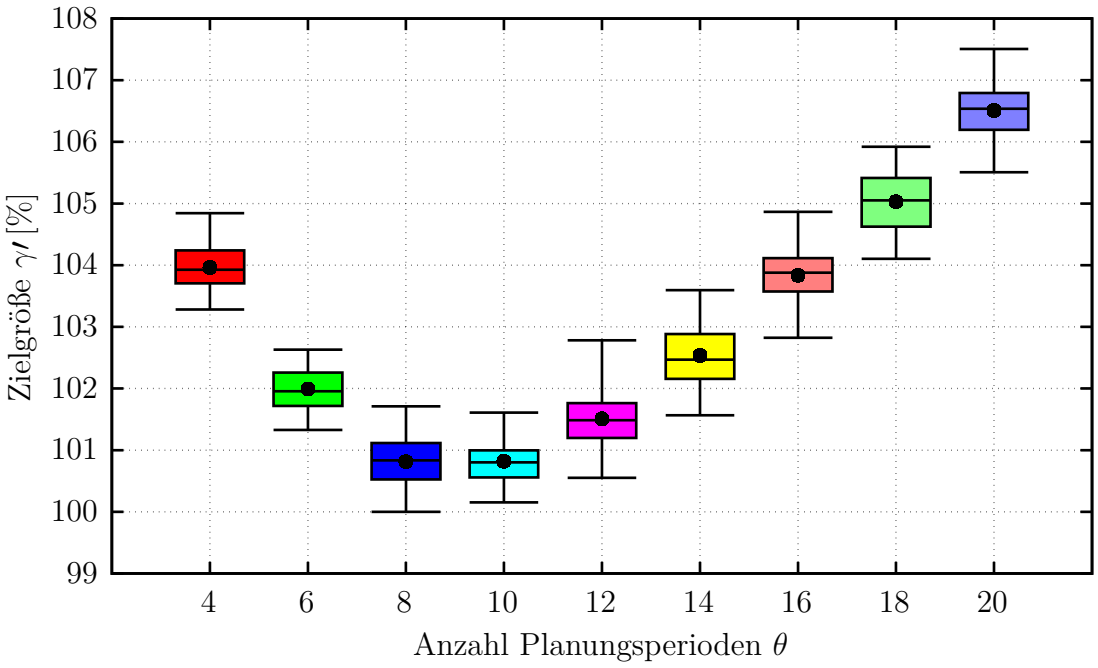


Abbildung 6.6: Vergleich unterschiedlicher Planungsperioden für M–424

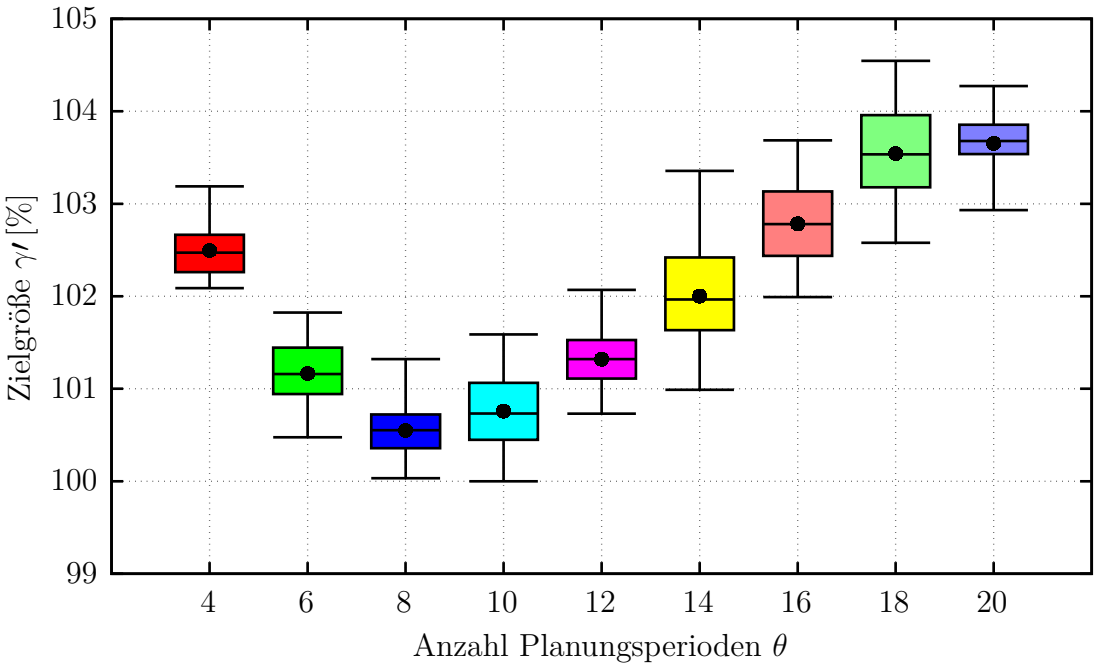


Abbildung 6.7: Vergleich unterschiedlicher Planungsperioden für L–824

6.3.2 Auswirkungen unterschiedlicher Planungsregeln

Für die Durchführung der Ablaufplanung stellt der Lösungsgenerator für jede Produktionsstufe den Einsatz unterschiedlicher Planungsregeln zur Verfügung (siehe Unterabschnitt 4.3.3). Allen bisherigen Untersuchungen liegt für jede Produktionsstufe die Regel **S/OPN (geringste Schlupfzeit zuerst)** zugrunde. Im Folgenden wird daher der Einfluss der unterschiedlichen Planungsregeln auf das Planungsergebnis untersucht.

Für jeden der drei Testdatensätze (S–224, M–424 und L–824) wird entsprechend eine vergleichende Studie durchgeführt. Für die Studien werden jeweils alle Planungsregeln systematisch ausprobiert. Für alle Produktionsstufen des jeweiligen Planungsproblems werden dabei immer die gleichen Planungsregeln angewendet. Das Kombinieren von unterschiedlichen Planungsregeln für die einzelnen Produktionsstufen würde den Rahmen dieser Arbeit überschreiten.

An dieser Stelle wird darauf hingewiesen, dass die Beurteilung der Ergebnisse nur auf Basis der in Abschnitt 6.1 definierten Gewichtungen der Zielgrößen (Formeln 6.4 bis 6.7) stattfinden kann. Es ist naheliegend, dass die Variationen der Planungsregeln bei einer anderen Zielgrößengewichtung abweichende Einflüsse auf das Planungsergebnis haben können. Auf weitergehende Untersuchungen in Bezug auf den Einfluss der Planungsregeln auf unterschiedliche Zielgrößengewichtungen muss an dieser Stelle verzichtet werden, da sie den Umfang der vorliegenden Arbeit überschreiten würden.

Die Einflüsse der unterschiedlichen Planungsregeln auf die Planungsergebnisse sind in Abbildung 6.8 (für S–224), Abbildung 6.9 (für M–424) und Abbildung 6.10 (für L–824)

dargestellt. Die Ergebnisse der einzelnen Studien zeigen insgesamt nur wenig Abhängigkeiten in Bezug auf das jeweilige Planungsproblem. So sind Trends und Tendenzen identifizierbar, welche Planungsregeln unabhängig von den zugrunde liegenden Planungsproblemen bessere Ergebnisse in Bezug auf die gewählten Zielgrößengewichte versprechen.

Die besten Ergebnisse werden bei allen Datensätzen mithilfe der EDD-Regel (frühster Termin zuerst) erzielt. Bei M–424 und L–824 wird auch das jeweils insgesamt beste Ergebnis ($\gamma\prime = 100\,\%$) mithilfe der EDD-Regel gefunden. Bei S–224 wird das insgesamt beste Ergebnis hingegen mit der S/OPN-Regel gefunden.

Die S/OPN-Regel liefert in Bezug auf die durchschnittlich erzielbaren Ergebnisse bei allen Datensätzen ebenfalls sehr gute Resultate. Die Durchschnittswerte von den mithilfe der S/OPN-Regel ermittelten Ergebnisse weichen bei allen Problemen nur um weniger als ca. 0,5 % von den Durchschnittswerten der EDD-Regel ab.

Der Einsatz der LPT-Regel (längste Operationszeit zuerst) liefert bei allen Problemen die schlechtesten Ergebnisse. Die Regeln SPT (kürzeste Operationszeit zuerst), FIFO (früheste Ankunftszeit zuerst) und LIFO (späteste Ankunftszeit zuerst) liegen in Bezug auf die Durchschnittsergebnisse bei allen Planungsproblemen im mittleren Bereich.

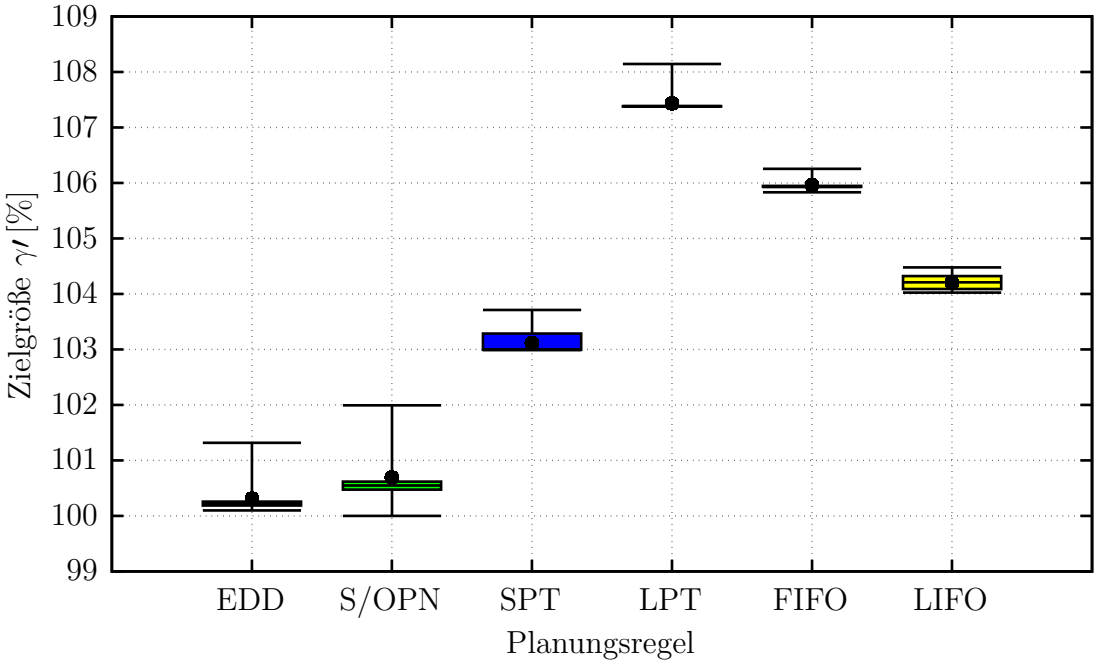


Abbildung 6.8: Vergleich der unterschiedlichen Planungsregeln für S–224

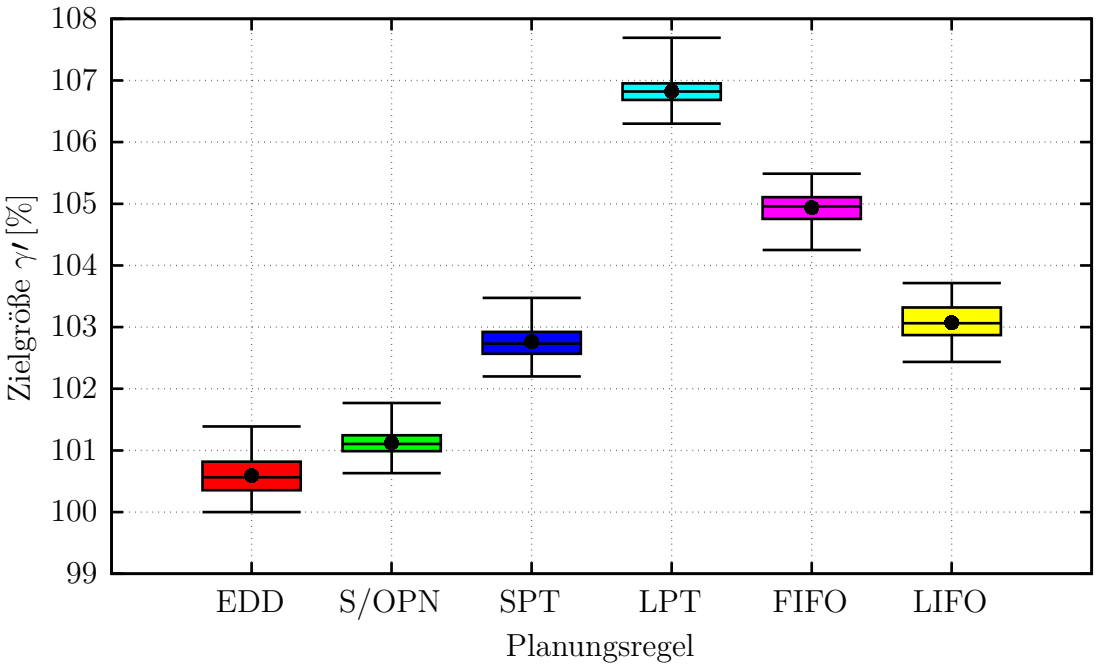


Abbildung 6.9: Vergleich der unterschiedlichen Planungsregeln für M–424

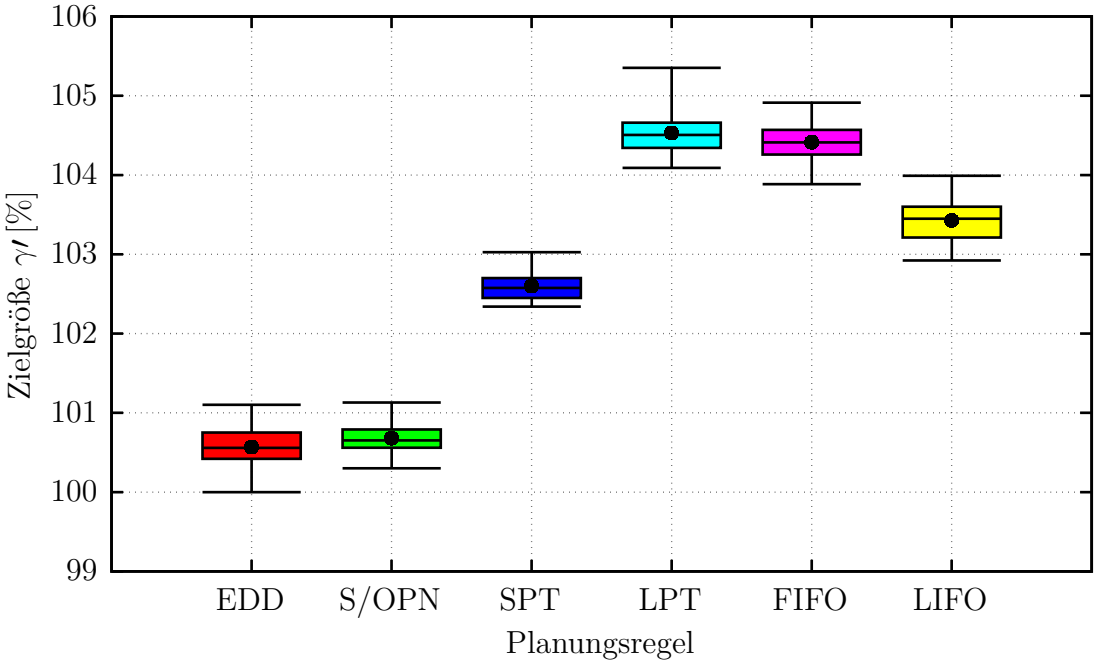


Abbildung 6.10: Vergleich der unterschiedlichen Planungsregeln für L–824

6.3.3 Analyse des Laufzeitverhaltens

Die Abbildungen 6.11 und 6.12 zeigen das Laufzeitverhalten des Planungsverfahrens basierend auf dem Genetischen Algorithmus jeweils auf Basis der absoluten Laufzeit von 4 min (Abbildung 6.11) und auf Basis der durchgeführten Iterationen (Abbildung 6.12). Beide Diagramme zeigen jeweils die Optimierung der Datensätze S–224, M–424 und L–824.

Das Abbruchkriterium des jeweiligen Optimierungsprozesses ist bei allen Beispielen (wie bei allen weiteren Untersuchungen in dieser Arbeit) das Erreichen der Zeitschwelle von 4 min. Die verwendeten Parameter des Genetischen Algorithmus sind jeweils der Tabelle 6.9 zu entnehmen. In jedem Iterationsschritt aus Abbildungen 6.11 und 6.12 werden folglich $N = 3200$ (Populationsgröße) einzelne Planungen durch den Lösungsgenerator durchgeführt.

Bei der Optimierung des Datensatzes S–224 werden innerhalb der vorgegebenen Zeit von 4 min ca. 640 Iterationen durchgeführt. Dies resultiert in einer Anzahl von insgesamt ca. $640 \cdot 3200 = 2\,048\,000$ Planungen, die dabei innerhalb von 4 min durchgeführt werden. Bei der Optimierung des Datensatzes M–424 werden innerhalb Zeit dagegen ca. 280 Iterationen durchgeführt. Dies resultiert in einer Anzahl von insgesamt ca. $280 \cdot 3200 = 896\,000$ Planungen. Bei der Optimierung des Datensatzes L–824 werden innerhalb der gleichen Zeit hingegen nur ca. 97 Iterationen durchgeführt. Dies resultiert in einer Anzahl von insgesamt ca. $97 \cdot 3200 = 310\,400$ Planungen.

Die Rechenzeiten, welche für die Berechnung und Auswertung eines einzelnen Ablaufplanes durch den Lösungsgenerator im Rahmen des jeweiligen Optimierungsprozesses benötigt werden, lassen sich dadurch ermitteln und sind in den Formeln 6.8 bis 6.10 gegeben.

$$T_{\text{S–224}} \approx \frac{4\,\text{min}}{640 \cdot 3200} \approx \frac{4\,\text{min}}{2\,048\,000} \approx 0{,}12\,\text{ms} \tag{6.8}$$

$$T_{\text{M–424}} \approx \frac{4\,\text{min}}{280 \cdot 3200} \approx \frac{4\,\text{min}}{896\,000} \approx 0{,}27\,\text{ms} \tag{6.9}$$

$$T_{\text{L–824}} \approx \frac{4\,\text{min}}{97 \cdot 3200} \approx \frac{4\,\text{min}}{310\,400} \approx 0{,}77\,\text{ms} \tag{6.10}$$

Die Rechenzeit, die für die Erstellung eines Ablaufplanes für den Datensatz S–224 benötigt wird, beträgt folglich ca. 0,12 ms (Formel 6.8). Für die Erstellung eines Ablaufplanes auf Basis des Datensatzes M–424 werden ca. 0,27 ms Rechenzeit benötigt (Formel 6.9). Für die Berechnung eines Ablaufplans für den Datensatz L–824 werden ca. 0,77 ms (Formel 6.10) benötigt.

Neben dem reinen Unterschied in der jeweiligen Laufzeit zeigen die Abbildungen 6.11 und 6.12 wesentliche Unterschiede in dem jeweiligen Lösungsverhalten. Die Optimierung des Datensatzes L–824 erstreckt sich über den gesamten Zeitraum von 4 min und nimmt dabei ca. 95 Iterationsschritte in Anspruch. Die Optimierung des Datensatzes M–424 zeigt dagegen nach ca. 60 Iterationsschritten ein Konvergenzverhalten. Allerdings wird für die Ausschöpfung von weiteren Potenzialen (Iterationsschritte 170 und 270) ebenfalls der gesamte Zeitraum von 4 min benötigt.

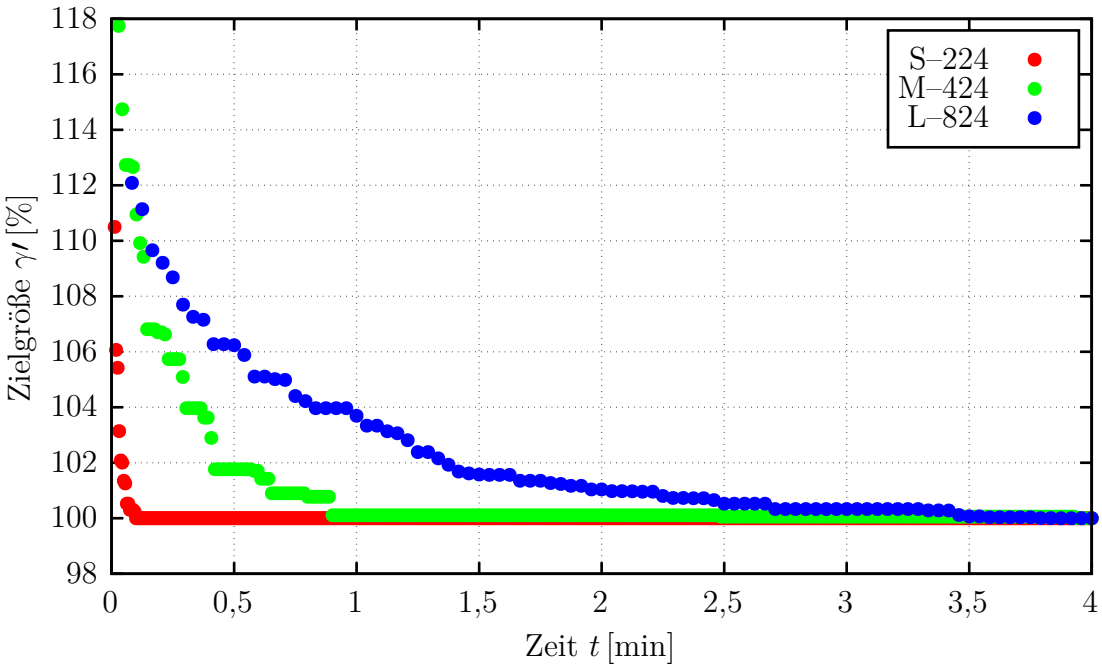


Abbildung 6.11: Laufzeitverhalten des Genetischen Algorithmus (zeitbasiert)

Der Optimierungsprozess auf Basis des Datensatzes S–224 ist hingegen nach nur ca. 20 Iterationsschritten bereits konvergiert. In den weiteren 620 Iterationsschritten findet entsprechend keine weitere Verbesserung der Zielgröße γ' mehr statt.

Unter den gegebenen Randbedingungen (Begrenzung der Optimierungszeit auf 4 min) und der zur Verfügung stehenden Hardware zeigt das Planungsverfahren bei der Optimierung des Datensatzes S–224 entsprechend noch ungenutzte Reserven. Bei dem Einsatz des Planungsverfahrens für reale Systeme der Größe S–224 könnte die Laufzeit folglich bei ansonsten gleichbleibenden Randbedingungen von 4 min auf z.B. 1 min reduziert werden. Eine Beeinflussung des Ergebnisses ist dabei in Bezug auf Abbildungen 6.11 und 6.12 nicht zu erwarten.

Auch bei der Optimierung von Datensätzen, die der Größe M–424 entsprechen, liegt die Vermutung nahe, dass die Rechenzeit auf beispielsweise 2 min (also auf ca. 140 Iterationsschritte) halbiert werden könnte. In diesem Fall wäre zwar damit zu rechnen, dass Optimierungspotenziale ungenutzt blieben, diese befinden sich allerdings in einer für die Praxis nicht relevanten Größenordnung (deutlich unter einem Prozent).

Bei der Optimierung des Datensatzes L–824 sind hingegen keine weiteren Reserven bezüglich der Laufzeit zu erwarten. Die Laufzeit von 4 min kann daher in Bezug auf Abbildungen 6.11 und 6.12 für Datensätze der Größe L–824 empfohlen werden.

Bei der Optimierung von Datensätzen, welche die Größe von L–824 überschreiten, ist mit einem weiteren Anstieg der benötigten Rechenzeit (vgl. Abbildungen 6.11 und 6.12) zu rechnen. Innerhalb einer vorgegebenen Optimierungszeit ist für größere Datensätze

entsprechend mit weniger Iterationsschritten zu rechnen. Für die Anwendung des Planungsverfahrens auf größere Datensätze wird daher (bei den gleichen Randbedingungen) eine Verlängerung der Rechenzeit empfohlen.

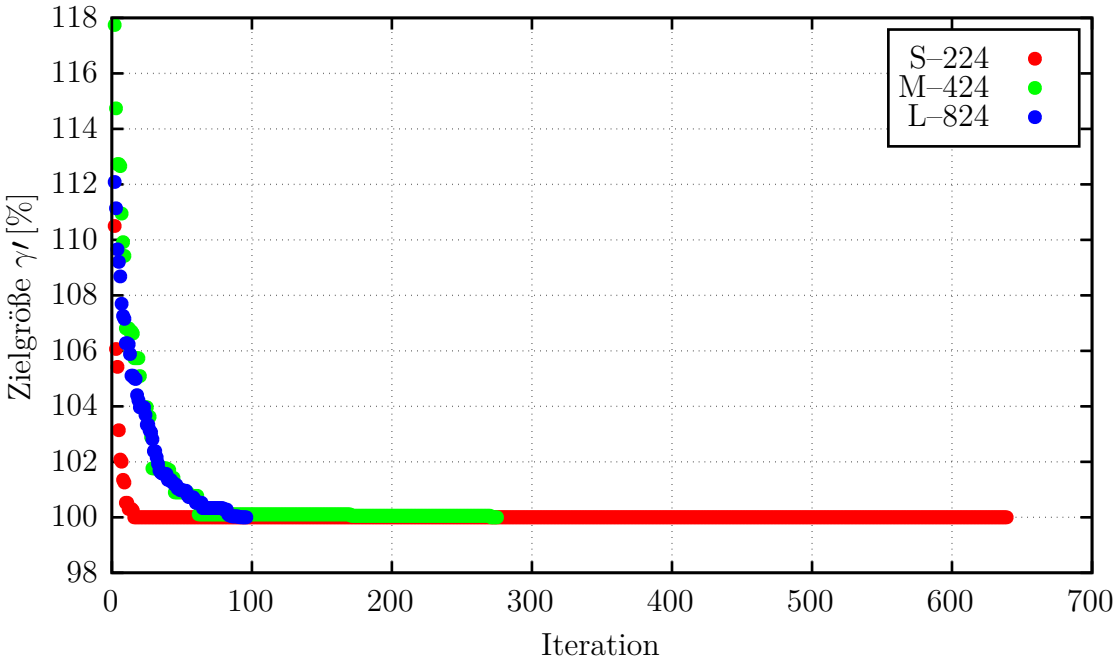


Abbildung 6.12: Laufzeitverhalten des Genetischen Algorithmus (iterationsbasiert)

6.3.4 Beurteilung der Planungsergebnisse

Die visuelle Beurteilung der Planungsergebnisse erfolgt anhand eines konkreten Ablaufplans auf Basis des Datensatzes L–824. In Abbildung 6.13 ist dafür der im Rahmen dieser Arbeit beste gefundene Ablaufplan auf Basis des Datensatzes L–824 dargestellt.

In Bezug auf Abbildung 6.7 liegt bei dem Ablaufplan eine Diskretisierung von $\theta = 10$ Planungsperioden zugrunde. Die Wahl der Planungsregel wird mit den in Abbildung 6.10 dargestellten Ergebnissen begründet. So liegt dem Ablaufplan aus Abbildung 6.13 die EDD-Regel zugrunde.

In Formel 6.11 ist die dem Ablaufplan aus Abbildung 6.13 zugehörige Entscheidungsmatrix dargestellt. Um die Zusammenhänge zwischen den einzelnen Produktionsstufen zu verdeutlichen sind beispielhaft die drei Fertigungsaufträge J_{178}, J_{78}, J_{216}, J_{88}, J_{111}, J_{397} und J_{31} farbig hervorgehoben. So lässt sich Abbildung 6.13 beispielsweise entnehmen, dass der Job J_{178} zunächst in der ersten Produktionsstufe K_1 an der Maschine $M_{1,2}$ eingeplant. Eine Bearbeitung in der zweiten Produktionsstufe K_2 ist laut dem Ablaufplan für J_{178} nicht vorgesehen ($\sigma_{2,178} = 0$), sodass die Bearbeitung von J_{178} in der dritten Stufe K_3 an der Maschine $M_{3,2}$ fortgesetzt wird.

Auch die Produktionsstufe K_4 und K_5 werden von J_{178} übersprungen ($\sigma_{4,178} = 0$ und $\sigma_{5,178} = 0$), daher findet die weitere Bearbeitung von J_{178} in den Produktionsstufen K_6

(Maschine $M_{6,1}$) und K_7 (Maschine $M_{7,2}$) statt. Der letzte Bearbeitungsschritt von J_{178} erfolgt schließlich an der Maschine $M_{8,2}$ (Stufe K_2).

Die Bearbeitung des Jobs J_{178} erstreckt sich über die drei Planungsperioden t_5 bis t_8. Die Wartezeiten zwischen den einzelnen Bearbeitungsschritten fallen relativ gering aus, sodass beispielsweise in Bezug auf die Durchlaufzeit von J_{178} durch die visuelle Beurteilung des Ablaufplans keine weitergehenden Optimierungspotenziale festgestellt werden.

In der Produktionsstufe K_5 sind an den beiden Maschinen $M_{5,1}$ und $M_{5,2}$ Lücken in Form von Leerlaufzeiten identifizierbar. Diese sind zurückzuführen auf die gewählten Bereiche der zufällig generierten Bearbeitungszeiten $p_{k,i,j}$ (siehe Tabelle 6.2). An den vorgelargerten Maschinen $M_{2,1}$, $M_{2,2}$, $M_{3,1}$ und $M_{4,2}$ sind die durchschnittlich benötigten Bearbeitungszeiten signifikant höher gewählt als an den Maschinen $M_{5,1}$ und $M_{5,2}$. Im Zusammenhang mit den zufällig generierten Werten von $\sigma_{k,i,j}$ ergibt sich für die beiden Maschinen $M_{5,1}$ und $M_{5,2}$ daher eine geringere Auslastung als für die weiteren Maschinen. Der Ursprung dieses Phänomens liegt folglich in der Struktur der zufällig erzeugten Daten und lässt sich nicht auf das Planungsverfahren zurückzuführen.

$$\delta_{k,i,\tau}\,(\text{L–824}) = \begin{pmatrix} 1 & 3 & 3 & 2 & 4 & 4 & 2 & 2 & 2 & 1 \\ 1 & 1 & 2 & 1 & 2 & 2 & 2 & 2 & 2 & 1 \\ 3 & 3 & 3 & 3 & 3 & 3 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 4 & 4 & 4 & 4 & 4 & 4 & 1 & 1 \\ 2 & 2 & 2 & 3 & 3 & 3 & 1 & 1 & 2 & 3 \\ 3 & 4 & 4 & 4 & 2 & 2 & 3 & 4 & 2 & 4 \\ 2 & 2 & 4 & 4 & 3 & 3 & 3 & 1 & 1 & 1 \\ 2 & 4 & 3 & 4 & 4 & 2 & 1 & 4 & 2 & 1 \\ 2 & 2 & 2 & 3 & 2 & 3 & 2 & 1 & 2 & 1 \\ 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 4 & 3 & 3 & 2 & 2 & 3 & 1 & 1 \\ 3 & 2 & 1 & 3 & 2 & 4 & 1 & 3 & 4 & 3 \\ 1 & 3 & 1 & 3 & 1 & 1 & 2 & 4 & 1 & 2 \\ 1 & 2 & 2 & 2 & 1 & 3 & 1 & 1 & 4 & 3 \end{pmatrix} \tag{6.11}$$

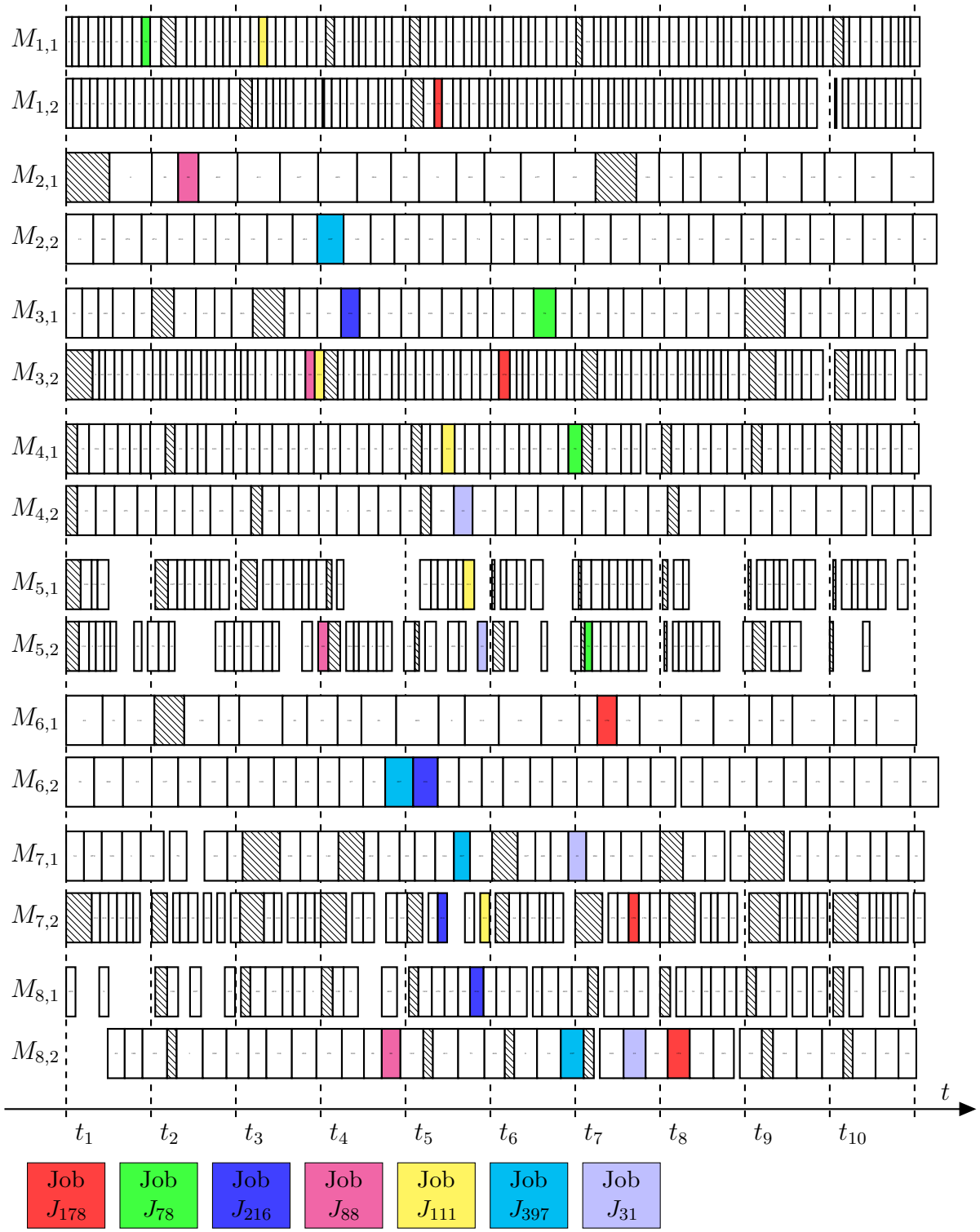


Abbildung 6.13: Bester gefundener Ablaufplan für L–824 (EDD-Regel und $\theta = 10$)

6.4 Fallstudie Holzverarbeitung als konkretes Industriebeispiel

Im Folgenden wird die Anwendung des ereignisdiskreten Planungsverfahrens auf die operative Produktionsplanung und -steuerung eines deutschen, mittelständischen Betriebs dargestellt.

6.4.1 Beschreibung des Produktionsprozesses

Bei der im Rahmen dieser Arbeit durchgeführten Fallstudie handelt es sich um einen Betrieb aus der industriellen Holzverarbeitung (siehe auch Stalinski et al., 2017a,b). In der Produktion werden hochautomatisiert Holzwerkstoffplatten zugeschnitten und mit unterschiedlichen Materialien beschichtet. Die Produkte finden Anwendung

- in der Weiterverarbeitung durch die Möbelindustrie,
- in der Weiterverarbeitung durch die Fußbodenindustrie,
- im Innenausbau,
- im Kabinenausbau von Schiffen,
- in Objektausbauten,
- usw.

Bei den Produkten handelt es sich immer um rechteckige Holzwerkstoffplatten. Je nach Bedarf der Kunden werden an den Platten die folgenden Bearbeitungsschritte durchgeführt:

- Zuschnitt der Holzwerkstoffplatten
- Verleimung und Verpressung von Beschichtungsmaterialien auf die Holzwerkstoffplatten
- Erst Zuschnitt und anschließende Beschichtung durch Verleimung und Verpressung

Für den Zuschnitt stehen zwei Sägeanlagen zur Verfügung ($M_{1,1}$ und $M_{1,2}$, Abbildung 6.14). Die Sägen unterscheiden sich im Wesentlichen durch ihre Größe, Geschwindigkeit und Handhabung. So steht eine große, vollautomatische Säge für die schnelle Produktion von hohen Losgrößen zur Verfügung (siehe Abbildung 6.15). Für kleinere Abmessungen und niedrigere Losgrößen steht eine kleinere Säge zur Verfügung.

Das Beschichten der Holzwerkstoffplatten kann an vier unterschiedlichen Pressenanlagen erfolgen ($M_{2,1}$, $M_{2,2}$, $M_{2,3}$ und $M_{2,4}$, Abbildung 6.14). Die Maschinen unterscheiden sich sowohl in Bezug auf ihre Form und Größe als auch bezüglich der jeweiligen Prozesstechnologien. Für das Beschichten werden die Trägerplatten zunächst über Leimrollen beleimt, um anschließend die Beschichtungsstoffe in der Presse unter Druck und Wärmezufuhr

aufzubringen. Nach dem Verpressen verlassen die Platten die Maschine über Förderbänder (siehe Abbildung 6.16).

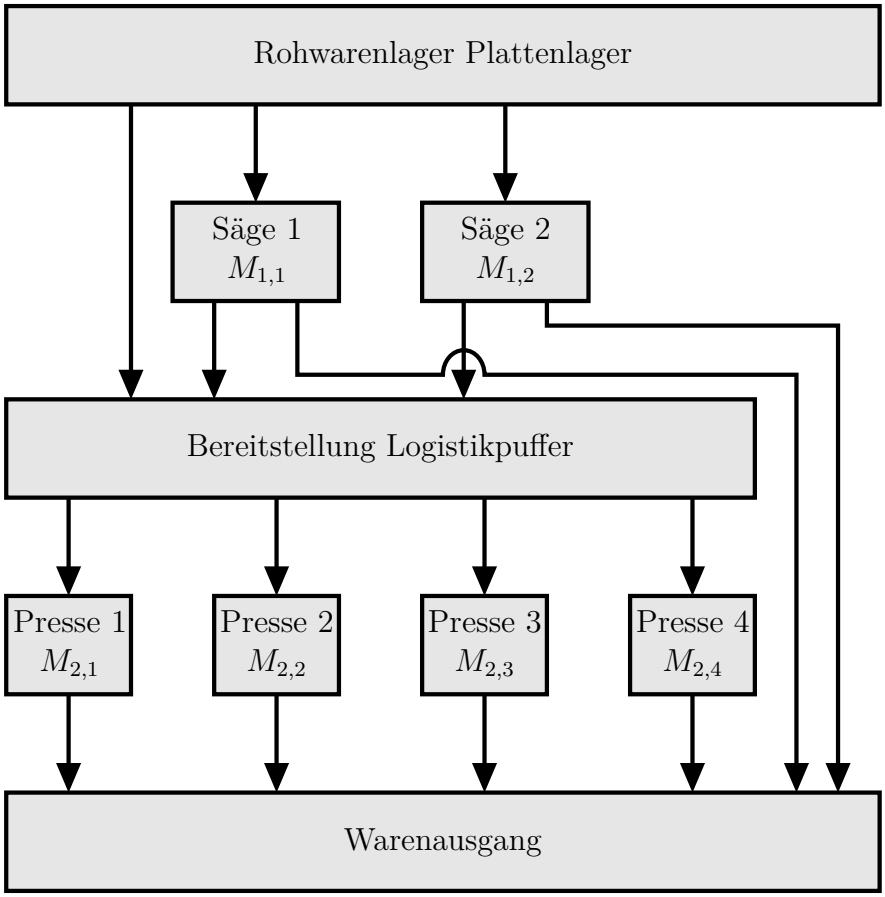


Abbildung 6.14: Schema des Produktionssystems der Fallstudie

Gängige Beschichtungsstoffe sind Furniere aus Echtholz (z.B. Eiche, Buche, Ahorn, Nuss, Kirsche), Dekore (z.B. Uni, Holzimitate), Folien (z.B. Dekor, Gegenzug, Blindlagen), Vinyl, PP und Kork (für Fußböden), HPL und viele weitere. Für die Trägerplatten kommen in der Regel die Materialien Span, MDF und HDF zum Einsatz. Auch Echtholzplatten, Tischlerplatten und Platten aus Leimholz werden verarbeitet.

Jedes hergestellte Produkt hat immer einen direkten Kundenbezug, so dass zu jeder Platte in der Produktion ein Fertigungsauftrag mit dem entsprechenden Kundenbezug gehört (Auftragsfertigung). Ohne die Bestellung eines Kunden wird entsprechend kein Produkt hergestellt.

Jeder Fertigungsauftrag liegt in dem ERP-System (Enterprise-Resource-Planning) des Unternehmens mit der entsprechenden Stückliste, dem Arbeitsplan sowie den weiteren Daten vor. Während der Produktion (z.B. nach dem Sägen oder vor dem Verpressen) liegt jeder Fertigungsauftrag in Form von Plattenstapeln vor. Der Transport und die Lagerung der Stapel erfolgt vollautomatisch auf motorisierten Rollenbahnen und Querförderwagen

(siehe Abbildung 6.17). Jeder der Plattenstapel ist für die automatisierte Abwicklung der logistischen Prozesse mit jeweils einem RFID-Tag ausgestattet.

Die Produktion ist an den für die Steuerung der Prozesse relevanten Schnittstellen mit der RFID-Technik ausgestattet. Hinter den jeweiligen Maschinen und vor den Logistikpuffern befinden sich RFID-Gates. Beim Überfahren dieser unsichtbaren Schranken durch die Fertigungsaufträge werden im ERP-System automatisch Bestandsmeldungen und Buchungen durchgeführt. Die Informationen über die Fortschritte und Positionen der Fertigungsaufträge, verbrauchte Mengen sowie die Lagerbestände liegen entsprechend in hohem Maß an Qualität und Quantität vor und bilden eine wichtige Grundlage für die Ablaufplanung.

In den meisten Fällen gelangen die Bestellungen der Kunden über eine softwarebasierte Schnittstelle (EDI) automatisch in das ERP-System des Betriebes. Die Arbeitsvorbereitung hat in diesem Fall nur noch eine kontrollierende Rolle. Aus den Bestellungen werden im System automatisch Fertigungsaufträge generiert.

Die Erstellung der Schnittpläne für den Zuschnitt auf den Sägen erfolgt durch einen Mitarbeiter der Arbeitsvorbereitung. Über digitale Schnittstellen zwischen dem ERP-System und einer Software für die Verschnittoptimierung werden die entsprechenden Auftragsdaten (Mengen, Materialien, Abmessungen, Termine, Verschnitt) zwischen den jeweiligen Systemen ausgetauscht. Unterschiedliche Fertigungsaufträge von verschiedenen Kunden werden dabei häufig in einem Schnittplan zusammengefasst.

Abbildung 6.15: Vollautomatische Sägeanlage mit Peripherie
(Quelle: `http://www.hobb.de`)

Abbildung 6.16: Vollautomatische Pressenanlage mit Auslaufband (Quelle: `http://www.hobb.de`)

Abbildung 6.17: Vollautomatisches Lagersystem mit motorisierten Rollenbahnen und RFID (Quelle: `http://www.hobb.de`)

6.4.2 Anwendung des Planungsverfahrens

Die Produktion besteht im Wesentlichen aus den beiden Produktionsstufen K_1 (Sägen) und K_2 (Pressen) (Formel 6.12). Es stehen jeweils zwei Sägen in der ersten Produktionsstufe (Formel 6.13) und vier Pressen in der zweiten Produktionsstufe (Formel 6.14) zur Verfügung.

$$c = 2 \tag{6.12}$$

$$m_1 = 2 \tag{6.13}$$

$$m_2 = 4 \tag{6.14}$$

Für die vorliegende Produktion lässt sich die in Formel 2.17 formulierte Problemklasse gemäß Formel 6.15 um die genaue Angabe der Maschinenumgebung spezifizieren.

$$\alpha = FF2\left((R2)^{(1)}, (R4)^{(2)}\right) \tag{6.15}$$

Bei der Erstellung der Schnittpläne für den Zuschnitt der Fertigungsaufträge wird jeder Fertigungsauftrag jeweils mit einer der beiden zur Verfügung stehenden Sägen verknüpft. Technologisch bedingt ist es nicht möglich, für einen Fertigungsauftrag nachträglich durch das Planungsverfahren eine andere Säge auszuwählen. In dem Optimierungsmodell müssen bzw. dürfen entsprechend alle Fertigungsaufträge der ersten Stufe immer nur mit jeweils einer Maschine kompatibel sein (Formel 6.16).

$$\sum_{i=1}^{2} \lambda_{1,i,j} = 1 \qquad \forall j \tag{6.16}$$

Die Sägeanlagen werden über einen Gabelstapler mit den jeweiligen Materialien versorgt. Nach dem Auflegen eines Plattenstapels durch den Gabelstapler kann direkt mit dem Zuschnitt begonnen werden. Dabei treten keine reihenfolgeabhängigen Rüstzeiten auf (Formel 6.17).

$$\sum_{f=1}^{o_{1,i}} \phi_{1,i,f,f'} = 0 \qquad \forall i, f' \tag{6.17}$$

Das Auftrennen eines Plattenstapels (eines Sägelaufs) zu mehreren Plattenstapeln unterschiedlicher Fertigungsaufträge (divergierender Materialfluss, siehe Unterabschnitt 2.2.2 c)) wird in der ersten Stufe durch die Bündelung der Jobs zu Batches berücksichtigt (Formel 6.18).

$$q_1 > 0 \tag{6.18}$$

Bei dem Start eines Jobs in der ersten Stufe werden entsprechend alle zu dem Schnittplan gehörenden Jobs mitgestartet. Diese Jobs können unterschiedliche Kundenbezüge haben und nach dem Zuschnitt unterschiedlich weiterverarbeitet werden.

Die Fertigungsaufträge eines zusammenhängenden Schnittplans können z.B. teilweise für die Weiterverarbeitung an den Pressen vorgesehen sein. In demselben Schnittplan können sich allerdings auch Fertigungsaufträge befinden, die ohne eine Weiterverarbeitung an den Pressen in den Warenausgang gehen. Es kann weiterhin nicht garantiert werden, dass Fertigungsaufträge, die nach dem gemeinsamen Zuschnitt an den Pressen beschichtet werden müssen, auf denselben Pressen (bzw. ohne Rüstzeit) bearbeitet werden können.

In Abhängigkeit der unterschiedlichen Trägerplatten, Beschichtungsmaterialien sowie der Maschinen können die Fertigungsaufträge an den Pressen mit unterschiedlichen Leimsystemen und Prozesstemperaturen verarbeitet werden. Weiterhin erfordert die Produktion mancher Fertigungsaufträge eine anschließende Reinigung der entsprechenden Maschine. Rüstzeiten treten in der zweiten Produktionsstufe entsprechend im Wesentlichen durch das Aufheizen und Abkühlen der Maschinen, das Wechseln des Leimsystems sowie durch Reinigungsarbeiten auf.

Entscheidend ist dabei, dass die Fertigungsaufträge häufig mit unterschiedlichen Leimsystemen verarbeitet werden dürfen. Durch das Planungsverfahren muss also festgelegt werden, mit welchem Leim die jeweiligen Jobs bearbeitet werden sollen. Auf Basis der unterschiedlichen Prozesstemperaturen, der unterschiedlichen Leime sowie der Reinigungsarbeiten werden für die zweite Produktionsstufe entsprechend die Auftragsfamilien formuliert (Formeln 6.19 bis 6.22).

$$o_{2,1} = 6 \tag{6.19}$$
$$o_{2,2} = 5 \tag{6.20}$$
$$o_{2,3} = 1 \tag{6.21}$$
$$o_{2,4} = 1 \tag{6.22}$$

Für die beiden Pressen $M_{2,1}$ und $M_{2,2}$ ergeben sich folglich 6 bzw. 5 Auftragsfamilien (Formeln 6.19 und 6.20). An den beiden Pressen $M_{2,3}$ und $2,4$ werden technologiebedingt keine Umrüstarbeiten vorgenommen. Es steht jeweils nur ein Leimsystem zur Verfügung und es wird jeweils nur mit einer Temperatur produziert. Daher muss jeweils nur eine Auftragsfamilie berücksichtigt werden (Formeln 6.21 und 6.22) und es treten keine Rüstzeiten auf.

Die zu erwartenden Zeitkonstanten für das Umrüsten der Anlagen $M_{2,1}$ und $M_{2,2}$ zwischen den jeweiligen Auftragsfamilien sind in den Formeln 6.23 und 6.24 gegeben.

$$\phi_{2,1,f_1,f_2}\,[\text{min}] = \begin{pmatrix} 0 & 20 & 480 & 480 & 480 & 480 \\ 20 & 0 & 480 & 480 & 480 & 480 \\ 240 & 240 & 0 & 120 & 120 & 0 \\ 240 & 240 & 120 & 0 & 0 & 120 \\ 240 & 240 & 120 & 120 & 0 & 120 \\ 240 & 240 & 120 & 120 & 120 & 0 \end{pmatrix} \tag{6.23}$$

$$\phi_{2,2,f_1,f_2}\,[\text{min}] = \begin{pmatrix} 0 & 20 & 120 & 120 & 640 \\ 20 & 0 & 120 & 120 & 640 \\ 120 & 120 & 0 & 20 & 640 \\ 120 & 120 & 20 & 0 & 640 \\ 240 & 240 & 240 & 240 & 0 \end{pmatrix} \tag{6.24}$$

Die Materialversorgung der Pressen (K_2) erfolgt für viele Fertigungsaufträge über die Sägen (K_1). Da bei den Sägen in der ersten Produktionsstufe keine Rüstzeiten auftreten, soll die Versorgung auf Basis der für die Pressen definierten Auftragsfamilien erfolgen. In Abhängigkeit der Auftragsfamilien, mit welchen die Jobs in der zweiten Produktionsstufe an den Pressen verarbeitet werden können, soll die Einplanung der Jobs in der ersten Stufe jeweils kontrolliert werden. Jeder Auftragsfamilie der zweiten Stufe soll daher eine Auftragsfamilie in der ersten entsprechen (Formeln 6.25 und 6.26).

$$o_{1,1} = \sum_{i=1}^{m_2} o_{2,i} + 1 = 14 \tag{6.25}$$

$$o_{1,2} = \sum_{i=1}^{m_2} o_{2,i} + 1 = 14 \tag{6.26}$$

Für die Jobs, die nach dem Zuschnitt nicht verpresst werden und damit das Werk nach der ersten Produktionsstufe verlassen, wird an den beiden Sägen jeweils eine weitere Auftragsfamilie angelegt. Mit diesem Mechanismus wird eine bedarfsgerechte Bestandsregelung für die Pressenanlagen durch die Zuschnittsägen ermöglicht.

In der zweiten Stufe (Pressen, K_2) findet kein Trennprozess statt. Der Fügeprozess an den Pressen ist auf das Verleimen der physischen Materialien beschränkt. Konvergierende Materialflüsse im Sinne von mehreren Fertigungsaufträgen, die vereint werden, finden daher nicht statt. Die Notwendigkeit der Stapelbearbeitung entfällt somit (Formel 6.27).

$$q_2 = 0 \tag{6.27}$$

Die Gewichtungen der einzelnen Teilziele der Optimierung (siehe Unterabschnitt 4.1.5) sind empirisch ermittelte Werte und werden von den bisherigen Untersuchungen übernommen (Formeln 6.28 bis 6.31).

$$w_1' = 1{,}0\,\frac{1}{\text{h}} \tag{6.28}$$

$$w_2' = 1{,}0\,\frac{1}{\text{h}} \tag{6.29}$$

$$w_3' = 1{,}5\,\frac{1}{\text{h}} \tag{6.30}$$

$$w_4' = 1{,}0\,\frac{1}{\text{h}} \tag{6.31}$$

Die zu erwartenden Bearbeitungszeiten der einzelnen Sägeläufe werden automatisch durch die entsprechende Software der Verschnittoptimierung ermittelt. Zusammen mit den

weiteren Daten der Verschnittoptimierung werden diese bei der Auftragserzeugung für die beiden Sägen $M_{1,1}$ und $M_{1,2}$ in das ERP-System transferiert. Für die Ablaufplanung durch das in dieser Arbeit entwickelte Verfahren sind die Bearbeitungszeiten der Sägeanlagen dementsprechend als gegeben anzusehen.

Für die Bearbeitung der Fertigungsaufträge durch die Pressenanlagen liegen keine Zeitvorgaben im ERP-System vor. Aus historischem Datenmaterial, welches über eine ausreichend lange Zeit und parallel zu dem laufenden Betrieb gesammelt wird, können die fehlenden Vorgabezeiten allerdings in ausreichender Qualität erzeugt werden. Über modellbasierte Auswertungen der gesammelten Daten werden in Abhängigkeit der einzelnen Pressenanlagen und der verschiedenen Auftragsfamilien die notwendigen Daten berechnet (siehe dazu Stalinski et al., 2019a). Folglich können an dieser Stelle auch die zu erwartenden Bearbeitungszeiten für die Pressenanlagen als gegeben vorausgesetzt werden.

6.4.3 Beurteilung der Ergebnisse

Das Planungsverfahren wird im Folgenden auf einen Datensatz angewendet, welcher auf den realen Betriebsdaten des Unternehmens basiert. Bei dem Datensatz handelt es sich um eine Momentaufnahme aus dem ERP-System, welche den aktuellen Ist-Zustand des Unternehmens zu einem bestimmten Zeitpunkt repräsentiert. Der Datensatz beinhaltet ca. 780 verplanbare Jobs (Formel 6.32) und ca. 570 verschiedene Materialien sowie deren Lagerbestände (Formel 6.33).

$$n \approx 780 \tag{6.32}$$

$$l \approx 570 \tag{6.33}$$

In Bezug auf die zu erwartende Komplexität des Suchraums liegt die Fallstudie laut den Formeln 6.34 und 6.35 zwischen den Datensätzen S–224 und M–424 (vgl. Tabelle 6.1).

$$L = 6 \cdot \theta \tag{6.34}$$

$$\kappa\left(\theta = 8\right) = 1{,}4 \cdot 10^{30} \tag{6.35}$$

Formel 6.36 liefert in Anlehnung an die bei den randomisierten Daten verwendete Schreibweise für den Datensatz der Fallstudie eine Bezeichnung. Das „R“ ist dabei als Abkürzung für „real“ zu interpretieren.

$$\text{R–2/2–4/14–14–6–5–1–1} \tag{6.36}$$

Wie bereits bei den vorherigen Studien wird zunächst untersucht, welche Diskretisierung θ und welche Planungsregel für den Einsatz des Verfahrens auf Basis der realen Daten die besten Ergebnisse liefert. In Abbildung 6.18 sind dafür die Einflüsse der unterschiedlichen Diskretisierungen $\theta \in \{8; 10; 12; 14; 16\}$ auf die Qualität der Planungsergebnisse dargestellt. Im Gegensatz zu allen bisherigen Untersuchungen (siehe Abbildungen 6.5 bis 6.7) zeigt Abbildung 6.18, dass die Anwendung des Planungsverfahrens auf die realen Daten der Fallstudie mit $\theta = 14$ die besten Resultate liefert.

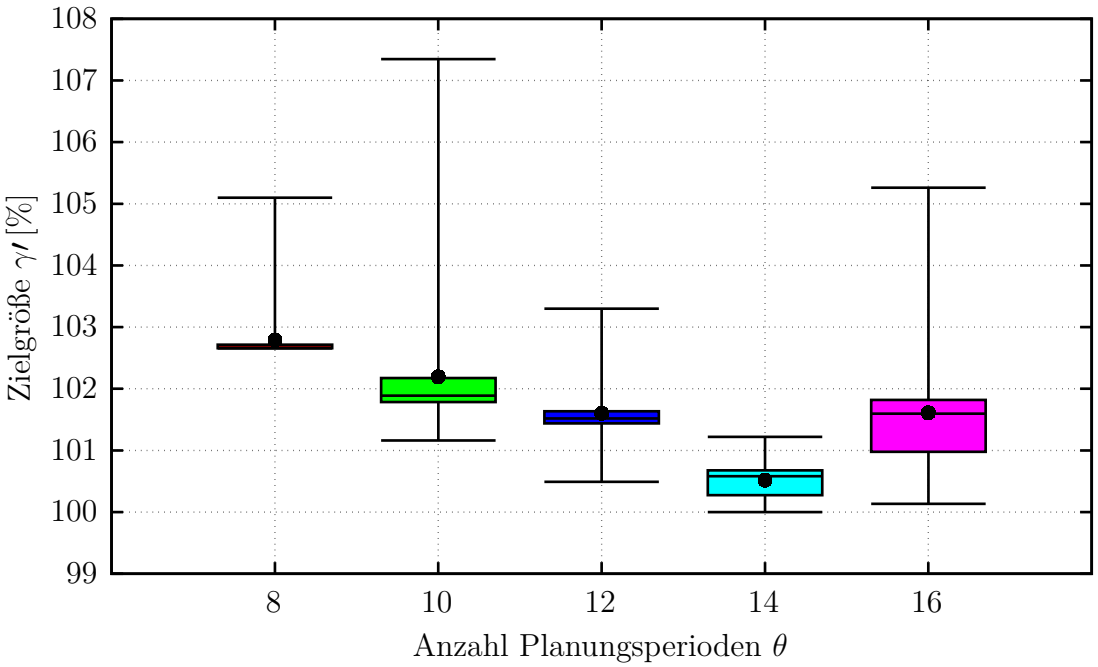


Abbildung 6.18: Vergleich unterschiedlicher Planungsperioden anhand realer Daten

In Abbildung 6.19 sind die Einflüsse der unterschiedlichen Planungsregeln auf die Planungsergebnisse dargestellt. Die Ergebnisse decken sich weitestgehend mit den Ergebnissen der bisherigen Untersuchungen (siehe Abbildungen 6.8 bis 6.10). So ist für die optimierte Anwendung des Planungsverfahrens bei der gewählten Zielgrößendefinition (Formeln 6.28 bis 6.31) die S/OPN-Regel auszuwählen.

In Abbildung 6.20 ist als Beispiel der beste im Rahmen dieser Arbeit gefundene Ablaufplan auf Basis des realen Datensatzes dargestellt. Es handelt sich dabei um einen Ablaufplan mit einem Planungshorizont von $H = 48\,\mathrm{h}$, welcher unter der Verwendung der S/OPN-Regel sowie mithilfe von $\theta = 14$ Planungsperioden berechnet wurde. Die dem Ablaufplan zugrunde liegenden Entscheidungsmatrix $\delta_{k,i,\tau}$ (Formel 6.37) wurde dabei durch den Genetischen Algorithmus mit den in Unterabschnitt 6.2.2 ermittelten Parametern optimiert.

$$\delta_{k,i,\tau} = \begin{pmatrix} 1 & 3 & 4 & 5 & 1 & 9 & 2 & 2 & 10 & 1 & 7 & 10 & 1 & 1 \\ 1 & 3 & 13 & 14 & 12 & 11 & 3 & 14 & 11 & 9 & 14 & 14 & 12 & 14 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 & 1 & 2 & 2 & 2 \\ 3 & 3 & 3 & 4 & 1 & 3 & 3 & 3 & 3 & 3 & 4 & 3 & 3 & 3 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{6.37}$$

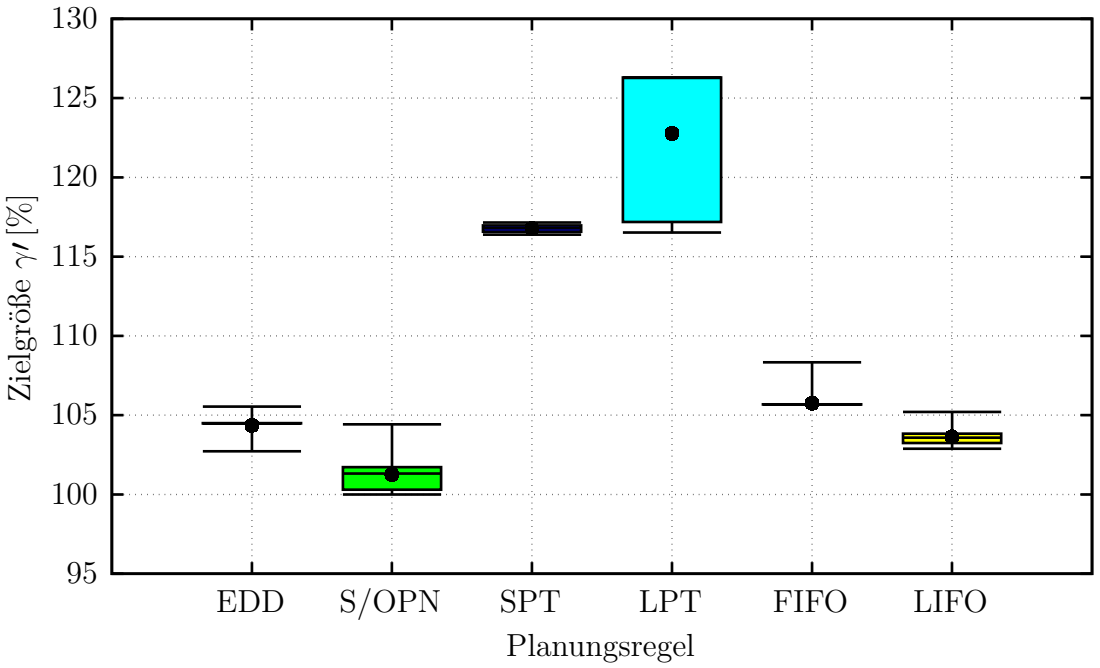


Abbildung 6.19: Vergleich der unterschiedlichen Planungsregeln anhand realer Daten

In dem Ablaufplan konnten durch das Planungsverfahren insgesamt ca. 460 der 780 zur Verfügung stehenden Jobs (Formel 6.32) eingeplant werden. Die Bearbeitungszeiten der einzelnen Jobs sind dabei teilweise so kurz, dass eine grafische Differenzierung nicht mehr möglich ist. Dies erklärt das Auftreten der schwarzen Balken in Abbildung 6.20.

Bei der Begutachtung des Ablaufplans fallen an den Maschinen $M_{1,2}$ und $M_{2,1}$ Lücken in Form von relativ kurzen Leerlaufzeiten auf. Dies ist darauf zurückzuführen, dass der Auftragsbestand von $n \approx 780$ nicht zwangsläufig genügend Jobs für eine vollständige Auslastung des gesamten Maschinenparks enthält. Dies erklärt weiterhin die sehr geringe Auslastung der Presse $M_{2,3}$.

Desweiteren ist davon auszugehen, dass in dem Datensatz durchaus weitere Jobs für die Einplanung an den entsprechenden Anlagen zur Verfügung stehen. Jobs, für deren Bearbeitung dieselben nur begrenzt verfügbaren Materialen benötigt werden, können um die zur Verfügung stehenden Ressourcen konkurrieren. Für Jobs, die zu Beginn der Planung zur Verfügung standen, kann daher ein Materialengpass entstehen. Durch die Berücksichtigung der begrenzten Ressourcen kommt es während der Planung daher zu Situationen, in denen die Einplanung von weiteren Jobs verhindert wird.

Abbildung 6.21 zeigt einen Ausschnitt aus dem in Abbildung 6.20 dargestellten Ablaufplan. Gezeigt wird der Zeitraum zwischen den Planungsperioden t_6 und t_7. Um den Durchlauf einzelner Fertigungsaufträge durch das Produktionssystem identifizieren und beurteilen zu können, sind in Anlehnung an Abbildung 6.13 einzelne Fertigungsaufträge farbig markiert. So kann beispielsweise anhand der Jobs J_{17}, J_{151}, J_{34}, J_{156}, J_{116}, J_{18} und J_{104} das Planungsergebnis beurteilt werden.

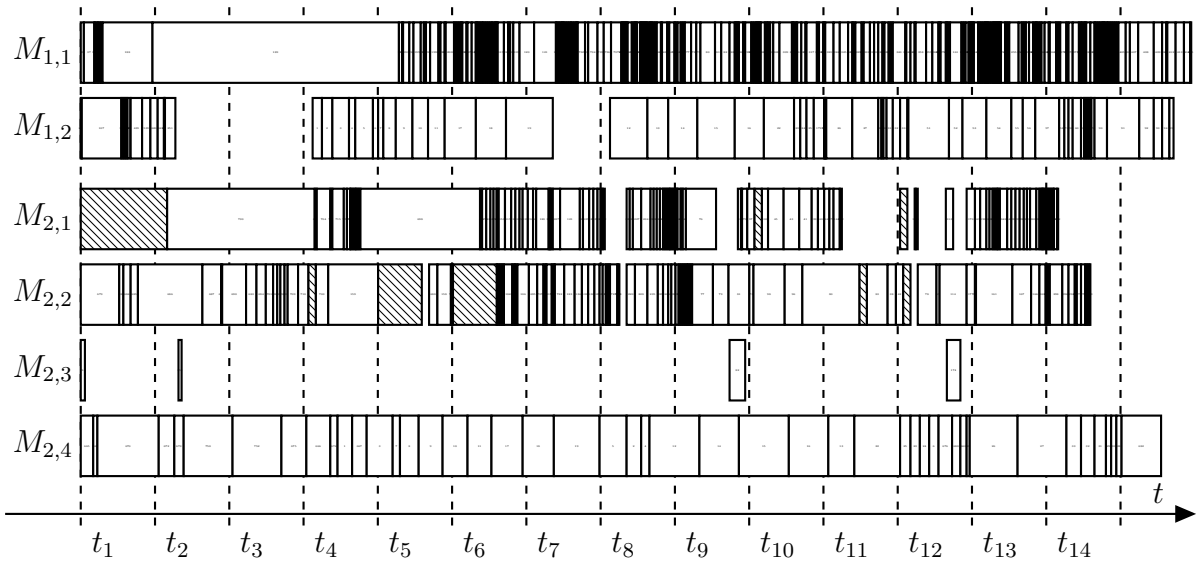


Abbildung 6.20: Ablaufplan auf Basis realer Daten (S/OPN-Regel, $H = 48\,\text{h}$ und $\theta = 14$)

Die Jobs J_{17} und J_{18} werden beispielsweise zunächst an der Säge $M_{1,2}$ geschnitten und anschließend an der Presse $M_{2,4}$ verpresst. Unnötige Wartezeiten treten für die Jobs in diesem Zusammenhang nicht auf.

Die Jobs J_{151}, J_{156} und J_{116} werden nach dem Zuschnitt auf der Säge $M_{1,1}$ durch die Presse $M_{2,1}$ weiterbearbeitet. Die Jobs J_{34} und J_{104} werden nach dem Zuschnitt auf der Säge $M_{1,1}$ nach einem Umrüstvorgang an der Presse $M_{2,2}$ verarbeitet.

In Abbildung 6.22 ist das Laufzeitverhalten des Planungsverfahrens auf Basis der realen Daten dargestellt. In Bezug auf die bisherigen Untersuchungen der Laufzeit (vgl. Abbildungen 6.11 und 6.12) zeigt das Verfahren im Zusammenhang mit den realen Daten ein ausgewogenes Laufzeitverhalten.

Insgesamt können unter den gegebenen Voraussetzungen und der verwendeten Hardware ca. 108 Iterationsschritte innerhalb der vorgegebenen Zeit von 4 min gerechnet werden. Bei einer Populationsgröße des Genetischen Algorithmus von $N = 3200$ entspreicht dies einer Anzahl von ca. $108 \cdot 3200 = 345\,600$ Ablaufplänen, die während eines Optimierungslaufs erzeugt und ausgewertet werden.

Der größte Anteil des Optimierungspotenzials wird bereits in den ersten 42 Iterationsschritten ausgeschöpft. Die Zielgröße wird dabei von ca. $\gamma\prime \approx 137\,\%$ auf ca. $\gamma\prime \approx 101\,\%$ optimiert. Zwischen den Iterationsschritten 70 und 90 finden schließlich noch die letzten Verbesserungen statt.

Insgesamt lässt der Verlauf der Zielgröße während des Optimierungsprozesses darauf schließen, dass sowohl die gewählten Parameter als auch die gewählten Randbedingungen mit der Anwendung auf Datensätze der Fallstudie abgestimmt sind.

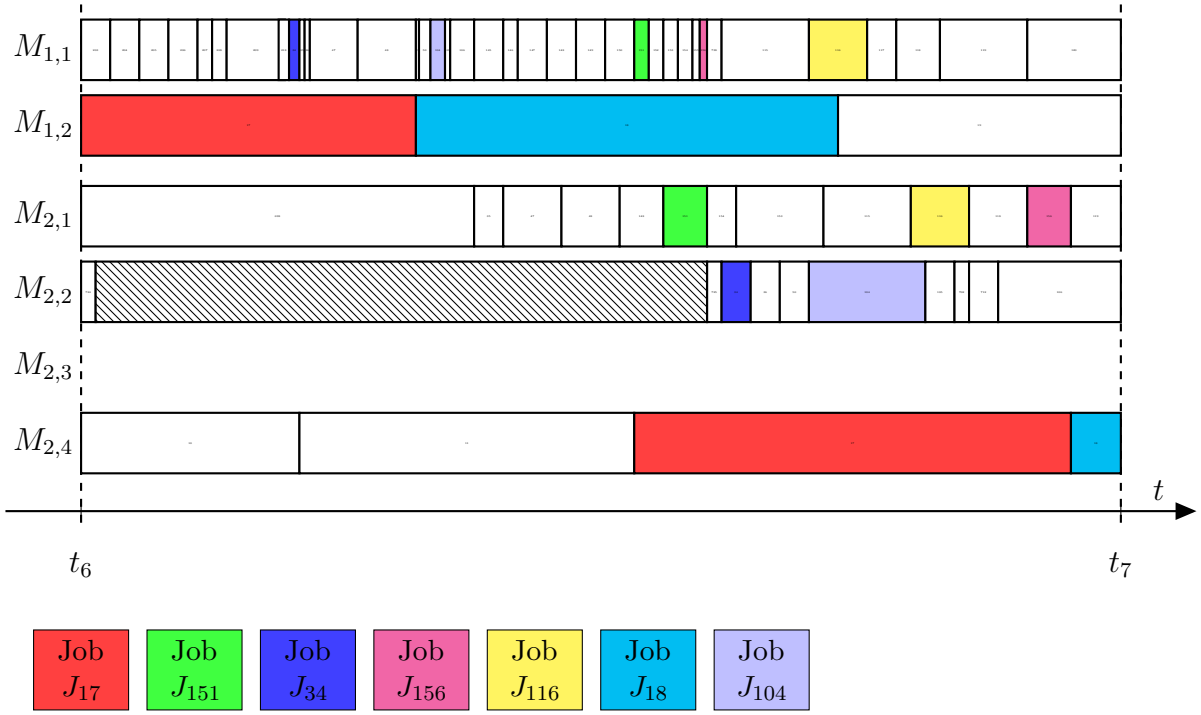


Abbildung 6.21: Ausschnitt aus dem Ablaufplan auf Basis realer Daten

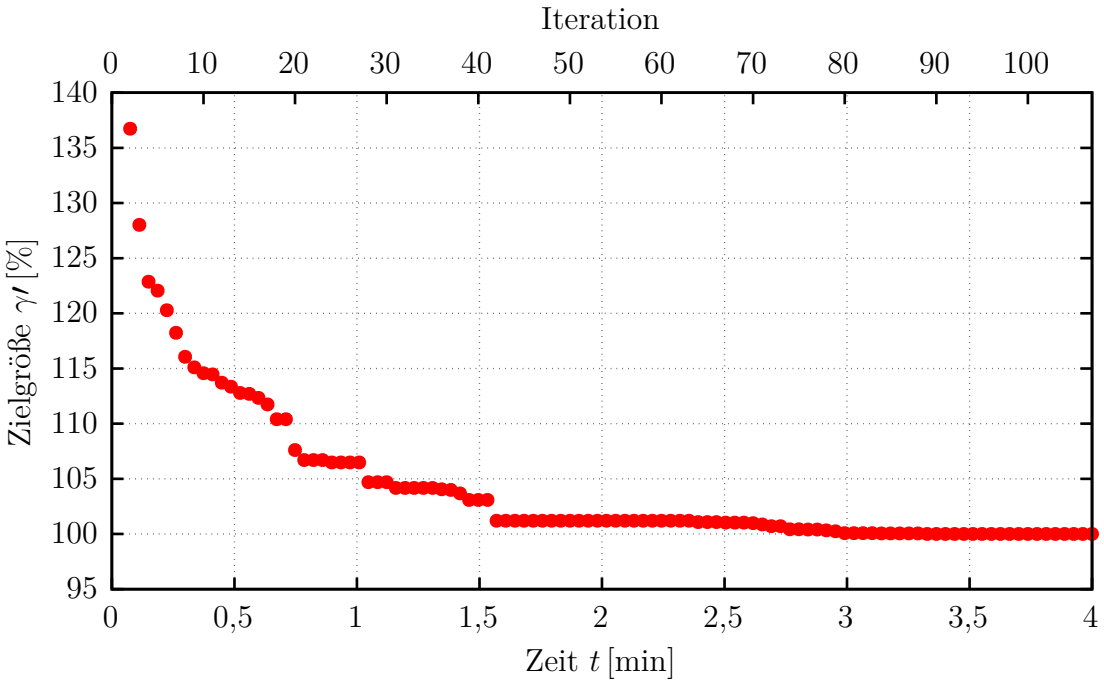


Abbildung 6.22: Laufzeitverhalten des Verfahrens auf Basis realer Daten (S/OPN, $\theta = 14$)

6.4.4 Einbindung des Planungsverfahrens in ein Entscheidungssystem

Das Planungsverfahren wurde in dem Produktionsbetrieb der Fallstudie in Form eines Entscheidungssystems für die Produktionsplanung und -steuerung implementiert. Dort begleitet es die Produktion und berechnet in einem Takt von 30 min laufend einen neuen Ablaufplan auf Basis stets aktueller Betriebsdaten.

Die Einbindung des Planungsverfahrens in den Produktionsprozess der Fallstudie ist in Abbildung 6.23 dargestellt. Jede Berechnung eines Ablaufplans beginnt mit dem Download eines Datensatzes aus dem ERP-System des Unternehmens sowie aus einer SQL-Datenbank.

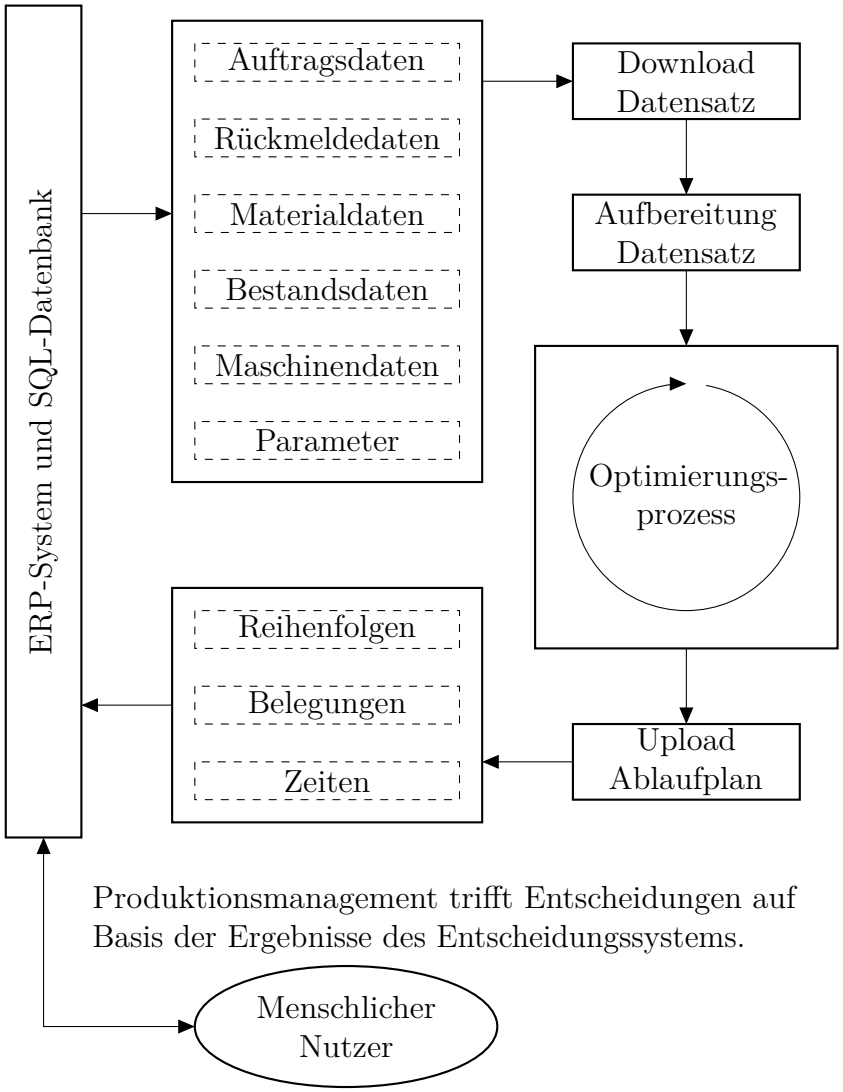


Abbildung 6.23: Implementierung des Verfahrens als Entscheidungssystem

Auf diese Weise werden zunächst die relevanten Daten bezüglich der Fertigungsaufträge (Kunden, Termine, Mengen etc.), der Rückmeldungen (Auftragsfortschritte, physische Positionen der Plattenstapel etc.), der Materialien (Abmessungen, Ausführungen etc.) sowie der Lagerbestände (Bestände in den Rohwarenlagern, Pufferbestände etc.) aus dem ERP-System akquiriert. Weitere Daten bezüglich des vorhandenen Maschinenparks (Lagerkapazitäten, Maschineneigenschaften etc.) sowie die Parameter des Planungsverfahrens liegen nicht im ERP-System vor und werden daher aus einer SQL-Datenbank importiert.

Anschließend werden die Daten derart aufbereitet, dass sie hinsichtlich ihrer Struktur dem mathematischen Modell des Planungsverfahrens entsprechen. Der Kern des Entscheidungssystems ist das im Rahmen dieser Arbeit entwickelte Planungsverfahren, so dass der Optimierungsprozess auf Basis der aufbereiteten Daten gestartet wird.

Nach dem Terminieren des Optimierungsprozesses werden die erzeugten Ergebnisse sowohl in das ERP-System als auch in die SQL-Datenbank exportiert. Dabei handelt es sich um die optimierten Bearbeitungsreihenfolgen und Maschinenbelegungen sowie um die zugehörigen Start- und Endzeiten der Fertigungsaufträge an den einzelnen Maschinen. Dort sind sie zugänglich für das Management der Produktion und dienen den menschlichen Entscheidern als Grundlage.

Das Entscheidungssystem ist auf diese Weise seit seiner Einführung in dem Unternehmen etabliert. Die Entscheidungen sind sowohl über die gewohnten Bedienoberflächen des ERP-Systems als auch über eine Web-Visualisierung auf Basis der SQL-Datenbank überall in dem Unternehmen zugänglich und sichtbar. Das System zeichnet sich durch eine hohe Zuverlässigkeit und Planungsqualität aus, so dass die Produktionsleitung in nahezu allen Fällen den Entscheidungen des Systems folgt.

6.5 Fazit und kritische Würdigung der Ergebnisse

Es folgt eine kritische Auseinandersetzung mit den in dieser Arbeit erzielten Ergebnissen. Die Bewertung der Ergebnisse erfolgt dabei

- bezüglich der in Kapitel 1.2 formulierten Ziele (Unterabschnitt 6.5.1),
- bezüglich des wissenschaftlichen Erkenntnisgewinns in Abgrenzung zur aktuellen Literatur (Unterabschnitt 6.5.2) sowie
- bezüglich der möglichen Größen von Planungsproblemen, die mit dem Verfahren (unter den gegebenen Voraussetzungen) noch gerechnet werden können (Unterabschnitt 6.5.3).

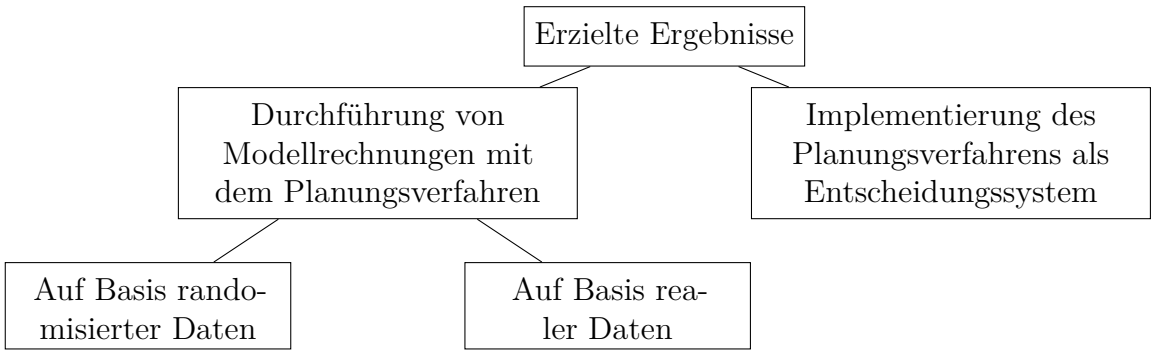


Abbildung 6.24: Erzielte Ergebnisse mit dem Planungsverfahren

6.5.1 Auswertung der erzielten Ergebnisse

Die in der vorliegenden Arbeit erzielten Ergebnisse basieren

- auf der Durchführung von Modellrechnungen auf Basis randomisierter sowie realer Daten (Abbildung 6.24, links) sowie
- auf der Implementierung des Planungsverfahrens als Entscheidungssystem in die operative Produktionsplanung und -steuerung eines realen Industriebetriebes (Abbildung 6.24, rechts).

Alle Ergebnisse der systematischen Modellrechnungen wurden auf einem Rechencluster erzielt, welches dem aktuellen Stand der Technik entspricht (Stand 2020). Die im Rahmen dieser numerischen Untersuchungen erzielten Ergebnisse lassen sich wie folgt zusammenfassen:

- Die programmiertechnische Umsetzung des Planungsverfahrens ermöglicht die parallelisierte Rechnung und damit eine Lastverteilung auf sämtliche zur Verfügung stehende CPU-Kerne des genutzten Rechenclusters. Ein volles Ausnutzen der gegebenen Rechenkapazität wird dadurch ermöglicht.
- Der Genetische Algorithmus liefert im Zusammenhang mit dem im Rahmen dieser Arbeit entwickelten Planungsverfahren nachweislich bessere Ergebnisse als der Ameisenalgorithmus.
- Die durch die durchgeführten Rechenstudien optimierten Parameter des Planungsverfahrens zeigen – wie zu erwarten – zumindest teilweise eine Abhängigkeit von der zugrunde liegenden Problemgröße. So unterscheidet sich die unter den gegebenen Randbedingungen (Rechenzeit, Hardware, Suchverfahren, weitere Verfahrensparameter etc.) gefundene optimale Anzahl der Planungsperioden für die kleinste Problemklasse S ($\theta = 16$) deutlich von dem ermittelten Optimum für die größeren Problemklassen M und L ($\theta = 8$ und $\theta = 10$).
- In Bezug auf die untersuchten Planungsregeln kann eine Abhängigkeit von der Problemgröße nicht festgestellt werden. Auch dies entspricht den Erwartungen. Hier

liefert die Planungsregel EDD (frühester Termin zuerst) bei allen randomisierten Testdaten die besten Ergebnisse.

- Die grafische Beurteilung eines Ablaufplans anhand praktischer Gesichtspunkte zeigt, dass die Planungsergebnisse vollumfänglich den Erwartungen aus der Praxis entsprechen.

- Die Ergebnisse der Studien auf Basis der realen Daten erweitern die durch die randomisierten Daten erlangten Erkenntnisse. Die Größe des hier betrachteten Produktionsprozesses liegt mit einer Anzahl von 6 Maschinen zwischen den Größen der Datensätze S (4 Maschinen) und M (8 Maschinen). Die optimale Anzahl an Planungsperioden bestätigt die zuvor erlangte Erkenntnis und liegt mit $\theta = 14$ wie zu erwarten zwischen den Optima der Datensätze S ($\theta = 16$) und M ($\theta = 8$).

- Im Gegensatz zu den randomisierten Daten (Planungsregel EDD) liefert bei dem realen Datensatz allerdings die Planungsregel S/OPN (geringste Schlupfzeit zuerst) die besten Ergebnisse. Die Produktionsprozesse der fiktiven Maschinenparks (randomisierte Daten) und des realen Datensatzes unterscheiden sich nicht in Bezug auf ihre Struktur. Es ist daher naheliegend, dass die Abweichung der optimalen Planungsregel auf die Zusammensetzung der Auftrags- und Maschinendaten (Termine, Mengen, Zeiten etc.) zurückzuführen ist.

- Die Analyse des Laufzeitverhaltens (siehe Abbildung 6.25) zeigt, dass vergleichsweise einfache Probleme (S, gelbe Kurve) die gegebenen Kapazitäten des Rechenclusters noch nicht voll ausnutzen. Erst die größeren Probleme L (grüne Kurve) und die reale Produktion (rote Kurve) zeigen die Grenzen des Verfahrens unter den gegebenen Randbedingungen. Für die Rechnung größerer Probleme wäre die Zuhilfenahme einer längeren Rechenzeit sinnvoll.

Die Praxistauglichkeit sowie der praktische Mehrwert des Planungsverfahrens werden durch die Implementierung in ein Entscheidungssystem für die operative Produktionsplanung und -steuerung eines realen Industriebetriebes deutlich. Für das Unternehmen der Fallstudie ergeben sich die folgenden konkreten Vorteile durch den Einsatz des Planungsverfahrens als Entscheidungssystem:

- Die Optimierung mit Fokus auf die logistischen Zielgrößen (siehe Unterabschnitt 4.1.5) führt zu einer verbesserten Termineinhaltung gegenüber dem Kunden. Diese Erhöhung der Liefertreue kann langfristig auf Kundenseite zu einer Erhöhung der Zufriedenheit und damit tendenziell auch zu einer Erhöhung der Kundenbindung führen.

- Weiterhin werden durch die Optimierung im Hinblick auf die logistischen Zielgrößen sowie durch die Berücksichtigung der entsprechenden Randbedingungen die Umlaufbestände kontrolliert. Die Auslastung der logistischen Puffer (Zwischenlager, Versandlager) wird dadurch in den technisch zulässigen Grenzen gehalten. Ein Überfüllen dieser Puffer – wie es durch einen menschlichen Planer passieren kann – und dadurch die Bildung von logistischen Blockaden ist nahezu ausgeschlossen.

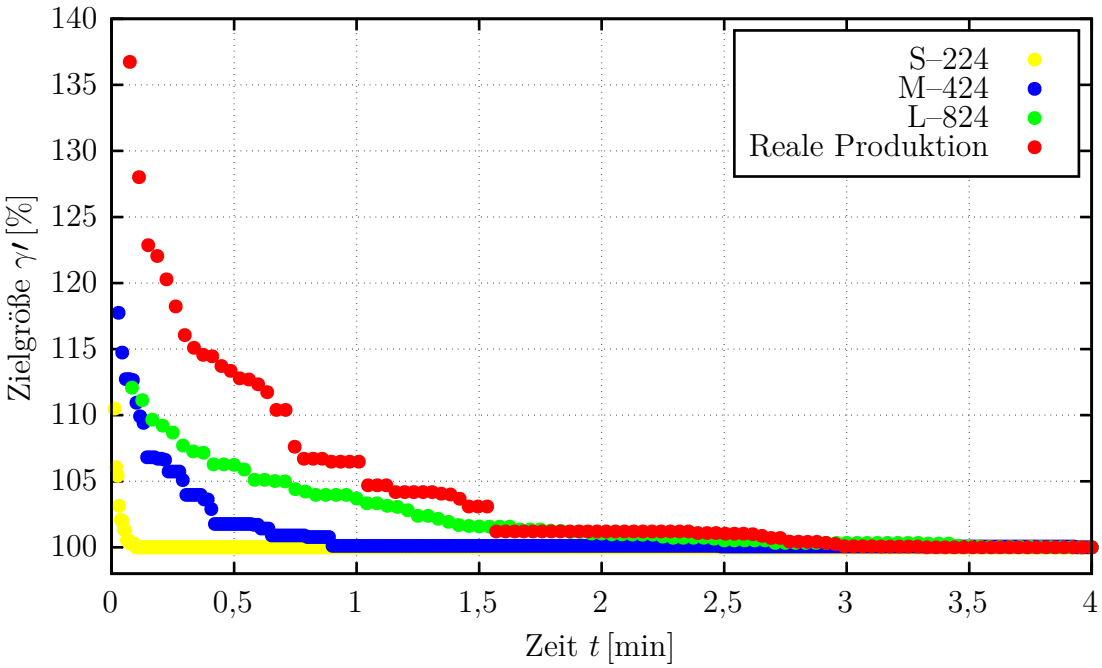


Abbildung 6.25: Laufzeitverhalten des Verfahrens auf Basis unterschiedlicher Daten

- Durch das hohe Reaktionsvermögen des Systems wird ein schnelles und angemessenes Reagieren auf kurzfristige Ereignisse und Situationen erreicht. Kurzfristige und häufig (mehrmals pro Stunde) eintretende Ereignisse wie z.B. Änderungen bzgl. der Materialverfügbarkeit oder Fortschritte in der Auftragsbearbeitung liegen sofort in Form von tabellarischen Daten im ERP-System vor. Das rollierende Entscheidungssystem ist an dieser Stelle in der Lage, auf Basis dieser neuen Informationen schnell genug sinnvolle Entscheidungen zu treffen. Dem Menschen allein fehlen dafür im Gegensatz zu dem System die Möglichkeiten und die Fähigkeiten.

- Das Entscheidungssystem bietet eine solide Unterstützung für die menschlichen Entscheider, so dass diese auch in Bezug auf ihren Stresslevel wesentlich entlastet werden. Das manuelle Aufbereiten der tabellarischen Betriebsdaten aus dem ERP-System durch den Menschen als Entscheidungsgrundlage erfordert dagegen viel Konzentration und übt Stress auf den Planer aus. Durch die systemseitige Unterstützung wird die Arbeit des Planers entsprechend erleichtert.

- Darüber hinaus bietet der Einsatz des Systems die Möglichkeit von Einsparungen oder Umverteilungen der verfügbaren Mitarbeiterkapazitäten. Mitarbeiter, die vor dem Einsatz des Entscheidungssystems vollumfänglich als Planer tätig waren, können – bedingt durch ihre Entlastung durch das System – mit weiterführenden Aufgaben (z.B. hierarchisch übergeordneten Entscheidungsprozessen) betraut werden.

- Da das System bei der Optimierung der Ablaufplanung stets auf das Gesamtsystem „Produktion“ fokussiert ist, ergeben sich deutliche Vorteile im Gegensatz zu der rein menschlichen Entscheidungsfindung. Die Praxis zeigt, dass Menschen ohne die Unterstützung durch ein Entscheidungssystem in der Regel nur ihren eigenen Bereich der Produktion überblicken und sich dadurch gewissermaßen immer nur selbst optimieren, ohne dabei den großen Zusammenhang überblicken zu können. Die Geschwindigkeit, die Qualität sowie die Sicherheit, mit welcher die Entscheidungen getroffen werden können, wird durch den Einsatz des Planungsverfahrens entsprechend deutlich erhöht.

Zusammenfassend kann festgehalten werden, dass die in Abschnitt 1.2 formulierten Ziele sowohl in Bezug auf die Ergebnisse der Modellrechnungen als auch in Bezug auf die Implementierung des Planungsverfahrens vollumfänglich erreicht wurden.

6.5.2 Abgrenzung des wissenschaftlichen Erkenntnisgewinns

Die in Unterabschnitt 6.5.1 aufbereiteten Forschungsergebnisse spiegeln den im Rahmen dieser Arbeit erzielten wissenschaftlichen Erkenntnisgewinn wieder. Dieser lässt sich mit den in Unterabschnitt 2.4.4 behandelten Ergebnissen bisheriger Forschungsarbeiten (siehe insbesondere Tabelle 2.1) in Beziehung setzen und entsprechend beurteilen.

In Tabelle 6.11 sind die wesentlichen Forschungsergebnisse sowohl dieser Arbeit als auch der in Unterabschnitt 2.4.4 analysierten Forschungsarbeiten nach Kategorien zusammengefasst. Alle Forschungsarbeiten verfolgen die Entwicklung bzw. Erprobung jeweils eines Lösungsansatzes für jeweils eine spezielle Klasse von Flexible Flow Shop Problemen. Ein Vergleich der Forschungsergebnisse der vorliegenden Arbeit mit den Ergebnissen der untersuchten Arbeiten wird daher über die jeweils verfolgte Forschungsfrage, das angewendete Lösungsverfahren sowie über die verwendete Methodik geführt.

Ein Vergleich der verfolgten Forschungsfragen lässt sich über die folgenden Kriterien festlegen:

Maschinenumgebung (α): Ist die mathematische Komplexität der in der jeweiligen Forschungsarbeit behandelten Maschinenumgebung gleichwertig oder höher als die Komplexität der in dieser Arbeit behandelten Maschinenumgebung, wird diese Kategorie mit einem „×“ markiert. Die Bewertung der Komplexität erfolgt gemäß der entsprechenden Werte für die Anzahl der Stufen (c), die Anzahl der Maschinen (m_k) und die Art der Maschinen (α_3), siehe Tabelle 2.1.

Randbedingungen (β): Nur wenn eine Forschungsarbeit sämtliche der im Rahmen dieser Arbeit berücksichtigten Randbedingungen und Restriktionen behandelt, erfolgt die Kennzeichnung mit einem „×“.

Zielsetzung (γ): Liegt der jeweiligen Forschungsarbeit die Optimierung einer multikriteriellen Zielsetzung (im Gegensatz zur Behandlung nur einer festen Zielgröße) zu Grunde, erfolgt die Markierung mit einem „×“.

Praxisbezug: Ist der Bezug zur Praxis (z.B. die Implementierung eines Verfahrens in eine reale Produktionsplanung und -steuerung) teil der jeweiligen Forschungsfrage, erfolgt die Kennzeichnung mit einem „×“.

Tabelle 6.11: Gegenüberstellung der erzielten Forschungsergebnisse

Publikation	α	β	γ	Praxisbezug	Metaheuristiken	Heuristiken	Simulation
Stalinski	×	×	×	×	×	×	×
Akrami et al. 2006	×	–	×	–	×	–	–
Alaykýran et al. 2007	–	–	–	–	×	–	–
Alisantoso et al. 2003	–	–	–	×	×	–	–
Behnamian et al. 2011	×	–	×	–	×	–	–
Chaaben et al. 2013	–	–	–	×	×	–	–
Chen et al. 2009	×	–	–	–	–	×	–
Chen et al. 2007	–	–	–	×	×	–	–
Costa et al. 2014	×	–	–	–	×	–	–
Ebrahimi et al. 2014	–	–	×	–	×	–	–
Engin et al. 2004	–	–	–	–	×	–	–
Ernst et al. 2019	–	–	–	–	–	×	–
Gondek 2011	–	–	–	×	–	×	–
Han et al. 2019	–	–	–	×	×	–	–
Janiak et al. 2007	–	–	×	–	×	–	–
Javadian et al. 2013	–	–	–	–	×	–	–
Jungwattanakit et al. 2009	–	–	×	–	×	–	–
Keshavarz et al. 2013	–	–	–	–	×	–	–
Kianfar et al. 2012	–	–	–	–	×	–	–
Klement et al. 2017	–	–	×	×	×	–	–
Laribi et al. 2016	–	–	–	–	–	×	–
Lin et al. 2020	–	–	×	–	×	–	–
Logendran et al. 2006	–	–	–	–	×	–	–
Luo et al. 2015	–	–	–	×	×	–	–
Naderi et al. 2009	–	–	×	–	×	–	–
Nahhas et al. 2015	–	–	–	×	×	–	×
Niu et al. 2012	–	–	–	–	×	–	–
Ramezanian et al. 2017	×	–	×	×	×	–	–
Ruiz et al. 2006	×	–	–	×	×	–	–
Sawik 2006	×	–	×	×	–	×	–
Shi et al. 2019	–	–	–	×	–	–	–
Tan et al. 2018	–	–	–	×	–	–	–
Tang et al. 2006	–	–	–	×	–	–	–
Wang et al. 2009	–	–	–	–	×	–	–
Wardono et al. 2004	–	–	–	–	×	–	–
Wu et al. 2003	×	–	–	×	×	–	–
Yaurima et al. 2009	×	–	–	×	×	–	–
Ying et al. 2006	–	–	–	–	×	–	–
Zhang et al. 2016	–	–	–	–	–	×	–

Der Vergleich über die jeweils angewendete Methodik sowie über das jeweils entwickelte bzw. untersuchte Lösungsverfahren wird über die folgenden Kriterien geführt:

- Verwendung von Metaheuristiken,
- Verwendung von Heuristiken,
- Verwendung von Simulationsmethoden als Teil des Lösungsverfahrens.

Die Entwicklung eines für die im Rahmen dieser Arbeit behandelten Problemklasse (die Kombination von α, β und γ) ist in Bezug auf den in Tabelle 6.11 durchgeführten Vergleich ein wesentlicher Fortschritt. Es lässt sich festhalten, dass die behandelte Problemklasse bei keiner dieser Forschungsarbeiten Teil der Forschungsfrage ist. Darüber hinaus stellt insbesondere die Kombination dieser Problemklasse mit der realen Anwendung (Praxisbezug) einen wesentlichen Forschungsfortschritt dar.

Das in dieser Arbeit entwickelte Lösungsverfahren beruht auf der Kombination von Metaheuristiken (Suchverfahren), Heuristiken (Planungsregeln) sowie einer Simulationsmethode. Die Kombination dieser Methoden als Teil eines Lösungsverfahrens stellt einen weiteren Fortschritt in der Entwicklung und Erforschung von Flexible Flow Shop Problemen dar.

Sowohl die im Rahmen dieser Arbeit behandelte Problemklasse als auch das im Rahmen dieser Arbeit entwickelte Verfahren sind neu und bislang unerforscht. Die in dieser Arbeit erzeugten mathematischen Ergebnisse zielen im Detail auf die Ausgestaltung und Parametrierung des neuen Verfahrens selbst ab. Da sowohl die Forschungsfrage als auch die verwendete Methodik im Detail eine Neuheit darstellen, bleibt ein direkter Vergleich dieser detaillierten Ergebnisse an dieser Stelle aus.

6.5.3 Abschätzung der Auslastung und Eingrenzung der Problemgröße

Das im Rahmen dieser Arbeit entwickelte Lösungsverfahren eignet sich für Aufgabenstellungen, welche die Voraussetzungen der hier untersuchten Problemklasse von Flexible Flow Shop Problemen erfüllen (siehe Abschnitt 2.2). Unter diesen Voraussetzungen kann das Verfahren prinzipiell auf beliebig große Probleme angewendet werden.

Für die im Rahmen dieser Arbeit analysierten Probleme lässt sich nach Formel 4.80 (siehe Unterabschnitt 4.2.3) die jeweilige Komplexität κ berechnen. Aus der Analyse des Laufzeitverhaltens (siehe beispielsweise Abbildung 6.25) lässt sich für jeden einzelnen dieser Datensätze die benötigte Zeit $t_{\text{Konvergenz}}$ für das Erreichen des jeweils gefundenen Optimums ablesen (Konvergenzzeit).

Im Rahmen der Studien zur Bestimmung der jeweils optimalen Anzahl an Planungsperioden θ (siehe Abbildungen 6.5 bis 6.7 und 6.18) wurden für die Datensätze S–224, M–424 und L–824 jeweils 9 und für den realen Datensatz 5 (also insgesamt 32) Berechnungsreihen durchgeführt. Die Variation der Planungsperioden θ stellt dabei gleichzeitig eine Variation der Komplexität κ dar (Beziehung zwischen θ und κ siehe Formel 4.80).

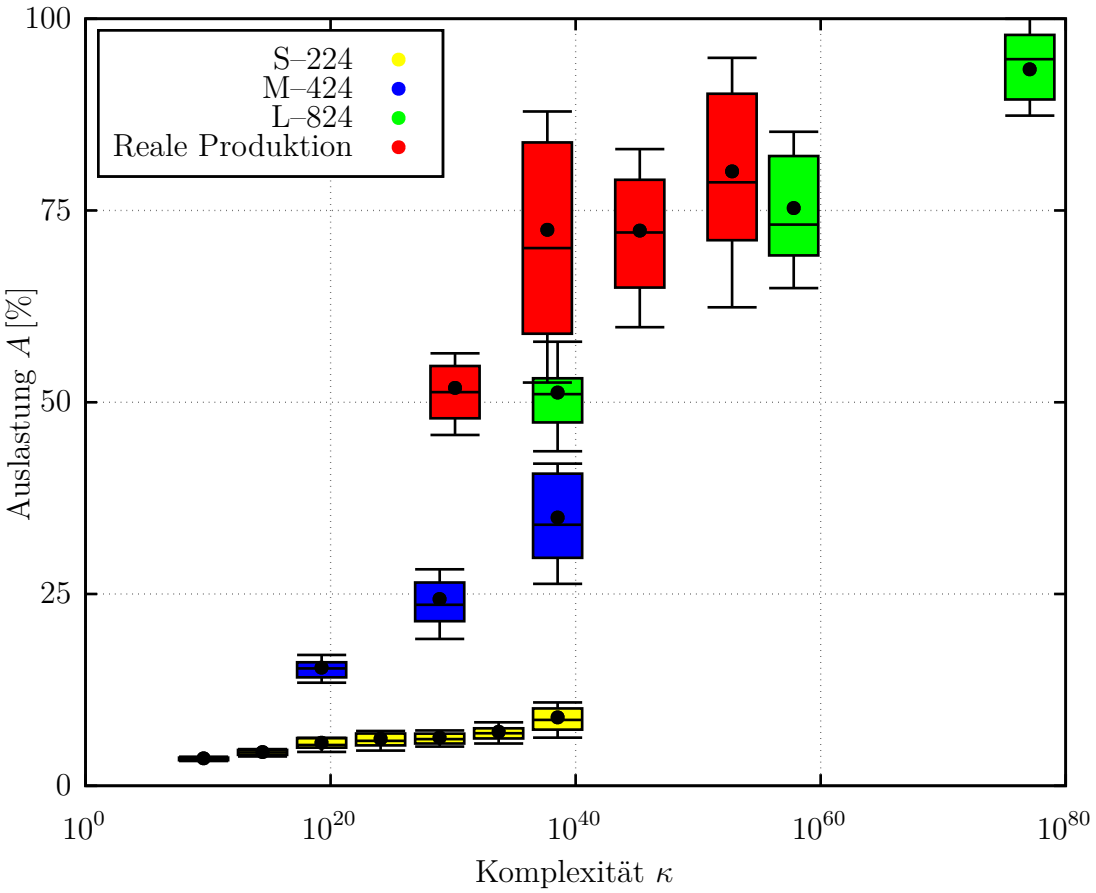


Abbildung 6.26: Auslastung des Verfahrens auf Basis unterschiedlicher Daten

Durch die statistische Absicherung der Rechenstudien (Wiederholrate von 50) ergeben sich folglich $32 \cdot 50 = 1600$ Verläufe des jeweiligen Laufzeitverhaltens, aus denen sich die jeweilige Konvergenzzeit $t_{\text{Konvergenz}}$ für unterschiedliche Komplexitäten κ ermitteln lässt.

Allen im Rahmen dieser Arbeit durchgeführten Rechenstudien liegen die gleichen Randbedingungen (verwendete Hardware, maximale Rechenzeit etc.) zu Grunde. Mit $t_{\text{Randbedingung}}$ als die maximal zur Verfügung gestellte Rechenzeit von $4\,\text{min} = 240\,\text{s}$ lässt sich darauf aufbauend die Auslastung A des Verfahrens nach Formel 6.38 bestimmen.

$$A = \frac{t_{\text{Konvergenz}}}{t_{\text{Randbedingung}}} \tag{6.38}$$

Die Ergebnisse dieser Untersuchung sind in Abbildung A.47 (siehe Anhang) komplett und in Abbildung 6.26 als Ausschnitt grafisch dargestellt. Die Abbildungen zeigen die statistische Verteilung (Wiederholrate von 50) der Auslastung A des Planungsverfahrens in Abhängigkeit der Komplexität κ.

Während Abbildung A.47 die Ergebnisse jeweils aller Variationen von θ enthält, sind in Abbildung 6.26 analog zu den Ergbnissen der θ-Studien (Erhöhung der Komplexität nur

soweit, bis optimale Ergebnisse erzielt werden) nur die als technisch sinnvoll anzusehenden Studien sichbar (bis $\theta\,(\mathrm{S}) = 16$, $\theta\,(\mathrm{M}) = 8$, $\theta\,(\mathrm{L}) = 8$ und $\theta\,(\text{Reale Produktion}) = 14$). Mit steigender Komplexität κ der untersuchten Datensätze ist in Abbildung 6.26 der Anstieg der benötigten Rechenzeit $t_{\text{Konvergenz}}$ bzw. der entsprechenden Auslastung A des Verfahrens deutlich zu erkennen.

Tabelle 6.12: Komplexitäten und Rechenzeiten der untersuchten Probleme

Datensatz	Perioden θ	Komplexität κ	Konvergenzzeit $\overline{t}_{\text{Konvergenz}}$	Auslastung $\overline{A}$
S–224	4	$4{,}3 \cdot 10^{9}$	7 s	3 %
S–224	6	$2{,}8 \cdot 10^{14}$	10 s	4 %
S–224	8	$1{,}8 \cdot 10^{19}$	15 s	6 %
S–224	10	$1{,}2 \cdot 10^{24}$	15 s	6 %
S–224	12	$7{,}9 \cdot 10^{28}$	15 s	6 %
S–224	14	$5{,}2 \cdot 10^{33}$	17 s	7 %
S–224	$\boldsymbol{16 = \theta_{\text{optimal}}}$	$\mathbf{3{,}4 \cdot 10^{38}}$	**22 s**	**9 %**
S–224	18	$2{,}2 \cdot 10^{42}$	33 s	14 %
S–224	20	$1{,}5 \cdot 10^{48}$	31 s	13 %
M–424	4	$1{,}8 \cdot 10^{19}$	36 s	15 %
M–424	6	$7{,}9 \cdot 10^{28}$	60 s	25 %
M–424	$\boldsymbol{8 = \theta_{\text{optimal}}}$	$\mathbf{3{,}4 \cdot 10^{38}}$	**84 s**	**35 %**
M–424	10	$1{,}4 \cdot 10^{48}$	101 s	42 %
M–424	12	$6{,}3 \cdot 10^{57}$	91 s	38 %
M–424	14	$2{,}7 \cdot 10^{67}$	86 s	36 %
M–424	16	$1{,}2 \cdot 10^{77}$	84 s	35 %
M–424	18	$5{,}0 \cdot 10^{86}$	86 s	36 %
M–424	20	$2{,}1 \cdot 10^{96}$	91 s	38 %
L–824	4	$3{,}4 \cdot 10^{38}$	52 s	52 %
L–824	6	$6{,}3 \cdot 10^{57}$	180 s	75 %
L–824	$\boldsymbol{8 = \theta_{\text{optimal}}}$	$\mathbf{1{,}2 \cdot 10^{77}}$	**223 s**	**93 %**
L–824	10	$2{,}1 \cdot 10^{96}$	223 s	93 %
L–824	12	$3{,}9 \cdot 10^{115}$	226 s	94 %
L–824	14	$7{,}3 \cdot 10^{134}$	223 s	93 %
L–824	16	$1{,}3 \cdot 10^{154}$	226 s	94 %
L–824	18	$2{,}5 \cdot 10^{173}$	226 s	94 %
L–824	20	$4{,}6 \cdot 10^{192}$	214 s	89 %
Reale Produktion	8	$1{,}4 \cdot 10^{30}$	125 s	52 %
Reale Produktion	10	$5{,}9 \cdot 10^{37}$	175 s	73 %
Reale Produktion	12	$1{,}7 \cdot 10^{45}$	175 s	73 %
Reale Produktion	$\boldsymbol{14 = \theta_{\text{optimal}}}$	$\mathbf{5{,}9 \cdot 10^{52}}$	**192 s**	**80 %**
Reale Produktion	16	$2{,}0 \cdot 10^{60}$	216 s	90 %

Zwischen der Komplexität κ des jeweiligen Planungsproblems und der Auslastung A des Planungsverfahrens ist grafisch näherungsweise zunächst ein linearer Zusammenhang identifizierbar. Aufgrund der logarithmischen Skalierung der Komplexität κ (x-Achse) lässt dies den Schluss auf einen näherungsweise logarithmischen Zusammenhang zwischen der Komplexität und der zu erwartenden Auslastung bzw. Rechenzeit des Verfahrens zu (siehe Formel 6.39).

$$A \sim \log(\kappa) \tag{6.39}$$

Die Ergebnisse aus Abbildung A.47 sind in Tabelle 6.12 zusammengefasst. Neben der Bezeichnung des jeweiligen Datensatzes, der jeweiligen Anzahl an Planungsperioden θ sowie der entsprechenden Komplexität κ sind in der Tabelle jeweils die mittlere Rechenzeit $\overline{t}_{\text{Konvergenz}}$ sowie die mittlere Auslastung $\overline{A}$ aufgeführt. Zusätzlich sind die Werte für die jeweils optimale Anzahl an Planungsperioden θ_{optimal} aufgeführt und die entsprechenden Tabellenzeilen in fett dargestellt.

Insgesamt lassen sich aus der Auswertung dieser Ergebnisse die folgenden Schlussfolgerungen ziehen:

- Die Auslastung steigt während der Erhöhung der Komplexität κ eines Datensatzes durch die Verfeinerung des Zeitgitters (Planungsperioden θ) zunächst proportional zu dem Logarithmus der Komplexität $\log(\kappa)$ an. Das Wachstum des benötigten Rechenaufwands ist dementsprechend grundätzlich signifikant schwächer als das Wachstum der Komplexität. Es gilt also näherungsweise Formel 6.40.

$$A \approx K \cdot \log(\kappa) \tag{6.40}$$

- Bei der Berechnung unterschiedlicher Datensätze mit der gleichen Komplexität (siehe beispielsweise $\kappa = 3{,}4 \cdot 10^{38}$ mit $\theta(\text{S}) = 16$, $\theta(\text{M}) = 8$ und $\theta(\text{L}) = 4$) steigt die Auslastung offensichtlich zusätzlich mit der Größe des Datensatzes. Bei dem oben beschriebenen proportionalen Zusammenhang wächst folglich die Steigung der zugehörigen Geradenfunktion (siehe Formel 6.40). Für die Steigung K gilt Formel 6.41.

$$K(\text{S}) < K(\text{M}) < K(\text{L}) \tag{6.41}$$

- Dieser Effekt ist möglicherweise auf den wachsenden Rechen- und Verwaltungsaufwand des PC-Systems (Speicherverwaltung, Speicherzuweisungen, Kopieren und Transferieren von Daten im Arbeitsspeicher etc.) bei steigender Problemgröße zurückzuführen. Diese rechnerinternen Verwaltungsoperationen nehmen mit dem Speicherbedarf der Datensätze zu und sind unabhängig von der rein mathematischen Komplexität κ. Der Speicherbedarf der Datensätze ist wiederum abhängig von der jeweiligen Problemgröße (Anzahl der Stufen c, Anzahl der Maschinen m_k, Anzahl der Auftragsfamilien $o_{k,i}$, Anzahl der Jobs n etc.).

- Dies kann desweiteren als eine Erklärung für den – im Gegensatz zu den randomisierten Datensätzen – steileren Verlauf der realen Datensätze herangezogen werden (siehe Abbildung 6.26, rot). Dabei wird der oben beschriebene Effekt des erhöhten Verwaltungsaufwands durch die größeren Datenmengen (insbesondere des Auftragsbestands) des realen Datensatzes noch weiter verstärkt. Damit gilt Formel 6.42 für die Steigung $K\,(\mathrm{R})$ der Auslastung der realen Produktion R.

$$K\,(\mathrm{R}) > K\,(\mathrm{L}) > K\,(\mathrm{M}) > K\,(\mathrm{S}) \tag{6.42}$$

- Wird die Komplexität eines Datensatzes durch die Variation der Planungsperioden θ über das gefundene Optimum hinaus weiter gesteigert (siehe Abbildungen 6.5 bis 6.7 und 6.18), ist in Bezug auf die Auslastung eine Form von Sättigungsverhalten erkennbar (siehe Abbildung A.47 Datenreihe M–424, blau). Dies ist ein Indiz für die Tatsache, dass für die Anzahl der Planungsperioden θ wirklich ein Optimum gefunden wurde.

- Auch bei dem Datensatz L–824 ist eine Form von Sättigungsverhalten erkennbar (Abbildung A.47, grün). Hier ist das Planungsverfahren bereits voll ausgelastet ($A \approx 100\,\%$). Es ist daher nicht auszuschließen, dass bei dem Datensatz L–824 unter Zuhilfenahme von mehr Rechenkapazität (Erhöhung der Rechenleistung oder Erhöhung der Rechenzeit) möglicherweise noch bessere Ergebnisse erzielt werden können.

In Bezug auf die Größe von Planungsproblemen, die mit diesem Planungsverfahren gerechnet werden können, lassen sich auf Basis dieser Ergebnisse die folgenden Schlussfolgerungen ziehen:

- Die Ergebnisse aus Abbildungen 6.26 und A.47 sowie aus Tabelle 6.12 zeigen, dass die Größe des Datensatzes L–824 unter den im Rahmen dieser Arbeit gegebenen Randbedingungen (Hardware, Rechenzeit etc.) das Planungsverfahren bereits voll auslastet bzw. möglicherweise bereits überlastet.

- Die Ergebnisse lassen allerdings den Schluss zu, dass eine Erhöhung der Rechenkapazität – durch Erhöhung der Rechnerleistung oder durch Erhöhung der Rechenzeit – trotzdem eine weitere Erhöhung der Problemgrößen zulässt. Diese Schlussfolgerung beruht auf den im Rahmen dieser Arbeit auf Basis von vier verschiedenen Datensätzen durchgeführten Studien mit schrittweiser Erhöhung der Komplexität κ (siehe Abbildung A.47).

- Nach heutigem Wissensstand sollten größere Probleme als L–824 also grundsätzlich gerechnet werden können.

- Für die Berechnung größerer Probleme sollte allerdings die Rechenkapazität (Rechnerleistung oder Rechenzeit) erhöht werden. Eine Hilfestellung zur Auslegung der benötigten Rechenkapazität kann Abbildung 6.26 geben.

- Größere Probleme sollten zunächst durch Simulationsrechnungen und anschließend auf Basis geeigneter realer Daten beurteilt und die jeweilige Lösungsqualität entsprechend überprüft werden. Dabei ist eine schrittweise Erhöhung der Problemgröße sowie die statistische Absicherung der Rechenstudien empfehlenswert. Das weiterführende Verhalten des Faktors K sollte dabei unbedingt beobachtet werden.
- Es ist zu beachten, dass für größere Probleme mit entsprechend höherer Komplexität die optimalen Parameterausprägungen des Planungsverfahrens (siehe Abschnitte 6.2 und 6.3) unter Umständen durch weiterführende Parameterstudien ermittelt werden sollten, um die bestmöglichen Planungsergebnisse erzielen zu können.

7 Zusammenfassung und Ausblick

Das Ziel der vorliegenden Arbeit war die Entwicklung eines Planungsverfahrens für die Lösung von bestimmten Flexible Flow Shop Problemen, welches für den Einsatz in der operativen Produktionsplanung und -steuerung realer Betriebe geeignet ist. Dieses Ziel wurde vollumfänglich erreicht.

Eine wesentliche Anforderung an das Planungsverfahren bestand darin, dass mit heute üblicher Rechnerhardware (Stand 2020) auch für größere Planungsprobleme mit mehreren Produktionsstufen mit jeweils mehreren Maschinen in hinreichend kurzer Zeit hochwertige, praktisch brauchbare Lösungen generiert werden können. Bei den im Rahmen dieser Arbeit behandelten Flexible Flow Shop Problemen handelte es sich um eine bislang weitestgehend unbehandelte Problemklasse. Die behandelte Problemklasse wird durch die nachfolgend aufgeführten Restriktionen charakterisiert.

Einerseits handelte es sich um die Berücksichtigung reihenfolgeabhängiger Rüstzeiten mit maschinenspezifischen Auftragsfamilien, die Berücksichtigung stufenspezifischer Auftragsstapel sowie um die Berücksichtigung begrenzter Pufferkapazitäten und Rohmaterialien. Weiterhin wurden Restriktionen berücksichtigt, durch welche nicht jeder Fertigungsauftrag in jeder Produktionsstufe bearbeitet werden musste und nicht jede Maschine in der Lage war, jeden Fertigungsauftrag zu bearbeiten. Darüber hinaus wurde die Anforderung gestellt, nahezu beliebig große Planungsprobleme mit beliebig vielen Produktionsstufen mit jeweils beliebig vielen, unterschiedlichen Maschinen durch das Verfahren behandeln zu können.

Als wichtige Randbedingungen für die Entwicklung des Verfahrens stand dabei die Forderung nach einer multikriteriellen Optimierung mit der beliebigen Gewichtung von Teilzielen im Vordergrund. Desweiteren wurden an das Verfahren Anforderungen in Bezug auf die Adaptivität an unterschiedliche Problemgrößen und Rechenleistungen gestellt. Darüber hinaus wurde an der Anforderung festgehalten, dass das Verfahren durch die Auswahl unterschiedlicher Planungsregeln sowohl nachvollziehbar als auch beeinflussbar bleibt.

Um das oben aufgeführte Ziel zu erreichen, wurde in Kapitel 2 zunächst die in dieser Arbeit behandelte Problemklasse mit den zur Beschreibung derartiger Sachverhalte üblichen Formalismen und Konventionen dargestellt. Eine umfangreiche Literaturrecherche zeigte, welche Werkzeuge und Methoden derzeit für die Lösung von Flexible Flow Shop Problemen angewendet werden. Eine weitergehende Literaturrecherche führte zu dem Ergebnis, dass die in dieser Arbeit behandelte Problemklasse sowie die angewendete Methodik noch weitestgehend unerforscht war. Vorherige Untersuchungen an der in dieser Arbeit

behandelten Klasse von Flexible Flow Shop Problemen konnten in der Literatur nicht nachgewiesen werden.

Um den gestellten Anforderungen gerecht zu werden, wurde in Kapitel 3 eine Methodik entwickelt, die sich insbesondere für Probleme mit reihenfolgeabhängigen Rüstzeiten eignet. Das Verfahren ist an Vorgehensweisen angelehnt, die sich auch in der manuellen Planung praktisch bewährt haben. Die Produktionsaufträge werden dabei an den einzelnen Maschinen in Auftragsfamilien klassifiziert, welche an einer Maschine jeweils ohne Umrüstung gefertigt werden können.

Entscheidungen über die Umrüsthäufigkeit und die Umrüstzeitpunkte der verschiedenen Maschinen beeinflussen die Produktionseffizienz sehr stark. Um diese Entscheidungen mit der erforderlichen Qualität treffen zu können, wurden leistungsfähige metaheuristische Optimierungsverfahren eingesetzt. Die Bearbeitungsreihenfolgen innerhalb der einzelnen Auftragsfamilien haben dagegen einen geringeren Einfluss auf die Produktionseffizienz. Zur Bestimmung dieser Bearbeitungsreihenfolgen kamen deshalb heuristische Verfahren zum Einsatz, die nur einen geringen Rechenaufwand erzeugen. Auf diese Weise wurde einerseits eine hohe Planungsgüte erzielt, andererseits wurde der Rechenaufwand auch für größere Planungsprobleme im beherrschbaren Rahmen gehalten.

In der einschlägigen Literatur konnten keine Veröffentlichungen gefunden werden, welche die oben skizzierte Methodik auf die Planung der hier behandelten Klasse von Flexible Flow Shop Problemen anwenden.

Für die Entwicklung des eigentlichen Planungsverfahrens wurde in Kapitel 4 zunächst ein gemischt-ganzzahliges Optimierungsmodell formuliert. Für die Erzeugung zulässiger Lösungen des Modells wurde anschließend ein Lösungsgenerator auf Basis einer ereignisdiskreten Simulationsmethode entwickelt. Dabei wurden unterschiedliche heuristische Planungsregeln implementiert, welche individuell für jede Produktionsstufe separat ausgewählt werden können.

Für die Anwendung von geeigneten Suchverfahren wurden in Kapitel 5 sowohl ein Genetischer Algorithmus als auch ein Ameisenalgorithmus entwickelt. Für die beiden Optimierungsalgorithmen wurden dabei zunächst die entsprechenden Grundlagen, Formalismen und Mechanismen erläutert und hergeleitet. Anschließend wurden die beiden Algorithmen jeweils mit dem Planungsverfahren gekoppelt.

In Kapitel 6 wurden schließlich umfangreiche Untersuchungen an dem in dieser Arbeit entwickelten Verfahren durchgeführt. Dafür wurden zunächst drei randomisierte Testdatensätze erzeugt, die sich bezüglich der Anzahl der Produktionsstufen unterschieden (S: small, 2 Stufen; M: medium, 4 Stufen; L: large, 8 Stufen). Darauf aufbauend wurde die Performance der beiden Optimierungsalgorithmen im Zusammenhang mit dem Planungsverfahren zunächst optimiert und anschließend miteinander verglichen.

Dabei stellte sich heraus, dass für die Parametrierung des Genetischen Algorithmus wesentlich umfangreichere Studien nötig waren. Es zeigte sich allerdings, dass die mit dem Genetischen Algorithmus erzielbaren Ergebnisse die Resultate des Ameisenalgorithmus insgesamt deutlich übertrafen. Nur bei dem kleinsten untersuchten Planungsproblem (S)

zeigte der Ameisenalgorithmus einen geringfügigen Vorteil gegenüber dem Genetischen Algorithmus. Desweiteren zeigte der Genetische Algorithmus bezüglich der Parameteroptimierung wesentlich geringere Einflüsse der Problemdaten als der Ameisenalgorithmus.

Ein weiterer Vorteil des Genetischen Algorithmus in Bezug auf die untersuchten Fragestellungen bestand darin, dass die optimierten Parameter deutlich weniger von den untersuchten Datensätzen abhingen als bei dem Ameisenalgorithmus. Alle weiteren Untersuchungen wurden deshalb mithilfe des Genetischen Algorithmus durchgeführt. In weiteren Studien wurde der Einfluss der zeitlichen Diskretisierung auf die Qualität der Planungsergebnisse untersucht. Die drei randomisierten Testdatensätze S, M und L wurden mit einer unterschiedlichen Anzahl an Planungsperioden bei gleichbleibendem Planungshorizont analysiert.

Anschließend wurden die Einflüsse der unterschiedlichen Planungsregeln auf die Planungsergebnisse an den drei Testdatensätzen untersucht. Es zeigte sich, dass die erzielbaren Resultate nur geringe Abhängigkeiten von den jeweiligen Problemdatensätzen aufwiesen. In Bezug auf die in dieser Arbeit untersuchte Gewichtung der Zielgrößen konnten daher Aussagen über eine zu bevorzugende Planungsregel getroffen werden.

Abschließend wurde die praktische Anwendbarkeit des Verfahrens anhand einer Fallstudie aus der industriellen Holzverarbeitung dargestellt. Das Planungsverfahren wurde in der Produktionsplanung und -steuerung eines produzierenden Unternehmens in Form eines Entscheidungsunterstützungssystems implementiert. Mithilfe des Planungsverfahrens wurden in dieser Arbeit entsprechende Ergebnisse auf Basis realer Datensätze des Unternehmens erzeugt und analysiert. Das System befindet sich in dem Betrieb seit ca. 1,5 Jahren (Stand 2020) erfolgreich im praktischen Einsatz und hat die konventionelle Planung de facto ersetzt.

Die Produktionsleitung folgt in nahezu allen Fällen den Empfehlungen des im Rahmen dieser Arbeit entwickelten automatischen Planungssystems. Basierend auf den Ergebnissen des Planungssystems können daher sinnvolle und gute Entscheidungen mit einer kurzen Reaktionszeit und einer hohen Sicherheit getroffen werden. Dadurch ergeben sich wesentliche Vorteile in Bezug auf die Entscheidungsprozesse im Rahmen der operativen Produktionsplanung und -steuerung.

Die Betriebsdaten des Unternehmens werden in den Datenbanken eines ERP-Systems in tabellarischer Form vorgehalten. Sie bilden die Entscheidungsgrundlage der operativen Produktionsplanung und -steuerung. Durch die Aufbereitung der Daten auf Basis des im Rahmen dieser Arbeit entwickelten Planungsverfahrens wird deren effiziente und vollständige Nutzung erst ermöglicht.

Weitere Vorteile für das Unternehmen spiegeln sich daher in einer geringen Fehlerquote und einer Entlastung der Mitarbeiter wider. Ständig auftretende Ereignisse wie beispielsweise die Anlieferung von Materialien oder die Fertigstellung von Fertigungsaufträgen erfordern ein ständiges Reagieren der operativen Produktionsplanung und -steuerung. Die Unterstützung durch das Entscheidungssystem bedeutet für den Menschen in diesen Situationen eine wesentliche Entlastung sowohl in Bezug auf den Stresslevel als auch in

Bezug auf die Fehleranfälligkeit. Aus Sicht des Unternehmens werden die Entscheidungen folglich schneller, besser und sicherer getroffen. Gleichzeitig kann der durch das System entlastete Mitarbeiter an anderen Stellen sinnvoll eingesetzt werden.

Das vorgestellte Planungsverfahren lässt sich auf vielfältige praktische Anwendungen übertragen. Insbesondere ist es für Produktionssysteme geeignet, bei welchen reihenfolgeabhängige Rüstzeiten auftreten und eine sinnvolle Bildung von Auftragsfamilien möglich ist. Das Verfahren eignet sich für Prozesse sowohl mit Trenn- als auch mit Fügeprozessen und berücksichtigt dabei die begrenzten Ressourcen in Bezug auf die Rohwarenlager sowie die Zwischenpuffer. Anwendungen finden sich beispielsweise in der Holz- und Möbelindustrie mit Lackierstraßen sowie mit Trenn- und Fügeverfahren, aber auch in weiteren Industriezweiten wie beispielsweise in der Metallverarbeitung.

Insgesamt konnte das Ziel, ein Planungsverfahren zu entwickeln, das sich für die Planung der weiter vorne definierten Flexible Flow Shop Probleme mit den aus der Praxis abgeleiteten Restriktionen eignet, in Bezug auf die untersuchten Beispiele im vollen Umfang erreicht werden. Die im Rahmen dieser Arbeit durchgeführten Untersuchungen eignen sich insbesondere als Basis für weitere Studien auch in Bezug auf die weitergehende Optimierung des Verfahrens.

Von hoher Bedeutung sind beispielsweise Analysen über die Einflüsse unterschiedlicher Zielgrößendefinitionen im Zusammenhang mit unterschiedlichen Planungsheuristiken anhand unterschiedlicher randomisierter sowie realer Daten. Auch die Entwicklung und Implementierung neuartiger Planungsregeln und Zielsetzungen stellen im Zusammenhang mit dem Verfahren einen weiteren interessanten Forschungszweig dar.

Beispielsweise stellen Zielsetzungen auf Basis von Energie- und Umweltfaktoren einen interessanten Forschungszweig dar. So könnte die in dieser Arbeit behandelte Problemklasse um die Optimierung der Energiekosten oder des Schadstoffausstoßes der Maschinen erweitert werden.

Auf Basis der im Rahmen dieser Arbeit durchgeführten Untersuchungen wurde eine Abschätzung der Auslastung des Planungsverfahrens in Abhängigkeit der unterschiedlichen Problemgrößen durchgeführt. Die Größe von Planungsproblemen, die mit der gegebenen Rechenkapazität gerechnet werden können, konnte darauf aufbauend bewertet werden. Die Erforschung noch größerer Probleme mit entsprechender Erhöhung der Rechenkapazität stellt folglich ein interessantes Forschungsgebiet dar, bei welchem die Ergebnisse dieser Arbeit als wichtige Grundlage herangzogen werden können.

Darüber hinaus sind Erweiterungen des Planungsverfahrens um zusätzliche Randbedingungen und Restriktionen wie beispielsweise die Berücksichtigung von Maschinenausfällen denkbar. Auch die operative Produktionsplanung und -steuerung mehrerer voneinander abhängiger Produktionsstandorte eines Unternehmens durch ein einziges, übergreifendes Planungsverfahren kann im Sinne der fortschreitenden Globalisierung eine bedeutende Forschungsrichtung darstellen. Dahingehende Weiterentwicklungen des in dieser Arbeit erarbeiteten Planungsverfahrens sind denkbar und sinnvoll.

Desweiteren bestehen auf Grundlage der vorliegenden Arbeit bedeutende Potenziale in Bezug auf die zusammen mit dem Planungsverfahren einsetzbaren Metaheuristiken.

Einerseits sind Weiterentwicklungen und Abwandlungen an den im Rahmen dieser Arbeit angewendeten Metaheuristiken denkbar. Andererseits gibt es eine Vielzahl wesentlich weniger populärer Metaheuristiken und Suchverfahren wie beispielsweise die Streusuche oder die Partikelschwarmoptimierung. Die Anwendungen dieser Metaheuristiken im Zusammenhang mit dem Planungsverfahren stellen ein vielversprechendes Forschungsgebiet dar.

Literaturverzeichnis

[Acero-Dominguez et al. 2004] Acero-Dominguez, M. J. ; Paternina-Arboleda, C. D.: Scheduling jobs on a k-stage flexible flow shop using a TOC-based (bottleneck) procedure. In: Systems and Information Engineering Design Symposium (Hrsg.): *Proceedings of the 2004 IEEE Systems and Information Engineering Design Symposium, 2004*, IEEE, 2004, S. 295–298. – ISBN 0-9744559-2-X

[Akrami et al. 2006] Akrami, B. ; Karimi, B. ; Moattar Hosseini, S. M.: Two metaheuristic methods for the common cycle economic lot sizing and scheduling in flexible flow shops with limited intermediate buffers: The finite horizon case. In: *Applied Mathematics and Computation* 183 (2006), Nr. 1, S. 634–645. – ISSN 00963003

[Alaykýran et al. 2007] Alaykýran, Kemal ; Engin, Orhan ; Döyen, Alper: Using ant colony optimization to solve hybrid flow shop scheduling problems. In: *The International Journal of Advanced Manufacturing Technology* 35 (2007), Nr. 5-6, S. 541–550. – ISSN 0268-3768

[Alisantoso et al. 2003] Alisantoso, D. ; Khoo, L. P. ; Jiang, P. Y.: An immune algorithm approach to the scheduling of a flexible PCB flow shop. In: *The International Journal of Advanced Manufacturing Technology* 22 (2003), Nr. 11-12, S. 819–827. – ISSN 0268-3768

[Allaoui et al. 2006] Allaoui, Hamid ; Artiba, Abdelhakim: Scheduling two-stage hybrid flow shop with availability constraints. In: *Computers & Operations Research* 33 (2006), Nr. 5, S. 1399–1419. – ISSN 03050548

[Almeder et al. 2013] Almeder, Christian ; Hartl, Richard F.: A metaheuristic optimization approach for a real-world stochastic flexible flow shop problem with limited buffer. In: *International Journal of Production Economics* 145 (2013), Nr. 1, S. 88–95. – ISSN 09255273

[Baker et al. 2019] Baker, Kenneth R. ; Trietsch, Dan: *Principles of sequencing and scheduling.* Second edition. Hoboken, NJ, USA : Wiley, 2019 (Wiley series in operations research and management science). – ISBN 9781119262589

[Behnamian et al. 2011] Behnamian, J. ; Fatemi Ghomi, S.M.T.: Hybrid flow-shop scheduling with machine and resource-dependent processing times. In: *Applied Mathematical Modelling* 35 (2011), Nr. 3, S. 1107–1123

[Blazewicz et al. 2019] Blazewicz, Jacek ; Ecker, Klaus H. ; Pesch, Erwin ; Schmidt, Günter ; Sterna, Malgorzata ; Weglarz, Jan: *Handbook on Scheduling: From Theory to Practice*. 2nd ed. 2019 (International Handbooks on Information Systems). – ISBN 978-3-319-99848-0

[Blum et al. 2016] Blum, Christian ; Raidl, Günther R.: *Hybrid Metaheuristics: Powerful Tools for Optimization*. Cham and s.l. : Springer International Publishing, 2016 (Artificial Intelligence). – ISBN 9783319308821

[Boll 2003] Boll, Helmut: *Evolutionäre Verfahren zur Optimierung von Produktionsplänen mittels impliziter Kooperation*. Aachen, RWTH Aachen, Dissertation, 2003

[Bracht et al. 2011] Bracht, Uwe ; Geckler, Dieter ; Wenzel, Sigrid: *Digitale Fabrik: Methoden und Praxisbeispiele*. Berlin u.a. : Springer, 2011 (VDI-Buch). – ISBN 978-3-540-89038-6

[Brucker 2007] Brucker, Peter: *Scheduling Algorithms*. Dordrecht : Springer, 2007. – ISBN 978-3-540-69515-8

[Brucker et al. 2012] Brucker, Peter ; Knust, Sigrid: *Complex Scheduling*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2012. – ISBN 978-3-642-23928-1

[Burke et al. 2014a] Burke, Edmund K. ; Kendall, Graham: Introduction. In: Burke, Edmund K. (Hrsg.) ; Kendall, Graham (Hrsg.): *Search Methodologies*. Boston, MA : Springer US, 2014, S. 1–18. – ISBN 978-1-461-46940-7

[Burke et al. 2014b] Burke, Edmund K. (Hrsg.) ; Kendall, Graham (Hrsg.): *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*. 2nd ed. 2014. Boston, MA : Springer US, 2014. – ISBN 978-1-461-46940-7

[Cevik Onar et al. 2016] Cevik Onar, Sezi ; Öztaysi, Basar ; Kahraman, Cengiz ; Yanik, Seda ; Senvar, Özlem: A Literature Survey on Metaheuristics in Production Systems. In: Talbi, El-Ghazali (Hrsg.) ; Yalaoui, Farouk (Hrsg.) ; Amodeo, Lionel (Hrsg.): *Metaheuristics for Production Systems*. 2016 (Operations Research/Computer Science Interfaces Series), S. 1–24. – ISBN 978-3-319-23349-9

[Chaaben et al. 2013] Chaaben, Nadia ; Mellouli, Racem ; Masmoudi, Faouzi: Evolutionary Metaheuristic Based on Genetic Algorithm: Application to Hybrid Flow Shop Problem with Availability Constraints. In: Jarboui, Bassem (Hrsg.) ; Siarry, Patrick (Hrsg.) ; Teghem, Jacques (Hrsg.): *Metaheuristics for production scheduling*. London and Hoboken, NJ : ISTE and Wiley, 2013 (Automation-control and industrial engineering series), S. 127–152. – ISBN 9781118731598

[Chen et al. 2009] Chen, Chun-Lung ; Chen, Chuen-Lung: Bottleneck-based heuristics to minimize total tardiness for the flexible flow line with unrelated parallel machines. In: *Computers & Industrial Engineering* 56 (2009), Nr. 4, S. 1393–1401. – ISSN 03608352

[Chen et al. 2007] Chen, Lu ; Bostel, Nathalie ; Dejax, Pierre ; Cai, Jianguo ; Xi, Lifeng: A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. In: *European Journal of Operational Research* 181 (2007), Nr. 1, S. 40–58. – ISSN 03772217

[Cheng et al. 2012] Cheng, T.C.E. ; Lin, B.M.T. ; Huang, H. L.: Resource-constrained flowshop scheduling with separate resource recycling operations. In: *Computers & Operations Research* 39 (2012), Nr. 6, S. 1206–1212. – ISSN 03050548

[Chernigovskiy et al. 2017] Chernigovskiy, A. S. ; Kapulin, D. V. ; Noskova, E. E. ; Yamskikh, T. N. ; Tsarev, R. Y.: Production scheduling with ant colony optimization. In: *IOP Conference Series: Earth and Environmental Science* 87 (2017), S. 062002. – ISSN 1755-1307

[Colorni et al. 1994] Colorni, Alberto ; Dorigo, Marco ; Maniezzo, Vittorio ; Trubian, Marco: Ant system for Job-shop Scheduling. In: *Belgian journal of operations research, statistics and computer science* (1994), Nr. 34

[Costa et al. 2014] Costa, Antonio ; Cappadonna, Fulvio A. ; Fichera, Sergio: A novel genetic algorithm for the hybrid flow shop scheduling with parallel batching and eligibility constraints. In: *The International Journal of Advanced Manufacturing Technology* 75 (2014), Nr. 5-8, S. 833–847. – ISSN 0268-3768

[Dempe et al. 2006] Dempe, Stephan ; Schreier, Heiner: *Operations Research: Deterministische Modelle und Methoden*. 1. Aufl. Wiesbaden : Teubner, 2006 (Teubner Studienbücher Wirtschaftsmathematik). – ISBN 3-519-00448-8

[Deroussi 2016] Deroussi, Laurent: *Metaheuristics for logistics*. London, England and Hoboken, New Jersey : ISTE and Wiley, 2016 (Computer Engineering Series). – ISBN 9781119136583

[Dhiflaoui et al. 2018] Dhiflaoui, Mondher ; Nouri, Houssem E. ; Driss, Olfa B.: Dual-Resource Constraints in Classical and Flexible Job Shop Problems: A State-of-the-Art Review. In: *Procedia Computer Science* 126 (2018), S. 1507–1515. – ISSN 18770509

[Domschke et al. 2005] Domschke, Wolfgang ; Drexl, Andreas: *Einführung in Operations Research: Mit 63 Tabellen*. 6., überarb. und erw. Aufl. Berlin u.a. : Springer, 2005 (Springer-Lehrbuch). – ISBN 3-540-23431-4

[Dorigo et al. 2005] Dorigo, Marco ; Blum, Christian: Ant colony optimization theory: A survey. In: *Theoretical Computer Science* 344 (2005), Nr. 2-3, S. 243–278. – ISSN 03043975

[Dorigo et al. 1997] Dorigo, Marco ; Gambardella, Luca M.: Ant colony system: A cooperative learning approach to the traveling salesman problem. In: *IEEE Transactions on Evolutionary Computation* 1 (1997), Nr. 1, S. 53–66

[Dorigo et al. 1991] Dorigo, Marco ; Maniezzo, Vittorio ; Colorni, Alberto: *Positive Feedback as a Search Strategy: Technical Report No. 91-016*. Politecnico di Milano : Dip .Electronico, 1991

[Dorigo et al. 1996] Dorigo, Marco ; Maniezzo, Vittorio ; Colorni, Alberto: Ant system: Optimization by a colony of cooperating agents. In: *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society* 26 (1996), Nr. 1, S. 29–41. – ISSN 1083-4419

[Dorigo et al. 2004] Dorigo, Marco ; Stützle, Thomas: *Ant Colony Optimization*. Cambridge, Mass : MIT Press, 2004. – ISBN 978-0-262-04219-2

[Du et al. 2016] Du, Ke-Lin ; Swamy, Madisetti N. S.: *Search and Optimization by Metaheuristics: Techniques and Algorithms Inspired by Nature*. Cham : Birkhäuser, 2016. – ISBN 978-3-319-41192-7

[Ebrahimi et al. 2014] Ebrahimi, M. ; Fatemi Ghomi, S.M.T. ; Karimi, B.: Hybrid flow shop scheduling with sequence dependent family setup time and uncertain due dates. In: *Applied Mathematical Modelling* 38 (2014), Nr. 9-10, S. 2490–2504

[Eley 2012] Eley, Michael: *Simulation in der Logistik: Einführung in die Erstellung ereignisdiskreter Modelle unter Verwendung des Werkzeuges "Plant Simulation"*. Berlin Heidelberg : Springer Berlin Heidelberg, 2012 (Springer-Lehrbuch). – ISBN 978-3-642-27372-8

[Emmons et al. 2013] Emmons, Hamilton ; Vairaktarakis, George: *International series in operations research & management science*. Bd. 182: *Flow shop scheduling: Theoretical results, algorithms, and applications*. New York : Springer, 2013. – ISBN 978-1-4614-5151-8

[Engelhardt 2015] Engelhardt, Philipp R.: *System für die RFID-gestützte situationsbasierte Produktionssteuerung in der auftragsbezogenen Fertigung und Montage*. München, Technische Universität München, Dissertation, 2015

[Engin et al. 2004] Engin, Orhan ; Döyen, Alper: A new approach to solve hybrid flow shop scheduling problems by artificial immune system. In: *Future Generation Computer Systems* 20 (2004), Nr. 6, S. 1083–1095

[Ernst et al. 2019] Ernst, Andreas ; Fung, Joey ; Singh, Gaurav ; Zinder, Yakov: Flexible flow shop with dedicated buffers. In: *Discrete Applied Mathematics* 261 (2019), S. 148–163

[Esquirol et al. 2008] Esquirol, Patrick ; Lopez, Pierre: Basic Concepts and Methods in Production Scheduling. In: Lopez, Pierre (Hrsg.): *Production scheduling*. London : ISTE, 2008 (Control systems, robotics and manufacturing series), S. 5–32. – ISBN 978-1-84821-017-2

[Fattahi et al. 2014] Fattahi, Parviz ; Hosseini, Seyed Mohammad H. ; Jolai, Fariborz ; Tavakkoli-Moghaddam, Reza: A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations. In: *Applied Mathematical Modelling* 38 (2014), Nr. 1, S. 119–134

[Gambardella et al. 1996] Gambardella, Luca M. ; Dorigo, Marco: Solving symmetric and asymmetric TSPs by ant colonies. In: *Proceedings of IEEE International Conference on Evolutionary Computation*, IEEE, 1996, S. 622–627. – ISBN 0-7803-2902-3

[Geiger 2005] Geiger, Martin J.: *Multikriterielle Ablaufplanung.* Hohenheim, Universität Hohenheim, Dissertation, 2005

[Gerdes et al. 2004] Gerdes, Ingrid ; Klawonn, Frank ; Kruse, Rudolf: *Evolutionäre Algorithmen: Genetische Algorithmen - Strategien und Optimierungsverfahren - Beispielanwendungen.* Wiesbaden : Vieweg+Teubner Verlag, 2004 (Computational Intelligence). – ISBN 978-3-528-05570-7

[Goldberg 1989] Goldberg, David E.: *Genetic algorithms in search, optimization, and machine learning.* Boston : Addison-Wesley, 1989. – ISBN 0-201-15767-5

[Gondek 2011] Gondek, Verena: *Hybrid Flow-Shop Scheduling mit verschiedenen Restriktionen: Heuristische Lösung und LP-basierte untere Schranken.* Duisburg/Essen, Universität Duisburg-Essen, Dissertation, 2011

[Gong et al. 2019] Gong, Guiliang ; Chiong, Raymond ; Deng, Qianwang ; Gong, Xuran: A hybrid artificial bee colony algorithm for flexible job shop scheduling with worker flexibility. In: *International Journal of Production Research* (2019), S. 1–15. – ISSN 0020-7543

[Gong et al. 2017] Gong, Xu ; van der Wee, Marlies ; Pessemier, Toon de ; Verbrugge, Sofie ; Colle, Didier ; Martens, Luc ; Joseph, Wout: Energy- and Labor-aware Production Scheduling for Sustainable Manufacturing: A Case Study on Plastic Bottle Manufacturing. In: *Procedia CIRP* 61 (2017), S. 387–392. – ISSN 22128271

[Graham et al. 1979] Graham, Ronald L. ; Lawler, Eugene L. ; Lenstra, Jan K. ; Kan, Alexander Hendrik George R.: Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey. In: Korte, Bernhard H. (Hrsg.) ; Johnson, Ellis L. (Hrsg.) ; Hammer, P. L. (Hrsg.): *Discrete Optimization II* Bd. 5. Amsterdam and New York and New York : Elsevier and North-Holland Pub. Co, 1979, S. 287–326. – ISBN 978-0-080-86767-0

[Günther et al. 2012] Günther, Hans-Otto ; Tempelmeier, Horst: *Produktion und Logistik.* Berlin/Heidelberg : Springer Berlin Heidelberg, 2012. – ISBN 978-3-642-25164-1

[Gutenschwager et al. 2017] Gutenschwager, Kai ; Rabe, Markus ; Spieckermann, Sven ; Wenzel, Sigrid: *Simulation in Produktion und Logistik*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2017. – ISBN 978-3-662-55744-0

[Gutjahr 2011] Gutjahr, Walter J.: Ant Colony Optimization: Recent Developments in Theoretical Analysis. In: Auger, Anne (Hrsg.) ; Doerr, Benjamin (Hrsg.): *Theory of Randomized Search Heuristics* Bd. 1. WORLD SCIENTIFIC, 2011, S. 225–254. – ISBN 978-981-4282-66-6

[Han et al. 2019] Han ; Lin ; Dong ; Shi: Flexible Flow Shop Scheduling Method with Public Buffer. In: *Processes* 7 (2019), Nr. 10, S. 681. – ISSN 2227-9717

[Han et al. 2018] Han, Zhonghua ; Ma, Xiaofu ; Lv, Zhe ; Sun, Yue: Hybrid flow shop scheduling with finite buffers. In: *International Journal of Simulation and Process Modelling* 13 (2018), Nr. 2, S. 156. – ISSN 1740-2123

[Hanisch et al. 2008] Hanisch, Christoph ; Zare Mehrjerdi, Yahia: RFID–enabled systems: A brief review. In: *Assembly Automation* 28 (2008), Nr. 3, S. 235–245. – ISSN 0144-5154

[Haouari et al. 2008] Haouari, Mohamed ; Hidri, Lotfi: On the hybrid flowshop scheduling problem. In: *International Journal of Production Economics* 113 (2008), Nr. 1, S. 495–497. – ISSN 09255273

[Heger et al. 2015] Heger, Jens ; Hildebrandt, Torsten: Simulationsbasierte Optimierung der Reihenfolgeplanung am Beispiel eines Liniensorters in der Automobilindustrie. In: Rabe, Markus (Hrsg.) ; Clausen, Uwe (Hrsg.): *Simulation in Production and Logistics 2015*. Stuttgart : Fraunhofer Verlag, 2015, S. 11–20. – ISBN 978-3-8396-0936-1

[Holland 1975] Holland, John H.: *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. 2. Cambridge, Mass. : MIT Press, 1975. – ISBN 0-262-58111-6

[Holthaus et al. 1997] Holthaus, Oliver ; Rajendran, Chandrasekharan: Efficient dispatching rules for scheduling in a job shop. In: *International Journal of Production Economics* 48 (1997), Nr. 1, S. 87–105. – ISSN 09255273

[Hui et al. 2000] Hui, Chi-Wai ; Gupta, Avaneesh ; van der Meulen, Harke A.: A novel MILP formulation for short-term scheduling of multi-stage multi-product batch plants with sequence-dependent constraints. In: *Computers & Chemical Engineering* 24 (2000), Nr. 12, S. 2705–2717. – ISSN 00981354

[Huisman et al. 2014] Huisman, Dennis ; Louwerse, Ilse ; Wagelmans, Albert P.: *Operations Research Proceedings 2013: Selected Papers of the International Conference on Operations Research, OR2013, organized by the German Operations Research Society (GOR), the Dutch Society of Operations Research (NGB) and Erasmus University Rotterdam, September 3-6, 2013*. Cham : Springer International Publishing, 2014

(Operations Research Proceedings, GOR (Gesellschaft für Operations Research e.V.)). – ISBN 978-3-319-07000-1

[Isenberg 2017] Isenberg, Florian: *Ein dreistufiger integrierter Planungsansatz zur mehrstufigen Losgrößen- und Reihenfolgeplanung in der spanenden Fertigung.* Paderborn, Universität Paderborn, Dissertation, 2017

[Isenberg 2016] Isenberg, Marc-André: *Produktionsplanung und -steuerung in mehrstufigen Batchproduktionen: Methoden der Loskomposition und -terminierung bei stufenspezifischen Auftragsfamilien.* Bremen, Universität Bremen, Dissertation, 2016

[Jaehn et al. 2019] Jaehn, Florian ; Pesch, Erwin: *Ablaufplanung: Einführung in Scheduling.* Springer Berlin Heidelberg, 2019. – ISBN 978-3-662-58779-9

[Janiak et al. 2007] Janiak, Adam ; Kozan, Erhan ; Lichtenstein, Maciej ; Oğuz, Ceyda: Metaheuristic approaches to the hybrid flow shop scheduling problem with a cost-related criterion. In: *International Journal of Production Economics* 105 (2007), Nr. 2, S. 407–424. – ISSN 09255273

[Jarboui et al. 2013] Jarboui, Bassem (Hrsg.) ; Siarry, Patrick (Hrsg.) ; Teghem, Jacques (Hrsg.): *Metaheuristics for production scheduling.* London and Hoboken, NJ : ISTE and Wiley, 2013 (Automation-control and industrial engineering series). – ISBN 9781118731598

[Javadian et al. 2013] Javadian, Nikbakhsh ; Amiri-Aref, Mehdi ; Hadighi, Ali ; Kazemi, Mohammad ; Moradi, Alireza: Flexible flow shop with sequence-dependent setup times and machine availability constraints. In: *International Journal of Management Science and Engineering Management* 5 (2013), Nr. 3, S. 219–226. – ISSN 1750-9653

[Jenabi et al. 2007] Jenabi, M. ; Fatemi Ghomi, S.M.T. ; Torabi, S. A. ; Karimi, B.: Two hybrid meta-heuristics for the finite horizon ELSP in flexible flow lines with unrelated parallel machines. In: *Applied Mathematics and Computation* 186 (2007), Nr. 1, S. 230–245. – ISSN 00963003

[Johnson 1954] Johnson, S. M.: Optimal two- and three-stage production schedules with setup times included. In: *Naval Research Logistics Quarterly* 1 (1954), Nr. 1, S. 61–68. – ISSN 00281441

[Joo et al. 2013] Joo, Byung J. ; Choi, Yong C. ; Xirouchakis, Paul: Dispatching Rule-based Algorithms for a Dynamic Flexible Flow Shop Scheduling Problem with Time-dependent Process Defect Rate and Quality Feedback. In: *Procedia CIRP* 7 (2013), S. 163–168. – ISSN 22128271

[Jungwattanakit et al. 2009] Jungwattanakit, Jitti ; Reodecha, Manop ; Chaovalitwongse, Paveena ; Werner, Frank: A comparison of scheduling algorithms for flexible flow shop problems with unrelated parallel machines, setup times, and dual

criteria. In: *Computers & Operations Research* 36 (2009), Nr. 2, S. 358–378. – ISSN 03050548

[Kacem 2013] Kacem, Imed: Genetic Algorithms for Solving Flexible Job Shop Scheduling Problems. In: Jarboui, Bassem (Hrsg.) ; Siarry, Patrick (Hrsg.) ; Teghem, Jacques (Hrsg.): *Metaheuristics for production scheduling*. London and Hoboken, NJ : ISTE and Wiley, 2013 (Automation-control and industrial engineering series), S. 19–44. – ISBN 9781118731598

[Kallrath 2013] Kallrath, Josef: *Gemischt-ganzzahlige Optimierung: Modellierung in der Praxis: Mit Fallstudien aus Chemie, Energiewirtschaft, Papierindustrie, Metallgewerbe, Produktion und Logistik*. 2., überarb. u. erw. Aufl. 2013. – ISBN 9783658006891

[Keshavarz et al. 2013] Keshavarz, Taha ; Salmasi, Nasser: Makespan minimisation in flexible flowshop sequence-dependent group scheduling problem. In: *International Journal of Production Research* 51 (2013), Nr. 20, S. 6182–6193. – ISSN 0020-7543

[Kianfar et al. 2012] Kianfar, K. ; Fatemi Ghomi, S.M.T. ; Oroojlooy Jadid, A.: Study of stochastic sequence-dependent flexible flow shop via developing a dispatching rule and a hybrid GA. In: *Engineering Applications of Artificial Intelligence* 25 (2012), Nr. 3, S. 494–506. – ISSN 09521976

[Kim et al. 2009] Kim, Yeong-Dae ; Joo, Byung-Jun ; Shin, Jong-Ho: Heuristics for a two-stage hybrid flowshop scheduling problem with ready times and a product-mix ratio constraint. In: *Journal of Heuristics* 15 (2009), Nr. 1, S. 19–42. – ISSN 1381-1231

[Kis et al. 2005] Kis, Tamás ; Pesch, Erwin: A review of exact solution methods for the non-preemptive multiprocessor flowshop problem. In: *European Journal of Operational Research* 164 (2005), Nr. 3, S. 592–608. – ISSN 03772217

[Klement et al. 2017] Klement, Nathalie ; Silva, Cristóvão ; Gibaru, Olivier: Solving a Discrete Lot Sizing and Scheduling Problem with Unrelated Parallel Machines and Sequence Dependent Setup Using a Generic Decision Support Tool. In: Lödding, Hermann (Hrsg.) ; Riedel, Ralph (Hrsg.) ; Thoben, Klaus-Dieter (Hrsg.) ; Cieminski, Gregor v. (Hrsg.) ; Kiritsis, Dimitris (Hrsg.): *Advances in production management systems*. Cham : Springer, 2017 (IFIP Advances in Information and Communication Technology), S. 459–466. – ISBN 978-3-319-66922-9

[Klemmt et al. 2011] Klemmt, Andreas ; Horn, Sven ; Weigert, Gerald: Simulationsgestützte Optimierung von Fertigungsprozessen in der Halbleiterindustrie. In: März, Lothar (Hrsg.) ; Krug, Wilfried (Hrsg.) ; Rose, Oliver (Hrsg.) ; Weigert, Gerald (Hrsg.): *Simulation und Optimierung in Produktion und Logistik*. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2011 (VDI-Buch), S. 49–64. – ISBN 978-3-6421-4536-0

[Kochhar et al. 1987] Kochhar, Sandeep ; Morris, Robert J.: Heuristic methods for flexible flow line scheduling. In: *Journal of Manufacturing Systems* 6 (1987), Nr. 4, S. 299–314. – ISSN 02786125

[Kozak 2019] Kozak, Jan: *Studies in computational intelligence*. Bd. volume 781: *Decision tree and ensemble learning based on ant colony optimization*. Cham, Switzerland : Springer, 2019. – ISBN 978-3-319-93751-9

[Kramer 2017] Kramer, Oliver: *Studies in computational intelligence*. Bd. 679: *Genetic Algorithm Essentials*. Cham and s.l. : Springer International Publishing, 2017. – ISBN 978-3-319-52155-8

[Krug et al. 2011] Krug, Wilfried ; Schwope, Markus: Simulationsbasierte Reihenfolgeoptimierung in der Produktionsplanung und -steuerung. In: März, Lothar (Hrsg.) ; Krug, Wilfried (Hrsg.) ; Rose, Oliver (Hrsg.) ; Weigert, Gerald (Hrsg.): *Simulation und Optimierung in Produktion und Logistik*. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2011 (VDI-Buch), S. 105–116. – ISBN 978-3-6421-4536-0

[Kurz et al. 2003] Kurz, Mary E. ; Askin, Ronald G.: Comparing scheduling rules for flexible flow lines. In: *International Journal of Production Economics* 85 (2003), Nr. 3, S. 371–388. – ISSN 09255273

[Kurz et al. 2004] Kurz, Mary E. ; Askin, Ronald G.: Scheduling flexible flow lines with sequence-dependent setup times. In: *European Journal of Operational Research* 159 (2004), Nr. 1, S. 66–82. – ISSN 03772217

[Laribi et al. 2016] Laribi, Imane ; Yalaoui, Farouk ; Belkaid, Fayçal ; Sari, Zaki: Heuristics for solving flow shop scheduling problem under resources constraints. In: *IFAC-PapersOnLine* 49 (2016), Nr. 12, S. 1478–1483. – ISSN 24058963

[Law et al. 1997] Law, Averill M. ; Kelton, W. D.: *Simulation modeling and analysis*. 2. edition. New York : McGraw-Hill, 1997 (McGraw-Hill Series in Industrial Engineering and Management Science). – ISBN 0-07-100803-9

[Lee et al. 2019a] Lee, Tian S. ; Loong, Y. T. ; Tan, S. C.: A hybrid genetic-gravitational search algorithm for a multi-objective flow shop scheduling problem. In: *International Journal of Industrial Engineering Computations* (2019), S. 331–348. – ISSN 19232926

[Lee et al. 2019b] Lee, Tian-Soon ; Loong, Ying-Tai: A review of scheduling problem and resolution methods in flexible flow shop. In: *International Journal of Industrial Engineering Computations* (2019), S. 67–88. – ISSN 19232926

[Lemessi et al. 2010] Lemessi, Marco ; Rehbein, Simeon ; Schulze, Thomas: Simulationsbasierte Optimierung von Farbgebungsanlagen. In: Zülch, Gert (Hrsg.): *Integrationsaspekte der Simulation: Technik, Organisation und Personal*. Karlsruhe : KIT Scientific Publ, 2010 (ASIM-Mitteilung), S. 567–574. – ISBN 978-3-86644-558-1

[Leu et al. 2002] Leu, Sou-Sen ; Hwang, Shao-Ting: GA-based resource-constrained flow-shop scheduling model for mixed precast production. In: *Automation in Construction* 11 (2002), Nr. 4, S. 439–452. – ISSN 09265805

[Leung 2004] Leung, Joseph Y-T. (Hrsg.): *Handbook of scheduling: Algorithms, models, and performance analysis*. Boca Raton : Chapman & Hall/CRC, 2004 (Chapman & Hall / CRC computer and information science series). – ISBN 978-1-584883-975

[Liaee et al. 1997] Liaee, Mohammad M. ; Emmons, Hamilton: Scheduling families of jobs with setup times. In: *International Journal of Production Economics* 51 (1997), Nr. 3, S. 165–176. – ISSN 09255273

[Lin et al. 2020] Lin, Chun-Cheng ; Liu, Wan-Yu ; Chen, Yu-Hsiang: Considering stockers in reentrant hybrid flow shop scheduling with limited buffer capacity. In: *Computers & Industrial Engineering* 139 (2020), S. 106154. – ISSN 03608352

[Lin et al. 2013] Lin, Danping ; Lee, C.K.M. ; Ho, William: Multi-level genetic algorithm for the resource-constrained re-entrant scheduling problem in the flow shop. In: *Engineering Applications of Artificial Intelligence* 26 (2013), Nr. 4, S. 1282–1290. – ISSN 09521976

[Linn et al. 1999] Linn, Richard ; Zhang, Wei: Hybrid flow shop scheduling: A survey. In: *Computers & Industrial Engineering* 37 (1999), Nr. 1-2, S. 57–61. – ISSN 03608352

[Liu et al. 2017] Liu, Weibo ; Jin, Yan ; Price, Mark: A new improved NEH heuristic for permutation flowshop scheduling problems. In: *International Journal of Production Economics* 193 (2017), S. 21–30. – ISSN 09255273

[Lödding 2016] Lödding, Hermann: *Verfahren der Fertigungssteuerung: Grundlagen, Beschreibung, Konfiguration*. 3. Auflage. Berlin, Heidelberg : Springer Vieweg, 2016 (VDI-Buch). – ISBN 978-3-662-48458-6

[Logendran et al. 2006] Logendran, Rasaratnam ; deSzoeke, Paula ; Barnard, Faith: Sequence-dependent group scheduling problems in flexible flow shops. In: *International Journal of Production Economics* 102 (2006), Nr. 1, S. 66–86. – ISSN 09255273

[Lopez 2008] Lopez, Pierre (Hrsg.): *Production scheduling*. London : ISTE, 2008 (Control systems, robotics and manufacturing series). – ISBN 978-1-84821-017-2

[Luo et al. 2015] Luo, Hao ; Zhang, Abraham ; Huang, George Q.: Active scheduling for hybrid flowshop with family setup time and inconsistent family formation. In: *Journal of Intelligent Manufacturing* 26 (2015), Nr. 1, S. 169–187. – ISSN 0956-5515

[März et al. 2011a] März, Lothar ; Krug, Wilfried: Kopplung von Simulation und Optimierung. In: März, Lothar (Hrsg.) ; Krug, Wilfried (Hrsg.) ; Rose, Oliver (Hrsg.) ; Weigert, Gerald (Hrsg.): *Simulation und Optimierung in Produktion und*

Logistik. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2011 (VDI-Buch), S. 41–48. – ISBN 978-3-6421-4536-0

[März et al. 2011b] März, Lothar (Hrsg.) ; Krug, Wilfried (Hrsg.) ; Rose, Oliver (Hrsg.) ; Weigert, Gerald (Hrsg.): *VDI-Buch*. Bd. 130: *Simulation und Optimierung in Produktion und Logistik: Praxisorientierter Leitfaden mit Fallbeispielen*. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2011. – ISBN 978-3-6421-4536-0

[Méndez et al. 2001] Méndez, C. A. ; Henning, G. P. ; Cerdá, J.: An MILP continuous-time approach to short-term scheduling of resource-constrained multistage flowshop batch facilities. In: *Computers & Chemical Engineering* 25 (2001), Nr. 4-6, S. 701–711. – ISSN 00981354

[Merkle et al. 2014] Merkle, Daniel ; Middendorf, Martin: Swarm Intelligence. In: Burke, Edmund K. (Hrsg.) ; Kendall, Graham (Hrsg.): *Search Methodologies*. Boston, MA : Springer US, 2014, S. 213–242. – ISBN 978-1-461-46940-7

[Minella et al. 2008] Minella, Gerardo ; Ruiz, Rubén ; Ciavotta, Michele: A Review and Evaluation of Multiobjective Algorithms for the Flowshop Scheduling Problem. In: *INFORMS Journal on Computing* 20 (2008), Nr. 3, S. 451–471. – ISSN 1091-9856

[Moore 1968] Moore, J. M.: An n Job, One Machine Sequencing Algorithm for Minimizing the Number of Late Jobs. In: *Management Science* 15 (1968), Nr. 1, S. 102–109. – ISSN 0025-1909

[Naderi et al. 2014] Naderi, Bahman ; Gohari, Sheida ; Yazdani, Mehdi: Hybrid flexible flowshop problems: Models and solution methods. In: *Applied Mathematical Modelling* 38 (2014), Nr. 24, S. 5767–5780

[Naderi et al. 2009] Naderi, Bashir ; Zandieh, Mostafa ; Khaleghei, Akram ; Roshanaei, Vahid: An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness. In: *Expert Systems with Applications* 36 (2009), Nr. 6, S. 9625–9633. – ISSN 09574174

[Nahhas et al. 2015] Nahhas, Abdulrahman ; Aurich, Paul ; Reggelin, Tobias: Lösung eines Hybrid-Flow-Shop-Maschinenbelegungsproblems mit simulationsbasierter Optimierung. In: Rabe, Markus (Hrsg.) ; Clausen, Uwe (Hrsg.): *Simulation in Production and Logistics 2015*. Stuttgart : Fraunhofer Verlag, 2015, S. 29–38. – ISBN 978-3-8396-0936-1

[Nawaz et al. 1983] Nawaz, Muhammad ; Enscore, E. E. ; Ham, Inyong: A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. In: *Omega* 11 (1983), Nr. 1, S. 91–95. – ISSN 03050483

[Nissen 1997] Nissen, Volker: *Einführung in Evolutionäre Algorithmen: Optimierung nach dem Vorbild der Evolution*. Wiesbaden : Vieweg+Teubner Verlag, 1997 (Computational Intelligence). – ISBN 978-3-528-05499-1

[Niu et al. 2012] Niu, Qun ; Zhou, Taijin ; Fei, Minrui ; Wang, Bing: An efficient quantum immune algorithm to minimize mean flow time for hybrid flow shop problems. In: *Mathematics and Computers in Simulation* 84 (2012), S. 1–25. – ISSN 03784754

[Nyhuis et al. 2012] Nyhuis, Peter ; Wiendahl, Hans-Peter: *Logistische Kennlinien: Grundlagen, Werkzeuge und Anwendungen*. 3. Aufl. 2012. Berlin, Heidelberg : Springer, 2012 (VDI-Buch). – ISBN 978-3-540-92838-6

[Parlier et al. 2020] Parlier, Greg H. ; Liberatore, Federico ; Demange, Marc: *Communications in computer and information science*. Bd. 1162: *Operations research and enterprise systems: 8th International Conference, ICORES 2019, Prague, Czech Republic, February 19-21, 2019, Revised selected papers*. Cham, Switzerland : Springer, 2020. – ISBN 978-3-030-37583-6

[Pinedo 2008] Pinedo, Michael L.: *Scheduling: Theory, algorithms, and systems*. Third Edition. Cham and Heidelberg and New York and Dordrecht and London : Springer, 2008. – ISBN 978-0-387-78934-7

[Pinedo 2016] Pinedo, Michael L.: *Scheduling: Theory, algorithms, and systems*. Fifth Edition. Cham and Heidelberg and New York and Dordrecht and London : Springer, 2016. – ISBN 978-3-319-26578-0

[Potts et al. 2017a] Potts, C. N. ; Strusevich, V. A.: Fifty years of scheduling: A survey of milestones. In: *Journal of the Operational Research Society* 60 (2017), Nr. sup1, S. S41–S68. – ISSN 0160-5682

[Potts et al. 2017b] Potts, C. N. ; van Wassenhove, L. N.: Integrating Scheduling with Batching and Lot-Sizing: A Review of Algorithms and Complexity. In: *Journal of the Operational Research Society* 43 (2017), Nr. 5, S. 395–406. – ISSN 0160-5682

[Potts et al. 2000] Potts, Chris N. ; Kovalyov, Mikhail Y.: Scheduling with batching: A review. In: *European Journal of Operational Research* 120 (2000), Nr. 2, S. 228–249. – ISSN 03772217

[Quadt et al. 2005] Quadt, D. ; Kuhn, H.: Conceptual framework for lot-sizing and scheduling of flexible flow lines. In: *International Journal of Production Research* 43 (2005), Nr. 11, S. 2291–2308. – ISSN 0020-7543

[Quadt et al. 2007] Quadt, Daniel ; Kuhn, Heinrich: A taxonomy of flexible flow line scheduling procedures. In: *European Journal of Operational Research* 178 (2007), Nr. 3, S. 686–698. – ISSN 03772217

[Rajendran et al. 1999] Rajendran, Chandrasekharan ; Holthaus, Oliver: A comparative study of dispatching rules in dynamic flowshops and jobshops. In: *European Journal of Operational Research* 116 (1999), Nr. 1, S. 156–170. – ISSN 03772217

[Ramezanian et al. 2017] Ramezanian, Reza ; Fallah Sanami, Sahar ; Shafiei Nikabadi, Mohsen: A simultaneous planning of production and scheduling operations in flexible flow shops: Case study of tile industry. In: *The International Journal of Advanced Manufacturing Technology* 88 (2017), Nr. 9-12, S. 2389–2403. – ISSN 0268-3768

[Ribas et al. 2015] Ribas, Imma ; Companys, Ramon: Efficient heuristic algorithms for the blocking flow shop scheduling problem with total flow time minimization. In: *Computers & Industrial Engineering* 87 (2015), S. 30–39. – ISSN 03608352

[Ribas et al. 2010] Ribas, Imma ; Leisten, Rainer ; Framiñan, Jose M.: Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. In: *Computers & Operations Research* 37 (2010), Nr. 8, S. 1439–1454. – ISSN 03050548

[Richter et al. 2017] Richter, Michael ; Glaser, Rolf ; Wenzel, Sigrid ; Jessen, Ulrich: Simulationsgestützte Entscheidungsunterstützung für das Produktionsmanagement einer Verzinkerei. In: Wenzel, Sigrid (Hrsg.) ; Peter, Tim (Hrsg.): *Simulation in Produktion und Logistik 2017*. Kassel : kassel university press, 2017, S. 479–488. – ISBN 978-3-7376-0192-4

[Rose et al. 2011] Rose, Oliver ; März, Lothar: Simulation. In: März, Lothar (Hrsg.) ; Krug, Wilfried (Hrsg.) ; Rose, Oliver (Hrsg.) ; Weigert, Gerald (Hrsg.): *Simulation und Optimierung in Produktion und Logistik*. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2011 (VDI-Buch), S. 13–20. – ISBN 978-3-6421-4536-0

[Ruiz et al. 2006] Ruiz, Rubén ; Maroto, Concepción: A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility. In: *European Journal of Operational Research* 169 (2006), Nr. 3, S. 781–800. – ISSN 03772217

[Ruiz et al. 2008] Ruiz, Rubén ; Stützle, Thomas: An Iterated Greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives. In: *European Journal of Operational Research* 187 (2008), Nr. 3, S. 1143–1159. – ISSN 03772217

[Ruiz et al. 2010] Ruiz, Rubén ; Vázquez-Rodríguez, José A.: The hybrid flow shop scheduling problem. In: *European Journal of Operational Research* 205 (2010), Nr. 1, S. 1–18. – ISSN 03772217

[Salvador 1973] Salvador, Michael S.: A Solution to a Special Class of Flow Shop Scheduling Problems. In: Elmaghraby, Salah E. (Hrsg.): *Symposium on the Theory of Scheduling and Its Applications*. Berlin and Heidelberg : Springer, 1973 (Lecture Notes in Economics and Mathematical Systems, Operations Research, Computer Science, Social Science), S. 83–91. – ISBN 978-3-540-06437-4

[Sastry et al. 2014] Sastry, Kumara ; Goldberg, David E. ; Kendall, Graham: Genetic Algorithms. In: Burke, Edmund K. (Hrsg.) ; Kendall, Graham (Hrsg.):

Search Methodologies. Boston, MA : Springer US, 2014, S. 93–118. – ISBN 978-1-461-46940-7

[Sawik 2006] Sawik, Tadeusz: Hierarchical approach to production scheduling in make-to-order assembly. In: *International Journal of Production Research* 44 (2006), Nr. 4, S. 801–830. – ISSN 0020-7543

[Schmidt et al. 2017] Schmidt, Thorsten ; Kühn, Mathias ; Genssler, Paul R.: Design of Project-oriented Calculation Models for Job Priorities by Using a Customized Genetic Algorithm. In: Wenzel, Sigrid (Hrsg.) ; Peter, Tim (Hrsg.): *Simulation in Produktion und Logistik 2017*. Kassel : kassel university press, 2017, S. 99–108. – ISBN 978-3-7376-0192-4

[Schuh et al. 2012] Schuh, Günther (Hrsg.) ; Stich, Volker (Hrsg.): *Produktionsplanung und -steuerung 1: Grundlagen der PPS*. 4. Aufl. 2012. Berlin Heidelberg : Springer Berlin Heidelberg, 2012. – ISBN 978-3-642-25423-9

[Shi et al. 2019] Shi, Haibo ; Qi, Yuanwei ; Sun, Liangliang ; Han, Zhonghua ; Zhang, Quan: Research on limited buffer scheduling problems in flexible flow shops with setup times. In: *International Journal of Modelling, Identification and Control* 32 (2019), Nr. 2, S. 93. – ISSN 1746-6172

[Stalinski et al. 2017a] Stalinski, David ; Scholz, Dieter: Digitale Steuerung in der Holzbearbeitung. In: *HK* (2017), Nr. 4, S. 114–115

[Stalinski et al. 2017b] Stalinski, David ; Scholz, Dieter: Prozessoptimierung durch digitale Fertigungssteuerung: Vorstellung eines Entwicklungsprojekts an einem Fallbeispiel aus der holzverarbeitenden Industrie. In: *ZWF* 112 (2017), Nr. 5, S. 301–304

[Stalinski et al. 2018a] Stalinski, David ; Scholz, Dieter: Model-based approach of a decision processing unit in a smart wood-processing company. In: *International Scientific Journal Industry 4.0* III (2018), Nr. 6, S. 8–12

[Stalinski et al. 2018b] Stalinski, David ; Scholz, Dieter: Prozessbegleitende Optimierung in der Produktionssteuerung: Ansatz für die Umsetzung einer praxistauglichen Selbstregelung. In: *ZWF* 113 (2018), Nr. 5, S. 277–280

[Stalinski et al. 2019a] Stalinski, David ; Scholz, Dieter: Modellbasierte Auswertung von Echtzeitdaten: Automatisierte Produktionsplanung und -steuerung in einem mittelständischen Industriebetrieb. In: *Werkstattstechnik* 109 (2019), Nr. 3, S. 174–178

[Stalinski et al. 2019b] Stalinski, David ; Scholz, Dieter: Simulationsbasiertes Entscheidungssystem für die Produktionsplanung und -steuerung in einem holzverarbeitenden Betrieb. In: Putz, Matthias (Hrsg.) ; Schlegel, Andreas (Hrsg.): *ASIM 2019*. Auerbach /Vogtl. : Wissenschaftliche Scripten, 2019, S. 345–354. – ISBN 978-3-95735-113-5

[Suerie 2005] Suerie, Christopher: *Lecture Notes in Economics and Mathematical Systems*. Bd. 552: *Time Continuity in Discrete Time Models: New Approaches for Production Planning in Process Industries*. Berlin, Heidelberg : Springer-Verlag Berlin Heidelberg, 2005. – ISBN 3-540-24521-9

[Tan et al. 2018] Tan, Yi ; Mönch, Lars ; FOWLER, JOHN W.: A hybrid scheduling approach for a two-stage flexible flow shop with batch processing machines. In: *Journal of Scheduling* 21 (2018), Nr. 2, S. 209–226. – ISSN 1094-6136

[Tang et al. 2006] Tang, L. ; Xuan, H.: Lagrangian relaxation algorithms for real-time hybrid flowshop scheduling with finite intermediate buffers. In: *Journal of the Operational Research Society* 57 (2006), Nr. 3, S. 316–324. – ISSN 0160-5682

[Taraz 2012] Taraz, Anusch: *Diskrete Mathematik: Grundlagen und Methoden*. Basel : Birkhäuser, 2012 (Mathematik kompakt). – ISBN 978-3-7643-8898-0

[Tavares et al. 2011] Tavares, Roberto F. ; Filho, Moacir G.: An ant colony optimization approach to a permutational flowshop scheduling problem with outsourcing allowed. In: *Computers and Operations Research* 38 (2011), Nr. 9, S. 1286–1293

[Tseng et al. 2008] Tseng, Chao-Tang ; Liao, Ching-Jong: A particle swarm optimization algorithm for hybrid flow-shop scheduling with multiprocessor tasks. In: *International Journal of Production Research* 46 (2008), Nr. 17, S. 4655–4670. – ISSN 0020-7543

[Turing 1937] Turing, Alan M.: On Computable Numbers, with an Application to the Entscheidungsproblem. In: *Proceedings of the London Mathematical Society* s2-42 (1937), Nr. 1, S. 230–265. – ISSN 00246115

[Vairaktarakis 2004] Vairaktarakis, George: Flexible Hybrid Flowshops. In: Leung, Joseph Y-T. (Hrsg.): *Handbook of scheduling*. Boca Raton : Chapman & Hall/CRC, 2004 (Chapman & Hall / CRC computer and information science series), S. 81–113. – ISBN 978-1-584883-975

[Vignier et al. 1999] Vignier, Antony ; Billaut, Jean-Charles ; Proust, Christian: Les problèmes d'ordonnancement de type flow-shop hybride. In: *RAIRO. Recherche opérationnelle* 33 (1999), Nr. 2, S. 117–183

[Waldherr 2015] Waldherr, Stefan: *Scheduling of flow shops with synchronous movement*. Osnabrück, Universität Osnabrück, Dissertation, 2015

[Waldherr et al. 2014] Waldherr, Stefan ; Knust, Sigrid: Two-stage scheduling in shelf-board production: A case study. In: *International Journal of Production Research* 52 (2014), Nr. 13, S. 4078–4092. – ISSN 0020-7543

[Waldherr et al. 2015] Waldherr, Stefan ; Knust, Sigrid: Complexity results for flow shop problems with synchronous movement. In: *European Journal of Operational Research* 242 (2015), Nr. 1, S. 34–44. – ISSN 03772217

[Wang 2005] Wang, Hong: Flexible flow shop scheduling: Optimum, heuristics and artificial intelligence solutions. In: *Expert Systems* 22 (2005), Nr. 2, S. 78–85. – ISSN 0266-4720

[Wang et al. 2009] Wang, Xianpeng ; Tang, Lixin: A tabu search heuristic for the hybrid flowshop scheduling with finite intermediate buffers. In: *Computers & Operations Research* 36 (2009), Nr. 3, S. 907–918. – ISSN 03050548

[Wardono et al. 2004] Wardono, Bagas ; Fathi, Yahya: A tabu search algorithm for the multi-stage parallel machine problem with limited buffer capacities. In: *European Journal of Operational Research* 155 (2004), Nr. 2, S. 380–401. – ISSN 03772217

[Weicker 2015] Weicker, Karsten: *Evolutionäre Algorithmen*. Wiesbaden : Springer Fachmedien Wiesbaden, 2015. – ISBN 978-3-658-09957-2

[Weigert et al. 2008] Weigert, Gerald ; Henlich, Thomas ; Klemmt, Andreas: Methoden zur Modellierung und Optimierung von Montageprozessen. In: Rabe, Markus (Hrsg.): *Advances in simulation for production and logistics applications*. Stuttgart : Fraunhofer IRB-Verl., 2008 (ASIM-Mitteilung), S. 479–488. – ISBN 978-3-8167-7798-4

[Werner 2011] Werner, Frank: Genetic algorithms for shop scheduling problems: a survey. In: *Preprint Series* 11 (2011), S. 1–66

[Werners 2013] Werners, Brigitte: *Grundlagen des Operations Research: Mit Aufgaben und Lösungen*. 3., überarb. Aufl. 2013. Berlin Heidelberg : Springer Berlin Heidelberg, 2013 (Springer-Lehrbuch). – ISBN 978-3-642-40101-5

[Wolpert et al. 1997] Wolpert, David H. ; Macready, William G.: No free lunch theorems for optimization. In: *IEEE Transactions on Evolutionary Computation* 1 (1997), Nr. 1, S. 67–82

[Wu et al. 2003] Wu, Yi ; Liu, Min ; Wu, Cheng: A genetic algorithm for solving flow shop scheduling problems with parallel machine and special procedure constraints. In: *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.03EX693)*, IEEE, 2003, S. 1774–1779. – ISBN 0-7803-7865-2

[Xu et al. 2003] Xu, Yuedong ; Sannomiya, Nobuo ; Tijan, Yajie: A New Method of Shifting Bottleneck with Tabu Search for Solving Flexible Flow Shop Problems. In: *Transactions of the Society of Instrument and Control Engineers* 39 (2003), Nr. 9, S. 865–871. – ISSN 0453-4654

[Yaurima et al. 2009] Yaurima, Victor ; Burtseva, Larisa ; Tchernykh, Andrei: Hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints and limited buffers. In: *Computers & Industrial Engineering* 56 (2009), Nr. 4, S. 1452–1463. – ISSN 03608352

[Ying et al. 2006] Ying, Kuo-Ching ; Lin, Shih-Wei: Multiprocessor task scheduling in multistage hybrid flow-shops: An ant colony system approach. In: *International Journal of Production Research* 44 (2006), Nr. 16, S. 3161–3177. – ISSN 0020-7543

[Zeng et al. 2018] Zeng, Zhiqiang ; Hong, Mengna ; Man, Yi ; Li, Jigeng ; Zhang, Yanzhong ; Liu, Huanbin: Multi-object optimization of flexible flow shop scheduling with batch process — Consideration total electricity consumption and material wastage. In: *Journal of Cleaner Production* 183 (2018), S. 925–939. – ISSN 09596526

[Zgurovsky et al. 2019] Zgurovsky, Michael Z. ; Pavlov, Alexander A.: *Studies in Systems, Decision and Control*. Bd. 173: *Combinatorial Optimization Problems in Planning and Decision Making: Theory and Applications*. Cham : Springer International Publishing, 2019. – ISBN 978-3-319-98976-1

[Zhang et al. 2016] Zhang, Cheng ; Shi, Zhongshun ; Huang, Zewen ; Wu, Yifan ; Shi, Leyuan: Flow shop scheduling with a batch processor and limited buffer. In: *International Journal of Production Research* 55 (2016), Nr. 11, S. 3217–3233. – ISSN 0020-7543

[Zhang et al. 2015] Zhang, Tao ; Xie, Shufang ; Rose, Oliver: Flexible Job-shop Scheduling with Dynamic Stochastic Machine Set. In: Rabe, Markus (Hrsg.) ; Clausen, Uwe (Hrsg.): *Simulation in Production and Logistics 2015*. Stuttgart : Fraunhofer Verlag, 2015, S. 21–28. – ISBN 978-3-8396-0936-1

[Zhong et al. 2013a] Zhong, Ray Y. ; Dai, Q. Y. ; Qu, T. ; Hu, G. J. ; Huang, George Q.: RFID-enabled real-time manufacturing execution system for mass-customization production. In: *Robotics and Computer-Integrated Manufacturing* 29 (2013), Nr. 2, S. 283–292. – ISSN 07365845

[Zhong et al. 2014] Zhong, Ray Y. ; Huang, George Q. ; Dai, Q. Y.: A Real-time Planning and Scheduling Model in RFID-enabled Manufacturing. In: *4th International Conference on Industrial Engineering and Operations Management* 4 (2014), S. 215–224

[Zhong et al. 2013b] Zhong, Ray Y. ; Li, Z. ; Pang, L. Y. ; Pan, Y. ; Qu, T. ; Huang, George Q.: RFID-enabled real-time advanced planning and scheduling shell for production decision making. In: *International Journal of Computer Integrated Manufacturing* 26 (2013), Nr. 7, S. 649–662

Veröffentlichungen

HK17 Stalinski, David ; Scholz, Dieter (2017): Digitale Steuerung in der Holzbearbeitung. In: Holz- und Kunststoffverarbeitung (2017), Nr. 4, S. 114–115

ZWF17 Stalinski, David ; Scholz, Dieter (2017): Prozessoptimierung durch digitale Fertigungssteuerung: Vorstellung eines Entwicklungsprojekts an einem Fallbeispiel aus der holzverarbeitenden Industrie. In: Zeitschrift für wirtschaftlichen Fabrikbetrieb 112 (2017), Nr. 5, S. 301–304

ZWF18 Stalinski, David ; Scholz, Dieter (2018): Prozessbegleitende Optimierung in der Produktionssteuerung: Ansatz für die Umsetzung einer praxistauglichen Selbstregelung. In: Zeitschrift für wirtschaftlichen Fabrikbetrieb 113 (2018), Nr. 5, S. 277–280

BUL18 Stalinski, David ; Scholz, Dieter (2018): Model-based approach of a decision processing unit in a smart wood-processing company. In: International Scientific Journal Industry 4.0 III (2018), Nr. 6, S. 8–12

WT19 Stalinski, David ; Scholz, Dieter (2019): Modellbasierte Auswertung von Echtzeitdaten: Automatisierte Produktionsplanung und -steuerung in einem mittelständischen Industriebetrieb. In: Werkstattstechnik 109 (2019), Nr. 3, S. 174–178

ASIM19 Stalinski, David ; Scholz, Dieter (2019): Simulationsbasiertes Entscheidungssystem für die Produktionsplanung und -steuerung in einem holzverarbeitenden Betrieb. In: Putz, Matthias (Hrsg.) ; Schlegel, Andreas (Hrsg.): ASIM 2019, Simulation in Produktion und Logistik (2019), S. 345–354

Lebenslauf

Der Lebenslauf ist in der Online-Version aus Gründen des Datenschutzes nicht enthalten.

A Anhang

A.1 Parameterstudien für den Genetischen Algorithmus

A.1.1 Ergebnisse der Parameterstudien für S–224

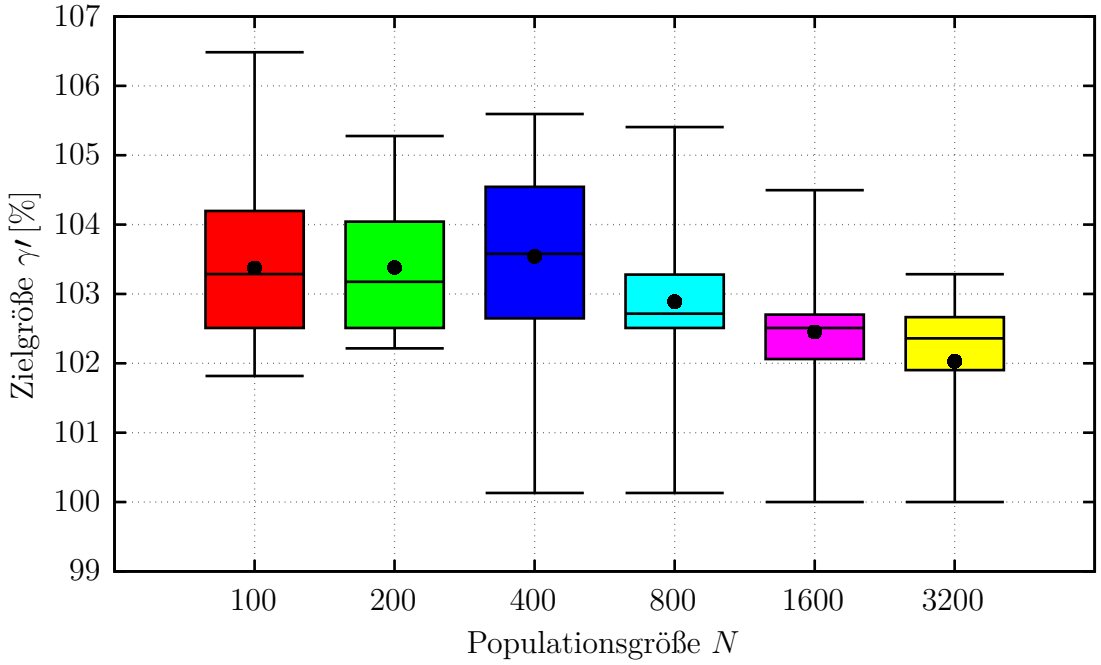


Abbildung A.1: Einfluss der Populationsgröße (Genetischer Algorithmus, S–224)

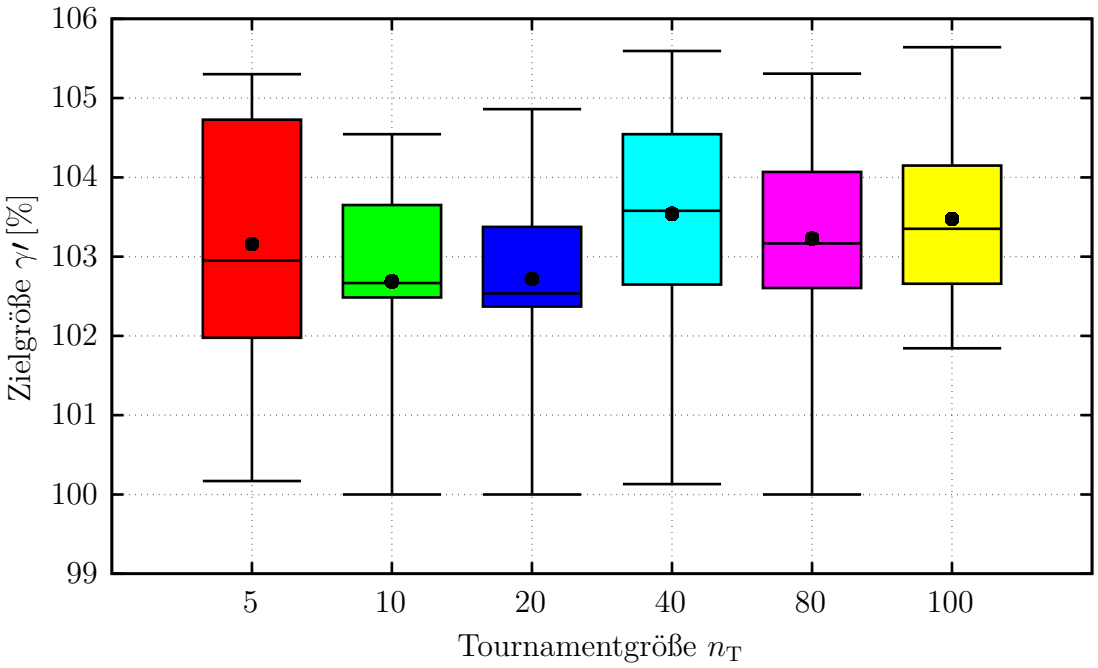


Abbildung A.2: Einfluss der Tournamentselektion (Genetischer Algorithmus, S–224)

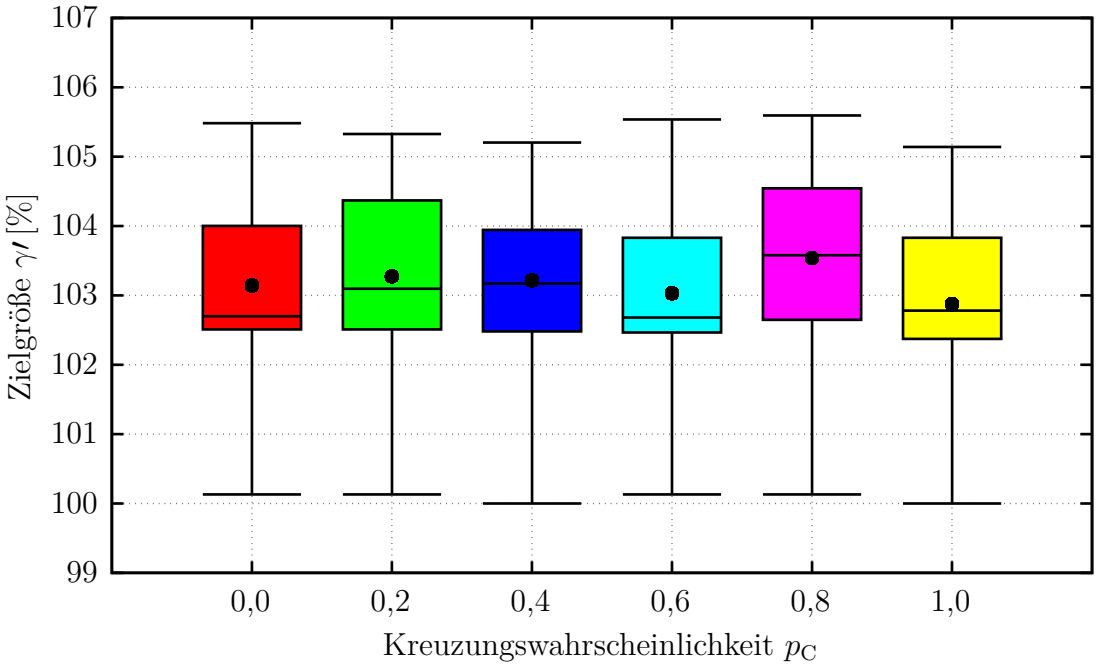


Abbildung A.3: Einfluss der Kreuzungswahrscheinlichkeit (Genetischer Algorithmus, S–224)

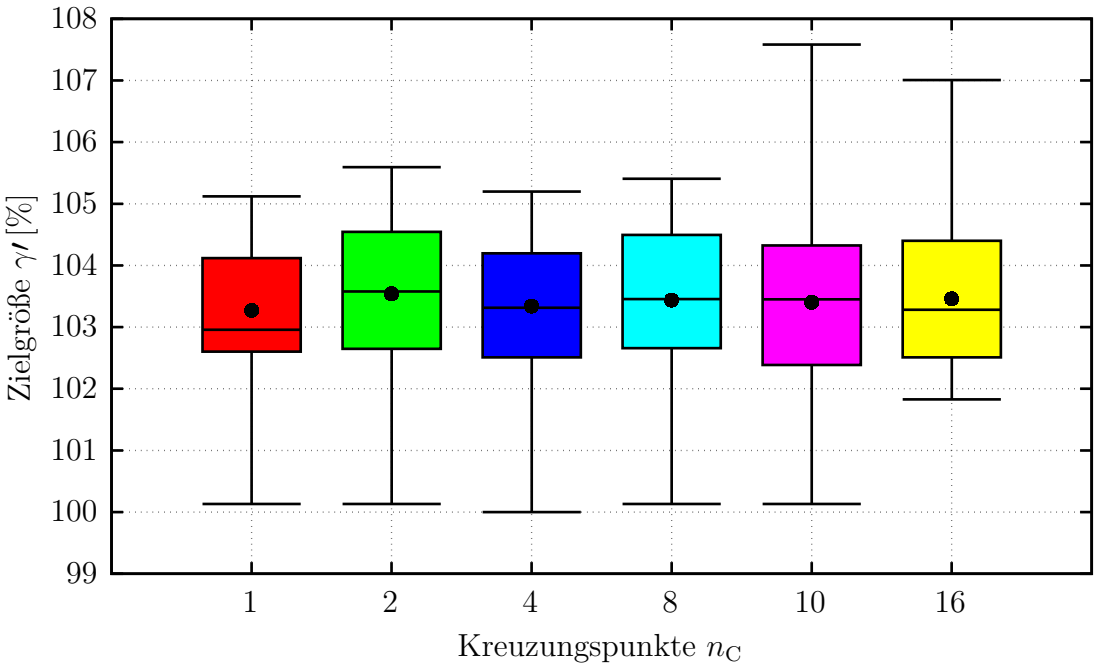


Abbildung A.4: Einfluss des N-Point Crossovers (Genetischer Algorithmus, S–224)

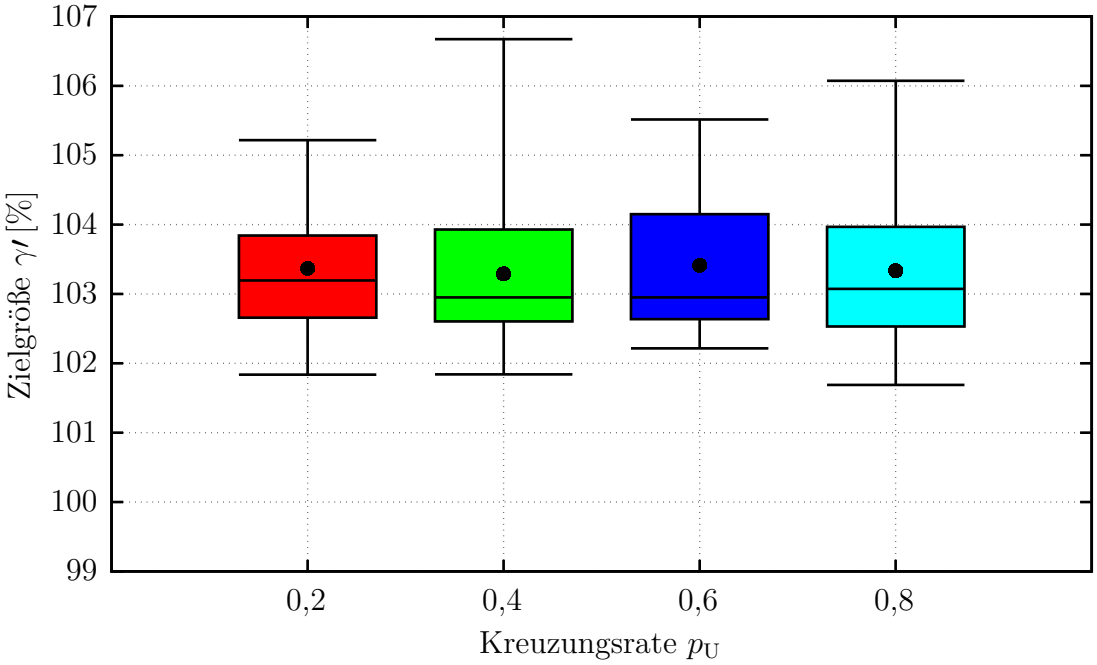


Abbildung A.5: Einfluss des Uniform Crossovers (Genetischer Algorithmus, S–224)

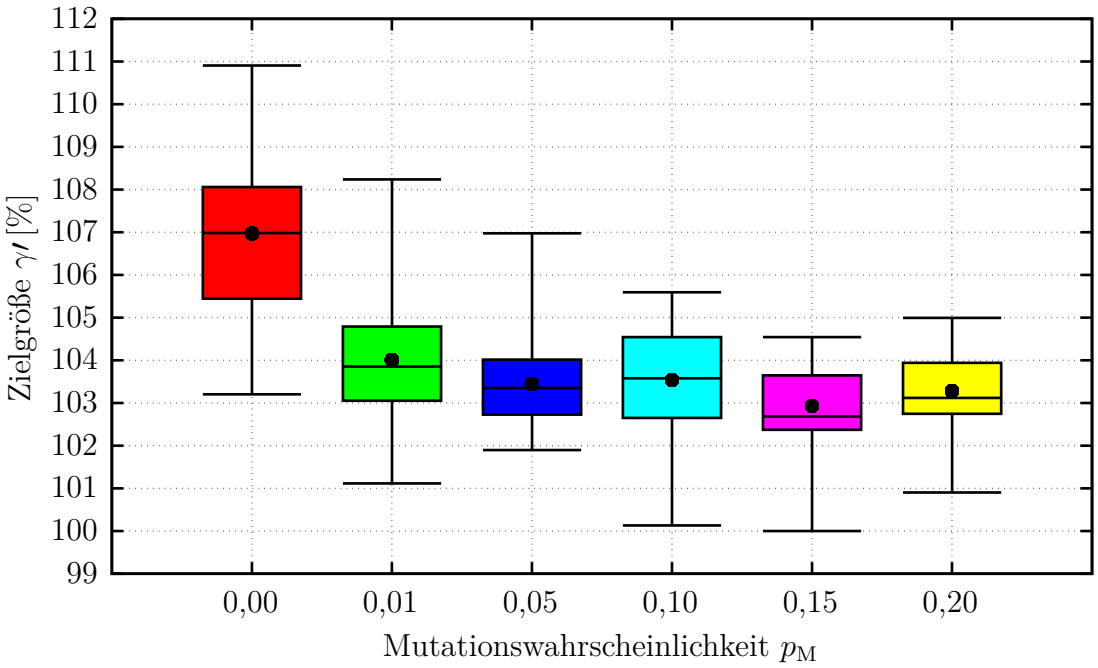


Abbildung A.6: Einfluss der Mutationswahrscheinlichkeit (Genetischer Algorithmus, S–224)

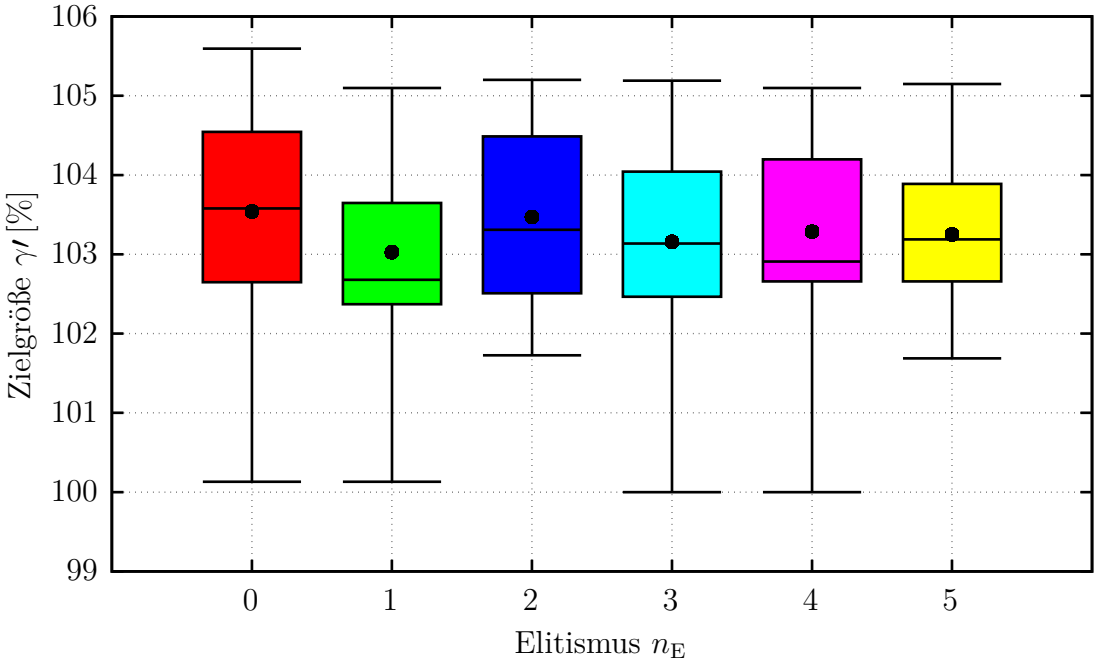


Abbildung A.7: Einfluss des Elitismus (Genetischer Algorithmus, S–224)

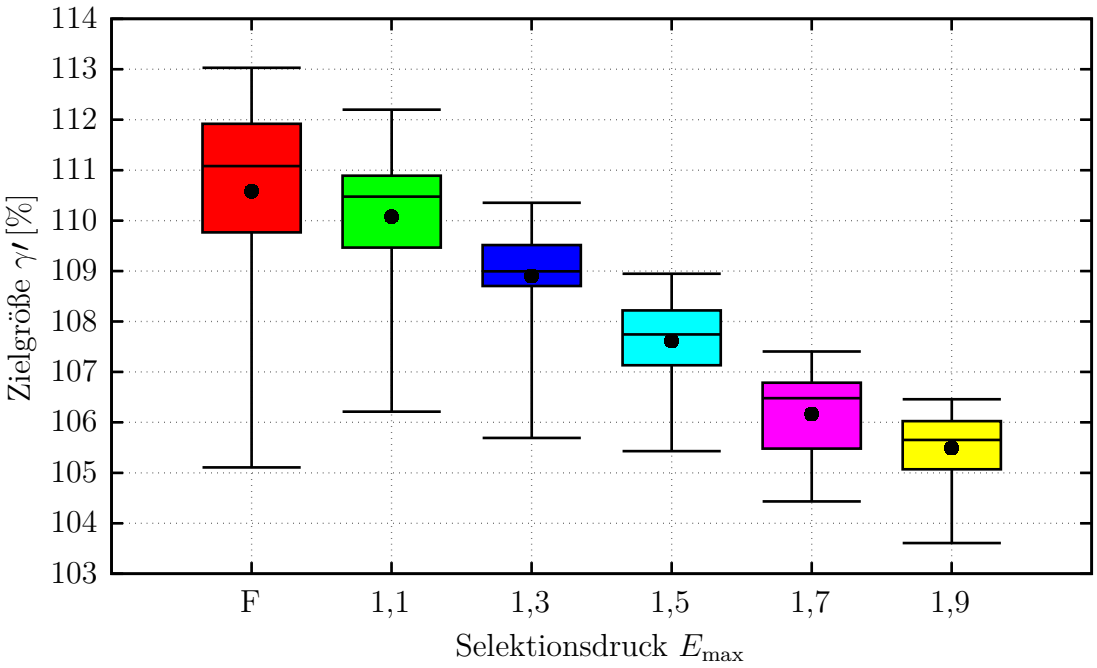


Abbildung A.8: Einfluss der Rouletteselektion (Genetischer Algorithmus, S–224)

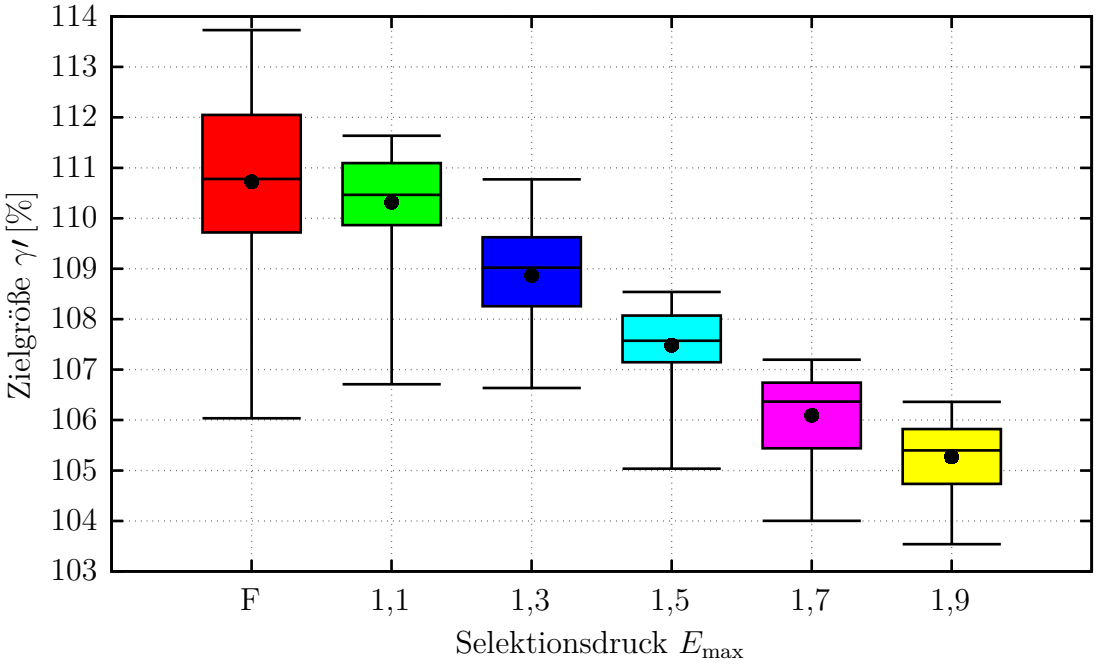


Abbildung A.9: Einfluss der SUS-Selektion (Genetischer Algorithmus, S–224)

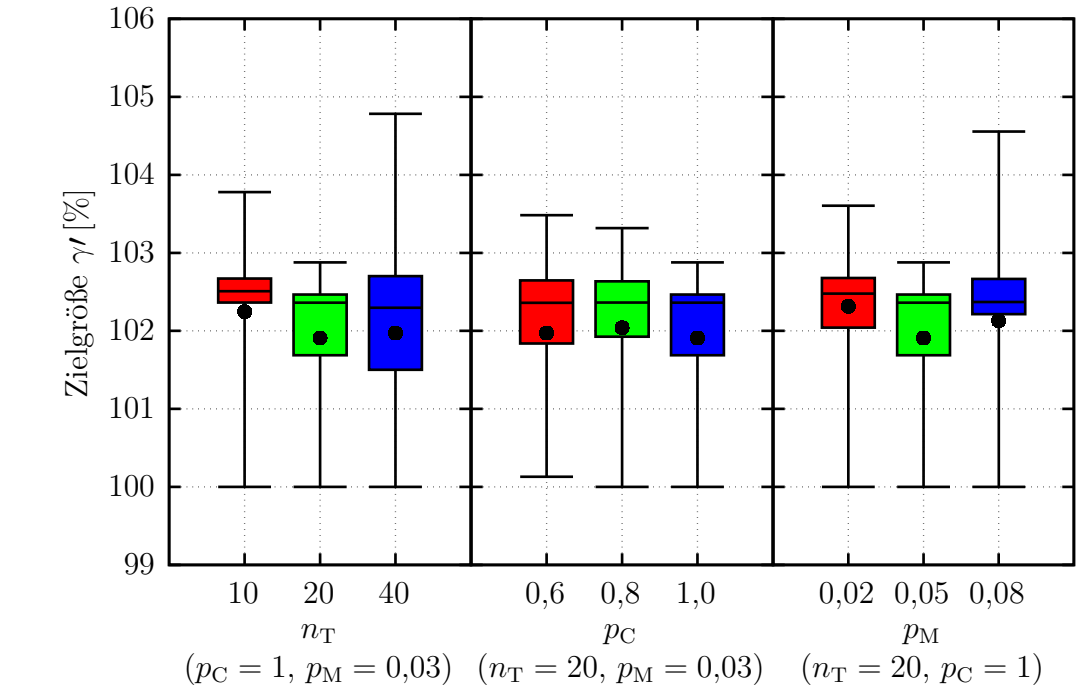


Abbildung A.10: Feintuning des Genetischen Algorithmus (S–224)

A.1.2 Ergebnisse der Parameterstudien für M–424

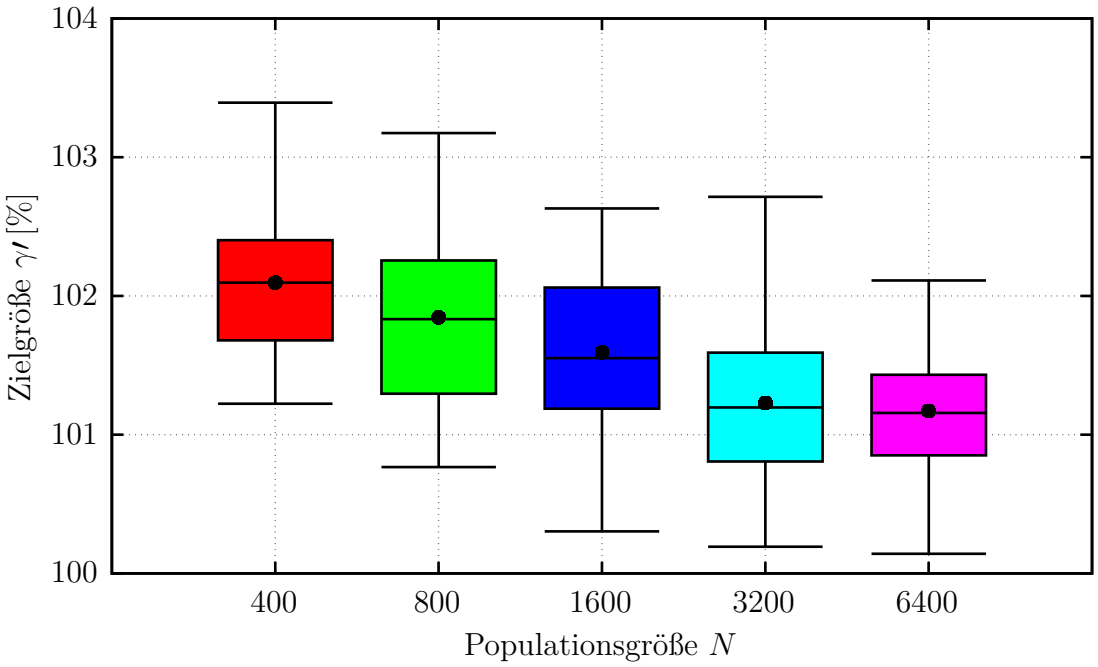


Abbildung A.11: Einfluss der Populationsgröße (Genetischer Algorithmus, M–424)

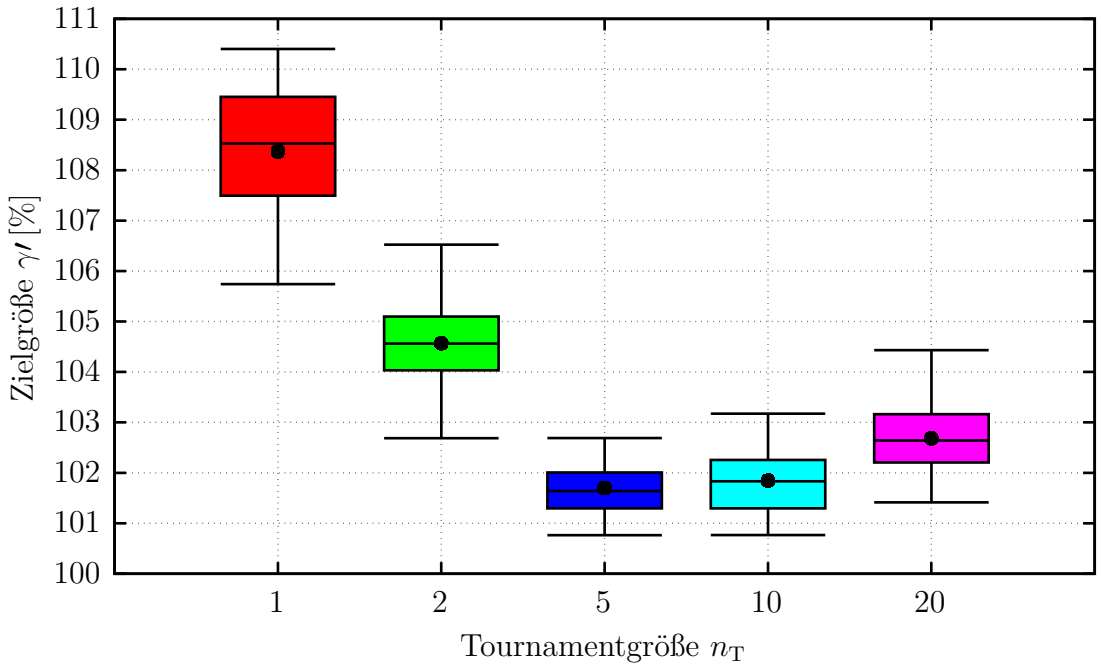


Abbildung A.12: Einfluss der Tournamentselektion (Genetischer Algorithmus, M–424)

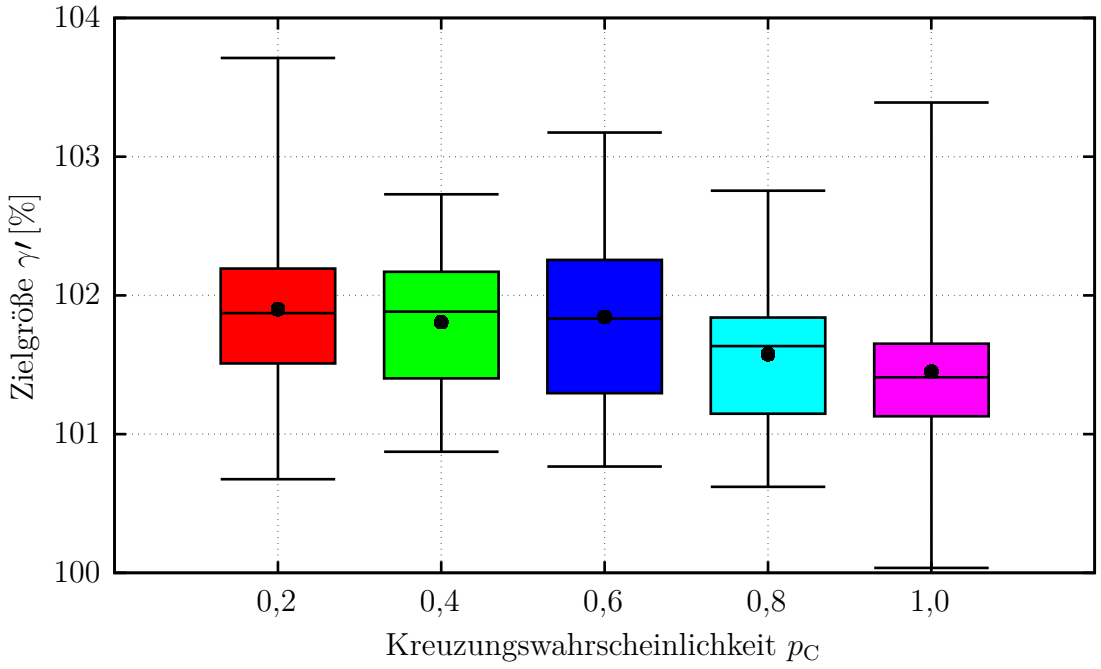


Abbildung A.13: Einfluss der Kreuzungswahrscheinlichkeit (Genetischer Algorithmus, M–424)

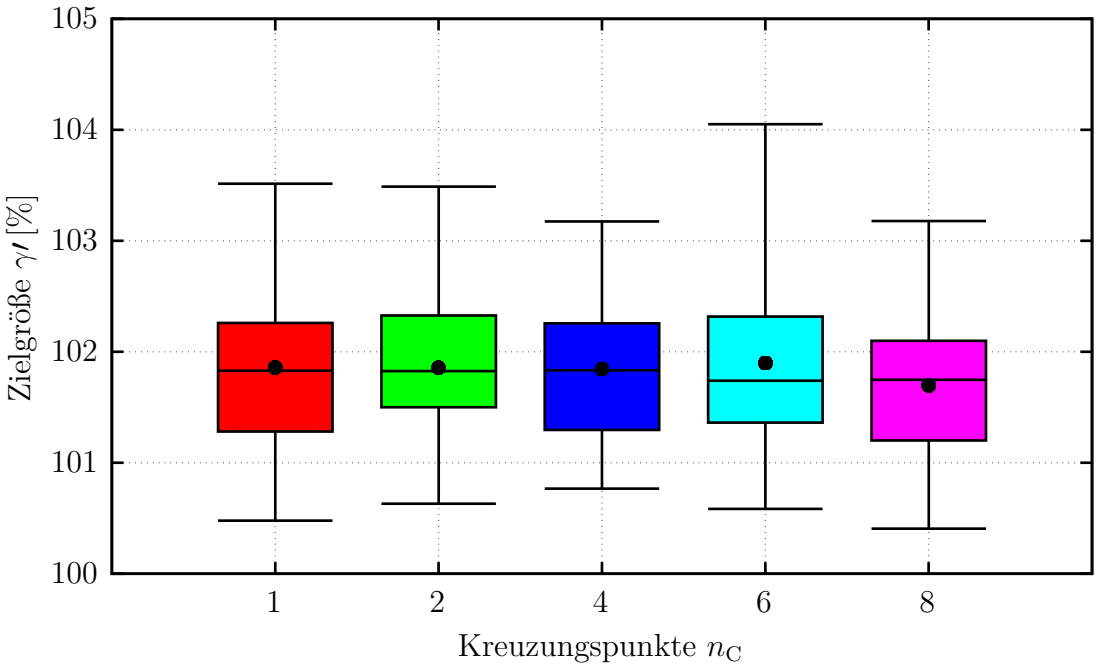


Abbildung A.14: Einfluss des N-Point Crossovers (Genetischer Algorithmus, M–424)

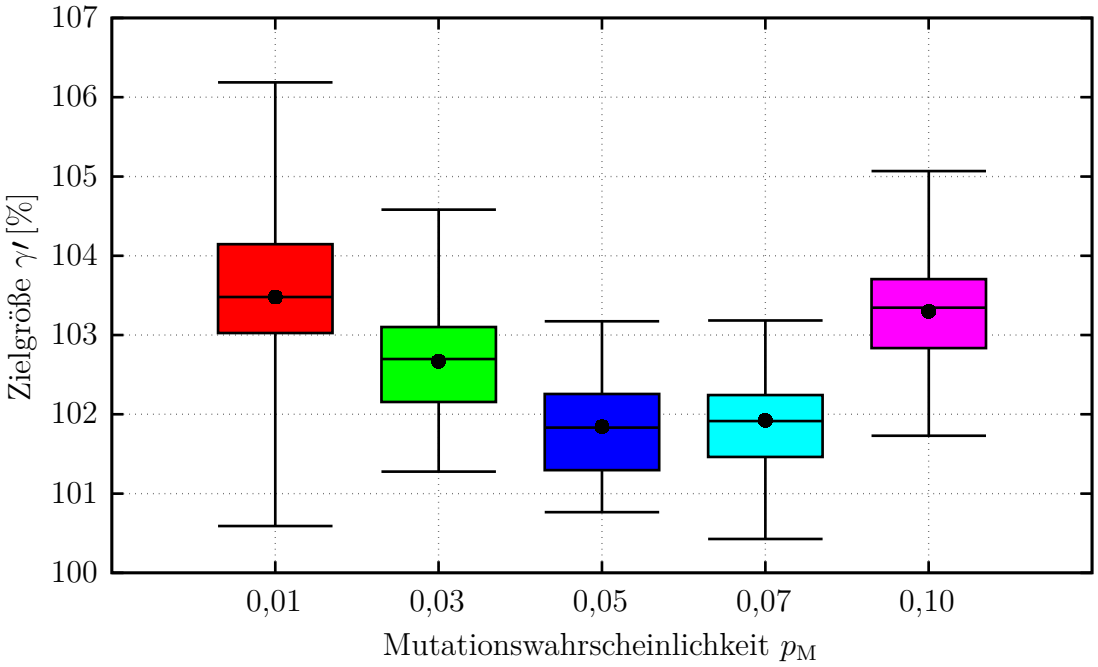


Abbildung A.15: Einfluss der Mutationswahrscheinlichkeit (Genetischer Algorithmus, M–424)

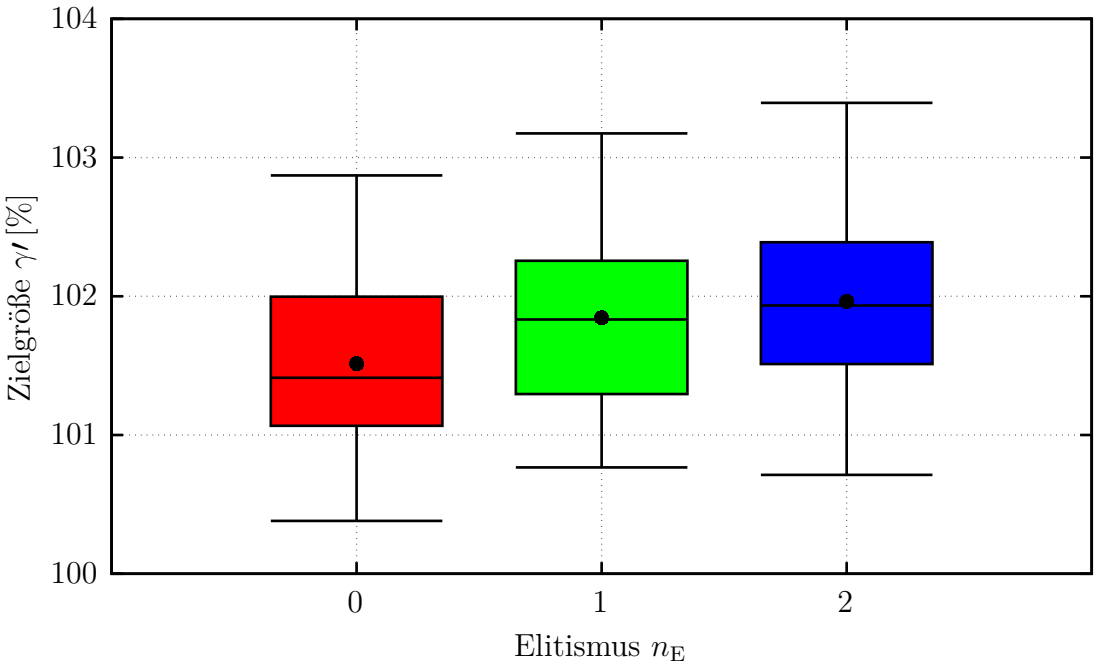


Abbildung A.16: Einfluss des Elitismus (Genetischer Algorithmus, M–424)

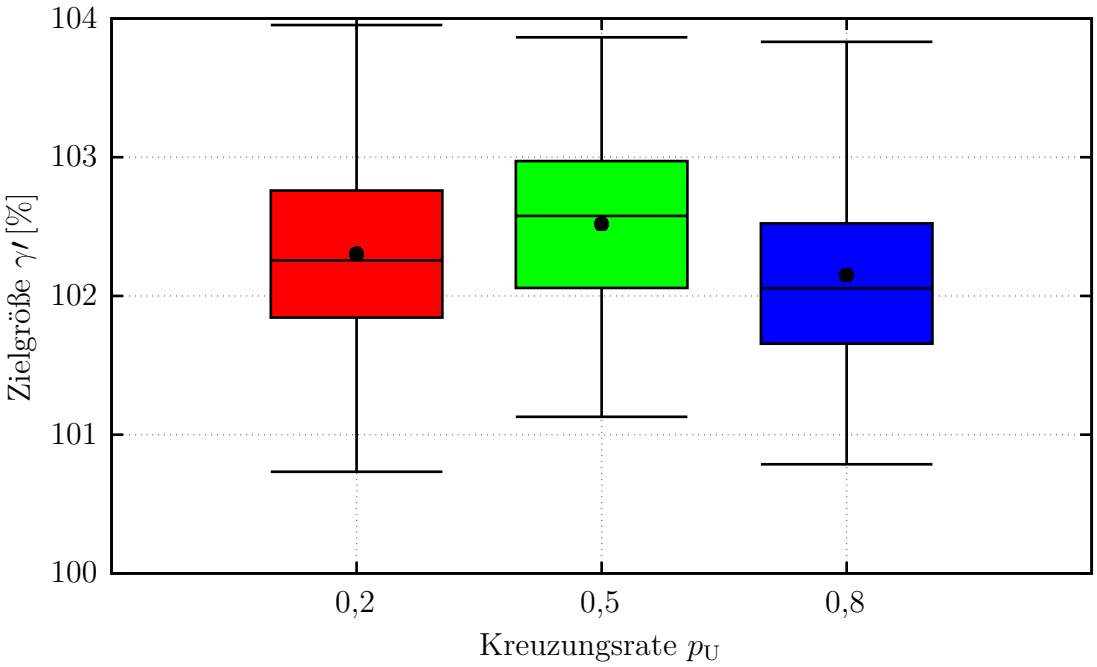


Abbildung A.17: Einfluss des Uniform Crossovers (Genetischer Algorithmus, M–424)

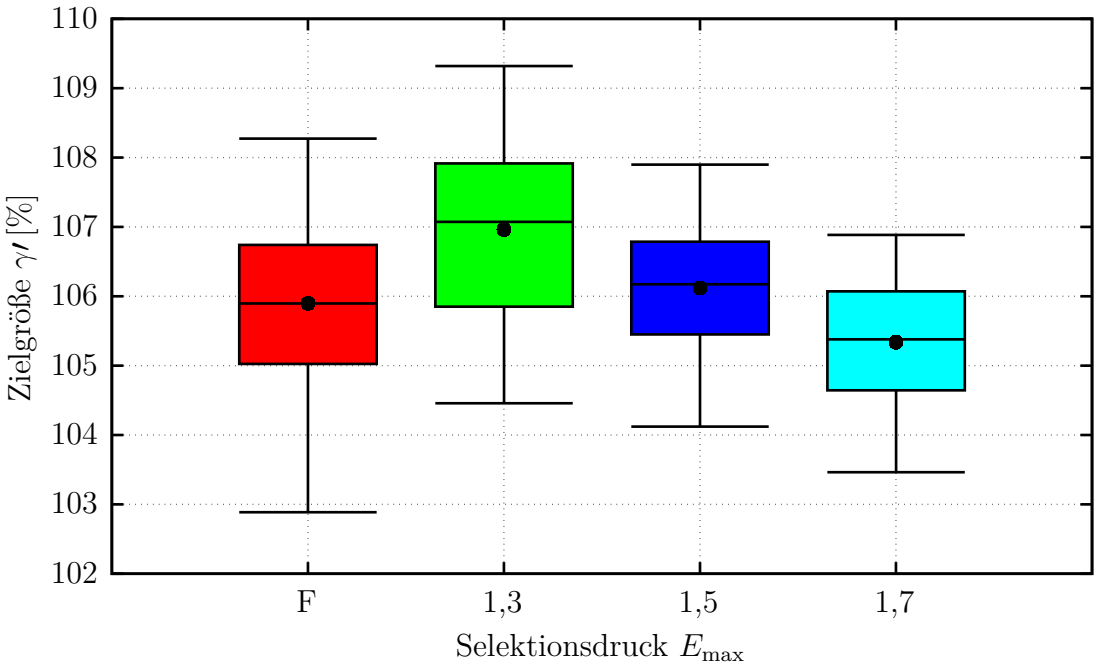


Abbildung A.18: Einfluss der Rouletteselektion (Genetischer Algorithmus, M–424)

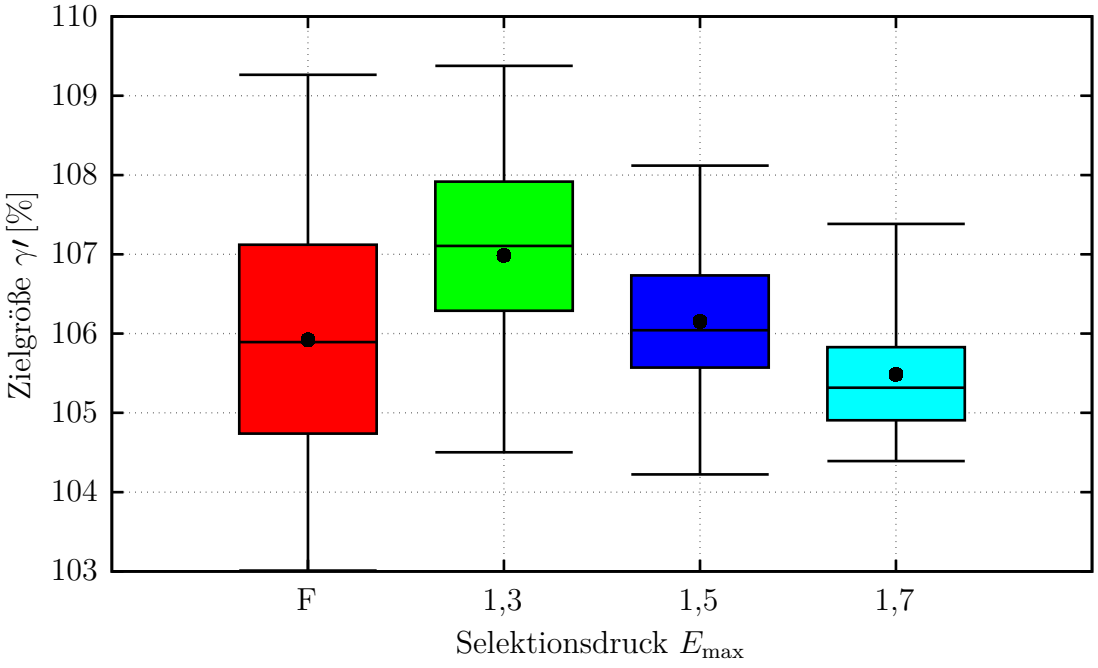


Abbildung A.19: Einfluss der SUS-Selektion (Genetischer Algorithmus, M–424)

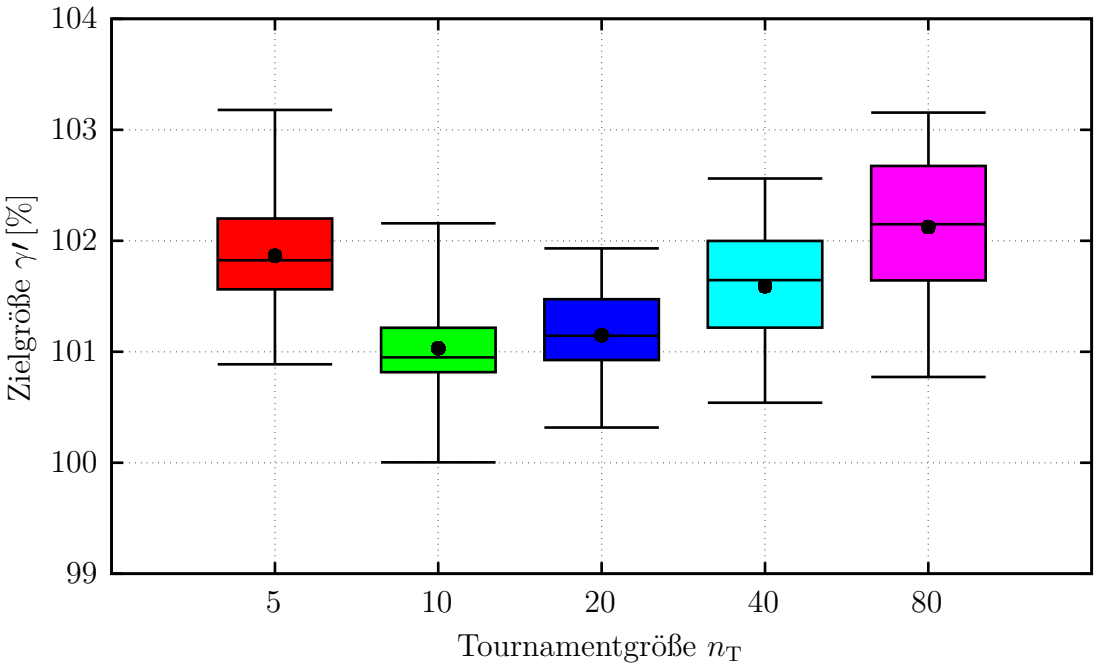


Abbildung A.20: Feintuning des Genetischen Algorithmus (M–424)

A.1.3 Ergebnisse der Parameterstudien für L–824

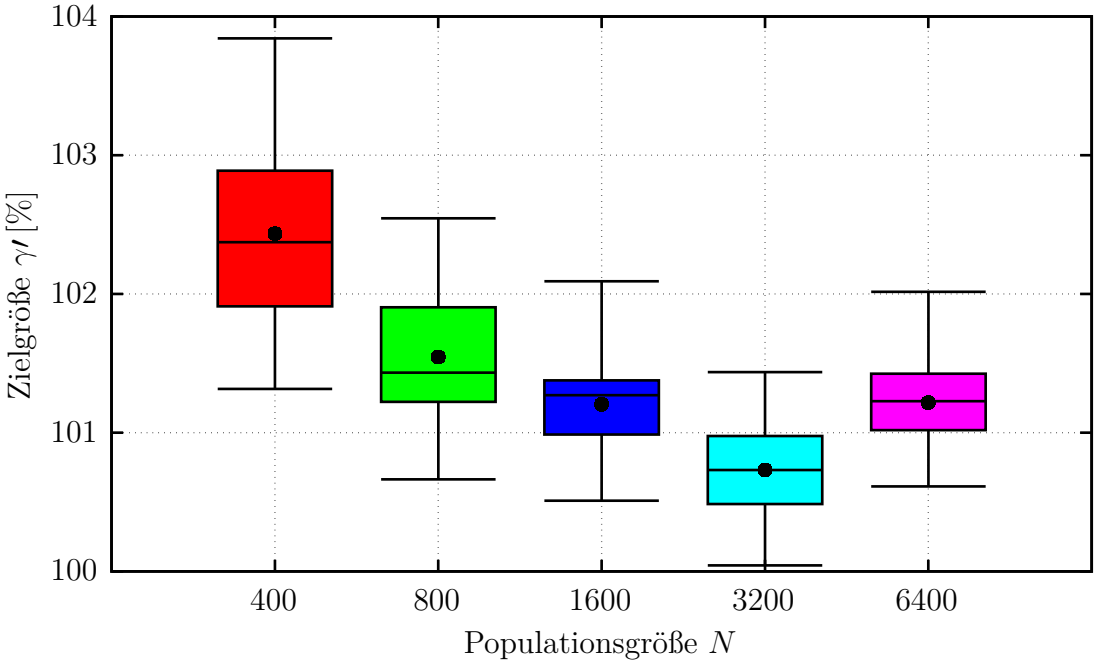


Abbildung A.21: Einfluss der Populationsgröße (Genetischer Algorithmus, L–824)

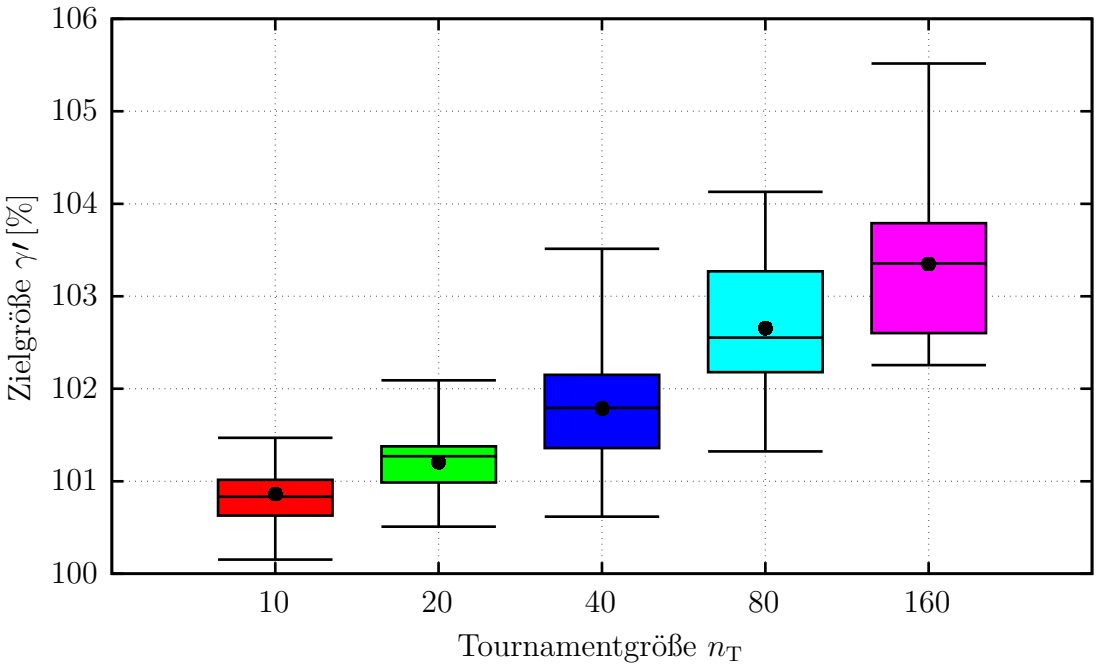


Abbildung A.22: Einfluss der Tournamentselektion (Genetischer Algorithmus, L–824)

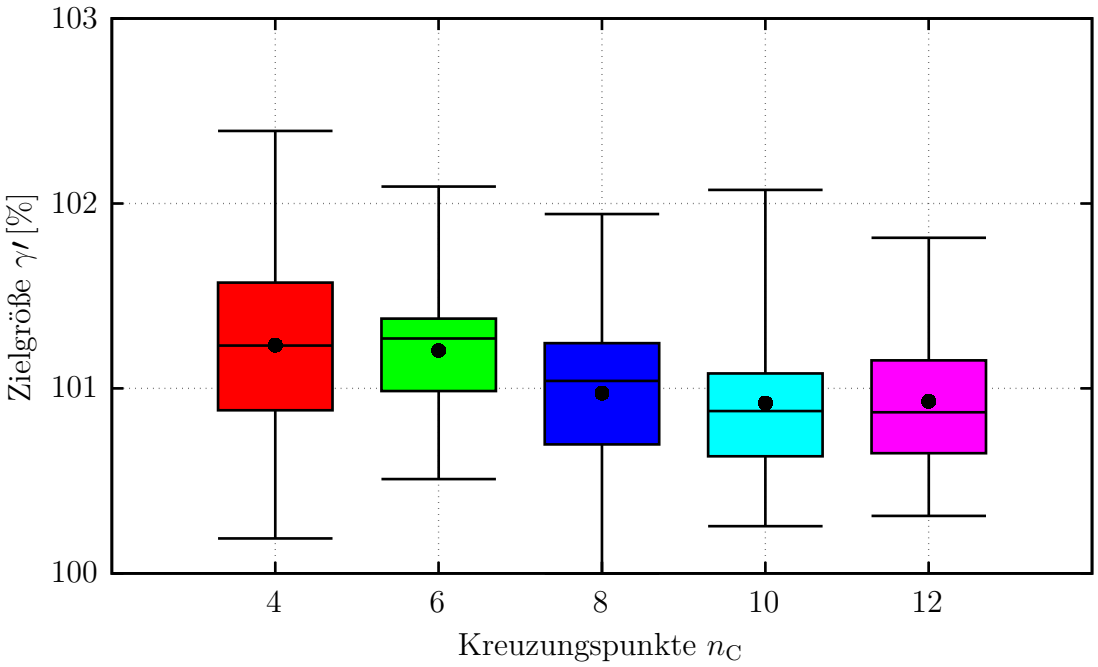


Abbildung A.23: Einfluss des N-Point Crossovers (Genetischer Algorithmus, L–824)

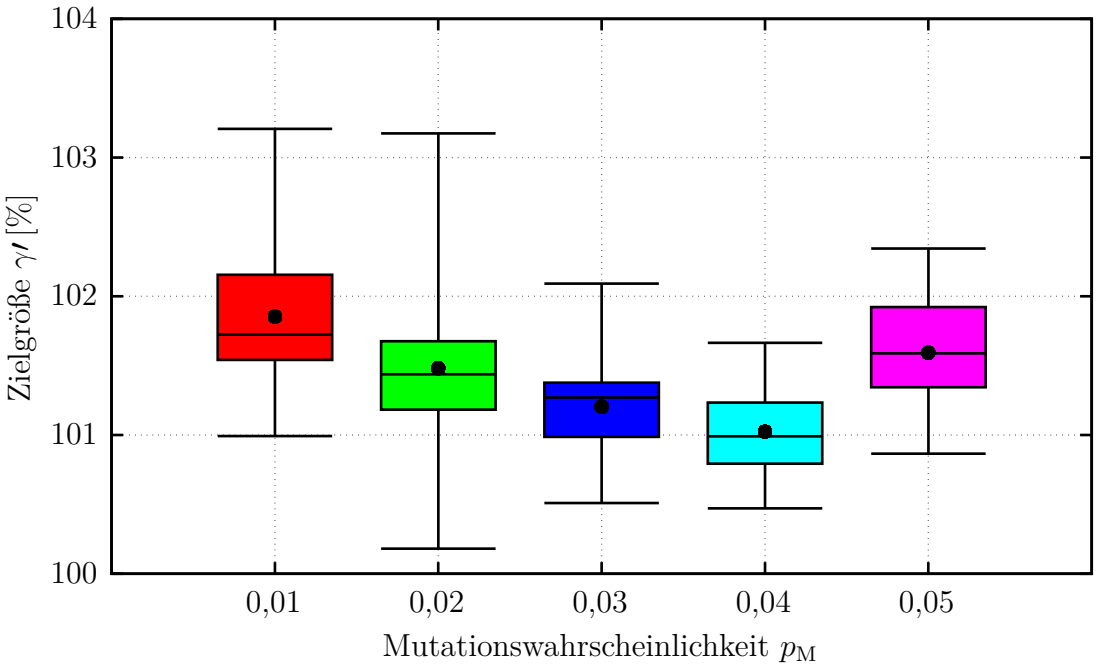


Abbildung A.24: Einfluss der Mutationswahrscheinlichkeit (Genetischer Algorithmus, L–824)

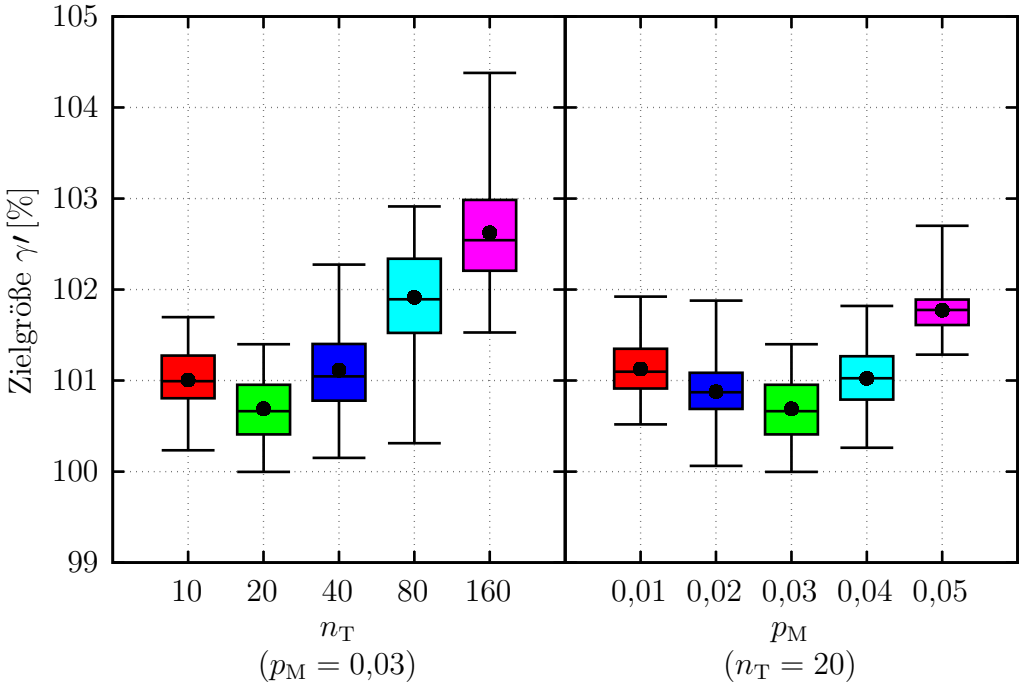


Abbildung A.25: Feintuning des Genetischen Algorithmus (L–824)

A.2 Parameterstudien für den Ameisenalgorithmus

A.2.1 Ergebnisse der Parameterstudien für S–224

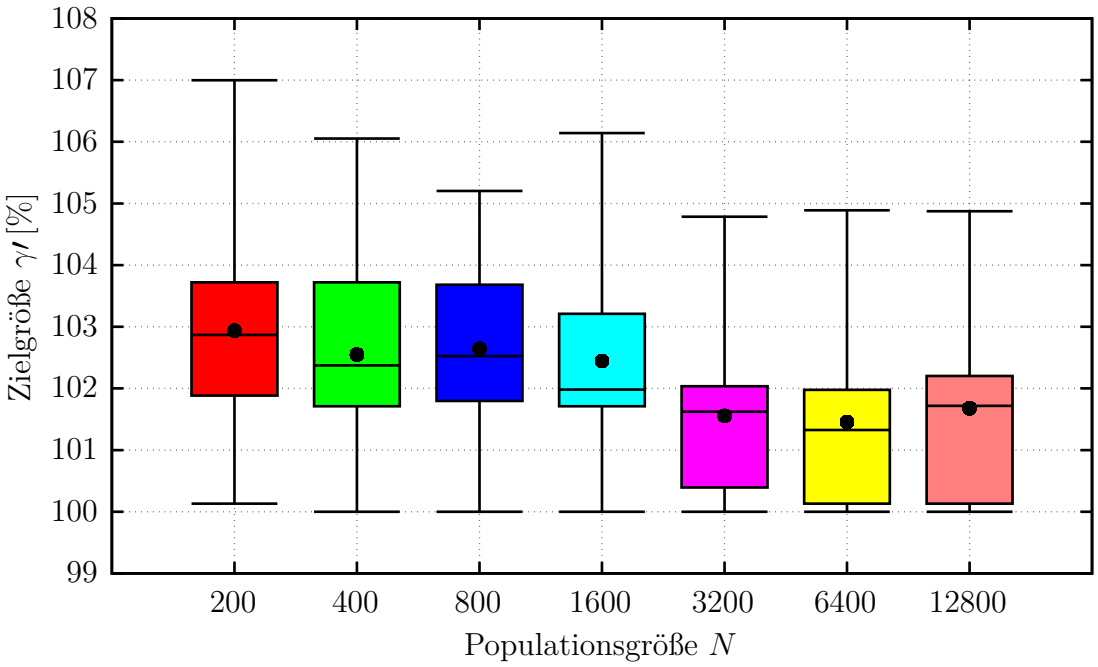


Abbildung A.26: Einfluss der Populationsgröße (Ameisenalgorithmus, S–224)

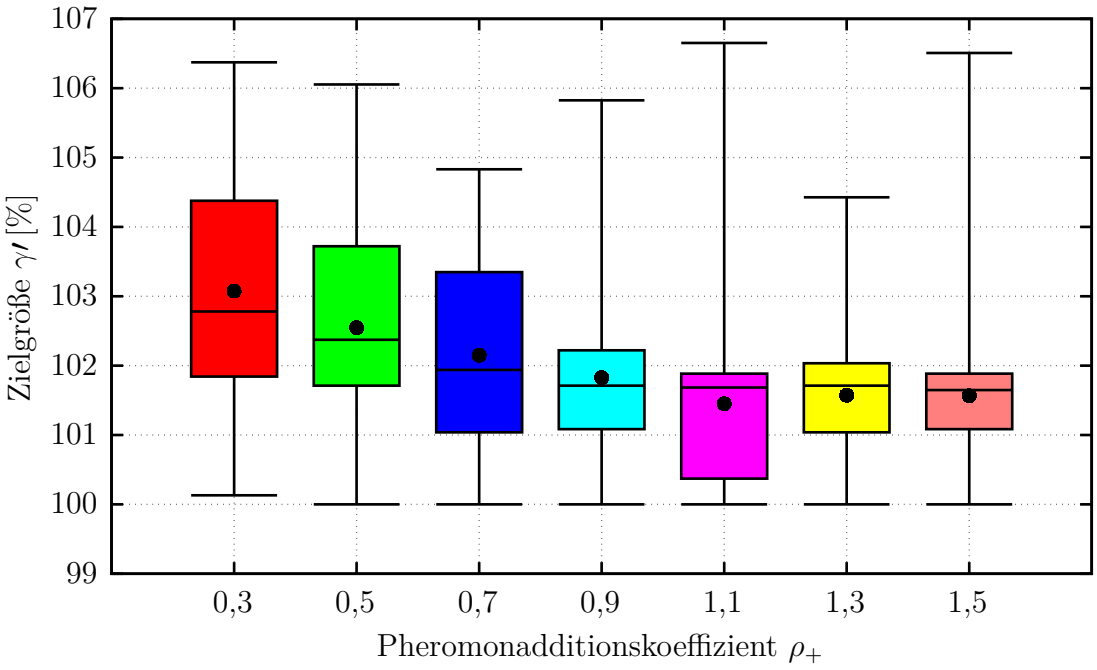


Abbildung A.27: Einfluss des Pheromonadditionskoeffizienten (Ameisenalgorithmus, S–224)

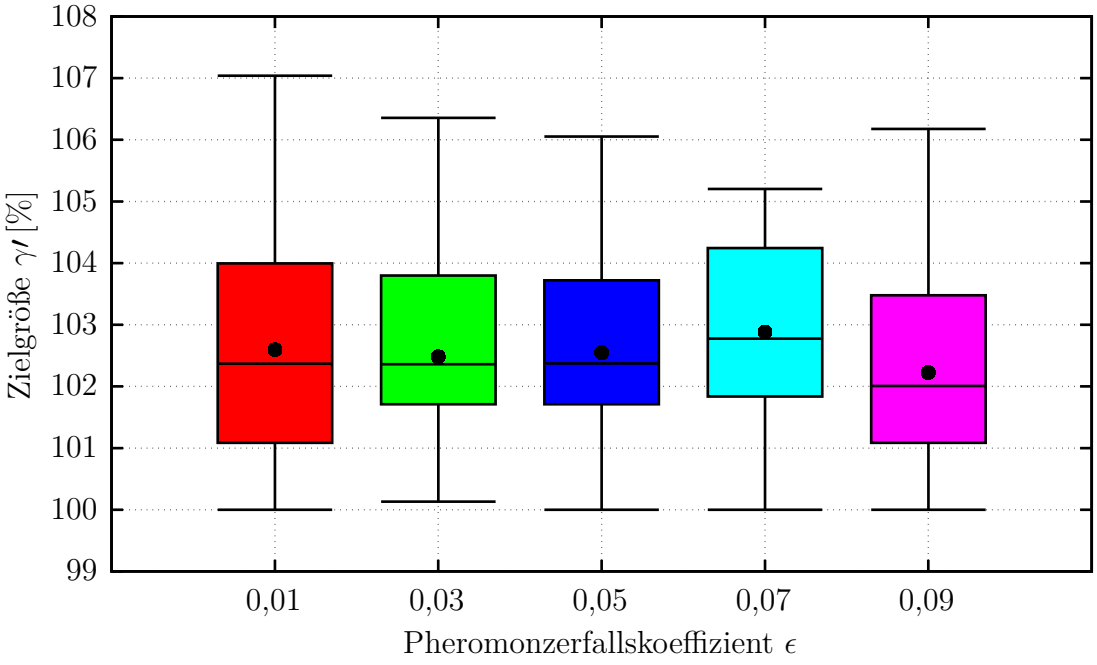


Abbildung A.28: Einfluss des Pheromonzerfallskoeffizienten (Ameisenalgorithmus, S–224)

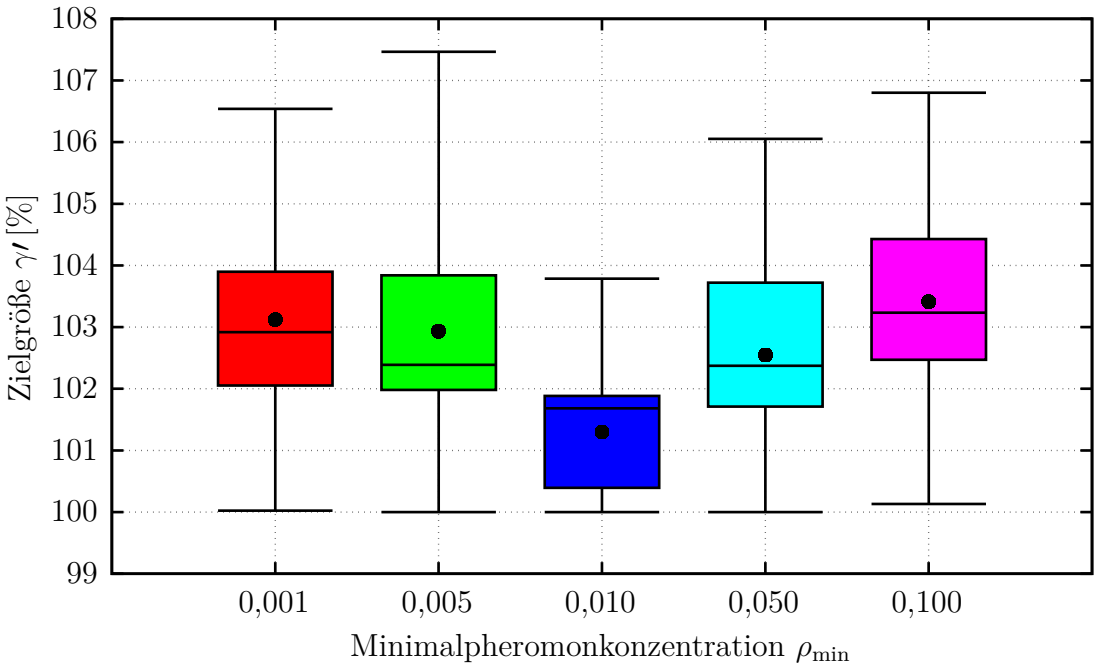


Abbildung A.29: Einfluss der Minimalpheromonkonzentration (Ameisenalgorithmus, S–224)

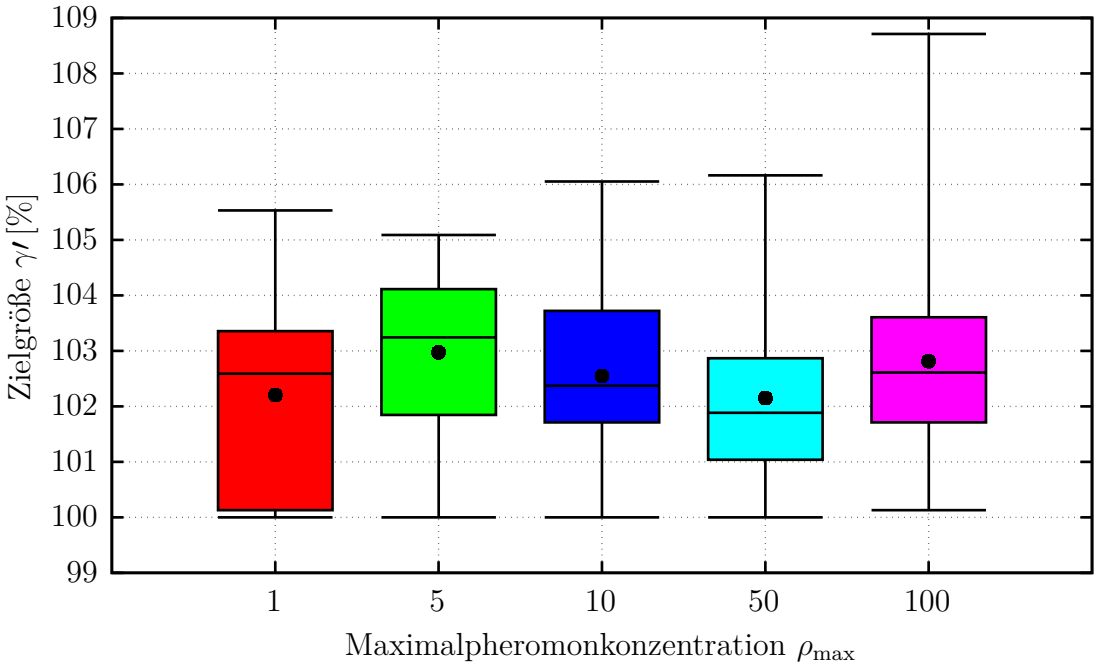


Abbildung A.30: Einfluss der Maximalpheromonkonzentration (Ameisenalgorithmus, S–224)

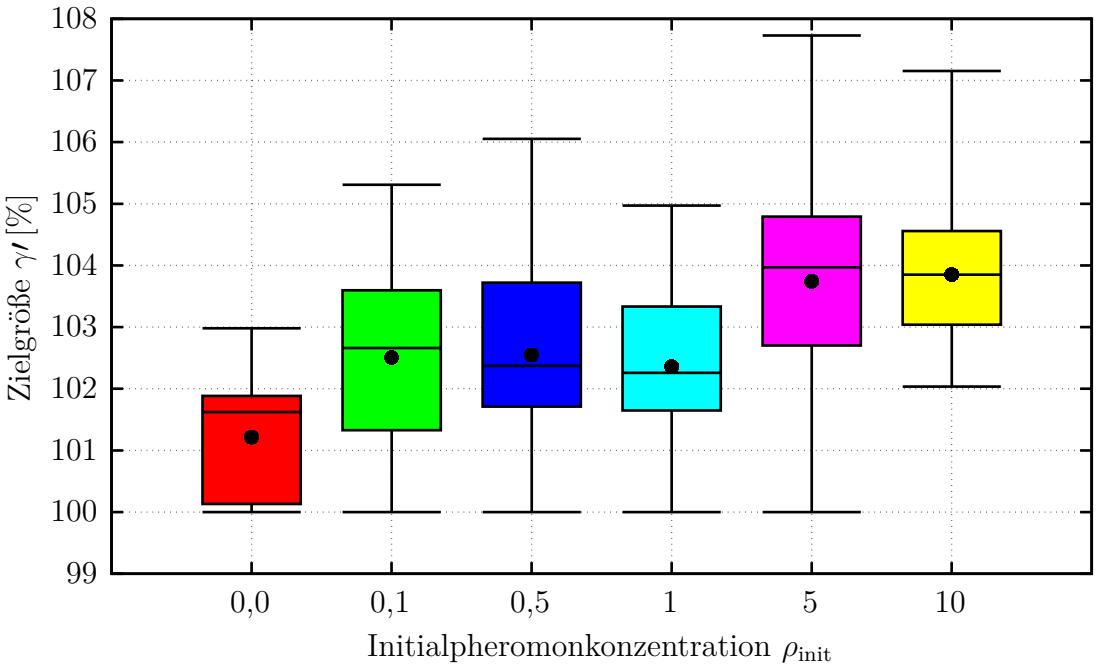


Abbildung A.31: Einfluss der Initialpheromonkonzentration (Ameisenalgorithmus, S–224)

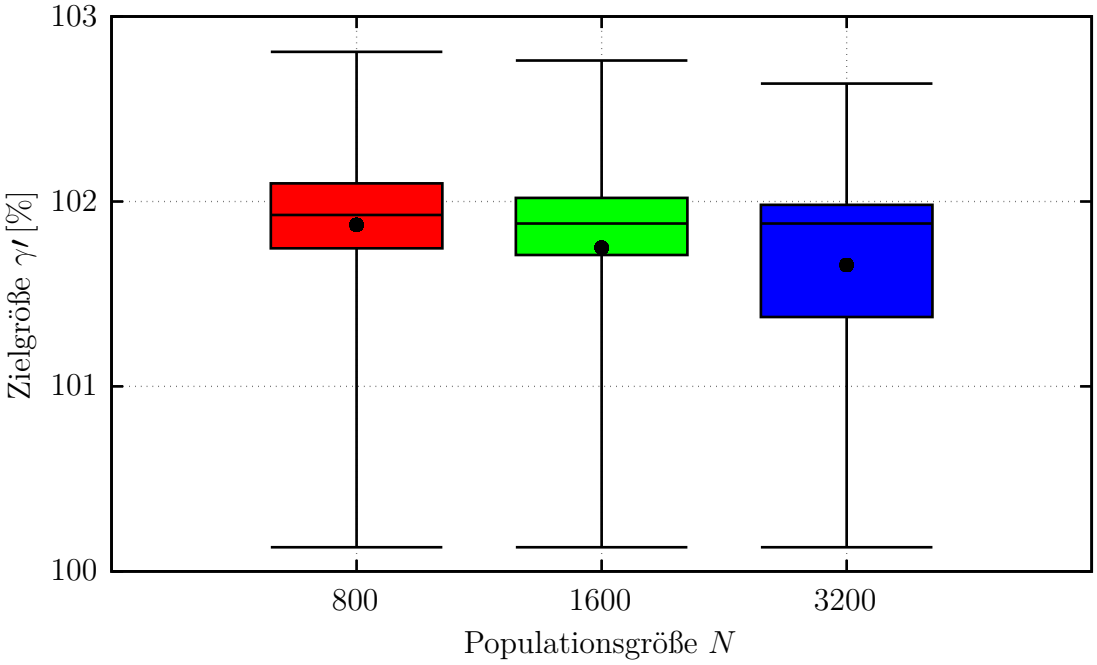


Abbildung A.32: Feintuning des Ameisenalgorithmus (S–224)

A.2.2 Ergebnisse der Parameterstudien für M–424

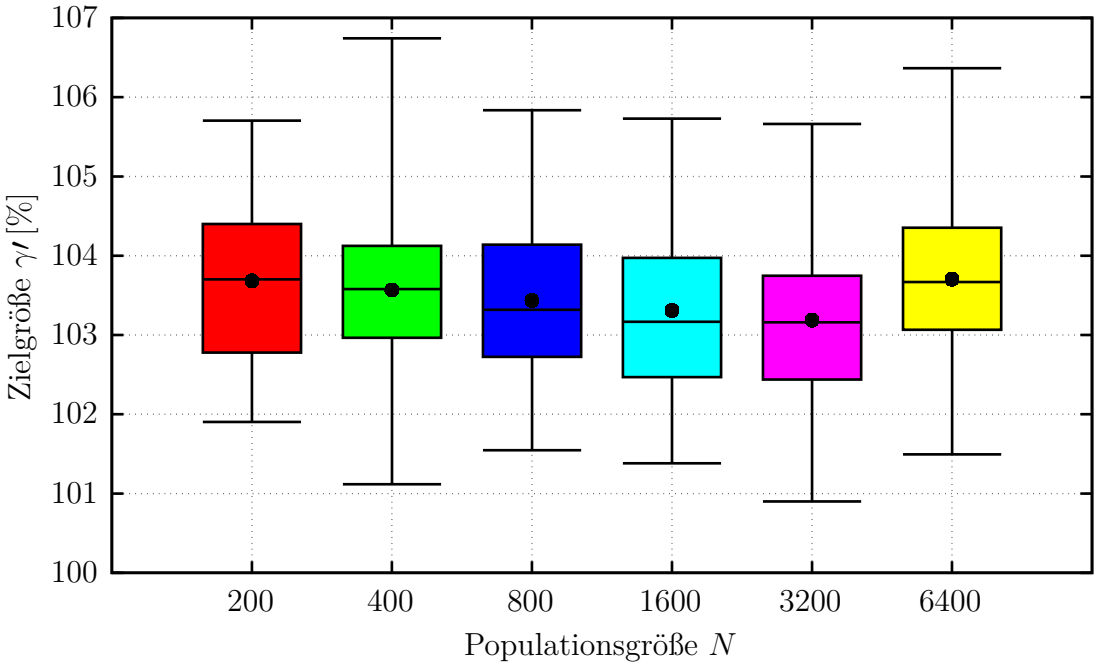


Abbildung A.33: Einfluss der Populationsgröße (Ameisenalgorithmus, M–424)

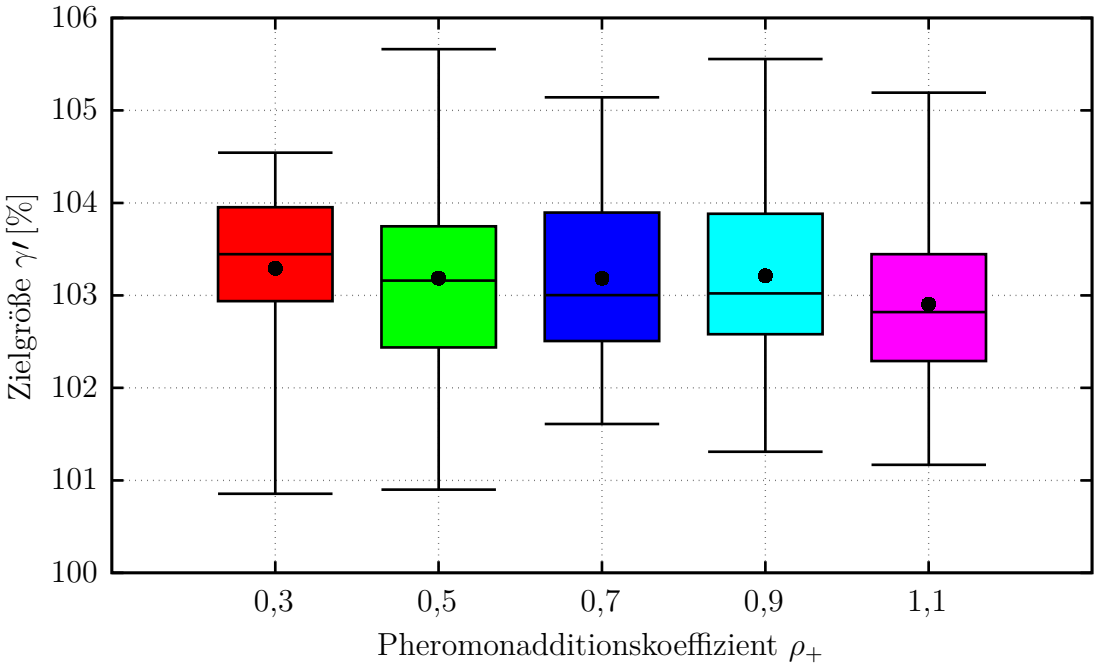


Abbildung A.34: Einfluss des Pheromonadditionskoeffizienten (Ameisenalgorithmus, M–424)

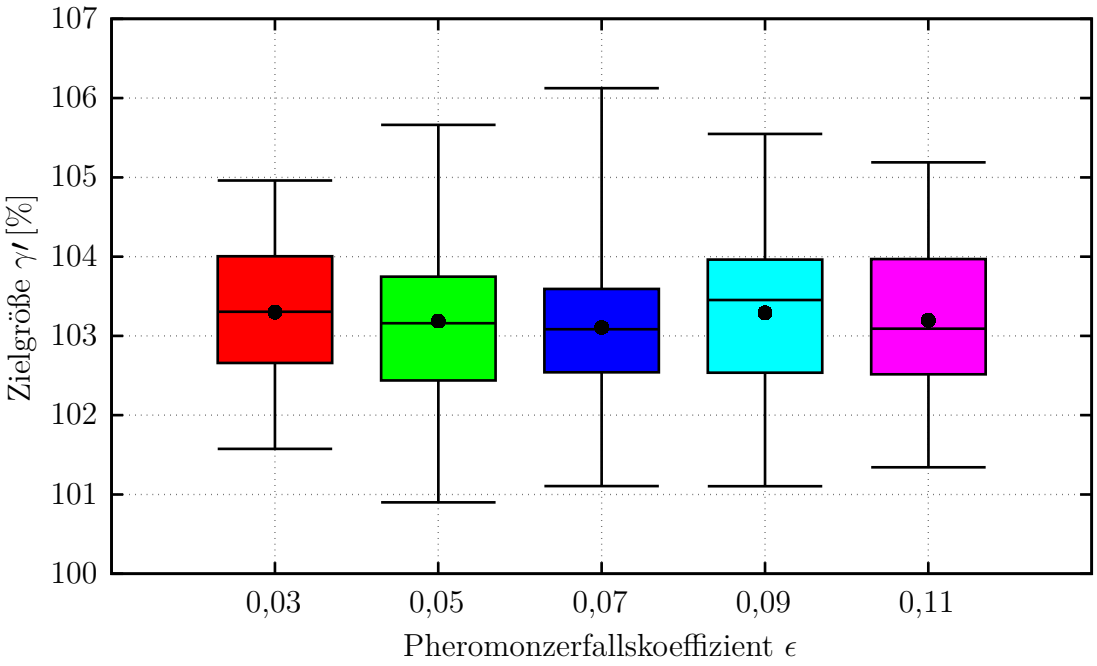


Abbildung A.35: Einfluss des Pheromonzerfallskoeffizienten (Ameisenalgorithmus, M–424)

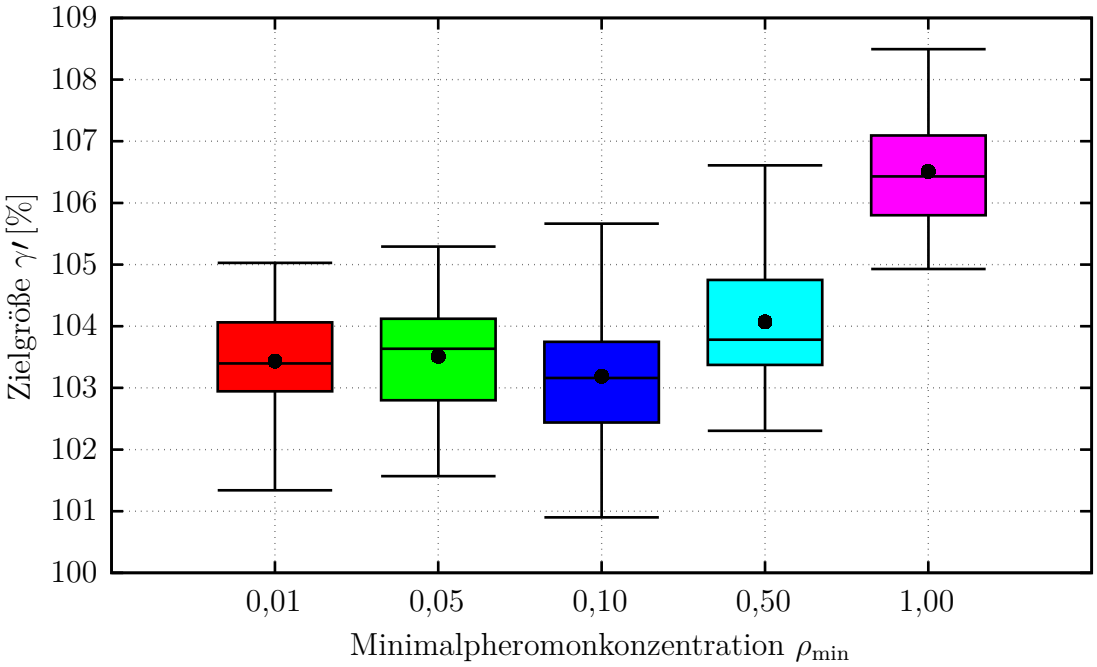


Abbildung A.36: Einfluss der Minimalpheromonkonzentration (Ameisenalgorithmus, M–424)

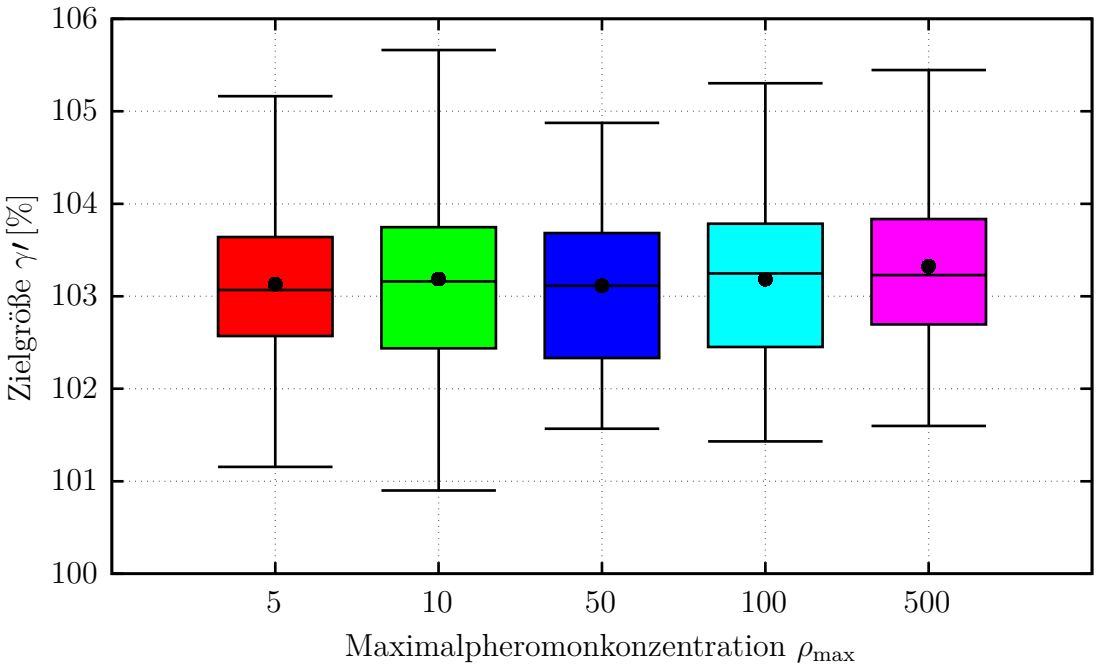


Abbildung A.37: Einfluss der Maximalpheromonkonzentration (Ameisenalgorithmus, M–424)

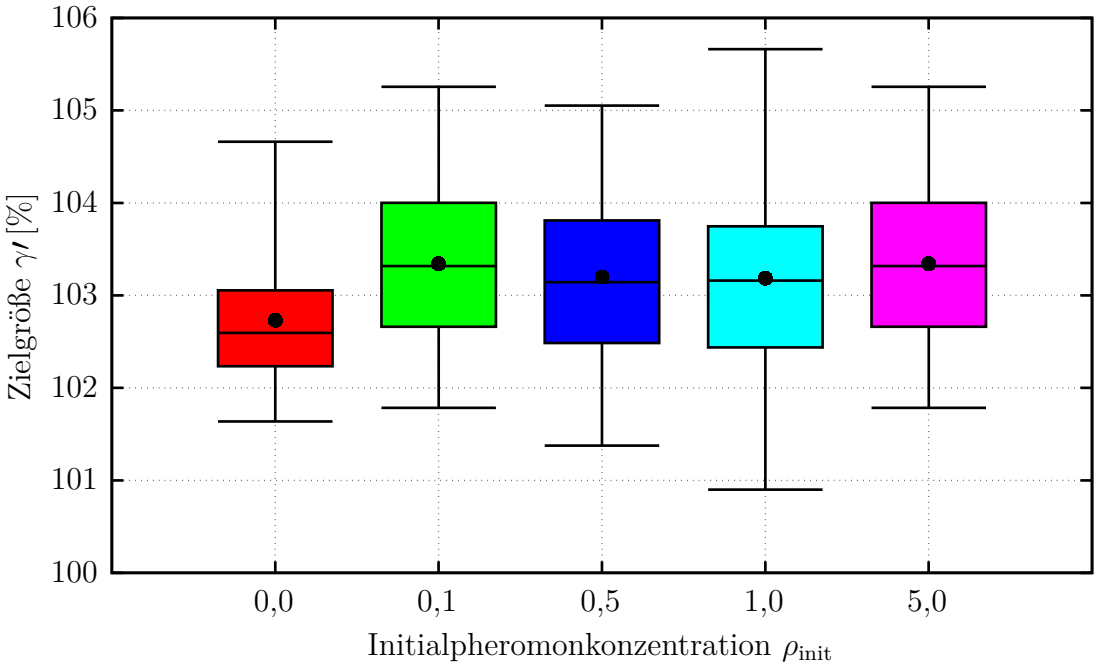


Abbildung A.38: Einfluss der Initialpheromonkonzentration (Ameisenalgorithmus, M–424)

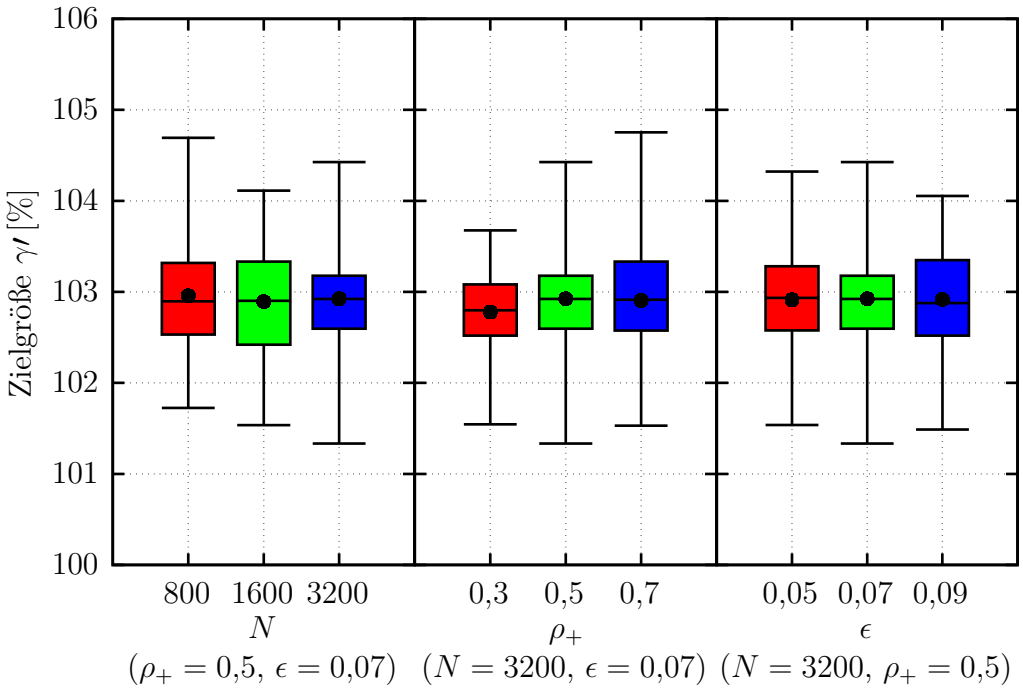


Abbildung A.39: Feintuning des Ameisenalgorithmus (M–424)

A.2.3 Ergebnisse der Parameterstudien für L–824

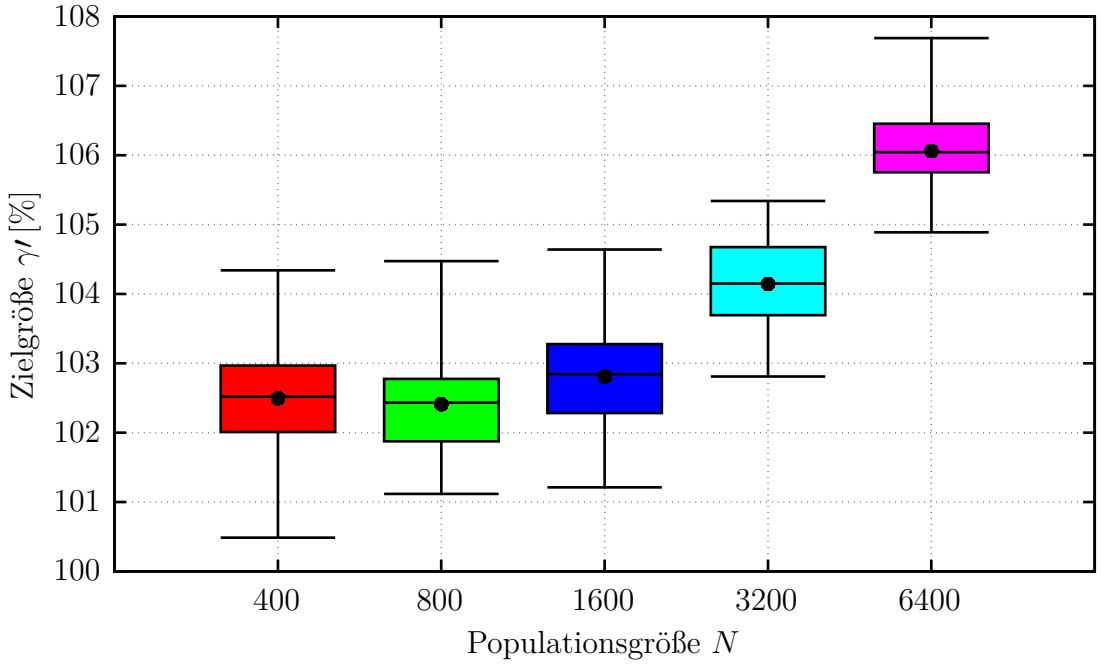


Abbildung A.40: Einfluss der Populationsgröße (Ameisenalgorithmus, L–824)

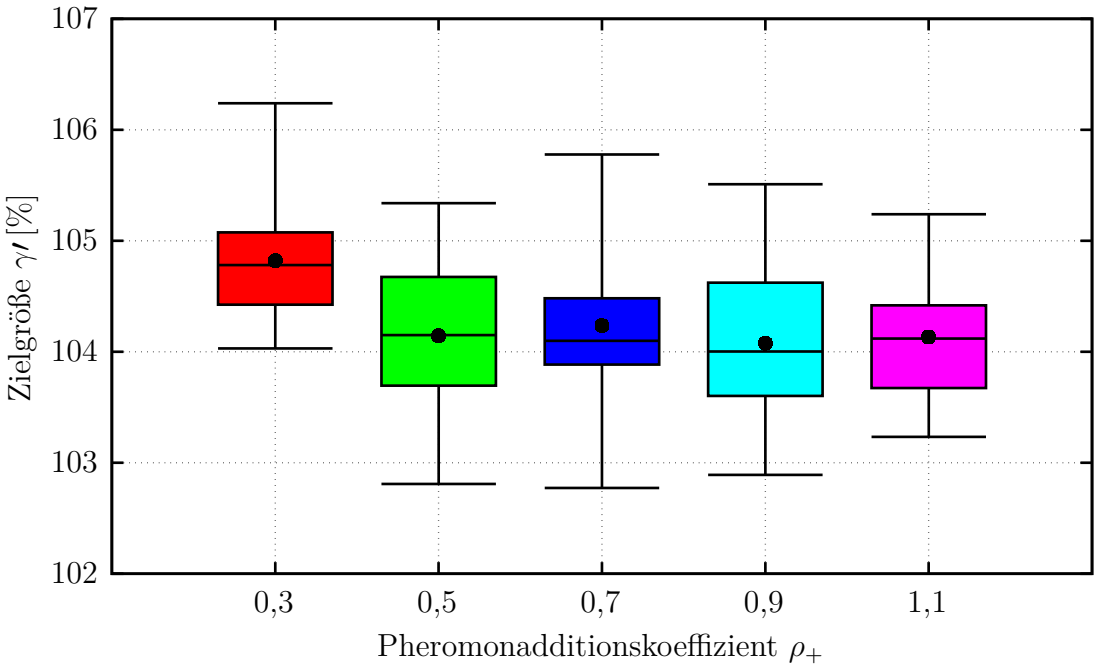


Abbildung A.41: Einfluss des Pheromonadditionskoeffizienten (Ameisenalgorithmus, L–824)

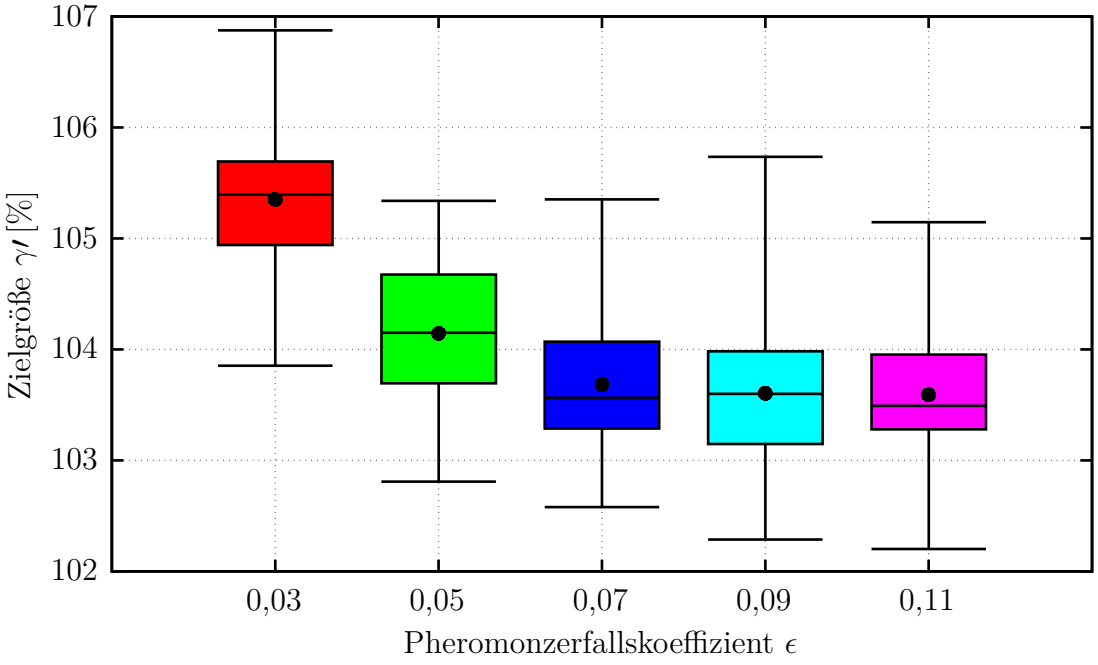


Abbildung A.42: Einfluss des Pheromonzerfallskoeffizienten (Ameisenalgorithmus, L–824)

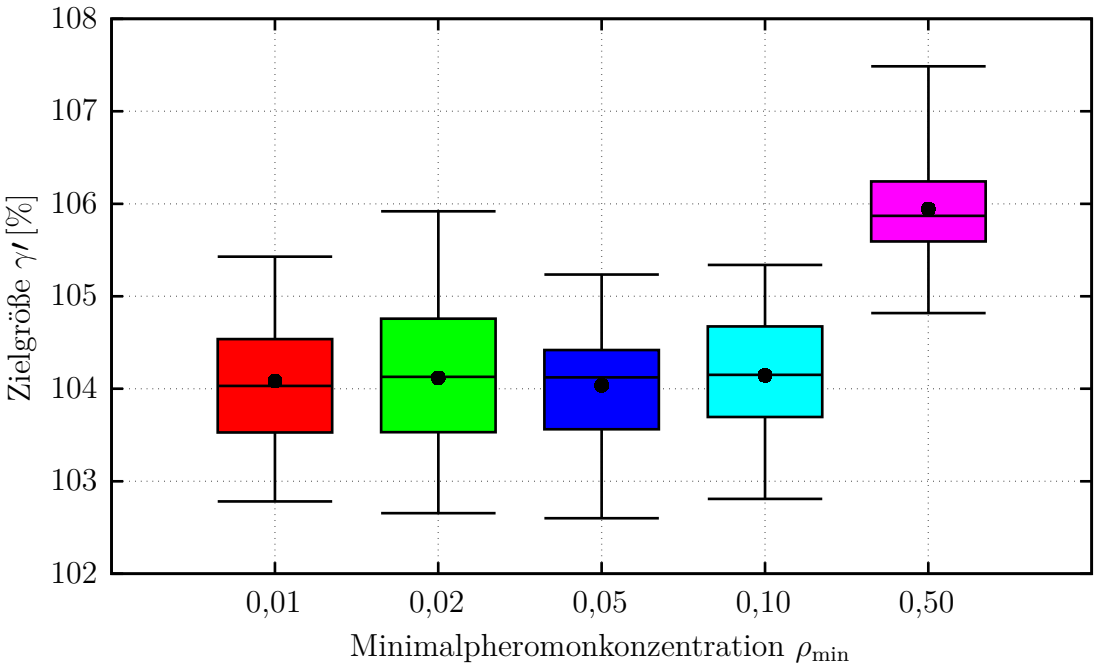


Abbildung A.43: Einfluss der Minimalpheromonkonzentration (Ameisenalgorithmus, L–824)

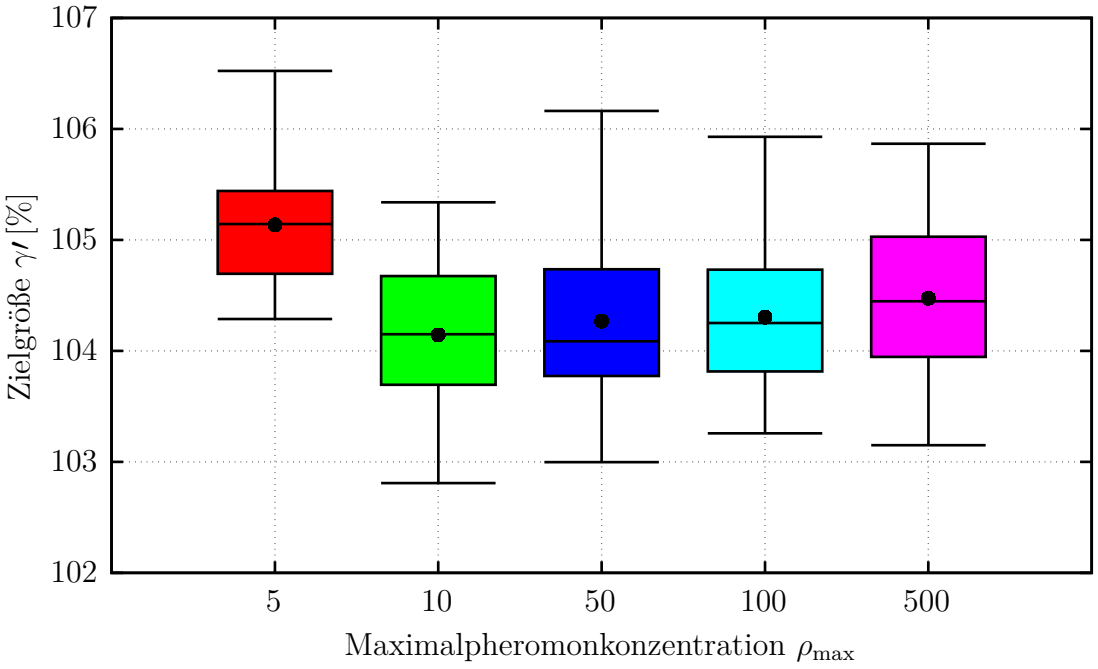


Abbildung A.44: Einfluss der Maximalpheromonkonzentration (Ameisenalgorithmus, L–824)

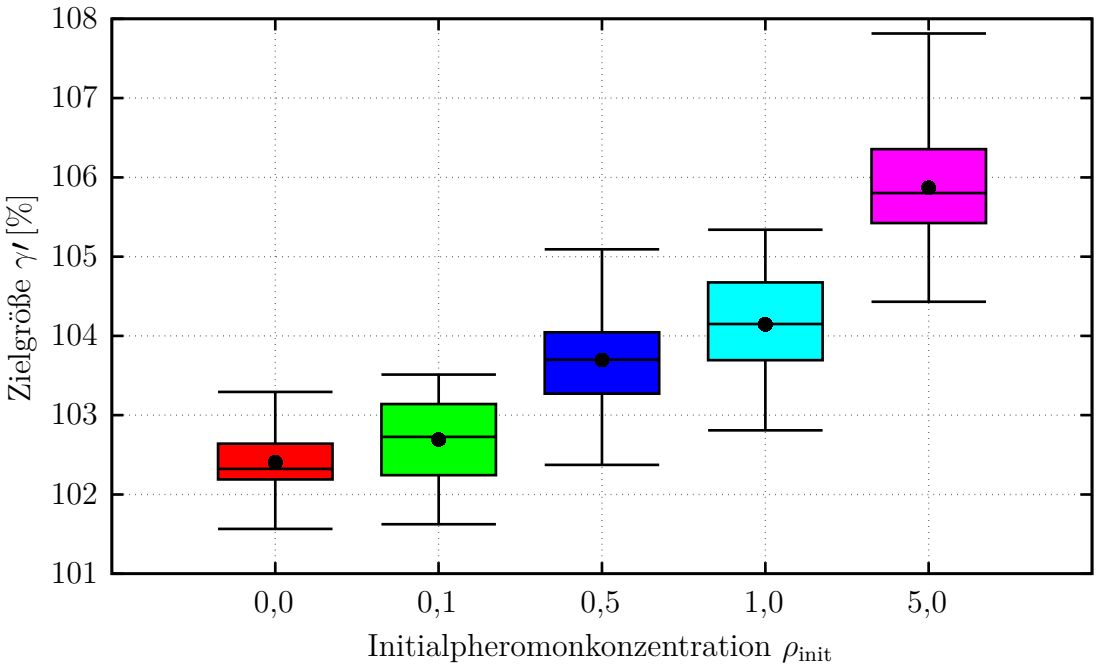


Abbildung A.45: Einfluss der Initialpheromonkonzentration (Ameisenalgorithmus, L–824)

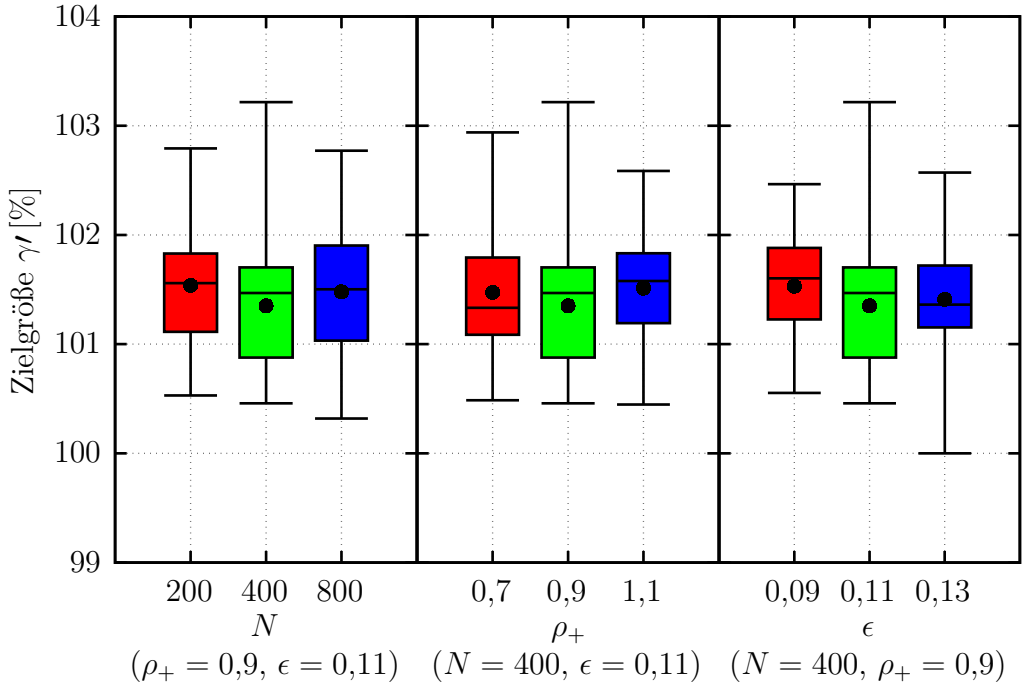


Abbildung A.46: Feintuning des Ameisenalgorithmus (L–824)

A.3 Abschätzung der Auslastung des Verfahrens

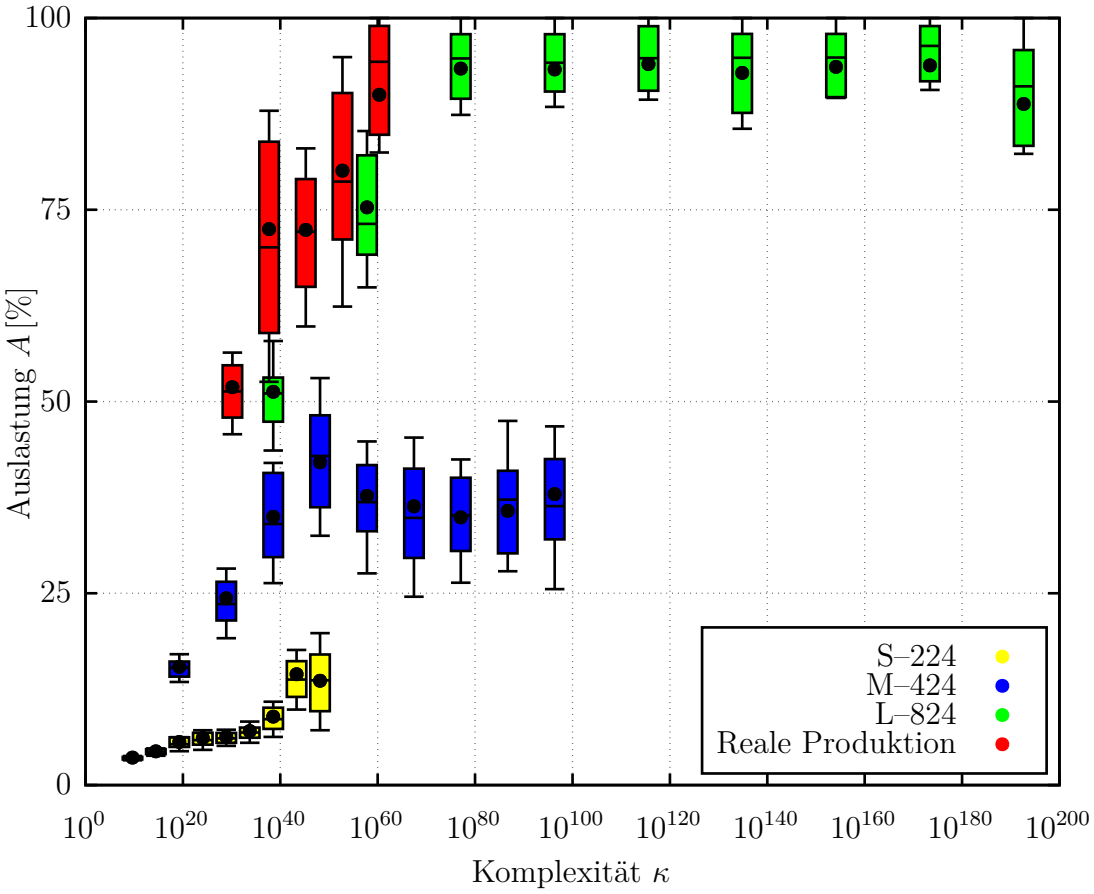


Abbildung A.47: Auslastung des Verfahrens auf Basis unterschiedlicher Daten