Robuste Kapazitätsgrobplanung bei Bedarfsunsicherheit unter Zuhilfenahme IT-gestützter Aggregation

Masterarbeit

von

Robert Bernerstätter

eingereicht am

Lehrstuhl Wirtschafts- und Betriebswissenschaften
der

Montanuniversität Leoben

Leoben, am 24. September 2014

Danksagung

Das Ende einer Arbeit ist die Zeit darüber zu rekapitulieren, welche Leute einem hilfreich zur Seite standen, um sie zu einem erfolgreichen Ende zu führen. Sei es mit professionellen Ratschlägen oder durch moralischen Beistand.

Ich will mich zuallererst bei meinen Betreuern Dr. Christian Rainer und DI Markus Gram, vom Lehrstuhl für Wirtschafts- und Betriebswissenschaften, bedanken. Sie haben mich mit ihrem Fachwissen und ihrer Geduld in dem letzten, für mich turbulenten Jahr, unterstützt.

Für die Betreuung bei meinem Industriepartner will ich DI (FH) Christian Grubmüller von der Produktionsplanung meinen Dank aussprechen. Ohne die rasche Umsetzung meiner Vorstellungen wäre das Ergebnis nicht so erfolgreich geworden.

Die Korrekturleserinnen und Korrekturleser dieser Arbeit dürfen natürlich nicht unerwähnt bleiben. Sie haben mir nicht nur bei dieser Arbeit, sondern bei allen Texten, die ich während meiner Studienzeit verfasst habe, so gut als möglich geholfen meine größte Schwäche zu überdecken.

Der Abschluss einer solchen Arbeit markiert auch einen wichtigen Kreuzungspunkt auf dem bisherigen Lebens- und Bildungsweg. In diesem Zusammenhang will ich all meine Freunde, die mich bis hier her unterstützt haben, dankend erwähnen. Ohne den gewichtigen Argumenten Einiger hätte ich mich wohl nicht für dieses Studium entschieden.

Ein besonderer Dank gilt meiner Familie für die moralische sowie finanzielle Unterstützung. Speziell sei meine Mutter Helga erwähnt, die mich laufend ermutigt und vorbehaltslos an meinen Erfolg geglaubt hat. Ohne ihre tatkräftige Unterstützung, vor allem in der Schullaufbahn vor der Universität, wäre ich kaum in der Situation diese Zeilen schreiben zu können. Danke Mama.

Zusammenfassung

Diese Masterarbeit beschäftigt sich zum einen mit der taktischen Kapazitätsplanung und zum anderen mit der robusten Planung bei nicht präzise prognostizierbarem Bedarf. Die Kapazitätsplanung auf taktischem Niveau wird als Kapazitätsgrobplanung bezeichnet. Sie hilft dem Planer abzuschätzen, ob ein Produktionsprogramm über einen Planungszeitraum von sechs bis achtzehn Monaten unter den Restriktionen der betrachteten Kapazitätsgröße umsetzbar ist. Dazu werden das Kapazitätsangebot der Produktion und der Kapazitätsbedarf des Produktionsprogramms ermittelt und gegenüber gestellt. Der Kapazitätsbedarf ist jedoch nicht immer eindeutig ermittelbar und unterliegt daher einer gewissen Unsicherheit. Um deren negativen Effekten wie Bedarfsunterdeckung oder Bestandsüberlauf entgegenzuwirken, wird mittels robuster Planung ein Produktionsplan entwickelt, der für verschiedene Szenarien eine allgemeine Gültigkeit besitzt.

Ein zentraler Punkt von Kapazitätsgrobplanung, der robusten Planung und folglich dieser Arbeit ist die Aggregation von Daten. Diese kann auf unterschiedliche Weise erfolgen. Im Fokus der Bedarfsunsicherheit und Robustheit wird die Fuzzy Mengenlehre angewandt, welche sehr gut Datenunsicherheiten abbildet. Die multivariaten Methoden sind weit verbreitete und einfach implementierbare Modelle, um Daten zu einheitlichen Gruppen zusammenzufassen oder Muster in Datenstrukturen zu erkennen.

Im Rahmen der praktischen Anwendung, wird ein multivariates Aggregationsmodell entwickelt. Dieses integriert robuste Ansätze, um Produktfamilien zu bilden, welche trotz Variation den Bedarfszahlen möglichst wenigen Änderungen ausgesetzt sind. Dieses Modell wird auf das Produktspektrums der AMAG rolling GmbH angewandt. Der entwickelte Ansatz wird in Form eines automatisch ablaufenden Programms im Grobplanungsprozess des Unternehmens eingesetzt. Das Programm ermöglicht eine Simulation verschiedener Szenarien und erhöht damit die Flexibilität, Genauigkeit und Robustheit der Kapazitätsgrobplanung.

Abstract

This master thesis deals with the issue of tactical capacity planning on the one hand and on the other with robust planning focused on poorly predictable demand. Capacity planning at the tactical level is also known as rough-cut capacity planning. It supports the planer to validate the production plan regarding the limiting capacity factor. The time scope is between six to eighteen months. To do so the available capacity of the production und the capacity demand of the production plan are determined and compared against each other. Due to uncertainty it is not always possible to predict the exact capacity demand. It is necessary to compensate for the negative effects of uncertainty in being stock-out or inventory overflow. Robust planning is one way of compensation by developing a production plan which remains valid for several scenarios.

Data aggregation constitutes a focal point of rough-cut capacity planning and robust planning. There a several ways to do data aggregation two of which are presented in closer detail. One is fuzzy set theory, which is well suited to cope with uncertainty in data and covers the issue of robustness. Multivariate methods are widely known and easy to implement models for grouping data in homogenous groups as well as to recognize patterns.

Within the scope of the industrial applicability a multivariate aggregation model has been developed. It incorporates a robust methodology to form product families. They should remain unaffected by demand variations. The developed model is applied as an automated computer program on the portfolio of the AMAG rolling inc. The program is part of the company's rough-cut planning process. It enables the planer to do simulations of different demand scenarios, hence increasing the flexibility, accuracy and robustness of the rough-cut capacity planning process.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich diese Arbeit selbstständig verfasst, andere als angegebene Quellen und Hilfsmittel nicht benutzt und mich auch sonst keiner unerlaubten Hilfsmittel bedient habe.

Leoben, am 24.09.2014

Robert Bernerstätter
Matrikelnummer: 0735041

Inhaltsverzeichnis

Danksagung ..... **iii**

Zusammenfassung ..... **iv**

Abstract ..... **v**

Eidesstattliche Erklärung ..... **vi**

Inhaltsverzeichnis ..... **vii**

Abbildungsverzeichnis ..... **x**

Tabellenverzeichnis ..... **xii**

Abkürzungsverzeichnis ..... **xiii**

1 Einleitung **1**

1.1 Aufgabenstellung und Problemabgrenzung 1

1.2 Aufbau der Arbeit 3

2 Kapazitätsgrobplanung **6**

2.1 Kapazitätsplanung 6

2.1.1 Bedarfsermittlung 8

2.1.2 Kapazitive Abstimmung 11

2.1.3 Kapazitätsplanungsverfahren 16

2.2 Taktische Grobplanung 19

2.2.1 Kapazitätsprofil 19

2.2.2 Produktionsprogrammplanung 21

2.2.3 Globale Belastungsfaktoren 23

2.3 Fazit der Kapazitätsgrobplanung 25

3 Robuste Planung **27**

3.1 Unsicherheit in der Planung 28

3.1.1 Auswirkungen von Unsicherheit 30

3.1.2 Reaktion auf Unsicherheit 32

3.2 Robustheitskriterien 35

3.2.1 Ergebnisrobustheit 35

3.2.2 Optimalitätsrobustheit 37
3.2.3 Zulässigkeitsrobustheit 38
3.2.4 Informationsrobustheit 39
3.2.5 Planungsrobustheit 40
3.3 Beurteilung und Wahl vom Modellen 41
3.3.1 Entscheidung bei Risiko 41
3.3.2 Entscheidung bei Ungewissheit 43
3.4 Planungsansätze 46
3.4.1 Rollierende Planung 46
3.4.2 Robuste stochastische Bedarfsermittlung 49
3.4.3 Robuste hierarchische Produktionsplanung 54
3.5 Fazit der robusten Planung 59

4 Datenaggregation 61
4.1 Fuzzy Mengenlehre 61
4.1.1 Fuzzy Produktfamilienbildung mit AHP 63
4.1.2 Produktionsplanung mit unsicherem Bedarf 66
4.2 Mulitvariate Methoden 71
4.2.1 Clusterverfahren 71
4.2.2 Gruppenbildung mittels Clusterung und genetischer Algorithmen 74

5 Robuste Produktfamilienbildung anhand eines Fallbeispiels 79
5.1 Stand des Planungsablaufes 79
5.1.1 Stand der Kapazitätsgrobplanung 80
5.1.2 Stand der Produktfamilienbildung 82
5.2 Alternativen und Algorithmusablauf 85
5.2.1 Entscheidungsfindung und Lösungsalternativen 85
5.2.2 Algorithmusablauf 88
5.3 Verfahrensbeschreibung 90
5.3.1 Datenvorbereitung 91
5.3.2 Genetischer Algorithmus 93
5.3.3 Clustervorgang 95
5.4 Fazit 97

6 Zusammenfassung und Ausblick 100
6.1 Ergebnisanalyse anhand der Forschungsfragen 101
6.2 Weiterer Forschungsbedarf 103

Literaturverzeichnis 104

Anhang ..... a
Dokumentation Programmcode ..... a
A. Planartikelermittlung ..... a
B. Hauptprogramm mit Attributermittlung ..... d
C. Hauptprogramm ohne Attributermittlung ..... n
D. Verknüpfung Auftragsdaten und Arbeitsplandaten ..... t
E. Vereinfachung des Arbeitsplans ..... w
F. Berechnung der Varianz der Nullstellen des Belastungsprofilpolynoms x
G. Berechnung der Distanzmatrix aufgrund der Anlagenfolge ..... z
H. Berechnung der Distanz zwischen zwei Anlagen ..... aa
I. Berechnung Anzahl unterschiedlichen Anlagenfolgen cc
J. Bestimmung der Attribute, die nicht beachtet werden sollten ..... ee
K. Population auf die richtige Größe bringen ..... ff
L. Durchführung des Clustervorgangs ..... gg
M. Berechnung der Fitness ll
N. Durchführung der fitnessproportionalen Selektion ..... oo
O. Durchführung einer n-Punkt Kreuzung ..... qq
P. Durchführung der genetischen Mutation ..... ss
Q. Ermittlung der Planartikel ..... tt
R. Einschränkung der Auftragsauswahl für die Planartikelermittlung vv
S. Protokollieren der Ergebnisse ..... yy
T. Ermittlung der Spaltenbezeichnung ..... bbb
U. Speichern des Ergebnisses ccc
Attributwichtigkeit auf Anlagen ..... fff
Protokolldatei ..... ggg
Datei der Auftragsdaten ..... hhh
Ergebnis Produktfamilie iii
Modellvariablen der aggregierten Planung ..... jjj

Abbildungsverzeichnis

Abbildung 1: Aufbau der Arbeit 4
Abbildung 2: Kapazitätsplanung 7
Abbildung 3: Berechnung der TEEP 7
Abbildung 4: Verlustzeiten - OEE vs. TEEP 8
Abbildung 5: Schritte der Bedarfsermittlung 9
Abbildung 6: Zeitliche Anpassung 12
Abbildung 7: Zeitlich-quantitative Anpassung 13
Abbildung 8: Zeitliche vs. quantitative Anpassung 13
Abbildung 9: Verbrauchsfunktion 14
Abbildung 10: Gesamtkosten intensitätsmäßige Anpassung 15
Abbildung 11: Gesamtkosten zeitlich-intensitätsmäßige Anpassung 16
Abbildung 12: Kapazitätsplanungsmethoden 18
Abbildung 13: Kapazitätsprofile entlang der Zeit 20
Abbildung 14: Kapazitätsprofile entlang des Wertstroms 21
Abbildung 15: Bausteine der Produktionsprogrammplanung 23
Abbildung 16: Belastungsentwicklung 25
Abbildung 17: Kapazitätsplanungsalternativen 26
Abbildung 18: Elastizität vs. Robustheit 28
Abbildung 19: Quellen für Planungsunsicherheit 29
Abbildung 20: Robustheitskriterien 35
Abbildung 21: Ergebnismatrix bei Risiko 41
Abbildung 22: Ergebnismatrix bei Ungewissheit 44
Abbildung 23: Konzept der rollierenden Planung 47
Abbildung 24: Optimalitätsrobustheit vs. Zulässigkeitsrobustheit 54
Abbildung 25: Abgestimmte Planungshorizonte 56
Abbildung 26: Struktogramm des taktischen Planungsablaufes 57
Abbildung 27: Konzept der robusten Planung 59
Abbildung 28: Trianguläre Zugehörigkeitsfunktion 63
Abbildung 29: Klassifikationsmatrix 64
Abbildung 30: Unterteilung des Clusterverfahren 72
Abbildung 31: 2-Punkt Kreuzung 76
Abbildung 32: Alte Konfigurationsdatei 84
Abbildung 33: Ablaufdiagramm des Programms 88
Abbildung 34: Auswahlmenü 89
Abbildung 35: SGE Auswahlliste 89

Abbildung 36: Populations-Generations-Matrix 94
Abbildung 37: Matrix Belastungsprofil 94
Abbildung 38: Kurvenfeld Produktfamilie mit Planartikel 95
Abbildung 39: Ausschnitt Planartikeldatei 97
Abbildung 40: Neue Konfigurationsdatei 98
Abbildung 41: Belastungsprofil - alte vs. neue Produktfamilie 102
Abbildung 42: Auszug aus Protokolldatei ..... ggg
Abbildung 43: Beispiel für Auftragsdatendatei ..... hhh
Abbildung 44: Auszug Planartikeldatei iii

Tabellenverzeichnis

Tabelle 1: Kapazitätsausgleichsmaßnahmen 11
Tabelle 2: Grundmodell der Produktionsprogrammplanung 21
Tabelle 3: Modellvariablen robuste stochastische Bedarfsermittlung 52
Tabelle 4: Variablen für AGGRPLAN 55
Tabelle 5: Zielfunktionsvariablen Fuzzybedarf 67
Tabelle 6: Nebenbedingungsvariablen Fuzzybedarf 68
Tabelle 7: Ausgewählte Proximitätsmaße 73
Tabelle 8: Clusterverfahren mit Berechnung 74
Tabelle 9: Terminologien Evolution vs. Gruppentechnologie 76
Tabelle 10: SGEs mit Gruppierungskriterien 83
Tabelle 11: Bewertung Verfahrensauswahl 86
Tabelle 12: Vergleich der Kriterienreihung 99
Tabelle 13: Attributeinfluss auf Anlagen ..... fff
Tabelle 14: Modellvariablen der aggregierten Planung ..... jjj

Abkürzungsverzeichnis

AG	Aktiengesellschaft
AHP	Analytic Hierarchy Process
AMAG	Austria Metall AG
AVO	Arbeitsvorgang
BOA	Belastungsorientierte Auftragsfreigabe
bzw.	beziehungsweise
C/C++	höhere Programmiersprachen
ebd.	ebendort
et al.	und andere
FMILP	Fuzzy Mixed Integer Linear Program
GE	Geldeinheit(en)
ggf.	gegebenenfalls
GmbH	Gesellschaft mit beschränkter Haftung
HAC	Hierarchische agglomerative Clusterung
inc.	Incorporated
KANBAN	Produktionsprozesssteuerungsmethode
KP	Kreuzungspunkt
MAA	Multi-Agenten-Ansätze
MATLAB	Firmenbezeichnung - Matrix Laboratory
MILP	Mixed Integer Linear Program
MRP	Material Requirement Planning (Materialbedarfsplanung)
MRP II	Manufacturing Resource Planning (Produktionsprogrammplanung)
MS	Microsoft
MTBF	Mean Time between Failures (durchschnittliche Zeit zw. zwei Fehlern)
MTTR	Mean Time to Repair (durchschnittliche Zeit zur Reparatur von Fehlern)
ODBC	Open Database Connectivity
OEE	Overall Equipment Effectivness
OEM	Original Equipment Manufacturer
S.	Seite
s.g.	so genannt(e)
SGE	Strategische Geschäftseinheit

SGF	Strategisches Geschäftsfeld
SQL	Structured Query Language
Stk.	Stück
TEEP	Total Effective Equipment Performance,
u.a.	unter anderen
vgl.	vergleiche
vl.	vielleicht
z.B.	zum Beispiel

1 Einleitung

Die vorliegende Arbeit befasst sich mit der Kapazitätsgrobplanung und legt den Fokus auf die robuste Planung. Zur technischen Umsetzung ist es nötig Datensätze zu Gruppen zusammenzufassen, daher beschäftigt sich die Arbeit weiters mit Möglichkeiten zu Datenaggregation. Die erarbeiteten Theorien sollen anhand eines Fallbeispiels angewandt werden, um eine robuste Datenaggregation zu erhalten.

Im folgenden Kapitel wird die Aufgabenstellung beschrieben und eine Problemabgrenzung vorgenommen. Für das Fallbeispiel erfolgt eine Vorstellung des Unternehmens und der speziellen Gegebenheiten, welche die Notwendigkeit der Arbeit bedingen. Ein grober Aufbau der Arbeit bildet den Abschluss des Kapitels.

1.1 Aufgabenstellung und Problemabgrenzung

Die Kapazitätsplanung hat zum Ziel den Kapazitätsbedarf dem Kapazitätsangebot eines Planungszeitraumes gegenüberzustellen, um zu überprüfen, ob der Bedarf vom Angebot gedeckt werden kann. Abhängig vom Planungszeitraum wird von der strategischen, der taktischen oder der operativen Kapazitätsplanung gesprochen.[1]

Die strategische Kapazitätsplanung umfasst langfristige Entscheidungen innerhalb eines Zeitraumes von mehreren Jahren. Dieser Planungsprozess beinhaltet Entscheidungen in der Größenordnung von Werksaus- oder Neubauten, Anlagenbeschaffungen oder einer alternativen Ressourcenquelle. Zielkonflikte ergeben sich in diesem multikriteriellen Entscheidungsprozess zwischen der Finanzierung, dem Durchsatz, der Durchlaufzeit und dem Risiko. Es kann alternativ vom Kapazitätsausbauplan, Anlagenbeschaffungsplan oder dem Ressourcenportfolioplan gesprochen werden.[2]

Auf taktischer Ebene befasst sich die Kapazitätsplanung mit Entscheidungen in einem mittelfristigen Betrachtungszeitraum von einem halben Jahr bis eineinhalb Jahren. Diese Form ist unter dem Begriff der Kapazitätsgrobplanung bekannt. Sie soll durch die Aggregation relevanter Betrachtungsgrößen eine Abschätzung liefern, ob ein Produktionsprogramm im folgenden Planungszeitraum umsetzbar ist. Mittels Maßnahmen, die in dieser Zeit umsetzbar sind, kann die Kapazität der Produktion erhöht werden, wodurch in dieser Planungsphase eine gewisse Flexibilität herrscht. Zu diesen Maßnahmen zählt u.a. die Einführung von Überstunden durch zusätzliche Schichten. Durch gesetzliche Regelungen und den höheren Kosten von Überstunden ist deren Einsatz limitiert, wodurch sich ein Zielkonflikt zwischen Kosten und Liefererfüllung ergibt. Wo es möglich ist, kann das Produktionsprogramm umgeplant werden. Damit wird die zeitliche Flexibilität genutzt und der Zielkonflikt, wenn nicht gelöst, wenigstens erheblich entspannt.[3]

Den kürzesten Planungshorizont hat die operative Ebene. Dieser reicht von wenigen Tagen bis kurz vor dem eigentlichen Produktionsstart. Ein viel verwendeter Begriff ist die

[1] Vgl. Steffen C. et al. (2013), S. 634.
[2] Vgl. Geng/Jiang (2009), S. 3639.
[3] Vgl. Chhaochhria/Graves (2013), S. 6860f.

Ablaufplanung. Hier werden alle Informationen, welche auf der taktischen Ebene noch in aggregierter Form vorliegen, vollständig disaggregiert.[4]

Die robuste Planung versucht speziell die Unsicherheit auf taktischer Ebene zu beherrschen. Die Ergebnisse sollen besser vorhersehbar und somit stabiler werden. Unsicherheit verursacht eine häufige Änderung des Plans. Diese Störungen können verringert oder verhindert werden. Dies geschieht durch die Ermittlung und Berücksichtigung der möglichen Abweichungen von Resultaten. Der Unsicherheit kann durch einen breit aufgestellten Plan, welcher mehrere Szenarien abbildet, begegnet werden.[5]

Bedarfsunsicherheit im Speziellen entsteht aufgrund der Volatilität des Marktes und durch steigende Konkurrenz. Da es vonseiten des Kunden eine steigende Erwartungshaltung im Bereich der Flexibilität gibt, werden immer kleinere Zeiträume, für kurzfristige Wünsche der Bestellmenge oder Spezifikation, zugestanden. Somit kommt es zu einer größeren Fluktuation im Bedarf und zu kurzfristigeren Planungszeiträumen.[6]

Im Rahmen der theoretischen Ausarbeitung sollen die Themenblöcke der Kapazitätsgrobplanung und der robusten Planung aufbereitet werden. Es soll die Unsicherheit im Allgemeinen und die Bedarfsunsicherheit im Speziellen berücksichtigt werden. Diese Themen bilden den betriebswirtschaftlichen Schwerpunkt der Arbeit. Die Aggregation, als unterstützende Maßnahme zur taktischen Planung, soll den Kern des technischen Schwerpunktes bilden. Neben der theoretischen Ausarbeitung der Themenblöcke soll mittels Beispielen eine Übersicht von existenten Einsatzgebieten der Verfahren gegeben werden.

In der praktischen Umsetzung soll eine Vorgehensweise gefunden werden, die eine robuste Produktfamilienbildung erlaubt. Dieses Verfahren muss in den Prozess der Kapazitätsgrobplanung der AMAG rolling GmbH integrierbar sein und die speziellen Gegebenheiten berücksichtigen. Das Produktportfolio der AMAG rolling ist durch die breite Aufstellung in allen Legierungsgruppen äußerst vielfältig. Dies unterscheidet die AMAG rolling von anderen Aluminiumwalzwerken, welche nur eine oder wenige Legierungsgruppen anbieten und in diesen einen entsprechend hohen Ausstoß haben. Die Planung ist vergleichsweiße einfach. Bei der AMAG rolling ergibt sich die potentielle Gefahr von langen Durchlaufzeiten durch häufige Umrüstvorgänge. Zur Gewährleistung der Flexibilität ist es nötig noch nicht ausgeschöpftes Optimierungspotential in der Planung zu nutzen. Ansätze der robusten Planung beinhalten ein solches Potential. Da die Produktfamilienbildung eine zentrale Rolle im Grobplanungsprozess des Unternehmens spielt, wird ein großes Potential in diesem Bereich angenommen. Die Aggregation zu Produktfamilien ist bereits mit einem Computerprogramm automatisiert, erfolgt aber nicht robust und ist somit noch anfällig für Änderungen in den Annahmen über Kundenaufträgen. Diese Verbesserungsmöglichkeiten im Programm sollen durch die neue Vorgehensweise und das erweiterte neue Programm ausgeschöpft werden. Das Ziel ist es den Kapazitätsgrobplanungsprozess der AMAG rolling durch das Programm zu unterstützen, zu verbessern und zu erleichtern. Die Integration und Anwendung beim

[4] Vgl. Bushuev (2013), S. 1050.

[5] Vgl. Van Landeghem/Vanmaele (2002), S. 773.

[6] Vgl. Gupta/Maranas (2003), S. 1219f.

Industriepartner soll mittels eines Computerprogramms erfolgen, welches die vorliegenden Daten verwendet. Die Ergebnisse müssen in der bisher üblichen bzw. mit dieser kompatiblen Form ausgegeben werden. Zusätzlich soll versucht werden die Bedienung des Programms für den Anwender einfach zu halten.

Forschungsfragen

Ergebend aus der Aufgabenstellung und Problemabgrenzung stellen sich verschiedene Fragen, welche in Rahmen dieser Arbeit beantwortet werden sollen. Am Ende wird ein Resümee der Ergebnisse betreffend der Forschungsfragen gezogen.

Ist es möglich, die für die Kapazitätsgrobplanung nötige Datenaggregation unter dem Gesichtspunkt der Robustheit vorzunehmen?

Kann mithilfe des Robustheitsmaßes die Auswahl der Aggregationskriterien erfolgen?

1.2 Aufbau der Arbeit

Zu Beginn stellte sich die Frage nach bewährten und alternativen Möglichkeiten zur Produktfamilienbildung. Im Mittelpunkt der Lösung stand eine umfangreiche Literaturrecherche. Zum Einsatz kamen bekannte Portale für wissenschaftliche Arbeiten, wie TAYLOR & FRANCIS ONLINE[7], SCIENCEDIREKT[8], SPRINGER LINK[9] oder allgemein GOOGLE SCHOLAR[10]. Die Suchbegriffe deckten die Themen robuste Planung, Produktfamilien, Kapazitätsgrobplanung, aggregierte Planung, taktische Planung und andere des Themengebietes ab.

Die Arbeit gliedert sich in weitere fünf Kapitel. Die Kapitel zwei bis vier beschäftigen sich mit der theoretischen Ausarbeitung der Thematik und Kapitel fünf mit der Umsetzung der Problemstellung der robusten Produktfamilienbildung anhand des Fallbeispiels. Abbildung 1 zeigt den Aufbau der Arbeit als graphischen Überblick.

Kapitel zwei erarbeitet das Thema der Kapazitätsgrobplanung. Dazu wird vorab die Kapazitätsplanung im Allgemeinen betrachtet. Wichtige Begriffe im Umfeld dieses Gebiets werden definiert. Weiters wird die Bedarfsplanung behandelt und passend kapazitive Anpassungsverfahren. Zum Schluss erfolgt ein grober Überblick über Kapazitätsplanungsverfahren. Die taktische Grobplanung beschäftigt sich mit dem zentralen Werkzeug des Kapazitätsprofils und stellt zwei Möglichkeiten zur Kapazitätsgrobplanung vor.

Die robuste Planung wird in Kapitel drei behandelt. Zu Beginn erfolgt eine begriffliche Abgrenzung zwischen Robustheit und Elastizität (engl. resilience). Es folgen die verschiedenen Kriterien, in denen Robustheit erlangt werden kann. Weiters erfolgt eine Unterscheidung zwischen der Bewertung von Modellen unter Risiko und Ungewissheit. Zum Ende werden gängige Planungsmodelle erörtert, an denen die Methoden der robusten Planung anwendbar sind.

[7] Vgl. TandF (2014)
[8] Vgl. Science Direct (2014)
[9] Vgl. Springerlink (2014)
[10] Vgl. GoogleScholar (2014)

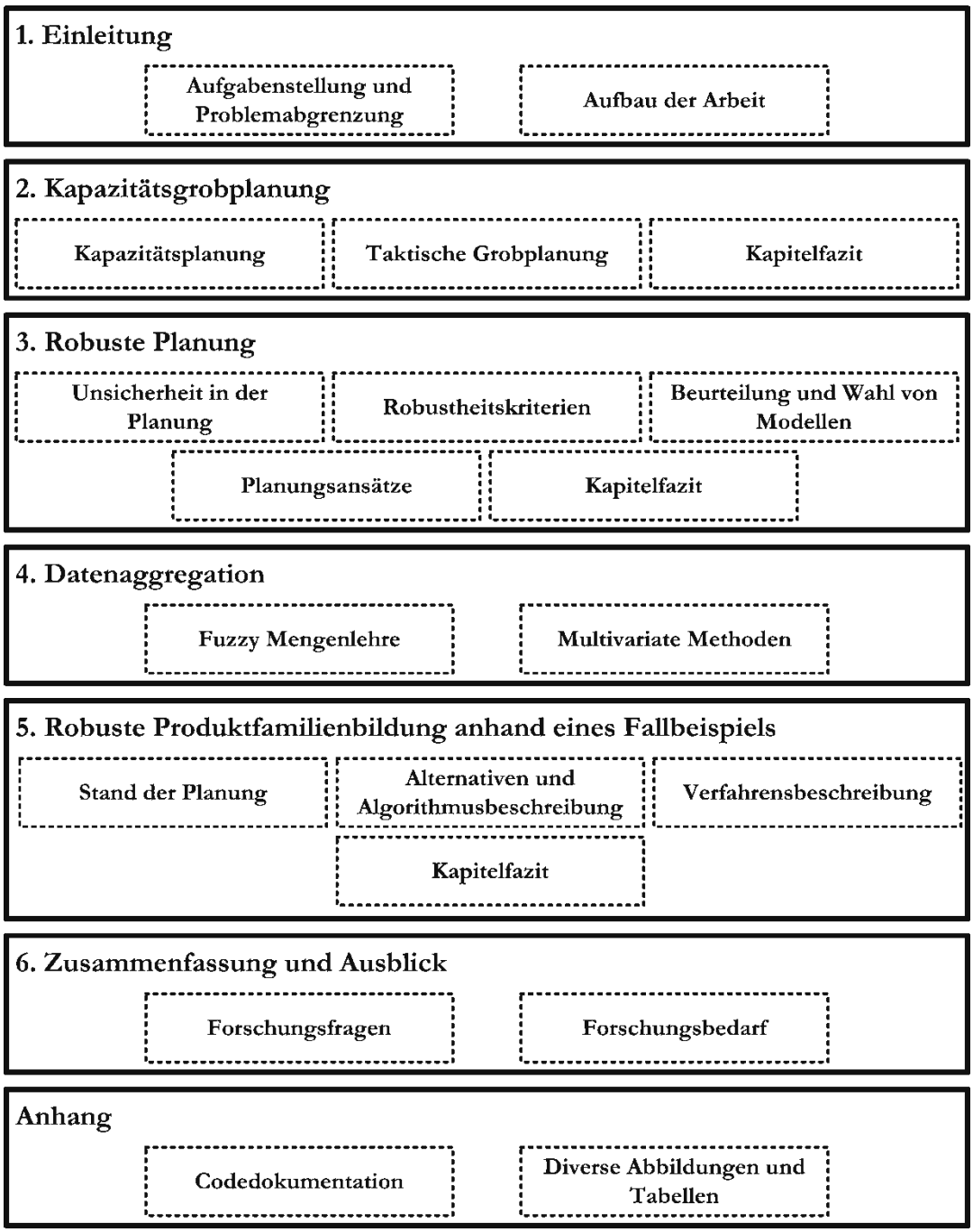


Abbildung 1: Aufbau der Arbeit[11]

Als ein zentrales Thema der Grobplanung beschäftigt sich Kapitel vier mit der Aggregation von Daten. Es werden zwei grundlegende Herangehensweisen vorgestellt. Die Fuzzy-Mengenlehre, welche sich gut für unscharf abgegrenzte Probleme und Unsicherheit eignet und multivariate Methoden, welche unproblematisch große Datenmengen zu Gruppen zusammenfassen. Für beide Möglichkeiten werden Anwendungsbeispiele gegeben.

Die Umsetzung der Aufgabenstellung erfolgt im praktischen Teil in Kapitel fünf, anhand eines Fallbeispiels. Es wird vorab die Ist-Situation der Planung beschrieben. Danach

[11] Quelle: Eigene Darstellung

werden Alternativen für die neue Vorgehensweise erörtert und diese in Form des Computeralgorithmus vorgestellt. Die genaue Beschreibung der Schlüsselstellen der neuen Vorgehensweise findet im letzten Teilkapitel statt.

Eine Zusammenfassung von den Erkenntnissen der Arbeit erfolgt im letzten Kapitel. Als Teil dieses Überblicks werden die Forschungsfragen beantwortet. Der weitere Forschungsbedarf, der sich aufgrund der Ergebnisse der Arbeit und des Umfanges des Themas ergibt, bildet den Abschluss der Arbeit.

Im Anhang findet sich die vollständige Dokumentation des Programmcodes mit den entsprechenden Codestellen. Weiters sind Abbildungen von diversen Exceldatenblättern und Tabellen zu finden, auf welche im Laufe der Arbeit verwiesen werden.

2 Kapazitätsgrobplanung

Die Kapazitätsgrobplanung unterstützt den Planer in der Erstellung des Produktionsplanes, indem sie sichtbar macht, ob die Annahmen betreffend der Kapazitätsbedarfe bei Personal, Anlagen oder externer Zulieferer plausibel sind.[12]

Das folgende Kapitel gibt einen Überblick über die Kapazitätsplanung und die taktische Grobplanung. Neben der Definition der Kapazität und den bestimmenden Kennzahlen von Arbeitssystemen sollen Anpassungsmaßnamen und Planungsverfahren erläutert werden. Die Grobplanung gibt einen Auszug der Verfahren im Allgemeinen mit einem Schwerpunkt auf den Faktor Kapazität im Speziellen.

2.1 Kapazitätsplanung

Die zentrale Betrachtungsgröße in der Kapazitätsplanung ist die Kapazität. Diese wird vorab definiert.

„Die Kapazität ist das quantitative und qualitative Leistungsvermögen einer Fertigungseinheit pro Zeiteinheit, definiert durch die auf Dauer maximal mögliche Leistung." [13]

Die Kapazitätsplanung setzt den Kapazitätsbedarf und das Kapazitätsangebot in Beziehung. Der Kapazitätsbedarf ergibt sich aus der Nachfrage von Aufträgen. Das Kapazitätsangebot wird in der Produktionsplanung bestimmt und kann auch wie diese in drei Planungsebenen, die strategische, taktische und operative, unterteilt werden. Strategische Entscheidungen sind langfristig und umfassen jene Festlegungen, welche das Leistungsvermögen des gesamten Produktionssystems bestimmen. Die taktische Ebene bestimmt konkrete Werte für die Kapazität des Personals, der Anlagen und Betriebsmittel. Sie hat einen mittelfristigen Zeithorizont von mehreren Monaten bis maximal zwei Jahren. Auf der operativen und kurzfristigsten Ebene findet die Kapazitätsbelegungsplanung statt.[14] Grafisch gesehen wird versucht das Belastungsprofil von der Kapazitätshüllkurve zu umschließen.

Abbildung 2 veranschaulicht diesen Vorgang. Sie zeigt auch gut die Eigenschaften der Definition der Kapazität von oben. Auf der Abszisse ist die Zeit und auf der Ordinate ist das Leistungsvermögen für fixe Zeiteinheiten aufgetragen.

[12] Vgl. Daniel et al. (1997), S. 239.

[13] Zsifkovits (2013), S. 151.

[14] Vgl. Steffen C. et al. (2013), S. 634f.

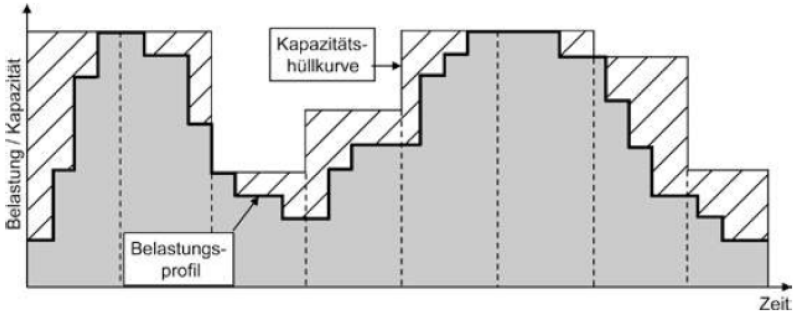


Abbildung 2: Kapazitätsplanung[15]

Es gibt verschiedene Maße für das Leistungsvermögen einer Anlage. Die gängigsten sind die Gesamtanlageneffektivität (OEE - Overall Equipment Effectivness) und die Totale effektive Anlagenproduktivität (TEEP - Total Effective Equipment Performance). Während die OEE die genutzte Zeit für die Gutausbringung der Planbelegungszeit gegenüberstellt, vergleicht die TEEP die genutzte Zeit für die Gutausbringung mit der Kalenderzeit. In der Regel ist sie kleiner als die Gesamtanlageneffektivität.[16] Die beiden Maße sind einander sehr ähnlich, da die TEEP eine Erweiterung der OEE ist. Mithilfe dieser beiden Größen können Probleme in der Kapazitätsausnutzung leicht identifiziert werden.[17]

Nachfolgend wird nur die TEEP genauer betrachtet. Aufgrund des Bezugs auf die Kalenderzeit und der genaueren Unterscheidung der Stillstandzeiten besitzt sie gegenüber der OEE Vorteile. Abbildung 3 zeigt das Zusammenspiel der Teilgrößen für die Berechnung der TEEP.

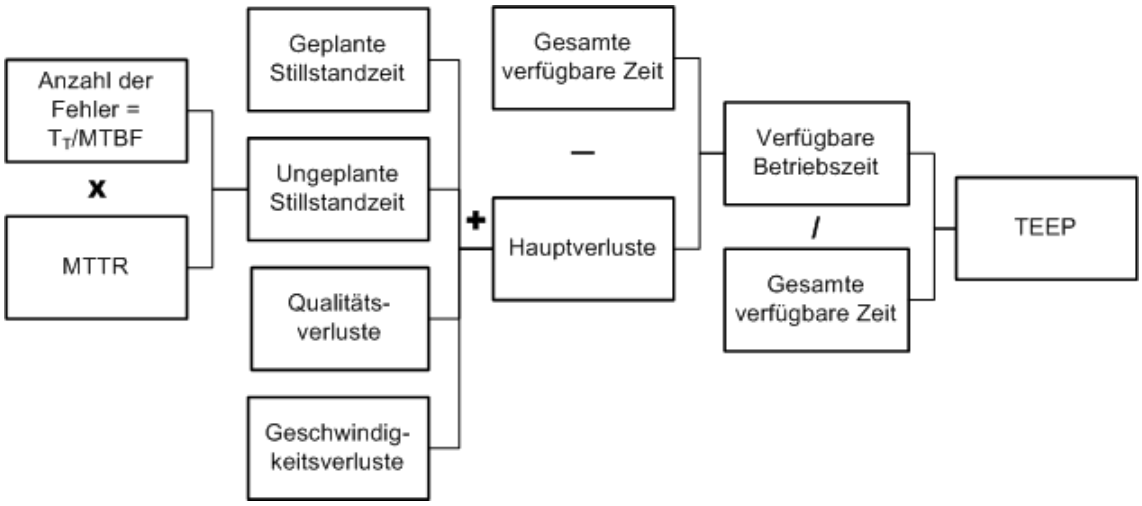


Abbildung 3: Berechnung der TEEP[18]

Die ungeplante Stillstandzeit soll möglichst gering gehalten werden. Dies wird durch verbesserte Instandhaltung erreicht, was jedoch die geplante Stillstandzeit durch die

[15] Quelle: Steffen C. et al. (2013), S. 634.
[16] Vgl. Jodlbauer (2008), S. 25.
[17] Vgl. Lanza et al. (2013), S. 32.
[18] Quelle: in Anlehnung an Muchiri/Pintelon (2008), S. 3522.

Wartung geringfügig erhöhen kann. Somit ist es wichtig, dass sich die Zeit in Summe verringert.

Wesentliche Maßgrößen in diesem Zusammenhang sind die durchschnittliche Zeit zwischen dem Auftreten von zwei Fehlern – MTBF (Mean Time between Failures) – und die durchschnittliche Zeit diese Fehler zu beheben – MTTR (Mean Time to Repair). Als Voraussetzung für die MTBF wird angenommen, dass die Fehler behebbar sind.[19]

Die Zusammensetzung der Verluste aus Abbildung 3 ist in Abbildung 4 zu sehen. Die Verluste durch Ausfälle, genauer die technischen Störungen sind die ungeplanten Stillstände die durch MTBF und MTTR abgebildet werden.

Die TEEP ist einheitenlos, im Wertebereich zwischen null und eins (0%-100%) und kann als Abschlagfaktor auf die gesamte verfügbare Kapazität gesehen werden.[20]

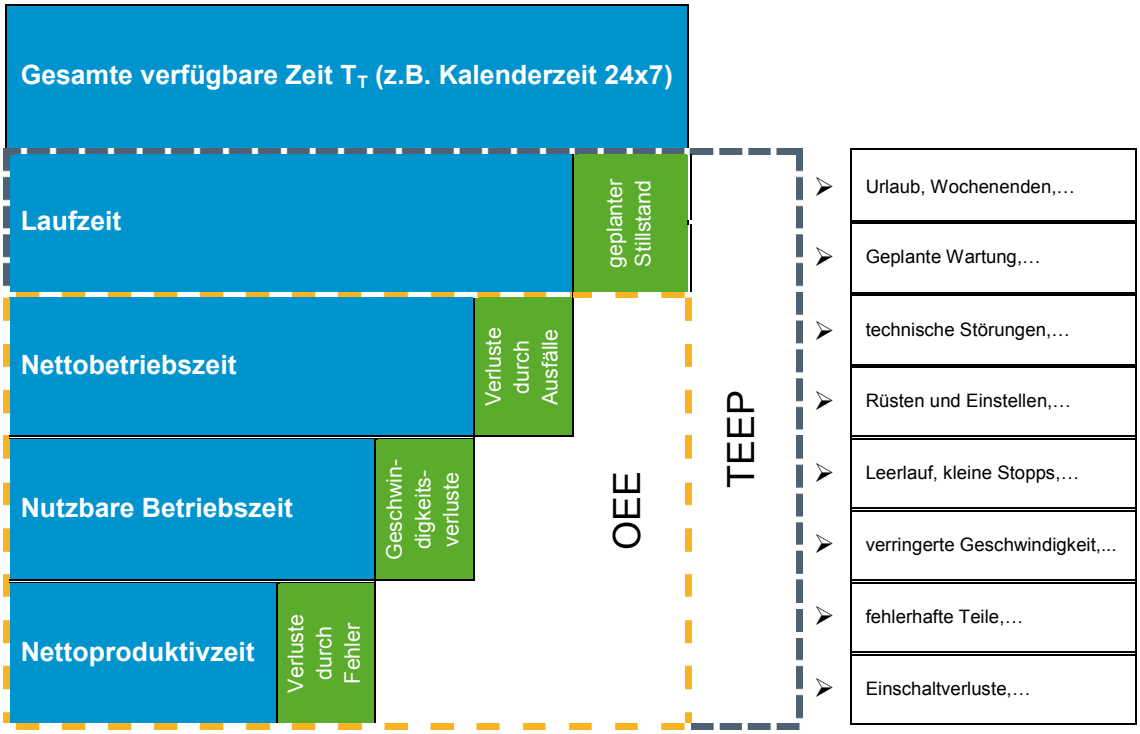


Abbildung 4: Verlustzeiten - OEE vs. TEEP [21]

Bei einem schlechten Wert für die TEEP kann es passieren, dass die benötigte Kapazität des Belastungsprofils nicht mehr durch die verfügbare Kapazität der Hüllkurve gedeckt werden kann. In einem solchen Fall sind Abstimmungs- und Anpassungsmaßnahmen nötig.

2.1.1 Bedarfsermittlung

Bevor Anpassungsmaßnahmen vorgenommen werden können, ist es nötig den Kapazitätsbedarf festzustellen. Die Kapazitätsbedarfsermittlung ist ein Teil der Produktionsbedarfsplanung. Das Ziel ist die Ermittlung der mittelfristig nötigen

[19] Vgl. Biedermann (2008), S. 42.
[20] Vgl. Jodlbauer (2008), S. 136.
[21] Quelle: in Anlehung an Nakajima (1988) zitiert nach Muchiri/Pintelon (2008), S. 3522 & Lanza et al. (2013), S. 32.

Ressourcen für die Realisierung eines Produktionsprogramms. Aufgrund des begrenzten Kapazitätsangebotes ergibt sich die einfache Notwendigkeit, den Bedarf durch die Einlastung von Aufträgen zu ermitteln. Für eine Planungsperiode müssen die terminierten Arbeitsgänge vorliegen. Aus diesen wird der Kapazitätsbedarf festgestellt, indem die Stückzeiten mit den Stückzahlen multipliziert werden. Mit dieser Bearbeitungszeit belastet ein Arbeitsgang eine Maschine in der Planungsperiode. Der Arbeitsgang ist nicht immer zwingend die Grundlage für die Ermittlung, es können auch Kapazitätsprofile oder Grobarbeitsplätze sein. Das Ergebnis der Kapazitätsbedarfsermittlung ist der Kapazitätsplan.[22]

Für die Bestimmung der Kapazitätsbelastung, müssen vorab die Stückzahlen ermittelt werden, die tatsächlich zu produzieren sind. Diese Bedarfsermittlung kann auf drei verschiedene Arten erfolgen. Deterministische Bedarfe sind bereits bekannt und ergeben sich auf Grund des Produktionsprogramms durch eine Stücklistenauflösung. Stochastische Daten werden mithilfe statistischer Verfahren ermittelt. Eine Einsatzsituation für stochastische Bedarfsermittlung liegt vor, wenn die Lieferzeiten kleiner als die Durchlaufzeiten sind. Im Rahmen der Grobplanung können bereits eingegangene Bestellungen, also deterministische Bedarfe, auf Grund von Erfahrungswerten hochgerechnet werden. Dadurch ergibt sich eine Kombination aus deterministischer und heuristischer Bedarfsermittlung. Die heuristische Form beruht auf Erfahrungswerten und wird meist nur bei Teilen mit geringem Wert angewandt.[23]

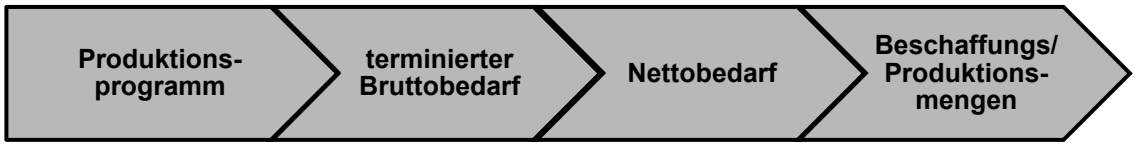


Abbildung 5: Schritte der Bedarfsermittlung[24]

Eine weitere Facette bei der Bedarfsermittlung ist die Bestandssituation im Unternehmen. Bereits vorhandene Bestände müssen nicht nochmals produziert werden und belasten daher keine Anlagen. Zusätzlich sind die Produktions- und Beschaffungszeiten in der Überlegung zu berücksichtigen, da diese den Zeitpunkt der Bedarfsrealisierung und daher die Belastung einer Anlage beeinflussen. Als letzter Aspekt muss die Bedarfsart und die Erzeugnisstruktur bekannt sein. Die zwei wichtigsten Bedarfsarten sind der Primärbedarf und der Sekundärbedarf. Der Primärbedarf ergibt sich aus dem Hauptproduktionsprogramm und ist jener der zum Verkauf bestimmt ist. Der Sekundärbedarf ergibt sich auf Grund der Erzeugnisstruktur durch eine Stücklistenauflösung. Er umfasst alle Rohstoffe, Einzelteile und Baugruppen die zur Herstellung des Primärbedarfs notwendig sind.[25] Somit kann die Bedarfsermittlung in vier Schritten abgeschlossen werden, welche in Abbildung 5 dargestellt werden.

Als erster Schritt muss ein Produktionsprogramm ermittelt werden, um daraus den Primärbedarf abzuleiten. In der nächsten Phase muss der terminierte Bruttobedarf eines

[22] Vgl. Schuh (2006), S. 47f.
[23] Vgl. Arnold et al. (2008), S. 329.
[24] Quelle: Eigene Darstellung
[25] Vgl. Zsifkovits (2013), S. 140.

Produktes ermittelt werden. Dieser ergibt sich aus dem Sekundärbedarf und einem Primärbedarf auf Grund der Notwendigkeit von Ersatzteilen. Der Zusatzbedarf stellt eine weitere Komponente dar und soll eventuellen Ausschuss abdecken. Mit Primärbedarf, Sekundärbedarf und Zusatzbedarf ergibt sich der terminierte Bruttobedarf eines Produktes in einer Periode. In weiterer Folge wird vom terminierten Bruttobedarf der disponible Lagerbestand subtrahiert. Dieser ergibt sich aus dem vorhandenen Bestand des Produktes und den noch ausstehenden Bestellung, verringert um einen reservierten Bestand und dem Sicherheitsbestand. Am Ende dieses Schrittes liegt der Nettobedarf vor, der jedoch nur dann größer 0 ist, wenn der Bruttobedarf größer als der disponible Lagerbestand ist. Als letzter Schritt müssen die Zeitrestriktionen der Produktion berücksichtigt werden. Dies geschieht indem der Nettobedarf um die Vorlaufzeit, welche die geplante Durchlaufzeit oder Beschaffungs- und Produktionszeit ist, vorgezogen wird. Damit ergeben sich die periodenbezogenen Beschaffungs- und Produktionsmengen. Vorlaufzeiten unterliegen jedoch einer gewissen Unsicherheit, da sie zum Planungszeitpunkt nicht immer bekannt sind. Es ergeben sich Probleme in der Planung die mit Hilfe robuster Ansätze kontrolliert werden können.[26]

Mit Hilfe der Bedarfe und der Arbeitspläne kann für die einzelnen Anlagen und Aggregate, unter Berücksichtigung der Bearbeitungszeiten, der Kapazitätsbedarf bestimmt werden.

[26] Vgl. Günther/Tempelmeier (2012), S. 187ff.

2.1.2 Kapazitive Abstimmung

Kapazitive Abstimmungsmaßnahmen werden nötig, wenn der Kapazitätsbedarf und das Kapazitätsangebot nicht mehr übereinstimmen. Es kann neben der Erhöhung des Angebotes auch nötig sein es zu reduzieren, da Überkapazitäten oft zu unnötigen Leerkosten führen.[27] Tabelle 1 gibt einen Überblick über die Abstimmungsmaßnahmen.

Tabelle 1: Kapazitätsausgleichsmaßnahmen[28]

Kapazitätsabstimmung					
		Kapazitätsanpassung		Belastungs-anpassung	Belastungsabgleich	
Reaktionszeit		Anpassung Arbeitskräfte	Anpassung Betriebsmittel		Zeitlicher Ausgleich	Technolog. Ausgleich
	kurz	Überstunden aufbau/-abbau Innerbetrieblicher Austausch von Arbeitskräften			Aufteilen der Lose Vorziehen/ Aufschieben von Aufträgen oder Einzelbedarfen	Ausweichen auf andere Betriebsmittel
	mittel	Zusätzliche Schicht Kurzarbeit	Wiedernutzung/ Stilllegung von Anlagen	Auftragsfremd-vergabe Fremdauftrags-annahme		
	lang	Einstellung/ Entlassung Personal	Beschaffen/ Abstoßen von Anlagen			

Bei der Kapazitätsanpassung wird die Kapazität an den Bedarf angepasst. Die meisten Möglichkeiten sind mittel- oder langfristig. Ein Austausch von Mitarbeitern zwischen den Bereichen ist nur bei ausreichender Qualifikation möglich. Belastungsanpassungen wählen den gegengesetzten Fall und passen den Bedarf der Kapazität an. Dies erfolgt durch die Vergabe von Aufträgen an externe Zulieferer oder die Annahme von Aufträgen von außen, um Leerkosten zu vermeiden. Der Belastungsausgleich erfolgt ausschließlich kurzfristig. Beim zeitlichen Ausgleich wird versucht die Belastung auf den Anlagen durch die Teilung und/oder Verschiebung von Aufträgen an die vorhandene Kapazität anzupassen. Beim technologischen Ausgleich werden Aufträge durch andere Anlagen gefertigt, sofern diese Kapazitäten frei haben und technologisch dazu im Stande sind.[29]

Die Anpassungsmaßnahmen können je nach ihrer Durchführbarkeit in kurz- mittel- und langfristige Maßnahmen unterteilt werden. GUTENBERG[30] stellt die Betriebsmittel in den Mittelpunkt der Anpassungsformen und unterteilt diese in zeitliche, quantitative und intensitätsmäßige Anpassungen.

[27] Vgl. Schuh (2006), S. 48.
[28] Quelle: in Anlehnung an Arnold et al. (2008), S. 332.
[29] Vgl. Arnold et al. (2008), S. 332.
[30] Vgl. Gutenberg (1983)

Zeitliche Anpassung

In diesen Fällen wird die Arbeitszeit variiert. Die Maschine verfügt in der Regelarbeitszeit $(0 \leq t \leq T)$ über eine maximale Kapazität $\bar{x}$. Es ergibt sich für die Produktionskosten die Kostenfunktion 2.1.

$$K(x) = K_F + \sum_{i=1}^{n} a_i \cdot x \cdot c_i \tag{2.1}$$

Die Fixkosten K_F enthalten die Löhne der Arbeitnehmer und die Abschreibungen von Gebäuden und Maschinen. Die variablen Kosten ergeben sich aus der mit den Preisen c_i bewerteten Faktoreinsatzfunktion. Diese berechnet sich aus dem Produktionskoeffizienten a_i, für die Menge des Werkstoffes i die je Produktionseinheit eingesetzt wird, und der Produktionsmenge x. Abbildung 6 zeigt den Kostenverlauf der zeitlichen Anpassung.

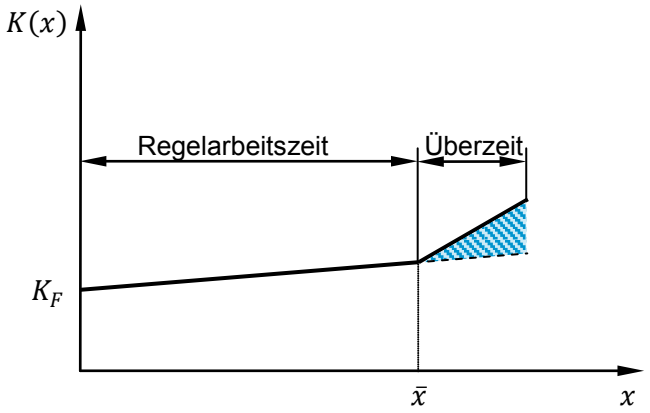


Abbildung 6: Zeitliche Anpassung[31]

Sobald die Regelarbeitszeit überschritten wird und Überstunden anfallen kommen zu den Kosten Überstundenzuschläge hinzu. Dieser Fall ist auf der Abszisse ab $\bar{x}$ zu sehen. Der Verlauf steigt durch der Überstundenzuschläge stärker an. Die blau schraffierte Fläche entspricht den Mehrkosten für die zeitliche Anpassung.[32]

Quantitative Anpassung

Im Fall der quantitativen Anpassung wird die Anzahl der eingesetzten Maschinen der geforderten Produktionsmenge angepasst. Es wird davon ausgegangen, dass es mehrere Maschinen des gleichen Typs mit einer ähnlichen Maximalkapazität $\bar{x}$ gibt. Somit würde die Produktionsmenge um Vielfache von $\bar{x}$ variieren, wenn man Maschinen zusätzlich in Betrieb nimmt oder stilllegt. Es handelt sich hierbei um keine Investitions- oder Deinvestitionsentscheidung, da sich die Anzahl der Maschinen im Betrieb nicht verändert. Es empfiehlt sich die quantitative Anpassung mit der zeitlichen Anpassung zu kombinieren, da wie in Abbildung 7 zu sehen ist, durch die Hinzunahme einer weiteren Maschine und die Anhebung der Produktionsmenge um ein weiteres Vielfaches von $\bar{x}$ sich sprungfixe Kosten k_f ergeben. Diese sprungfixen Kosten unterliegen zusätzlich einer

[31] Quelle: in Anlehnung an Steven (2012), S. 77.
[32] Vgl. ebd., S. 76f.

Remanenz. Für die Produktionskosten ergibt sich die Kostenfunktion (**2.2**, wobei z ganze Vielfache der Maximalkapazität sind. Diese Bedingung wird durch (**2.3** gewährleistet.[33]

$$K(x) = K_F + \sum_{i=1}^{n} a_i \cdot x \cdot c_i + z \cdot k_f \tag{2.2}$$

$$\forall x \neq z \cdot \bar{x} \qquad z = \left\lfloor \frac{x}{\bar{x}} \right\rfloor + 1 \tag{2.3}$$

Die Kostenfunktion 2.2 stimmt bis auf den Term, welcher die sprungfixen Kosten berechnet, mit jener von 2.1 überein. Die Vielfachen der Maximalkapazität werden in 2.3 berechnet, wobei immer auf ganzzahlige Werte abgerundet wird.

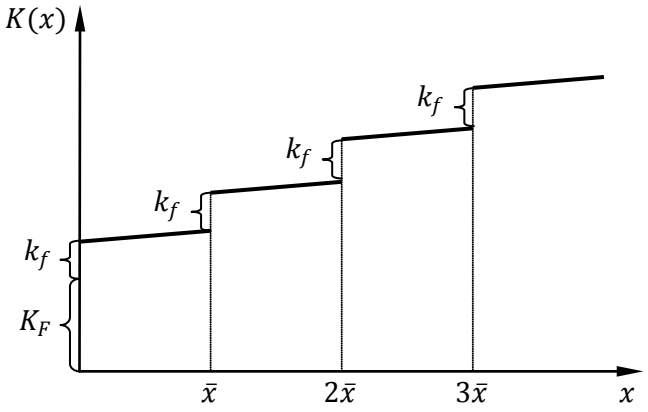


Abbildung 7: Zeitlich-quantitative Anpassung[34]

Der Punkt x^* an dem eine zusätzliche Maschine hinzu genommen werden soll, anstatt mit Überstunden die geforderte Produktionsmenge zu erreichen, kann mittels einer Break-Even-Analyse bestimmt werden. Abbildung 8 bereitet diese graphisch auf.

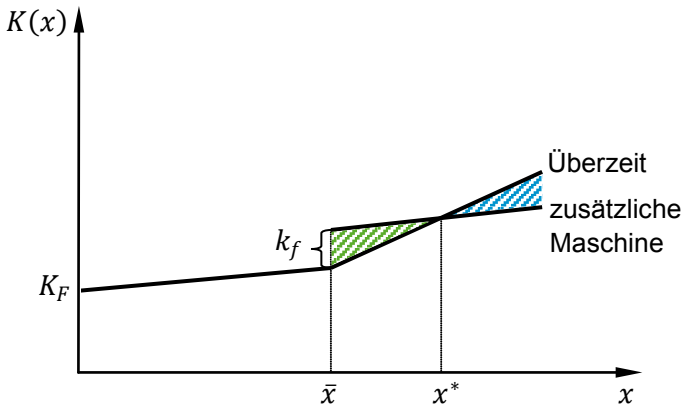


Abbildung 8: Zeitliche vs. quantitative Anpassung[35]

[33] Vgl. Steven (2012), S. 77f.

[34] Quelle: in Anlehnung an Berndt/Cansier (2007), S. 101.

[35] Quelle: in Anlehnung an Steven (2012), S. 79.

Die grün schraffierte Fläche stellt die anfänglichen Mehrkosten der quantitativen Anpassung gegenüber der zeitlichen Anpassung dar. Ab dem Punkt x^* überwiegen die Mehrkosten der zeitlichen Anpassung, welche der blau schraffierten Fläche entsprechen, gegenüber der quantitativen Anpassung.[36]

Intensitätsmäßige Anpassung

Die intensitätsmäßige Anpassung wird durch eine Variation der Produktionsgeschwindigkeit d realisiert. Die Produktionsgeschwindigkeit ist die Produktionsmenge oder die Leistung je Zeiteinheit. Durch die Variation einer technischen Größe, die die Produktionsmenge beeinflusst, wird diese Anpassungsform in die Praxis umgesetzt. Sie muss innerhalb der technischen Möglichkeiten der Anlage zwischen der Untergrenze d_{min} und der Obergrenze d_{max} erfolgen. Die Produktionskoeffizienten hängen von der Produktionsgeschwindigkeit ab. Der Zusammenhang wird durch die s.g. Verbrauchsfunktion in Abbildung 9 verdeutlicht.

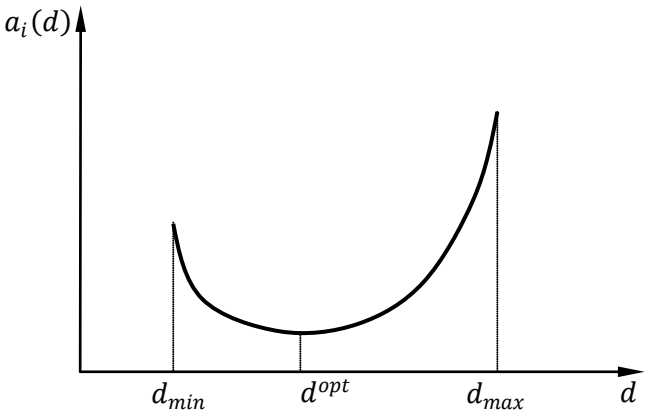


Abbildung 9: Verbrauchsfunktion[37]

Es gibt einen idealen Wert d^{opt} der Produktionsgeschwindigkeit bei dem der Wert des Verbrauchs des Einsatzfaktors i je Produkteinheit minimal ist. Bei Geschwindigkeiten unterhalb d^{opt} läuft die Maschine im unwirtschaftlichen Bereich was zu einem erhöhten Bedarf an Werkstoff i führt. Sobald die Maschine mit einer Geschwindigkeit über d^{opt} gefahren wird, liegt eine Überbeanspruchung vor. Die folgenden Gleichungen beschreiben die Gesamtkosten 2.4, den Faktoreinsatz 2.5 und die variablen Stückkosten 2.6.[38]

$$K(x) = K_F + \sum_{i=1}^{n} a_i(d) \cdot x \cdot c_i \tag{2.4}$$

$$i = 1 \dots n \qquad r_i = a_i(d) \cdot x \tag{2.5}$$

[36] Vgl. Steven (2012), S. 78f.
[37] Quelle: in Anlehnung ebd., S. 80.
[38] Vgl. Steven (2012), S. 80f.

$$k_v(d) = \frac{K(x) - K_F}{x} = \sum_{i=1}^{n} a_i(d) \cdot c_i \tag{2.6}$$

Der graphische Verlauf der Gesamtkosten 2.4 ist in Abbildung 10 zu sehen. Diese setzen sich aus den Fixkosten K_F und dem mit Faktorpreisen c_i bewerteten Faktoreinsatz zusammen. Dieser ergibt sich laut 2.5 aus dem von der Produktionsgeschwindigkeit abhängigen Produktionskoeffizienten $a_i(d)$ und der Stückzahl x. Die variablen Stückkosten 2.6 haben einen analogen Verlauf zur Verbrauchsfunktion aus Abbildung 9 und nehmen ihr Minimum ebenfalls bei d^{opt} an. Sie ergeben sich aus der Summe der mit Faktorpreisen bewerteten Produktionskoeffizienten.[39]

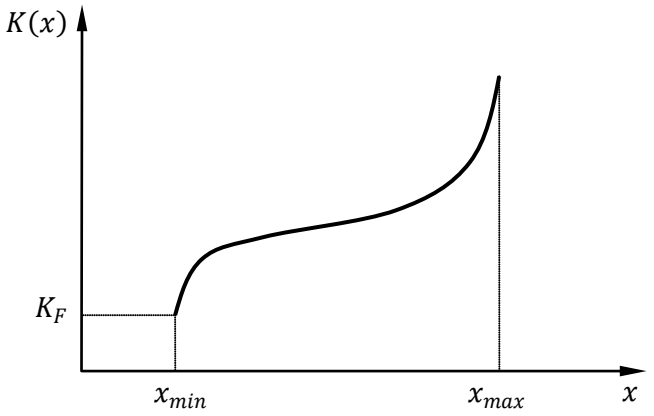


Abbildung 10: Gesamtkosten intensitätsmäßige Anpassung[40]

Der ungewünschte Effekt der nicht-linearen Kosten unterhalb von d^{opt} lässt sich vermeiden, indem man die intensitätsmäßige Anpassung mit der zeitlichen Anpassung kombiniert. Die Anlage wird mit der Produktionsgeschwindigkeit d^{opt} gefahren und die Produktionszeit wird so weit reduziert, dass die Produktionsmenge noch erreicht wird. Dadurch bleiben die Stückkosten bis d^{opt} auf dessen Niveau konstant und die Gesamtkosten werden, wie in Abbildung 11 zu sehen ist, linearisiert. Erst oberhalb von d^{opt} ist eine zeitliche Anpassung nicht mehr möglich und es kommt zum verstärkten Anstieg der Gesamt- und Stückkosten.[41]

[39] Vgl. Steven (2012), S. 80ff.

[40] Quelle: in Anlehnung an Steven (2012), S. 82.

[41] Vgl. Steven (2012), S. 82f.

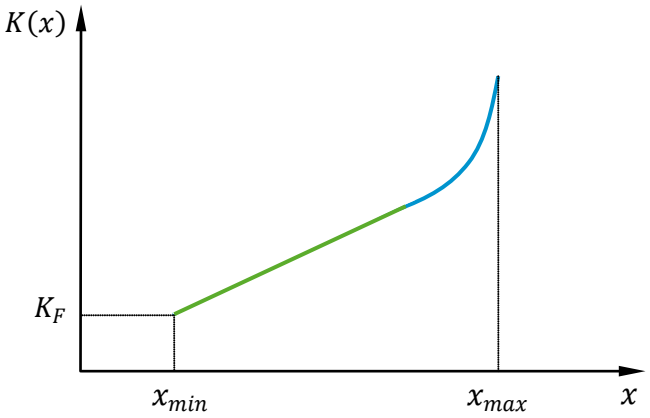


Abbildung 11: Gesamtkosten zeitlich-intensitätsmäßige Anpassung[42]

Diese Form der Anpassung ist für die meisten Industriezweige typisch. In einigen Fällen, wie im Hochofenprozess der Stahlindustrie, ist die Produktionsunterbrechung der zeitlichen Anpassung nicht möglich. Eine Anpassung der Produktionsmenge ist nur intensitätsmäßig möglich. Der ungünstige nicht-lineare Verlauf der Gesamtkosten lässt sich jedoch durch Intensitätssplitting vermeiden. Dabei werden zwei Produktionsmengen und folglich zwei Geschwindigkeiten gewählt die ungefähr zu einem ähnlichen Verlauf wie der grünen Gerade in Abbildung 11 führen. Diese beiden Geschwindigkeiten werden so kombiniert, dass immer die gewünschte Menge erzeugt wird ohne die Maschine anhalten zu müssen. Dadurch wird der Verlauf in diesem Bereich linear und die Stückkosten konstant. Für alle Anpassungsformen gilt jedoch, dass sie dem Erzeugungsprozess angepasst und meist miteinander kombiniert werden müssen, um die Kosten für die Produktion zu minimieren.[43]

2.1.3 Kapazitätsplanungsverfahren

Die Möglichkeiten zur Kapazitätsplanung richten sich nach den Unternehmenszielen in den Punkten der Flexibilität der Auftragsfristigkeit und der Flexibilität der quantitativen Kapazität. Daraus ergeben sich zwei Hauptplanungsparadigmen: Die Planung in die unbegrenzte Kapazität und in die begrenzte Kapazität. Diese beiden Konzepte können auch für die Grobplanung verallgemeinert werden. Im Falle der unbegrenzten Kapazität wird die Belastung entlang der Zeitachse der Fristigkeit berechnet ohne die Kapazität der Arbeitsplätze zu berücksichtigen, daher die Bezeichnung der unbegrenzten Kapazität. Im Vordergrund steht eine termingerechte Einhaltung der Auftragsfälligkeiten. Eine möglichst gute oder konstante Auslastung steht im Hintergrund. Ein typisches Einsatzgebiet ist die Werkstattfertigung oder Kundenauftragsproduktion.[44]

Bei der Grobplanung werden die Belastungsprofile mit einem Endtermin versehen und nach einer gewünschten Priorität eingeplant. Es ergibt sich ein Gesamtressourcenprofil für einen Arbeitsplatz. Sollte es zu einer Überlast kommen muss der Planer entsprechend des zeitlichen Umfangs der Planung reagieren. Ist dieser Umfang mittelfristig so dient die Erkenntnis aus dieser Planung oft der Entscheidung über die Annahme oder die

[42] Quelle: in Anlehnung an Steven (2012), S. 82.
[43] Vgl. Steven (2012), S. 82f.
[44] Vgl. Schönsleben (2007), S. 275f.

Ablehnung eines Auftrages. Der Planer kann, falls der Lieferzeitpunkt es zulässt, ggf. den Endtermin des Auftrages verschieben.[45]

Im Falle der begrenzten Kapazität wird eine Überlast nicht erlaubt. Die Kapazität soll möglichst gut ausgenutzt werden. Dazu wird eine Änderung des Start- und/oder Endtermins akzeptiert. Das Einsatzgebiet dieses Planungsparadigmas sind Industriezweige in denen die begrenzte Kapazität als gegebenes Hauptproblem akzeptiert werden muss. Typisch hierfür ist die Prozessindustrie.[46]

Die Grobplanung muss davon ausgehen, dass die Fälligkeitstermine flexibel sind. Es werden immer die kumulierten Kapazitäten und Vorbelastungen betrachtet. Ein kumuliertes Ressourcenprofil wird dem jeweils anderen periodentreu aufgesetzt. Der sich ergebende, früheste Starttermin ist jene Periode nach der noch zusätzliche Kapazitäten zur Verfügung stehen. Analog dazu ist der früheste Endtermin das Ende jener Periode an der die Kapazität, die Vorbelastung und die neue Belastung bleibend übersteigt, also auf Dauer zusätzliche Kapazität zu Verfügung steht. Da es sich um eine Grobplanung handelt, können der Ausgleich lokaler Über- oder Unterlast der späteren Arbeitssteuerung überlassen werden.[47]

[45] Vgl. Schönsleben (2007), S. 722ff.

[46] Vgl. ebd., S. 276

[47] Vgl. ebd., S. 724ff.

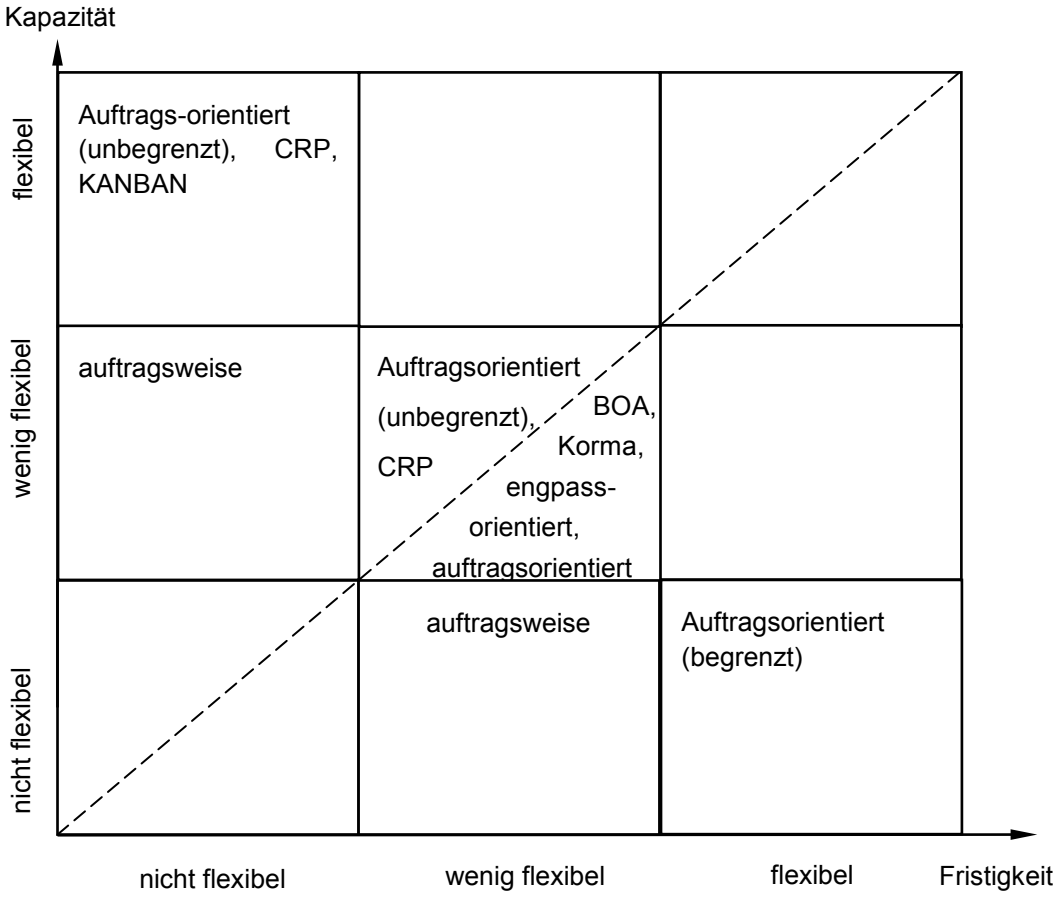


Abbildung 12: Kapazitätsplanungsmethoden[48]

Generell kann gesagt werden, dass die Planung in die unbegrenzte Kapazität bei flexiblen Kapazitäten und unflexiblen Auftragsfälligkeitsterminen angewandt wird. Die Planung in die begrenzte Kapazität wird hingegen bei unflexiblen Kapazitäten und gezwungener maßen flexiblen Auftragsfälligkeitsterminen eingesetzt. Je nach Kombination aus diesen beiden Flexibilitätsbereichen sind diverse Kapazitätsplanungsmethoden besser oder weniger gut geeignet. Abbildung 12 gibt einen Überblick über diese Methoden, wobei jene oberhalb der unterbrochenen Linie die der unbegrenzten Kapazität und jene unterhalb die Möglichkeiten der begrenzten Kapazität aufzeigen. Ein spezieller Bereich in der Abbildung ist der Sektor links unten, wo keine Flexibilität in keiner der beiden Dimensionen existiert. Ein Kapazitätsausgleich in welcher Form auch immer ist nicht möglich. Das Planungsproblem wäre schwer, bis nicht lösbar. Umgekehrt ist die Situation in den drei Sektoren oben rechts. Die Flexibilität ist in beiden Dimensionen so groß, dass jeder Auftrag angenommen werden kann. Meist wird diese Kapazität teuer durch Überkapazitäten mittels Auslagerungen oder Überstunden erkauft. Die beiden Sektoren welche für eine auftragsweise Planung stehen, müssen auf Grund ihrer gänzlichen Inflexibilität in wenigstens einer Dimension, Auftrag für Auftrag einzeln eingeplant werden. Wegen der extremen Zeitintensivität der Planung wird eine große Wertschöpfung bei diesen Aufträgen vorausgesetzt. Entlang der Diagonale von rechts unten nach links

[48] Vgl. Schönsleben (2007), S. 689.

oben gibt es auf Grund der ausreichenden Kapazität in wenigstens einer der Dimensionen mehrere Planungsverfahren. Der Einsatz von Computeralgorithmen ist in diesem Fall sehr gut geeignet.[49] Für eine genauere Erläuterung dieser Methoden wird eine entsprechende Fachliteratur[50] empfohlen, da eine Beschreibung an dieser Stelle den Umfang der Arbeit sprengen würde.

Die Kapazitätsplanung kann auch zur taktischen Grobplanung verallgemeinerst werden. Das folgende Unterkapitel soll dieses Gebiet behandeln und mittels einer Beschreibung von Anwendungsbeispielen erläutern.

2.2 Taktische Grobplanung

Zeiträume und Fristen für die Planung sind nach Industriezweig und Unternehmen verschieden aber es haben sich gewissen Richtwerte allübergreifend etabliert. Wichtige Terminologien sind strategisch, taktisch und operativ. Strategische Zeithorizonte sind langfristig und umfassen einen Planungszeitraum von fünf, manchmal auch zehn Jahren. Mittelfristige Planungszeiträume werden auch unter dem Begriff taktisch gefunden und haben einen Umfang von einem halben Jahr bis zwei, seltener fünf Jahren. Für die operative oder kurzfristige Planung gilt, dass der Zeitraum wenigstens so lange sein muss wie die längste Auftragsdurchlaufzeit.[51]

Die Grobplanung wird herangezogen um eine Abschätzung über die Machbarkeit von Plänen zu erhalten ohne dabei zu sehr ins Detail gehen zu müssen.

Der wichtigste Planungsansatz auf dieser Ebene ist die aggregierte Produktionsplanung. Ziel ist es die Bedarfe in einem gegebenen, mittelfristigen, Zeitraum zu erfüllen. Daraus ergibt sich die Aufgabe die Produktionsrate und die Kapazitäten von Arbeitskräften und Anlagen zu ermitteln und zu verteilen. Die aggregierte Produktionsplanung spielt sich in einem Zielkonflikt zwischen Genauigkeit und Komplexität des Modells ab.[52]

2.2.1 Kapazitätsprofil

Das Kapazitätsbelastungsprofil zeigt den zeitlichen Verlauf des Kapazitätsangebotes und des Kapazitätsbedarfs. Über-, und Unterlasten sind in einer graphischen Darstellung sehr gut zu sehen.[53] Für die Planung, ob grob oder fein, stellt es ein wichtiges Hilfsmittelt dar.

Der Kapazitätsbedarf kann vom Primärbedarf abgeleitet werden. Dieser ergibt sich wiederum je nach Fertigungsart aus anderen Quellen. Im Falle der Massenfertigung dient die Nachfrageprognose als Quelle. In der Serienfertigung wird das Produktionsprogramm herangezogen und in der Einzelfertigung der Terminplan.[54]

Eine dauerhafte Überlast kann auf einen Engpass hindeuten. In der Grobplanung können Überkapazitäten oder Überlast auch auf eine mangelhafte Planung hinweisen. Eine

[49] Vgl. Schönsleben (2007), S. 276ff.
[50] Vgl. Gudehus (2010); Klaus Erlach (2010); Zsifkovits (2013)
[51] Vgl. Gudehus (2010), S. 210.
[52] Vgl. Bushuev (2013), S. 1050.
[53] Vgl. Dyckhoff (2006), S. 326.
[54] Vgl. Schneeweiß (2002), S. 192.

Möglichkeit dies zu beheben ist ein alternativer Ansatz bei der Ermittlung der Bedarfe. Durch eine Verschiebung von Produkten in andere Produktfamilien ist eine Glättung des Kapazitätsgrobprofils möglich. Sollten vorab Engpässe erkennbar sein können Aufträge entweder abgelehnt, oder eine Optimierung an der Engpassstelle durchgeführt werden. Durch den Planungshorizont von sechs bis vierundzwanzig Monaten besteht meist ausreichend Zeit diese Maßnahmen umzusetzen. Dazu zählen die Reduktion der Bearbeitungszeit je Teil, Verringerung der Ausfallzeiten und Rüstzeiten sowie Erhöhung der Einsatzzeiten durch Zusatzschichten oder durchgearbeitete Pausen.[55]

Abbildung 13 zeigt typische Kapazitätsprofile eines Arbeitsplatzes entlang der Zeitachse. Diese Betrachtungsform ist typisch in der Grobplanung von Arbeitsplätzen. Eine Alternative des Kapazitätsprofiles ist die Betrachtung entlang des Wertstromes. Abbildung 14 zeigt schematisch mögliche Ausprägungsformen sich ergebender Kapazitätsprofile.

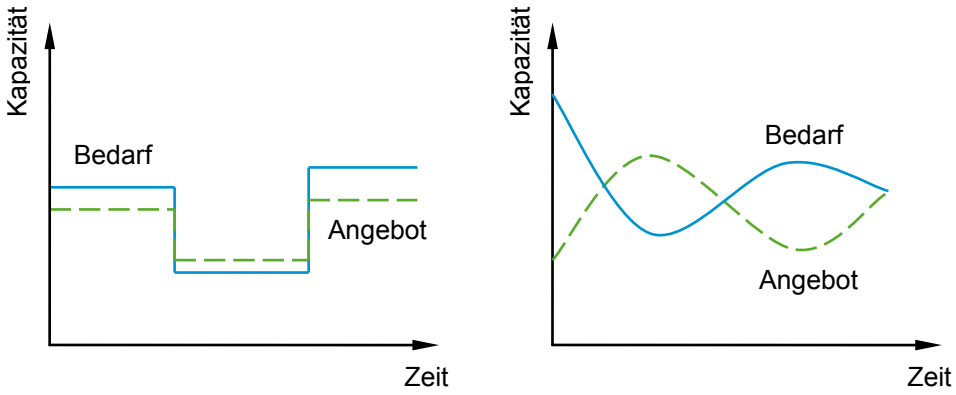


Abbildung 13: Kapazitätsprofile entlang der Zeit[56]

Das Kapazitätsangebot eines Prozesses wird entlang des Fertigungsflusses aneinandergereiht. Dadurch ergibt sich ein Kanal dessen Durchmesser das Kapazitätsangebot darstellt. Im Idealfall ist dieser Kanal an jeder Stelle gleich groß, was jedoch sehr unwahrscheinlich ist. Im Anwendungsfall wird dem s.g. Schrittmacherprozess die geringste Kapazität zur Verfügung gestellt. Dieser Fall ist in Abbildung 14 bei Punkt 1 zu sehen. Dadurch wird sicher gestellt, dass der Materialfluss nie abreißt indem die vorgelagerten Prozesse in der Lage sind genügend Material zu liefern und die nachgelagerten Prozesse die nötige Kapazität haben, um den eintreffenden Materialfluss schnell genug abzuarbeiten. Diese und weitere Methoden, wie zum Beispiel die Engpasssteuerung, werden meist in der operativen Auftragseinsteuerung verwendet.

[55] Vgl. Erlach (2010), S. 144f.

[56] Quelle: in Anlehnung an Schönsleben (2007), S. 30.

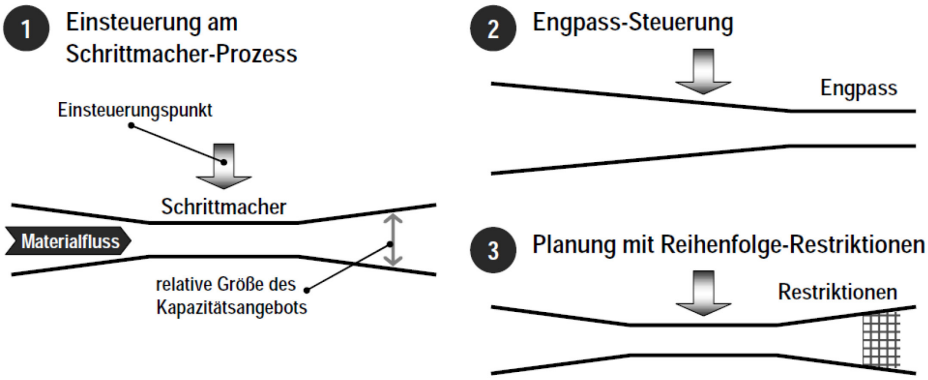


Abbildung 14: Kapazitätsprofile entlang des Wertstroms[57]

Im Falle von Punkt 3, der Reihenfolgebildung nach Restriktionen, sind der Schrittmacherprozess und der Engpass ident. Es wird anhand von diesem, unter Berücksichtigung der Reihenfolgerestriktionen von flussabwärts liegenden Prozesse, eingesteuert. [58]

2.2.2 Produktionsprogrammplanung

Die Produktionsprogrammplanung umfasst einen mittelfristigen Zeithorizont von einem Quartal bis zu einem Jahr. Die prinzipiellen Anforderungen werden in Abbildung 15 dargestellt. Das Ziel der Produktionsprogrammplanung ist die Maximierung des Deckungsbeitrages unter Berücksichtigung diverser Nebenbedingungen.

Tabelle 2: Grundmodell der Produktionsprogrammplanung[59]

Variable	Bedeutung
$\boldsymbol{\delta_{it}}$	Deckungsbeitrag von Produkt i in Periode t [GE].
$\boldsymbol{x_{it}}$	Produktionsmenge von Produkt i in Periode t [Stk].
$\boldsymbol{a_{ji}}$	Produktionskoeffizient für die Herstellung einer Einheit von Produkt i benötigte Kapazität der Maschine j.
$\boldsymbol{R_{jt}}$	Verfügbare Kapazität der Maschine j in Periode t.
$\boldsymbol{\underline{D}_{it}}$	Mindestproduktionsmenge des Produktes i in Periode t.
$\boldsymbol{D_{it}}$	Effektive Nachfrage nach Produkt i in Periode t.

Gleichung 2.7 formuliert die Maximierung des Deckungsbeitrages über alle Produkte und Perioden eines Betrachtungszeitraumes T. Die Einhaltung der durch die Maschinenkapazitäten gegeben Restriktionen wird durch Gleichung 2.8 sichergestellt.

[57] Quelle: Erlach (2010), S. 251.
[58] Vgl. Erlach (2010), S. 250ff.
[59] Vgl. Kistner/Steven (2001), S. 192.

Formel 2.9 ist die Absatzbedingung und 2.10 die Nichtnegativitätsbedingung für die Produktionsmengen.

$$\max \sum_{t=1}^{T} \sum_{i=1}^{I} \delta_{it} x_{it} \tag{2.7}$$

$$j = 1 \dots J;\ t = 1 \dots T \qquad \sum_{i=1}^{I} a_{ji} x_{it} \leq R_{jt} \tag{2.8}$$

$$i = 1 \dots I;\ t = 1 \dots T \qquad \underline{D}_{it} \leq x_{it} \leq D_{it} \tag{2.9}$$

$$i = 1 \dots I;\ t = 1 \dots T \qquad x_{it} \geq 0 \tag{2.10}$$

Dieses Modell ist die rudimentärste Darstellung und kann durch weitere Nebenbedingungen den Anforderungen im Unternehmen angepasst werden.[60]

Als Planungsvorgang auf taktischem Niveau erfolgt die Aggregation nicht nur auf der zeitlichen Ebene, sondern auch durch eine Zusammenfassung von Produkten zu Produktfamilien. In der taktischen Grobplanung dienen Produktfamilien zur Vereinfachung des Planungsprozesses. Man unterscheidet dabei Ablauf-, Fertigungs- und Teilefamilien. Ablauffamilien haben einen ähnlichen Wertstrom durch die Fertigung, Fertigungsfamilien sind in Einzelheiten ähnlich und eine Teilefamilie fasst Produkte mit gleichartigen Teilen zusammen. Gemein ist allen Familienarten, dass eine Bearbeitung verschiedener Produkte der Familie mit nur geringen Rüstkosten einhergeht.[61] Daher muss jedes Unternehmen individuell entscheiden auf Grund welcher produktionsrelevanter Ähnlichkeitskriterien die Produktfamilien gebildet werden. Die Zusammenfassung der Produkte zu Familien kann entweder bottum-up oder top-down erfolgen. Im ersten Fall werden aus einzeln vorliegenden Produkten die Familien gebildet. Im anderen Fall werden zuerst die Produktfamilien anhand derer Gruppierungskriterien modelliert und die Produkte danach zugeteilt.[62]

Neben der zeitlichen Gliederung des Produktionsprozesses umfasst die Produktionsprogrammplanung auch die mittelfristige Kapazitätsglättung. Es wird versucht den Kapazitätsbedarf eines längeren Zeitraums möglichst gleichmäßig auf das Kapazitätsangebot zu verteilen. Eine Art der Beeinflussung ist die Anpassung der Personalkapazität z.B. bei saisonal schwankenden Belastungen an verschiedenen Arbeitsstationen oder die Anwendung von Kapazitätsanpassungsmaßnahmen.[63]

[60] Vgl. Kistner/Steven (2001), S. 192f.
[61] Vgl. Wannenwetsch (2010), S. 565.
[62] Vgl. Erlach (2010), S. 38ff.
[63] Vgl. Schneeweiß (2002), S. 143f.

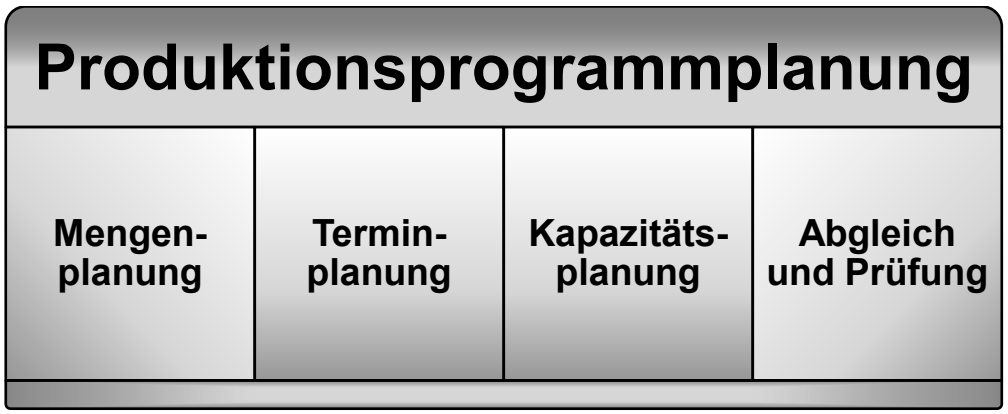


Abbildung 15: Bausteine der Produktionsprogrammplanung[64]

Neben der Mengenplanung, die aus den Primärbedarfen die Sekundärbedarfe ermittelt, beinhaltet die Produktionsprogrammplanung auch die Termin- und Kapazitätsplanung. Hier werden die zeitlichen Abläufe koordiniert. Als begrenzende Größe, sind die verfügbaren Kapazitäten von Personal und Anlagen zu berücksichtigen. Die Durchlaufterminierung bestimmt den Start- und Endtermin, diverse Pufferzeiten und den kritischen Pfad in der Fertigung. Die Fertigungsaufträge werden auf Grund ihrer Ressourcenverbräuche (Rohmaterial und Kapazitäten) auf ihre Durchführbarkeit hin überprüft. Diese Grobplanung wird in regelmäßigen Abständen durchgeführt. Aus dem Ergebnis folgt der Maschinenbelegungsplan. Dieses Vorgehen wird als Sukzessivplanung bezeichnet. Ein großer Nachteil dieser Planung ist ein Abstimmungsproblem zwischen Losgrößenbildung und Maschinenbelegungsplan. Um diesem Problem entgegen zu wirken werden alternative Kapazitätsplanungsverfahren verwendet, wie in Kapitel 2.1.3 beschrieben. Eine weitere Möglichkeit die Abläufe zu optimieren ist die Verkürzung von Durchlaufzeiten. Dazu zählen u.a. die Erstellung von Produktfamilien um die Umrüstvorgänge zu verringern und die Rüstzeiten zu minimieren.[65]

Einer der wichtigsten Schritte ist der Abgleich von Kapazitätsbedarf und Kapazitätsangebot. Der Kapazitätsbedarf ergibt sich durch die Ermittlung der Start- und Endtermine. Verschiedene Kapazitätsanpassungsverfahren wurden in Kapitel 2.1.2 bereits erörtert. Als Abschluss der Produktionsprogrammplanung soll im Sinne der taktischen Planung überprüft werden ob zu den Grobstartterminen alle nötigen Ressourcen zu Verfügung stehen. Dadurch können Engpässe frühzeitig ermittelt werden.[66]

2.2.3 Globale Belastungsfaktoren

Globale Belastungsfaktoren sind Schätzwerte für die Belastung, aus denen in Folge die zeitliche Entwicklung des Ressourcenbedarfs für ein Hauptproduktionsprogramm bestimmbar ist. Aus der Produktion einer Einheit eines Erzeugnisses in allen Produktionssegmenten oder Arbeitssystemen ergeben sich ein pauschaler Erfahrungswert für die Kapazitätsbelastung und deren Abschätzung. Die Vorgehensweise soll anhand eines Beispiels erklärt werden. Es wird angenommen, dass der Produktionsplan für die

[64] Quelle: Eigene Darstellung
[65] Vgl. Wannenwetsch (2010), S. 560ff.
[66] Vgl. ebd., S. 566f.

Enderzeugnisse fixiert ist und nun gemäß der Aufgabe der Kapazitätsgrobplanung überprüft werden soll, ob dieser Plan realisierbar ist. Mithilfe linearer Algebra ist eine Überprüfung sehr einfach. Im Beispielsystem gibt es $K = 2$ Produkte, $T = 6$ Perioden und $J = 3$ Produktionssegmente.[67] Ein Produktionssegment umfasst eine oder mehrere Maschinen bzw. Arbeitsplätze.[68] 2.11 zeigt die Matrix P mit der Dimension $K \times T$. Sie enthält die Produktionsmenge je Produkt und Periode.

$$P = \begin{bmatrix} 100 & 80 & 120 & 100 & 120 & 60 \\ 40 & 0 & 60 & 0 & 40 & 0 \end{bmatrix} \tag{2.11}$$

Matrix 2.12 zeigt die produktbezogenen Kapazitätsbelastungsfaktoren in Form der Matrix F mit der Dimension $J \times K$. Die Einträge sind globale Kapazitätsbelastungsfaktoren f_{jk}, welche die gesamte Belastung des Produktionssegments j kumulieren, die sich ergibt, wenn Erzeugnis k einschließlich seiner Vorprodukte produziert wird.

$$F = \begin{bmatrix} 0{,}4 & 1{,}6 \\ 0{,}6 & 2{,}4 \\ 2{,}0 & 2{,}0 \end{bmatrix} \tag{2.12}$$

Durch die Aggregation in der Phase der Grobplanung werden bei den Kapazitätsbelastungsfaktoren die Vorprodukte mit berücksichtigt. Es wird angenommen, dass die Kapazitätsbelastungsfaktoren periodenunabhängig sind und mit den Produktionsmengen linear steigen. In den meisten Fällen werden diese Faktoren aufgrund von Erfahrungswerten geschätzt.[69]

Die periodenbezogene Belastung B aus 2.13 ergibt sich durch die Matrizenmultiplikation von F und P und hat die Dimension $J \times T$. Jede Zeile stellt die zeitliche Entwicklung des Kapazitätsbedarfs eines Produktionssegments, gegeben durch das Produktionsprogramm, dar.

$$B = \begin{bmatrix} 104 & 32 & 144 & 40 & 112 & 24 \\ 156 & 48 & 216 & 60 & 168 & 36 \\ 280 & 160 & 360 & 200 & 320 & 120 \end{bmatrix} \tag{2.13}$$

In Abbildung 16 ist die Belastungsschwankung im zeitlichen Verlauf klar ersichtlich. Die Gültigkeit des Produktionsprogrammes kann überprüft werden, indem die Periodenkapazitäten der einzelnen Arbeitssysteme mit dem Bedarf verglichen werden. Sollten die Periodenkapazitäten bei 200 liegen, so kommt es bei Produktionssegment 3 häufig zu einer Überlast, während Produktionssegment 1 häufig unterlastet ist.[70]

Diese Methode liefert sehr einfach einen Überblick über die Entwicklung der Kapazitätsauslastung. Aufgrund dieser Werte können nun Maßnahmen zur Glättung der Schwankungen ergriffen werden. Es sind auch Kapazitätsanpassungsmaßnahmen, wie in Kapitel 0 beschrieben, denkbar.

[67] Vgl. Günther/Tempelmeier (2012), S. 164f.
[68] Vgl. Englberger/Herrmann (2013), S. 635.
[69] Vgl. Günther/Tempelmeier (2012), S. 165.
[70] Vgl. Günther/Tempelmeier (2012), S. 165f.

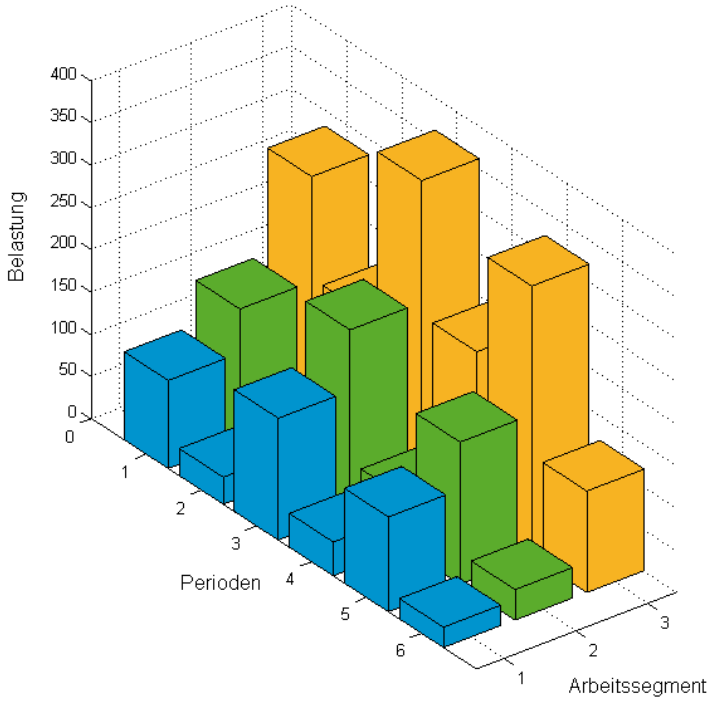


Abbildung 16: Belastungsentwicklung[71]

Die Bestimmung der Kapazitätsbelastungsfaktoren ist schwierig. In der Regel werden Schätzwerte aufgrund früherer Erfahrungen und der Annahme der Periodenunabhängigkeit verwendet. Diese Annahme stellt sich bei genauerer Betrachtung als falsch heraus. Die Angabe periodenspezifischer Faktoren ist nur begrenzt möglich, da sie starken Schwankungen unterliegen. Sie sind einer großen Unsicherheit ausgesetzt und können nur mithilfe verschiedener Szenarien modelliert werden. Es kommen Verfahren der robusten Planung zum Einsatz. Diese werden in Kapitel 3 beschrieben. Mit Hilfe von Simulationen über die verschiedenen Szenarien können repräsentativere Kapazitätsbelastungsfaktoren bestimmt werden.[72]

2.3 Fazit der Kapazitätsgrobplanung

Die Kapazitätsgrobplanung ist mit ihrem mittelfristigen Planungshorizont ein wichtiger Prozess in der Planung eines jeden Unternehmens. Durch die Abschätzung des Kapazitätsbedarfs und dem Vergleich mit dem Kapazitätsangebot können vorab Lieferausfälle verhindert werden. Der einfachste, wenn gleich wirtschaftlich ungünstigste, Fall ist es Aufträge vorweg abzulehnen. Die besseren Alternativen werden in Abbildung 17 gezeigt.

[71] Quelle: in Anlehnung an Günther/Tempelmeier (2012), S. 166.

[72] Vgl. Englberger/Herrmann (2013), S. 638.

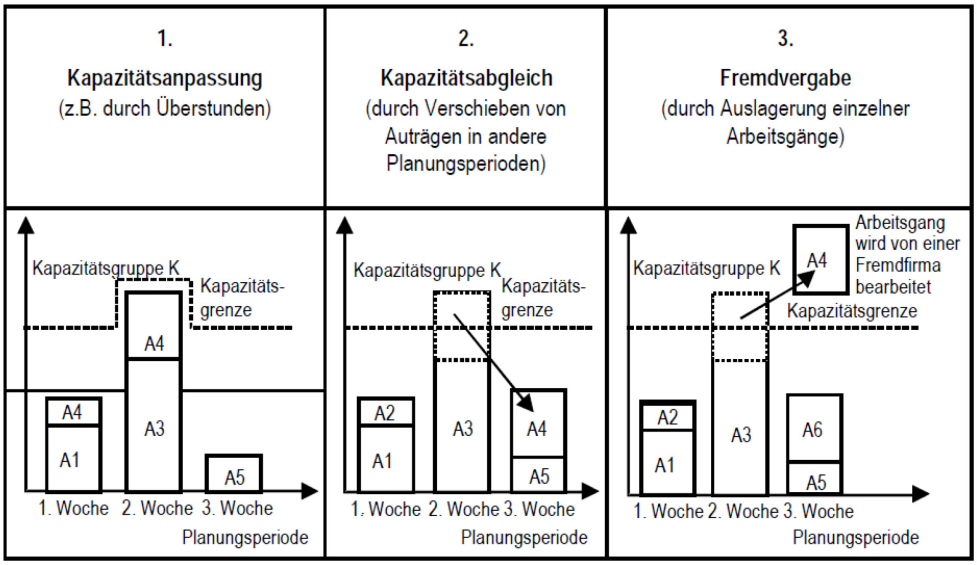


Abbildung 17: Kapazitätsplanungsalternativen[73]

Alternative 1 zeigt die Möglichkeit der Erweiterung des Kapazitätsangebotes, Möglichkeit 2 ist die Umplanung von Produktionsaufträgen und die letzte Wahl ist eine Drittvergabe an eine Fremdfirma. Alle diese Alternativen benötigen für eine Abschätzung der Machbarkeit, der Planung und der Umsetzung eine Vorlaufzeit. Sei es eine Einigung mit Arbeitnehmervertretern im ersten Fall, eine Umstrukturierung oder Neuverhandlungen mit dem Kunden im zweiten oder eine Angebotsstellung an die externe Firma beim dritten Punkt. Diese Vorlaufzeit ist mit dem Planungshorizont von mehreren Monaten durch die Grobplanung gegeben. Eines der wichtigsten Hilfsmittel für die taktische Grobplanung ist die Aggregation der zu planenden Einheiten. In den meisten Fällen handelt es sich, um die Zusammenfassung von Produkten zu Produktfamilien für welche die Summe der Kennzahlen auf ein repräsentatives Produkt hochgerechnet wird. Dieses bildet alle Produkte in der Familie in der Planung stellvertretend ab.[74]

Somit wird die Anwendung eines Profils erleichtert. Es zeigt den Verlauf des Kapazitätsangebotes und des Bedarfes entlang der Zeit oder des Wertstromes des Prozesses an. Es ist einfach eine Über- oder Unterlast festzustellen und einen Glättungsbedarf zu identifizieren.

Der größte Nachteil einer solchen Aggregation ist die Verzerrung der Daten und die dadurch entstehende Unsicherheit. Dieser Umstand macht es nötig die Grobplanung mit robusten Ansätzen zu kombinieren, um eine zuverlässige Aussage für einen längeren Planungshorizont von mehreren Monaten zu erhalten.

[73] Quelle: Schuh (2006), S. 49.
[74] Vgl. Neumann (1996), S. 196ff.

3 Robuste Planung

In einer eng vernetzten und globalisierten Wirtschaft werden Prognosen über Bedarfe laufend schwieriger. Einer der Gründe ist ein volatiler Absatzmarkt, der kürzere Durchlaufzeiten und eine hohe Flexibilität bei Bestellungen verlangt. Unter solchen Umständen ist eine mittel- bis langfristige Planung herausfordernd. Zukünftige Bedarfe erfahren Lieferanten meist in Form vager Prognosen. Innerhalb der Schwankungsbreite der Prognose können sich mit unterschiedlicher Eintrittswahrscheinlichkeit verschiedene Szenarien ergeben. Es gilt somit die entstehende Unsicherheit zu kompensieren.[75]

Die robuste Planung ist ein Ansatz, welcher versucht, für verschiedene Prognoseszenarien einen einheitlichen, auf alle Szenarien anwendbaren Plan zu erstellen.[76]

In diesem Kapitel werden die robuste Planung und die damit verbundenen Gebiete behandelt. Dazu gehören die Formen der Unsicherheit, die Arten der Robustheit und der Einsatz der robusten Planung an sich.

Diese Arbeit beschäftigt sich ausschließlich mit der Robustheit und nicht mit der Elastizität (engl. resilience). Meistens werden Robustheit und Elastizität unbewusst untereinander als Synonym verwendet. Das ist u.a. dem Fall zuschulden, da die Übersetzung des englischen Begriffes mit Widerstandsfähigkeit erfolgt, weshalb in dieser Arbeit von Elastizität gesprochen wird. Es wird versucht vorab eine begriffliche Abtrennung vorzunehmen.

Das Konzept der Elastizität ist in verschiedenen wissenschaftlichen Gebieten bekannt. In der Supply Chain versteht man darunter die Fähigkeit des Systems in den ursprünglichen Zustand oder einen besseren zurückzukehren, nachdem es gestört wurde. Die Begrifflichkeit der Elastizität wird verwendet, um eine Supply Chain auf ihre Reaktion auf schwerwiegende Störungen hin zu untersuchen.[77]

Robuste Supply Chains zeichnen sich aus, indem sie auf Störungen nicht durch eine Veränderung ihrer Leistung reagieren, bzw. die Veränderung einen vorab definierten Wert nicht überschreitet. Es kann argumentiert werden, dass eine robuste Supply Chain nur unter geringen Störungen noch einwandfrei reagiert, eine elastische auch bei groben Störungen. Dadurch ist eine elastische Supply Chain robust, eine robuste Supply Chain jedoch nicht immer elastisch.[78]

Abbildung 18 zeigt diese Definition, indem die Robustheit erweitert durch Agilität zur Elastizität führt. Zwei wichtige Aspekte der Robustheit, auf welche im Laufe dieser Arbeit eingegangen wird, sind in der Abbildung zu sehen, nämlich die Prognosen zukünftiger Ereignisse und die Reaktion auf Änderungen. Eine weitere Vertiefung dieses Ansatzes von WIELAND[79] würde jedoch den Rahmen dieser Arbeit sprengen.

[75] Vgl. Scholl (2001), S. 90f.
[76] Vgl. Gebhard/Kuhn (2007), S. 168.
[77] Vgl. Spiegler et al. (2012), S. 6163.
[78] Vgl. ebd., S. 6196.
[79] Vgl. Wieland/Wallenburg (2013).

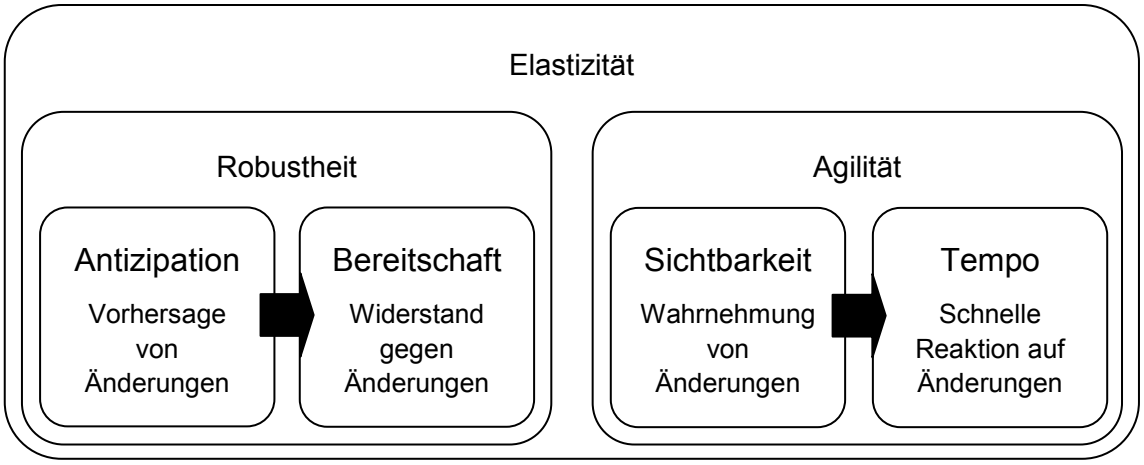


Abbildung 18: Elastizität vs. Robustheit[80]

In dieser Arbeit wird die Definition verwendet, dass ein System robust ist, falls es auf Änderungen in definierten Parametern mit akzeptablen Änderungen des Outputs reagiert. Es müssen für jedes System das Ausmaß der akzeptablen Änderung und die zu beobachtenden Parameter definiert werden. Bei dieser Definition ist die Elastizität eine Maßeinheit für die Robustheit des Systems. Eine Supply Chain kann so robust gestaltet werden, dass sie als elastisch gilt. Die Leistung, gemessen an definierten Parametern, kehrt nach einer Störung zu ihrer ursprünglichen Form zurück.[81]

3.1 Unsicherheit in der Planung

Um einen Produktionsprozess planen zu können, sind verschiedene Informationen nötig. Die Differenz aus den benötigten und den vorhandenen Informationen drückt sich als Unsicherheit aus.[82]

Unsicherheit ergibt sich aufgrund der ungenauen Kenntnis eines Systems, wobei das System aus den statischen Komponenten - seinen Daten - und deren dynamischen Interaktionen besteht. Die Daten können Preisentwicklungen bei Rohstoffen, Ausfälle von Anlagen und Mitarbeitern oder Bestellungen der Kunden sein. Politische- oder Umwelteinflüsse bedingen z.B. die Preisentwicklung und sind ein Beispiel für die dynamische Interaktion und der Interdependenzen der Daten untereinander, welche ein System ausmachen. Die Anzahl der Daten und die möglichen Beziehungen ergeben die Komplexität eines Systems. Ein weiterer Faktor für die Unsicherheit liegt in der Stärke der Schwankungen von Daten.[83] Diese wird versucht durch Prognosen vorherzusagen, was jedoch mit einem wachsenden Zeithorizont immer schwieriger wird und dadurch die Unsicherheit erhöht. Man spricht von einem Prognose- oder Szenariotrichter.[84]

Aufgrund dieser Ausführungen wird die Unsicherheit in dieser Arbeit als ein Produkt der Komplexität des Systems und des Betrachtungszeitraumes, in dem das System verwendet

[80] Quelle: Wieland/Wallenburg (2013), S. 304.
[81] Vgl. Spiegler et al. (2012), S. 6169.
[82] Vgl. Galbraith (1973) zitiert nach Mula et al. (2006), S. 271.
[83] Vgl. Gupta/Maranas (2003), S. 1220ff.
[84] Vgl. Vollmuth (2008), S. 438.

wird, definiert. Abbildung 19 zeigt mögliche Quellen für die Planungsunsicherheit entlang der Versorgungskette.

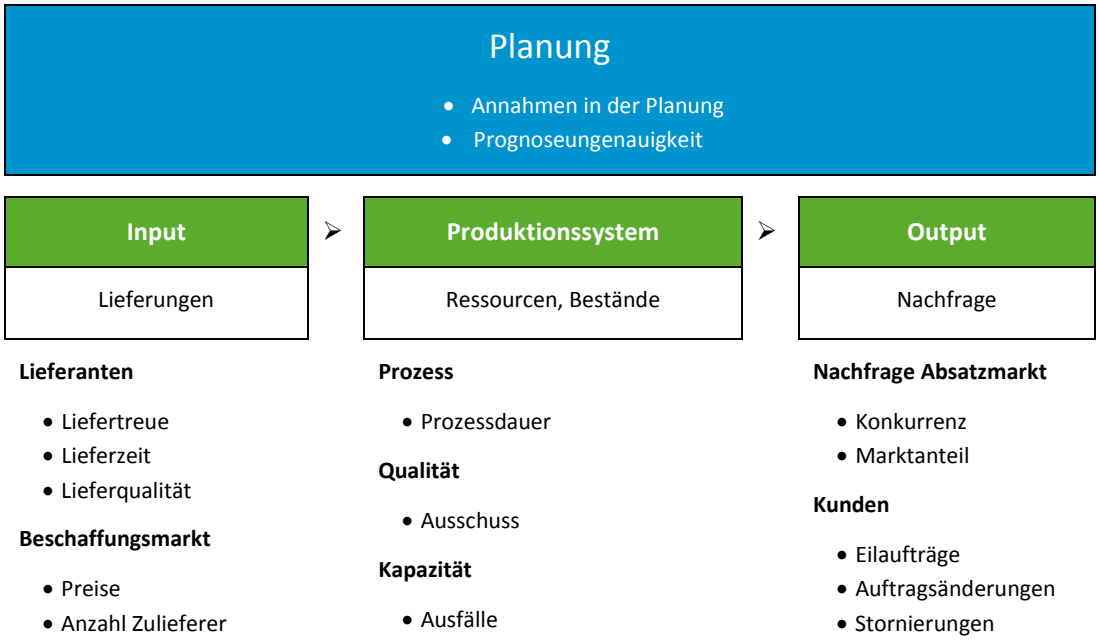


Abbildung 19: Quellen für Planungsunsicherheit [85]

Unsicherheit wird auf verschiedene Art und Weise untergliedert. HO[86] trifft zwei Unterscheidungen: die Unsicherheit durch Einflussfaktoren der Umwelt und jene durch Einflussfaktoren des Systems. Die Umweltfaktoren sind nur begrenzt bis gar nicht durch das Unternehmen zu beeinflussen.

Wissensstand über die Informationen

Es handelt sich hier um eine fundamentale Unterscheidung und beruht auf der Kenntnis über die Eintrittswahrscheinlichkeiten von Szenarien. Sind die Wahrscheinlichkeiten bekannt, spricht man von Risiko, sind sie unbekannt spricht man von Ungewissheit.[87]

Zeitliche Betrachtung

Kurzfristige Unsicherheit: Diese entsteht im s.g. Tagesgeschäft durch stornierte Bestellungen oder durch Vorreihung von Bestellungen aufgrund einer Änderung in der Priorisierung. Eine weitere Quelle bilden Anlagen durch deren Ausfall oder das Personal durch Erkrankungen.[88]

Langfristige Unsicherheit: Sie entsteht durch Preisfluktuationen bei Rohmaterialien oder dem Endprodukt. Saisonelle Schwankungen im Bedarf oder im Produktionsoutput unterliegen ebenfalls einem langfristigen Betrachtungszeitraum.[89]

[85] Quelle: in Anlehnung an Gebhard (2009), S. 22.
[86] Vgl. C.-J. Ho (1989)
[87] Vgl. Scholl (2001), S. 43.
[88] Vgl. Gupta/Maranas (2003), S. 1220.
[89] Vgl. Gupta/Maranas (2003), S. 1220.

Ort der Entstehung

Die Begrifflichkeiten können variieren. In dieser Arbeit wird von extern und intern bedingter Unsicherheit gesprochen. HO spricht von Unsicherheit auf Grund der Einflussfaktoren der Umwelt und jener durch Einflussfaktoren des Systems.[90]

Extern bedingte Unsicherheit: Diese Art der Unsicherheit ist auf Situationen zurück zu führen, die außerhalb des Einflussbereiches des Unternehmens liegen. Somit sollte die Versorgungs- und Abnahmeseite der Versorgungskette betrachtet werden. Die Lieferunsicherheit entsteht durch Abweichungen in der Liefertreue und Lieferzeit, was zu Bestandsschwankungen führen kann. Starke Verzögerungen können sich auf die Liefertreue des eigenen Unternehmens auswirken und den Marktanteil negativ beeinflussen, wodurch die Abnahmeseite zu betrachten ist. Die Nachfrageunsicherheit ergibt sich durch eine schlecht prognostizierbare Abnahmemenge und einen unsicheren Abnahmezeitpunkt. Sollten Prognosen diese beiden Faktoren sehr schlecht vorhersagen, führt dies entweder zu einer Unterversorgung der Nachfrage oder zu einem Überlaufen des Bestandes.[91]

Intern bedingte Unsicherheit: Sie entstehen u.a. durch den Planungsvorgang innerhalb des Unternehmens. Bei der hierarchischen Planung werden Daten abhängig von der Planungsebene aggregiert. Durch jede Aggregation kommt es zu einem Informationsverlust, welcher einer der zentralen Faktoren für Unsicherheit ist. Primäres Problem ist die Aggregation der Zeit, was wiederum zur kurzfristig bedingten Unsicherheit führt.[92] Weiters sind der Produktionsausstoß, die Qualität und die Bearbeitungs- und Prozesszeiten ein wichtiger Faktor für interne Unsicherheit.[93] Die letzten Faktoren können ebenfalls den langfristig bedingten Unsicherheitsquellen zugerechnet werden.

3.1.1 Auswirkungen von Unsicherheit

Sollte man Unsicherheit nicht beachten und versuchen zu kompensieren, so kann es dazu führen, dass man die Bedarfe des Marktes nicht deckt. Hier unterscheidet man den bereits geltend gemachten und nicht geltend gemachten Bedarf. Ersterer sind Bestellungen, welche bei nicht Erfüllung zum Verlust des Kunden und eventuellen Strafzahlungen sowie einem Schaden der Reputation führen kann. Nicht geltend gemachte Bedarfe sind das Marktpotential, welches nicht ausgeschöpft werden kann und der Konkurrenz am Markt überlassen wird. Beide Formen führen am Ende zum Verlust von Marktanteilen und folglich zu Umsatz- und Gewinneinbußen[94].

Peitschenschlageffekt

In der Literatur meist unter Bullwhip Effekt zu finden, ist der Peitschenschlageffekt eine direkte Ursache von Nervosität und folglich von Unsicherheit. Der Effekt beschreibt das

[90] Vgl. C.-J. Ho (1989) zitiert nach Mula et al. (2006), S. 271.
[91] Vgl. Gebhard (2009), S. 22f
[92] Vgl. Gebhard (2009), S. 23f.
[93] Vgl. Mele et al. (2007), S. 723.
[94] Vgl. Gupta/Maranas (2003), S. 1220f.

Phänomen, dass sich kleine Entscheidungen in einer N-Tier Supply Chain stromauf- oder abwärts, zeitlich versetzt, in der Auswirkung auf die Supply Chain verstärken.[95]

Es werden vier Gründe für diesen Effekt unterschieden. Als erster sei die Verarbeitung von Nachfragesignalen genannt. Vorgelagerte Ebenen in der Supply Chain haben einen höheren Bedarf als ihre Abnehmer. Dieser Effekt kommt zu Stande, da jeder Teilnehmer entlang der Supply Chain Prognosen anstellt, die nur den eigenen Tier 1 Verbraucher berücksichtigen. Zusätzlich zur Prognose wird je nach Unternehmen ein Sicherheitszuschlag zur prognostizierten Menge addiert. Da der jeweilige Tier 1 Verbraucher seinen gemeldeten Bedarf auf ähnliche Art bestimmt schaukelt sich, durch den Prognosen inhärenten Fehler und den Sicherheitszuschlägen, die Überproduktion von jedem Lieferanten auf. Ein ähnlicher Effekt tritt ein wenn die Fehler und folglich die Überproduktion realisiert werden. Durch den angestrebten Lagerabbau und der Unterproduktion kann es zu Fehlmengen kommen. Ein weiterer Grund ist die Bündelung von Bestellungen. Die Bestellstrategie von Teilnehmern in der Supply Chain ist nicht aufeinander abgestimmt. Während einige eine Bestellrhythmusstrategie anwenden, setzen andere eine Bestellpunktstrategie ein. Bei der Bestellrhythmusstrategie besteht zusätzlich die Möglichkeit, dass die Zeitintervalle nicht aufeinander abgestimmt sind.[96]

Der dritte Grund des Bullwhip Effektes ist der Engpasspoker, weitläufig als Hamsterkäufe bekannt. Im Falle eines Mangels oder Engpasses eines Produktes kommt es zu vermehrten Bestellungen, um sich gegen zukünftige Mängel zu rüsten. Dadurch erhöht sich der Bedarf der vorgelagerten Teilnehmer der Supply Chain aufgrund unzureichender Prognosen. Ein Engpass kann durch diese Überbestellungen jedoch noch stärker bzw. öfter auftreten, was den Effekt nochmals verstärkt. Als letzter Grund sind Preisschwankungen als Grund für den Bullwhip Effekt anzuführen. Dadurch sehen sich Teilnehmer der Supply Chain gezwungen größere Mengen zu einem günstigen Preis zu bestellen. Diese erhöhten Bestellmengen führen wiederum zu falschen Prognosen der vorgelagerten Supply Chain Mitglieder. Die Auswirkungen des Bullwhip Effekts sind u.a. stark fluktuierende Lagerbestände, schlechte Kapazitätsausnutzung und Lieferschwierigkeiten durch Fehlmengen. Weiters kann es zu Qualitätsproblemen kommen, hohen Material- und Versandkosten, längeren Durchlaufzeiten und Überstunden.[97]

Bei allen Versionen kann Unsicherheit als Hauptgrund für den Bullwhip Effekt angeführt werden. Durch das Informationsdefizit werden für das Unternehmen nachteilige Entscheidungen getroffen.

Planungsnervosität

Die Planungsnervosität kann als Resultat des Bullwhip Effekts gesehen werden. Sie tritt auf, wenn bereits getroffene Entscheidungen und fixierte Vorgaben im Laufe der Planung wiederholt verändert und angepasst werden.[98]

Besonders anfällig sind computergestützte Standardsoftwaretools wie MRP Systeme. Dem Unternehmen nicht angepasste Computersysteme reagieren sehr empfindlich auf

[95] Vgl. Kistner/Steven (2001), S. 333f.
[96] Vgl. Kaipia et al. (2006), S. 98f.
[97] Vgl. ebd., S. 100f.
[98] Vgl. Scholl (2001), S. 140.

Veränderungen in den Systemparametern. Algorithmen können in kürzester Zeit auf neue Gegebenheiten reagieren und immer wieder neue Pläne generieren, bereits getroffene und umgesetzte Maßnahmen abändern und folglich zur Nervosität in der Supply Chain beitragen. Neben der Verwendung verlässlicher Prognosen zur Verringerung der Nervosität können Zeithorizonte für die Planung erweitert, gefrorene Perioden eingeführt und Losgrößen reduziert werden.[99]

Als weitere mögliche Quelle soll die fehlende Abstimmung der Produktionsparameter angeführt werden. Losgrößen oder Produktionsbatches in einer Fabrik sind nicht auf jene einer anderen abgestimmt. Besonders kritisch ist dieses Faktum, wenn es sich um die gleiche Produktionsstufe des gleichen Produktes handelt, die jedoch an verschiedenen Standorten durchgeführt wird. Ein Grund für ein solches Vorgehen kann in einer besseren Kapazitätsauslastung der jeweiligen Fabrik liegen. Die Folgen für die Stabilität der gesamten Supply Chain können jedoch extrem sein und drücken sich durch Schwierigkeiten in der Zeitplanung anderer Supply Chain Teilnehmer, durch längere Transportzeiten oder durch schlechtere Verfügbarkeit von Komponenten aus.[100]

3.1.2 Reaktion auf Unsicherheit

Der Unsicherheit kann proaktiv oder reaktiv begegnet werden. Der proaktive Ansatz will die Auswirkungen der Unsicherheit schon vorab zu kompensieren. Dabei wird versucht bei bekannten Aktivitäten oder Perioden mit Unsicherheit ein gewisses Maß an Flexibilität in Form von Zusatzzeit einzuplanen. Das Ziel des reaktiven Ansatzes ist es Szenarien zu entwickeln, die den Störungen in einem Plan entgegen wirken. Die Planung wird neu durchgeführt, sobald die Störung eingetreten ist, was den Plan erheblich ändern kann. Diese Reaktion wird angewandt, wenn Störungen und deren Eintritt nicht mit ausreichender Gewissheit vorhergesehen werden können, also Ungewissheit vorliegt oder wenn die Auswirkungen zu massiv sind, um proaktiv Zusatzzeit einzuplanen[101].

Modellgestützter-/Simulationsansatz

In diesem Fall wird die Unsicherheit als Störung abgebildet, welche mit einem dynamischen System der Versorgungskette interagiert. Ein relativ neuer Ansatz ist, die Versorgungskette durch Softwareagenten abzubilden. Dabei wird jeder Teilnehmer in der Versorgungskette und jede Einflussmöglichkeit durch ein Programm, einem s.g. Agenten, dargestellt, welcher mit den anderen Programmen interagiert und somit die Dynamik des Systems darstellt. Durch Stellgrößen wird das Verhalten der Agenten beeinflusst, wodurch verschiedene Szenarien simuliert werden können. Somit wird das Verhalten der Versorgungskette abgebildet und ein Ergebnis kann ermittelt werden.[102]

Die Modellierung über s.g. Fuzzymengen ist ebenfalls relativ neu. Durch mehrere „wenn – dann“ Bedingungen wird das Planungsproblem abgebildet. Dadurch entsteht eine umfangreiche Menge an Regeln, die durch s.g. Zugehörigkeitsfunktionen miteinander interagieren. Somit soll die als Modell schwer erfassbare Realität des Planungsumfeldes

[99] Vgl. Kaipia et al. (2006), S. 100.
[100] Vgl. Volling et al. (2013), S. 107.
[101] Vgl. Hans et al. (2007), S. 564.
[102] Vgl. Mele et al. (2007), S. 724.

abstrahiert und abgebildet und folglich planbar gemacht werden. Die Unsicherheit ist einem solchen System inhärent, da die Regeln durch die Zugehörigkeitsfunktionen manchmal mehr oder weniger wahr sind.[103]

Je nach Einsatz kann dieser Ansatz proaktiv oder reaktiv sein. Versucht man vorab das System durch die Simulation so zu gestalten, dass es vor den meisten durch Unsicherheit verursachten Störungen gefeit ist, so ist der Ansatz proaktiv und dadurch robust. Wird die Simulation den äußeren Gegebenheiten immer wieder angepasst, um die beste Lösung zu ermitteln und die Planung entsprechend zu ändern, so ist der Einsatz reaktiv.

Mehrwertige Planungsansätze

Es werden alle erdenklichen Ausprägungen in die Planung übernommen. Unsicherheiten in der Nachfrage oder des Bestandes können durch verschiedene Szenarien und Verteilungen ausgedrückt werden. Dadurch ergeben sich stochastische Optimierungsmodelle, wie z.B. jene mit wahrscheinlichkeitsrelaxierten Nebenbedingungen. Diese s.g. Chance Constrained-Modelle erlauben, es einem Plan auch in einigen Fällen unzulässig zu sein, indem sie die strikte Einhaltung der Nebenbedingung nicht fordern, sondern vom Plan nur verlangen, dass er die Nebenbedingungen mit einer gewissen Wahrscheinlichkeit einhält. Es spielt keine Rolle, wie regelmäßig die Störungen auftreten - mehrwertige Planungsansätze können diese abbilden. Eine weitere Art der mehrwertigen Ansätze sind Kompensationsmodelle, die den reaktiven Umgang mit Unsicherheiten wählen. Nach der Verletzung der Nebenbedingungen durch eine veränderte Umweltlage werden vorher nicht festgelegte Kompensationsmaßnahmen vorgenommen, um den Plan wieder mathematisch korrekt zu gestalten. Die Kosten für diese Maßnahmen werden jedoch bei der Planerstellung antizipiert.[104]

Analytischer Ansatz

Es wird versucht, das Planungsproblem möglichst gut als lineares Programm abzubilden. Dabei kann es nötig sein, die diversen Erweiterungen eines linearen Programms, wie das ganzzahlige lineare Programm, einzusetzen. Nicht lineare Programme werden auf Grund ihres komplexen Lösungsverfahrens versucht durch robuste Ansätze zu vermeiden.[105]

Lineare Programme

Ein lineares Programm soll eine lineare Zielfunktion unter der Einhaltung von linearen Nebenbedingungen minimieren oder maximieren.

Die Zielfunktion 3.1 hat die Gestalt:

$$z = F(x_1, \dots, x_n) = \sum_{j=1}^{n} c_j x_j \tag{3.1}$$

[103] Vgl. Mula et al. (2006), S. 277f.
[104] Vgl. Gebhard (2009), S. 29f.
[105] Vgl. Mula et al. (2006), S. 274f.

Die Nebenbedingungen 3.2 sind entweder Gleichungen oder Ungleichungen der Form:

$$i = 1, \dots, m \qquad \sum_{j=1}^{n} a_{ij} x_j \{\leq \mid = \mid \geq\} b_i \qquad (3.2)$$

Die Variablen x_j müssen positiv sein und das gesamte Gleichungssystem linear. Eine Kombination von Variablen x_j, die die Nichtnegativitätsbedingung und die Nebenbedingung erfüllt, ist eine gültige Lösung des Gleichungssystems. Die Menge der gültigen Lösungen wird zulässiger Bereich genannt. Ziel ist eine Optimierung des Systems, indem eine Lösung gefunden wird, die die Zielfunktion minimiert oder maximiert.[106]

Ganzzahlige lineare Programme

Der Aufbau des Gleichungssystems entspricht jenem der linearen Programme mit der zusätzlichen Einschränkung, dass die Lösungen für x_j ganzzahlig sein müssen. Dadurch steigt der Lösungsaufwand. Der Speicherbedarf und die Laufzeit können exponentiell steigen. Meist reicht es aus, das System mit den Verfahren der linearen Programme zu lösen und die Lösungen auf-, bzw. abzurunden, solange die Nebenbedingungen eingehalten werden. Sollte der Anspruch auf Optimalität bestehen, so sind andere Lösungsverfahren, wie das Schnittebenenverfahren oder Branch-and-Bound, anzuwenden.[107]

Mit Hilfe der oben erwähnten Prinzipien können bekannte Planungsansätze durch die Berücksichtigung von Unsicherheit erweitert werden. Die hierarchische Produktionsplanung beachtet die Unsicherheit durch zusätzliche Parameter. Meist ist der Bedarf oder der Produktionsausstoß mit Unsicherheit behaftet. In Form der Kapazitätsplanung wird die Unsicherheit z.B. durch stochastische Programmierung und verschiedener Bedarfsszenarien abgebildet. In MRP II wird Unsicherheit z.B. durch ganzzahlige lineare Programme abgebildet, wobei von unsicherem Bedarf und unsicherer Durchlaufzeit ausgegangen wird.[108]

Um die Modelle bedarfsgerecht anzuwenden, ist es nötig zu wissen in welchem Bereich man Robustheit erlangen will. Dazu kann ein Plan anhand verschiedener Kriterien bewertet werden. Es obliegt dem Planer, bzw. dem Ziel des Unternehmens, auf welche Art und Weise Robustheit erlangt werden soll.

[106] Vgl. Koop/Moock (2008), S. 35ff.
[107] Vgl. ebd., S. 153.
[108] Vgl. Mula et al. (2006), S. 274ff.

3.2 Robustheitskriterien

Mit Hilfe von Robustheitskriterien kann ein Plan hinsichtlich seiner Robustheit beurteilt werden. Abhängig vom Planungsansatz kommen diese Kriterien stärker oder weniger stark zur Geltung. Abbildung 20 gibt einen Überblick über die Kriterien, welche im Folgenden beschrieben werden.

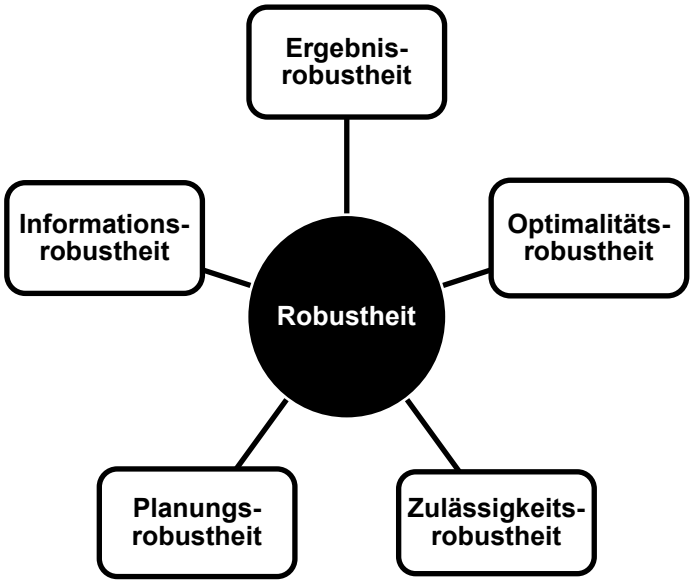


Abbildung 20: Robustheitskriterien[109]

Viele Quellen unterscheiden jedoch nur die Optimalitäts- und die Zulässigkeitsrobustheit. In dieser Arbeit wird noch zusätzlich ein Überblick über die Planungs-, Informations- und Ergebnisrobustheit gegeben. In den folgenden Abschnitten werden die Robustheitskriterien genauer beschrieben.

3.2.1 Ergebnisrobustheit

Im Zentrum der Betrachtung steht das zu erzielende Ergebnis eines Planes, wie etwa der Gewinn, der Umsatz oder der Nutzen. Die Ergebnisrobustheit versucht unter allen Szenarien s ein Ergebnis z zu erreichen, welches maximal ist. Dies führt in den meisten Fällen jedoch zu einem Zielkonflikt, da eine Maßnahme, welche in einem Szenario zu einem maximalen Ergebnis führt, in anderen Szenarien das Ergebnis schmälert. Es muss also ein Anspruchsniveau A definiert werden, welches vom erstellten Plan x für jedes Szenario s erreicht werden soll.[110]

$$s = 1 \dots S \qquad z_s(x) \geq A \tag{3.3}$$

Wird Gleichung 3.3 erfüllt, so ist der Plan hinsichtlich des Anspruchsniveaus total ergebnisrobust. Durch den Plan wird das vorgegebene Anspruchsniveau eines

[109] Quelle: in Anlehnung an Scholl (2001), S. 99f

[110] Vgl. Scholl (2001), S. 99f.

Maximierungsziels also erreicht oder überschritten. Im Falle eines Fixierungsziels[111], wenn ein Wert exakt erreicht werden soll, ist der Plan sogar total ergebnisstabil, da das Ergebnis vom betrachteten Szenario konstant ist.

Ein Plan x ist hinsichtlich des Anspruchsniveaus A relativ ergebnisrobust, wenn Abweichungen gering sind oder A weitgehen eingehalten wird. Die folgenden drei Gleichungen sollen diese etwas weit gefasste Definition konkretisieren.

$$W(Z(x) \geq A) \geq \alpha \tag{3.4}$$

$$\max_{s \in S} A - z_s(x) \leq \Delta_{max} \tag{3.5}$$

$$\sum_{s=1}^{S} p_s \cdot \max\{A - z_s(x), 0\} \leq V_{max} \tag{3.6}$$

Gleichung 3.4 legt fest, dass das Anspruchsniveau mit einer Wahrscheinlichkeit α nicht unterschritten werden darf. Dabei ist $Z(x)$ die Zufallsvariable für die Ergebnisse z_s eines Plans x. Gleichung 3.5 beschränkt die maximale Unterschreitung von A auf einen Grenzwert Δ_{max}. Gleichung 3.6 besagt, dass die erwartete Unterschreitung des Anspruchsniveaus nicht größer als V_{max} sein darf, wobei p_s die Eintrittswahrscheinlichkeit für Szenario s ist. Die Festlegung des Anspruchsniveaus ist, sofern systembedingt nicht vorgegeben, eine gewisse Herausforderung. Wird der Wert zu tief angesetzt werden, zu viele Pläne als ergebnisrobust eingestuft und wird er zu hoch angesetzt, erreicht vielleicht kein Plan diese Vorgabe. Die Auflösung des Problems kann durch die Anwendung verschiedener Anspruchsniveaus zur Beurteilung der Ergebnisrobustheit erfolgen oder durch die Vorgabe einer Wahrscheinlichkeit α. Die Pläne werden im letzeren Fall dahingehend beurteilt, welche Ergebnishöhe mit der Wahrscheinlichkeit erreichbar ist. Für die Planer besteht der Vorteil darin, dass eine solche Wahrscheinlichkeit eng mit der Risikoeinstellung verbunden ist und daher leichter festgelegt werden kann als das Anspruchsniveau. Risikoscheue Planer wählen einen Wert nahe eins, während risikofreudige einen niedrigeren verwenden. Die Ergebnisrobustheit kann mit allen Streuungsmaßen, wie der Varianz, der Spannweite oder dem Quartilsabstand beurteilt werden.[112]

[111] Vgl. Scholl (2001), S. 45.

[112] Vgl. Scholl (2001), S. 100f.

3.2.2 Optimalitätsrobustheit

Ein Plan x ist optimalitätsrobust, wenn seine Ergebnisse für jedes Szenario s nicht, bzw. möglichst gering vom szenariooptimalen Wert abweichen. Diese Abweichungen werden als Regret bezeichnet.

$$s = 1, \dots, S \qquad ar_s(x) = z_s^* - z_s(x) \tag{3.7}$$

$$s = 1, \dots, S \qquad rr_s(x) = 1 - \frac{z_s(x)}{z_s^*} \tag{3.8}$$

Gleichung 3.7 definiert den absoluten Regret $ar_s(x)$ für ein Szenario s mit der Differenz des szenariooptimalen Werts z_s^* und den Wert $z_s(x)$, welcher durch den Plan x erreicht wird. Gleichung 3.8 spezifiziert den relativen Regret $rr_s(x)$ für ein Szenario s als relative Abweichung des durch den Plan x erzielten Wertes $z_s(x)$ vom szenariooptimalen Wert z_s^*. Sind alle Regretwerte null so ist der Plan total optimalitätsrobust und perfekt. Dieser Fall tritt in der Regel nicht auf, weshalb nach einem relativ optimalitätsrobusten Plan gesucht wird. Dieser weicht nur geringfügig vom Optimum der Szenarien ab, bzw. Abweichungen sind unwahrscheinlich.[113] Ein solcher Plan soll die folgenden Eigenschaften erfüllen:

$$s = 1, \dots, S \qquad ar_s(x) \leq AR \tag{3.9}$$

$$s = 1, \dots, S \qquad rr_s(x) \leq RR \tag{3.10}$$

$$s = 1, \dots, S \qquad W(ar_s(x) > 0) \leq \beta \tag{3.11}$$

Die Gleichungen 3.9 und 3.10 beschränken die Abweichungen für alle Szenarien von ihren Szenarienoptima mit einem Maximalwert, jeweils für den absoluten Regret (AR) und den relativen Regret (RR). Die Werte sollten klein gewählt werden. Gleichung 3.11 besagt, dass Abweichungen nur mit einer kleinen Wahrscheinlichkeit β auftreten dürfen. Für den Fall, dass die Eintrittswahrscheinlichkeiten der Szenarien unbekannt sind, ist die Anzahl der Szenarien mit positiven Regretwerten zu beschränken. Die Festlegung der erlaubten Abweichung und der angemessenen Wahrscheinlichkeiten stellt eine Schwierigkeit dar. Relative Regretwerte von über 100% zeigen auf einem Blick unerwünschte Ereignisse oder Entwicklungen, wie einen Verlust an. Sollten sehr kleine szenariooptimale Werte vorliegen, steht einem kleinen absoluten Regret ein unproportional großer relativer Regret gegenüber. Hier müssen Skalierungsmaßnahmen ergriffen werden, um einen aussagekräftigen relativen Regret zu erhalten.[114]

Da in der Praxis der szenariooptimale Wert nur selten bekannt ist, wird der Regret oft anhand von Abschätzungen oder szenariospezifischer Anspruchsniveaus berechnet. Im Falle, dass ein einheitliches Anspruchsniveau für alle Szenarien gefunden, bzw. vorgegeben wird, liegt sogar Ergebnisrobustheit vor. Die Optimalitätsrobustheit betrachtet die Verteilung der Regretwerte und die relative Güte der Ergebnisse bezogen auf das Umfeld

[113] Vgl. Scholl (2001), S. 102.
[114] Vgl. Scholl (2001), S. 103.

der Szenarien, während die Ergebnisrobustheit die Verteilung der Ergebniswerte beurteilt und den Schwerpunkt auf absolute Ergebnishöhen legt.[115]

3.2.3 Zulässigkeitsrobustheit

Ein Plan ist zulässigkeitsrobust, wenn er für jedes Szenario gilt, bzw. die Handlungen wie geplant und mit nur geringen Modifikationen ausführbar ist. Von totaler Zulässigkeitsrobustheit wird gesprochen, wenn der Plan für alle erdenklichen Umweltlagen ohne unvorhergesehene Anpassungen durchgeführt werden kann. Man spricht hier von Stabilität. Es wird auch akzeptiert, wenn es bereits vorab definierte Teilpläne gibt, die je nach Szenario zum Einsatz kommen können.[116]

Relativ zulässigkeitsrobuste Pläne beinhalten Anpassungsmaßnahmen für die Flexibilität, welche bei der Planausführung mit nur geringer Wahrscheinlichkeit und geringem Maß eingesetzt werden müssen. Die Anpassungsmaßnahmen dürfen jedoch nicht zu einer maßgeblichen Verschlechterung des Ergebnisses führen. Wird dieser Aspekt bei der Berechnung der szenarioabhängigen Ergebnisse bereits einbezogen, so erfolgt eine Bewertung bereits bei der Ergebnis- und Optimalitätsrobustheit. Die Zulässigkeitsrobustheit ist somit nur bedingt eigenständig einsetzbar, um die Robustheit eines Planes zu bewerten. In der Realität gibt es kaum unzulässige Pläne, die nicht durch erheblichen Aufwand zulässig gemacht werden können. Überhaupt sollte erwähnt werden, dass die mathematischen Modelle aufgrund ihrer Restriktionen Pläne sehr schnell als unzulässig erklären. Darauf muss in der betrieblichen Praxis Rücksicht genommen werden. Die meisten unzulässigen Pläne können mit einem gewissen Aufwand zu zulässigen werden, wobei mit gewissen Ergebnisminderungen zu rechnen ist.[117]

Zwischen der Zulässigkeitsrobustheit und der Optimalitätsrobustheit gibt es einen der Natur der Kriterien innewohnenden Zielkonflikt. Systeme können nicht gleichzeitig beides sein. Je optimalitätsrobuster ein Plan ist, desto geringer ist die Zulässigkeitsrobustheit und vice versa. Dieser Zielkonflikt kann sich auch monetär niederschlagen. Wird eine fast totale Zulässigkeitsrobustheit angestrebt, um eine Erfüllung der Aufträge unter allen möglichen Szenarien zu gewährleisten, steigen die Kosten ab einem gewissen Punkt weit über mögliche Strafkosten für eine nicht Einhaltung von Lieferverpflichtungen. Ein Unternehmen muss daher einen Punkt finden, ab dem eine Nichterfüllung monetär akzeptabel ist. Es handelt sich dabei meist um äußerst unwahrscheinliche Szenarien.[118]

Eine Möglichkeit, um die Zulässigkeitsrobustheit zu testen, ist es Simulationsergebnisse mit Beobachtungen aus Experimenten zu vergleichen. Dabei wird die saisonale Entwicklung verschiedener simulierter Variablen beobachtet. Dieses Ergebnis wird mit weiteren aus einer Reihe von Experimenten verglichen. Hierbei wird eine große Bandbreite von Bedingungen, wie verschiedene Einsatzorte oder Managementmethoden, abgedeckt.[119]

[115] Vgl. Scholl (2001), S. 103f.
[116] Vgl. Scholl (2001), S. 104.
[117] Vgl. Scholl (2001), S. 104f.
[118] Vgl. Leung/Wu (2004), S. 511f.
[119] Vgl. Confalonieri et al. (2010), S. 960f.

3.2.4 Informationsrobustheit

Die Informationsrobustheit eines Plans bezieht sich auch auf Szenarien, die bei der Erstellung nicht eingeflossen sind. Sie besteht dann, wenn der Plan auch für jene Szenarien gilt, die nicht oder nicht ausreichend zum Planungszeitpunkt berücksichtigt wurden. Diese Szenarien waren zum Planungszeitpunkt jedoch bereits absehbar. Informationen werden aus zweierlei Gründen nicht in die Planung einbezogen. Bekannte Informationen kommen nicht vor, da sich dadurch ein Kostenvorteil bei der Planung durch eine Verringerung des Aufwandes ergibt. Die Ermittlung von beschaffbaren Informationen erfolgt nicht, da dies ebenfalls mit hohen Kosten verbunden sein kann.[120]

Ein Plan wird mittels Informationsstand A erstellt, wobei zum Planungszeitpunkt der bestmögliche Informationsstand B wäre, der jedoch nur mit maximalem Aufwand zur Informationsbeschaffung ermittelt werden hätte können. Ein Informationsstand ist eine Menge an Szenarien s und den dazugehörigen Eintrittswahrscheinlichkeiten p_s. Sollten konkrete Wahrscheinlichkeitswerte nicht einem jeden Szenario zuordenbar sein, so spricht man von partieller Information. Je unvollständiger die vorliegenden Wahrscheinlichkeitsinformationen sind, desto mehr Informationsstände können als möglich abgeleitet werden. Ein Plan gilt als informationsrobust, wenn er bei Informationsstand B als fast genauso ergebnis-, optimalitäts- und zulässigkeitsrobust eingestuft wird wie unter Informationsstand A. In der Betrachtung unterscheiden sich die Informationsstände nur in ihrer Eintrittswahrscheinlichkeit. Es gilt $p_s(A) = 0$ falls ein in B betrachtetes Szenario s nicht in A berücksichtigt wird. $p_s(B) = 0$ hingegen bedeutet, dass Szenario s bei genauerer Informationslage als unrealistisch betrachtet wird, selbst wenn es zuvor in A berücksichtigt wurde. Mit den bisherigen Robustheitskriterien ist es möglich die totale Informationsrobustheit zu definieren. Ist bei Informationsstand B die Wahrscheinlichkeit der Erreichung des vorgegebenen Anspruchsniveaus mindestens der maximale relative Regret und die Wahrscheinlichkeit für Unzulässigkeit höchstens so hoch wie bei Informationsstand A, so ist der Plan total informationsrobust. Die relative Informationsrobustheit ist gegeben, wenn der Robustheitsverlust bei verschiedenen Ausprägungen gering ist.[121]

Wie schon bei den vorhergehenden Robustheitskriterien ist es auch bei der Informationsrobustheit kritisch, das zentrale Bewertungsmerkmal, hier den Informationsstand, zu ermitteln. Denn der bestmögliche Informationsstand B ist meist nur dann bekannt, wenn er auch gleich als Planungsgrundlage dient. Der beste Informationsstand wird nur dann nicht verwendet, wenn der Aufwand zur vollständigen Bewertung zu groß wäre. Dann würde der verwendete Informationsstand A aus einer Teilmenge von Szenarien aus B gebildet. Die Planung würde mit A durchgeführt und anhand von B beurteilt. Implizit wird dabei auch die Qualität der Informationsaggregation festgestellt. In Fällen, wo B nicht bekannt ist, wird mit einem Schätzwert gearbeitet. Da in strategischen und teils auch taktischen Planungsprozessen aufgrund des Zeithorizonts nie die gesamte Umweltlage erfasst und prognostiziert werden kann, ist eine Unkenntnis von B nicht mit weiter von Nachteil. Die Informationsrobustheit beurteilt somit Pläne auf ihr

[120] Vgl. Scholl (2001), S. 105.
[121] Vgl. Scholl (2001), S. 105ff.

Verhalten hinlänglich unvorhergesehener Störungen. Bei großer Risikoscheu sollten auch unwahrscheinliche Szenarien in die Planung eingebunden werden, speziell jene, die bei Eintreten äußerst negative Folgen nach sich ziehen könnten.[122]

3.2.5 Planungsrobustheit

Die Planungsrobustheit ist ein wichtiges Kriterium in dynamischen Planungen, wie der rollierenden Planung. Dabei wird ein Plan zum Zeitpunkt t für die Periode t und den Folgeperioden bis $T-1$ aufgestellt. Alle Entscheidungen für Periode t und jene innerhalb des gefrorenen Horizonts bis zum nächsten Planungsschritt sind unveränderlich. Anpassungen können beim nächsten Planungsschritt für die folgenden Perioden vorgenommen werden, falls es der neue Informationsstand aufgrund der Umweltlage erfordert. Totale Planungsrobustheit liegt vor, wenn der Plan mit all seinen Teilplänen gültig ist und Entscheidungen aus Periode t für Periode $t+1$ gültig bleiben. Von relativer Planungsrobustheit wird gesprochen, wenn Änderungen an Teilplänen nur im geringen Umfang mit geringer Wahrscheinlichkeit nötig sind und diese nur kleine negative Auswirkungen auf das Ergebnis haben. Gründe für eventuelle Planänderungen können sich aus unvorhergesehenen Umweltentwicklungen ergeben oder aus anderen Berwertungsgrundlagen und neuen Informationen. Somit ist eine Anpassung direkt auf schlechte Prognosen und der daraus folgenden mangelnden Stabilität zurückzuführen. Die Bewertung der Auswirkungen der Planungsrobustheit hängt stark mit dem Zeithorizont zusammen, da Entscheidungen weit in der Zukunft kaum negativen Einfluss haben und nur zu Planänderungen und eventueller Planungsnervosität führen. Unmittelbare negative Kosteneinwirkungen sind zum Beispiel jene Zusatzkosten für verdorbene Produkte, die aufgrund nicht realisierter Produktion verderben, Konventionalstrafen bei Verstreichen des Liefertermins. Weiters der Imageverlust oder ein möglicher vermeidbarer Umrüstaufwand auf Anlagen, wenn unvorhergesehen auf andere Produktfamilien umgestellt werden muss, Kosten für Stornierungen, Expresslieferungen oder einer Eilproduktion mit den anfallenden Kosten für Überstunden. Viele dieser Fälle haben je nach Zeitfortschritt eine beschränkte Bindung und werden als partiell realisiert angesehen, wenn eine Abwendung eines Teils der negativen Folgen noch nicht zu spät ist, beispielsweise wenn noch ein Teil der verderblichen Produkte verwendbar oder bereits für den geplanten Einsatzzweck verdorbene Produkte alternativ verwertbar sind. Es zeigt sich hier, dass sich die Planungsnervosität von der Endproduktebene auf die Vorproduktebene verstärkt negativ durchschlagen kann.[123]

Die Planungsrobustheit kann auch im Nachhinein für bereits realisierte Entscheidungen bewertet werden. Strategische Entscheidungen, die nach Umsetzung nicht mehr geändert werden können und somit den Handlungsspielraum einschränken, sind hinsichtlich ihrer Planungsunsicherheit als negativ einzustufen, wenn sich z.B. eine Standortentscheidung oder ein Produktionsverfahren als ungeeignet herausstellen. Zeigt sich somit später, dass es nötig ist, die Entscheidungen und folglich die Auswirkungen aus diversen Gründen rückgängig zu machen, so ist die Planungsrobustheit auch im Nachhinein als negativ zu

[122] Vgl. Scholl (2001), S. 107.
[123] Vgl. Scholl (2001), S. 108f.

beurteilen. Es ist jedoch nicht immer eindeutig ermittelbar wodurch eine Rücknahme von Entscheidungen zu Stande kommt.[124]

Sollte der Planer mehrere Robustheitskriterien auswerten, steht er vor der Entscheidung die am besten geeignete Alternative, für die ihm vorliegende Situation, zu wählen. Um diese Entscheidung treffen zu können, hat er eine Vielzahl von Methoden zu seiner Verfügung.

3.3 Beurteilung und Wahl vom Modellen

Ein Entscheidungsträger muss Handlungsalternativen, welche die robuste Planung anhand von Szenarien zur Verfügung stellt, beurteilen und die für ihn Beste wählen. Dafür stellen die Methoden der Entscheidungstheorie verschiedene Ansätze zu Verfügung, um Präferenzwerte zu erhalten. Anhand dieser werden die Pläne der Handlungsalternativen sortiert und einer ausgewählt.

3.3.1 Entscheidung bei Risiko

Entscheidungen bei Risiko kennzeichnen sich dadurch, dass die Eintrittswahrscheinlichkeit p_s des Szenarios s bekannt ist. Die Ergebnismatrix in Abbildung 21 zeigt die Zielfunktionswerte $z(x,s)$ verschiedener Pläne x unter den Szenarien s. Die Eintrittswahrscheinlichkeit p_s wird für die Berechnung der Präferenzwerte durch die verschiedenen Kriterien benötigt.[125]

		Szenarien						
		s_1	s_2	⋯	⋯	⋯	⋯	s_S
Planalternativen	x_1	$z(x_1,s_1)$	$z(x_1,s_2)$	⋯	⋯	⋯	⋯	$z(x_1,s_S)$
	x_2	$z(x_2,s_1)$	$z(x_2,s_2)$	⋯	⋯	⋯	⋯	$z(x_2,s_S)$
	⋮	⋮	⋮	⋱	⋱	⋱	⋱	⋮
	⋮	⋮	⋮	⋱	⋱	⋱	⋱	⋮
	x_X	$z(x_X,s_1)$	$z(x_X,s_2)$	⋯	⋯	⋯	⋯	$z(x_X,s_S)$
		p_1	p_2	⋯	⋯	⋯	⋯	p_S
		Eintrittswahrscheinlichkeiten						

Abbildung 21: Ergebnismatrix bei Risiko [126]

[124] Vgl. Scholl (2001), S. 109f.
[125] Vgl. Scholl (2001), S. 43.
[126] Quelle: in Anlehnung an Zimmermann (2008), S. 16.

Erwartungswert-Kriterium

Des Erwartungswertkriterium 3.12 bestimmt jenen Plan x, welcher den größten Erwartungswert aufweist. Da wirtschaftliche Ereignisse meist nur einmalig auftreten, ist es möglich, dass das tatsächliche Ergebnis vom Erwartungswert stark abweicht.

$$\arg\max_{x \in X} \sum_{s=1}^{S} p_s \cdot z(x_x, s_s) \tag{3.12}$$

Weiters muss beachtet werden, dass die robuste Planung von risikoscheuen Entscheidungsträgern ausgeht. Risiken werden nur eingegangen, wenn der Gewinn groß genug ist. Der Gewinn wird weniger stark positiv bewertet, als der mögliche Verlust negativ beurteilt wird. Der Erwartungswert hingegen präferiert beide Möglichkeiten gleich stark. Somit wird ein risikoneutraler Entscheider unterstellt. Das Kriterium kann als relativ informationsrobust angesehen werden, da alle Szenarien beachtet werden und geringe Änderungen einzelner Szenarien somit weniger stark ins Gewicht fallen. Es werden alle Ergebnisse $z(x_x, s_s)$ berücksichtigt und die gewichtete Summe wird maximiert. Es kann somit von einer gewissen Ergebnisrobustheit gesprochen werden, da für alle Szenarien ein möglichst hohes Ergebnis erzielt werden soll. Abweichungen können nicht kontrolliert werden und somit ist keine Aussage über den Grad der Ergebnisrobustheit möglich. Für die robuste Planung ist das Kriterium eher ungeeignet, da es mögliche Risiken von Entscheidungen nicht berücksichtigt. Negative und positive Auswirkungen können sich in der gesamten Berechnung ausgleichen und somit bleiben die Risiken der negativen unberücksichtigt.[127]

Erwartungswert-Varianz-Kriterium

Dieses Kriterium bezieht neben dem Erwartungswert 3.13 auch die Varianz 3.14 zur Ermittlung des zu wählenden Plans ein. Wie in Gleichung 3.15 ersichtlich, muss der präferierte Wert maximiert werden. Mit dem Parameter λ ist es möglich, die Risikoaffinität des Entscheiders einzustellen. Bei einem negativen Wert ist der Entscheider als risikoscheu einzustufen, da die Abweichungen als Risiko angesehen werden. Bei einem positiven Wert hingegen ist der Entscheider risikofreudig und die Abweichungen werden als Chance betrachtet. [128]

$$\mu(x_x) = \sum_{s=1}^{S} p_s \cdot z(x_x, s_s) \tag{3.13}$$

$$\sigma^2(x_x) = \sum_{s=1}^{S} p_s \cdot \left(\mu(x_x) - z(x_x, s_s)\right)^2 \tag{3.14}$$

$$\arg\max_{x \in X} \mu(x_x) + \sigma^2(x_x) \cdot \lambda \tag{3.15}$$

[127] Vgl. Scholl (2001), S. 124ff.

[128] Vgl. Scholl (2001), S. 126.

Über die Ergebnis-, bzw. Optimalitätsrobustheit ist keine eindeutige Aussage möglich, da die große Problematik in der Wahl des Parameters λ liegt. Es kann zu irrationalen Entscheidungen kommen, weil durch eine entsprechende Wahl des Parameters klar dominante Pläne den dominierten benachteiligt werden. Für die Informationsrobustheit gilt ähnliches wie für das Erwartungswert-Kriterium. Da alle Alternativen und Szenarien betrachtet werden, gilt das Erwartungswert-Varianz-Kriterium als informationsrobust.[129]

Regret-Erwartungswert-Kriterium

Das Ziel besteht darin durch die Minimierung der Regretwerte optimalitätsrobuste Lösungen zu erhalten. Dabei kann der absolute Regret 3.7 oder der relative Regret 3.8 verwendet werden.

$$\arg\min_{x\in X} \sum_{s=1}^{S} p_s \cdot z_s^* - \sum_{s=1}^{S} p_s \cdot z(x_x, s_s) \tag{3.16}$$

$$\arg\min_{x\in X} 1 - \sum_{s=1}^{S} p_s \cdot \frac{z(x_x, s_s)}{z_s^*} \tag{3.17}$$

Beim absoluten Regret-Erwartungswert-Kriterium 3.16 werden die Erwartungswerte der absoluten Regretwerte minimiert. Beim relativen Regret-Erwartungswert-Kriterium 3.17 werden die Erwartungswerte der relativen Regretwerte minimiert. Dies erfolgt letztlich durch eine Maximierung der Erwartungswerte des Quotienten in 3.17. Die Grundhaltung des Entscheiders wird als risikoscheu angenommen. Da mit dem Regretwert eine Bewertungsgrundlage für die Optimalitätsrobustheit verwendet wird, ist dieses Kriterium gut geeignet, um optimalitätsrobuste Lösungen zu ermitteln. Die Lösungen sind auf Grund der Risikoscheu auch ergebnisrobuster als jene des Erwartungswert-Kriteriums. Die Informationsrobustheit ist ebenfalls positiv zu bewerten, da sich geringe Fehler in den Daten nur gering auf das Gesamtergebnis auswirken. Tendenziell werden Pläne präferiert, die in ungünstigen Fällen besonders gute Ergebnisse liefern und nicht nur jene die in allen Szenarien gute Ausgänge haben. Somit ist es sehr gut geeignet, um optimalitätsrobuste Lösungen bei geringer Risikoscheu, zu ermitteln.[130]

3.3.2 Entscheidung bei Ungewissheit

Entscheidungen unter Ungewissheit werden getroffen, wenn die Eintrittswahrscheinlichkeit p_s eines Szenarios s unbekannt ist. Die Abwesenheit der Eintrittswahrscheinlichkeiten kann aus einer unvollständigen Datenbasis resultieren oder auf Grund einer allgemeinen Unerfahrenheit des Entscheidungsträger in der Materie. Diese verunmöglicht es ihm, eine realistische Einschätzung der Wahrscheinlichkeiten abzugeben. Es ergibt sich eine alternative Ergebnismatrix, welche in Abbildung 22 zu sehen ist. Ohne die Eintrittswahrscheinlichkeiten ergehen sich andere Entscheidungskriterien.[131]

[129] Vgl. Gebhard (2009), S. 63.
[130] Vgl. Scholl (2001), S. 132f.
[131] Vgl. Laus et al. (2012), S. 49.

		Szenarien						
		s_1	s_2	⋯	⋯	⋯	⋯	s_S
Planalternativen	x_1	$z(x_1, s_1)$	$z(x_1, s_2)$	⋯	⋯	⋯	⋯	$z(x_1, s_S)$
	x_2	$z(x_2, s_1)$	$z(x_2, s_2)$	⋯	⋯	⋯	⋯	$z(x_2, s_S)$
	⋮	⋮	⋮	⋱	⋱	⋱	⋱	⋮
	⋮	⋮	⋮	⋱	⋱	⋱	⋱	⋮
	x_X	$z(x_X, s_1)$	$z(x_X, s_2)$	⋯	⋯	⋯	⋯	$z(x_X, s_S)$

Abbildung 22: Ergebnismatrix bei Ungewissheit [132]

Maximin-Kriterium

Für jede Planalternative wird der schlechteste Wert ermittelt und aus diesen Werten wird jene Alternative mit dem besten Ergebnis gewählt. Es wird somit für den schlechtesten Fall noch der beste Ausgang gewählt.

$$\arg\max_{x \in X} \min_{s \in S} z(x_x, s_s) \tag{3.18}$$

Das Maximin-Kriterium 3.18 geht von einer extrem pessimistischen Planungssituation oder einem sehr risikoscheuen Entscheidungsträger aus. Auf Grund dieser sehr einseitigen Betrachtung kann es zu sehr irrationalen Entscheidungen kommen, die den Gewinn stark verringern.[133]

Bewertet hinsichtlich der Eignung zum Einsatz in der robusten Planung muss festgehalten werden, dass ergebnisrobuste Lösungen nur bei einem sehr niedrigen Anspruchsniveau erreicht werden. Bei hoher Unsicherheit kommt es zu stark schwankenden szenariooptimalen Ergebnissen. Somit ist mit erheblichen Abweichungen von den szenariooptimalen Werten zu rechnen, wenn das Maximin-Kriterium angewandt wird. Daher ist dieses Kriterium als wenig optimalitätsrobust einzustufen. Informationsrobustheit besteht nur, wenn die pessimistische Planungssituation realistisch ist, also dem Informationsstand entspricht. Ansonsten gilt das Kriterium als wenig informationsrobust.[134]

Hurwicz-Kriterium

Im Falle des Hurwicz-Kriteriums 3.19 wird von einem Entscheider ausgegangen, der weder übermäßig pessimistisch noch optimistisch ist. Der Entscheider kann das Ergebnis durch

[132] Quelle: in Anlehnung an Zimmermann (2008), S. 15.
[133] Vgl. Rommelfanger/Eickemeier (2002), S. 51f.
[134] Vgl. Scholl (2001), S. 136.

den Optimismusparameter λ beeinflussen. Bei $\lambda = 0$ würde das Maximin-Kriterium vorliegen.[135]

$$0 \leq \lambda \leq 1 \qquad \arg\max_{x \in X} \left(\lambda \cdot \max_{s \in S} z(x_x, s_s) + (1 - \lambda) \cdot \min_{s \in S} z(x_x, s_s) \right) \tag{3.19}$$

Es wird jene Planungsalternative gewählt, die den besten gewichteten Durchschnittswert aus dem besten und schlechtesten Ergebnis hat. Die Durchschnittsbildung macht es nötig, dass die Werte der Nutzenmatrix metrisch und nicht ordinal sein müssen. Die Bestimmung eines idealen Optimismusparameters ist ein Problem des Kriteriums, die Ungleichverteilung der einbezogenen Szenarien ein weiteres. Bei letzterem besteht die Herausforderung darin, alle Szenarien in die Ermittlung des Planungswertes möglichst gleichberechtigt zu betrachten, da durch die Definition bereits der Fokus auf den beiden Extrema liegt. [136]

Hinsichtlich der robusten Planung ist das Kriterium für kleine λ als ergebnisrobust einzustufen. Die Informationsrobustheit ist hoch, da nicht extreme Szenarien keine Änderungen bewirken können. Der Extremszenarien muss man sich aber relativ sicher sein.[137]

Savage-Niehans-Kriterium

Das Savage-Niehans-Kriterium verwendet zur Bestimmung der optimalen Planungsalternative den absoluten Regret 3.7 und nicht direkt die Ergebnisse der Alternativen.[138]

$$\arg\min_{x \in X} \max_{s \in S} \left(\max_{y \in Y} z(x_y, s_s) - z(x_x, s_s) \right) \tag{3.20}$$

$$\arg\min_{x \in X} \max_{s \in S} \left(ar(x_x, s_s) \right) \tag{3.21}$$

Bestimmt wird der maximale Regret jedes Szenarios s und gewählt wird jener Plan x, der das Minimum liefert. Vereinfacht man 3.20, so ist in 3.21 zu sehen, dass es sich um das Maximin-Kriterium angewandt auf den absoluten Regret handelt.[139]

Im Hinblick auf die robuste Planung muss festgehalten werden, dass relativ optimalitätsrobuste Ergebnisse erwartbar sind, die jedoch bei stark schwankenden szenariooptimalen Werten nicht sehr ergebnisrobust sind. Ansonsten gilt dieselbe Beurteilung wie beim Maximin-Kriterium. Selbstverständlich ist es möglich, den relativen Regret zu verwenden. Der Vorteil besteht darin, dass Ergebniseinbußen bei ungünstigen Szenarien stärker gewichtet werden wie bei günstigen. Diese Einstellung ist im Sinne der robusten Planung.[140]

[135] Vgl. Zimmermann (2008), S. 21f.
[136] Vgl. Rommelfanger/Eickemeier (2002), S. 52f.
[137] Vgl. Scholl (2001), S. 137.
[138] Vgl. Laus et al. (2012), S. 85.
[139] Vgl. Rommelfanger/Eickemeier (2002), S. 55.
[140] Vgl. Scholl (2001), S. 138.

Die Ansätze der robusten Planung können auch in bekannte Planungsmethoden aufgenommen werden. Nach der Identifikation der Unsicherheiten, wird in die linearen Modelle der Robustheitsansatz integriert.

3.4 Planungsansätze

In den folgenden Unterkapitel werden verschiedene Planungsansätze aus der Literatur vorgestellt und wie dabei das Konzept der Robustheit zur Anwendung kommt. Die Anschluss- und rollierende Planung sind bekannte Konzepte, die mit einigen einfachen Überlegungen robuster gestaltet werden können.

LEUNG und WU[141] stellen ein Konzept vor, in dem eine robuste Planung für die aggregierte Produktionsplanung durchgeführt wird. Es wird dabei ein Augenmerk auf die Unsicherheit der Daten gelegt.

GEBHARD[142] entwickelt ein Konzept für die gesamte hierarchische Produktionsplanung bei Unsicherheit. Da der Schwerpunkt dieser Arbeit jedoch auf der mittelfristigen taktischen Planung liegt, wird nur dieser Teil aus GEBHARDS Arbeit vorgestellt.

3.4.1 Rollierende Planung

Bei der rollierenden Planung handelt es sich um eine Planungsvorschreibung, wo zum Zeitpunkt t Entscheidungen für zukünftige Perioden getroffen werden. Abbildung 23 soll das Konzept mit den Terminologien verdeutlichen. Die Planungsläufe werden mit τ bezeichnet, die jeweils am Beginn von Periode t durchgeführt werden. Im Beispielfall wird bei $\tau 1$ zu Periode $t1 = 1$ die Planung von $t1, \dots t1 + T - 1$ durchgeführt. T wird als Planungshorizont oder Planreichweite bezeichnet. Es sind jedoch nur die ersten D Perioden fixiert, welche als Planabstand bezeichnet werden. Die weiteren $T - D$ Perioden sind nur vorläufig geplant. Der nächste Planungslauf erfolgt nach Ablauf des Planabstands. Bei $\tau 2$ startet die Planung zu Periode $t2 = t1 + D$ und wird für den Planungshorizont $t2, \dots, t2 + T - 1$ durchgeführt. Die Perioden $t5$ bis $t8$, welche zum Planungsdurchlauf $\tau 1$ noch vorläufig waren, werden jetzt fix geplant. Weiterhin vorläufig werden die Perioden $t9$ bis $t11$ geplant, falls jedoch neue Informationen vorliegen, werden diese einbezogen. Der Gesamtplanungszeitraum T_{max} ist in diesem Fall $t = 24$. Für den Fall, dass der Planabstand dem Planungshorizont gleich ist, spricht man von der Anschlussplanung[143].

Für das Ende der Planung gibt es verschiedene Ansätze. Sollte der Planungshorizont das Ende des Gesamtplanungszeitraumes erreichen, können alle Perioden des Planungshorizontes fix geplant werden. Es besteht jedoch auch die Möglichkeit den Planungshorizont entsprechend zu verringern wie im Beispiel von Abbildung 23 bei $\tau 5$, wo der Planungshorizont nur mehr acht Perioden statt der ursprünglichen zwölf hat.[144]

[141] Vgl. Leung/Wu (2004).
[142] Vgl. Gebhard (2009).
[143] Vgl. Herrmann (2011), S. 249.
[144] Vgl. Scholl (2001), S. 34.

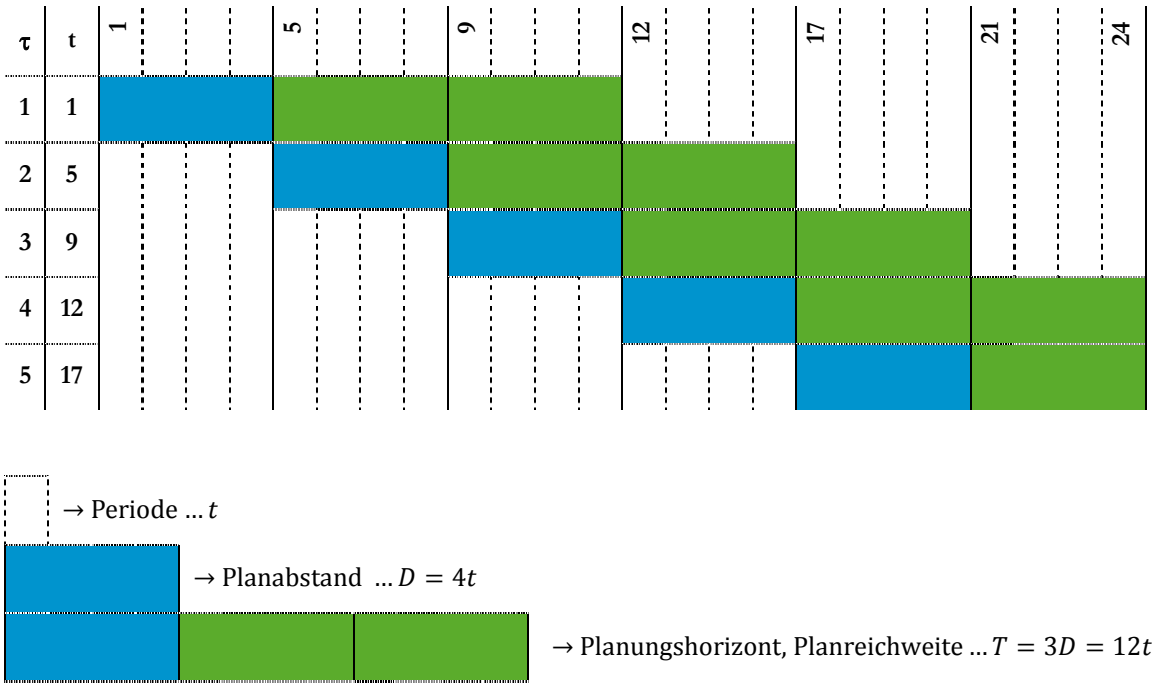


Abbildung 23: Konzept der rollierenden Planung[145]

Robustheitsbetrachtung

Aufgrund der Natur der rollierenden Planung müssen bereits getroffene Entscheidungen nach Ablauf des Planabstandes reevaluiert werden. Die Informationen werden gegen Ende des Planungshorizontes immer stärker mit Unsicherheit behaftet. Um die Planung in ihrem Ergebnis genau zu erhalten, sollte der Planabstand gering gehalten werden. Ein zu geringer würde jedoch die Nervosität der Planung erhöhen. Es sind daher verschiedene Aspekte zu berücksichtigen.[146]

- **Zeitlich-vertikale Interdependenz:** Die rollierende Planung kann Alternativen über den Planungszeitraum hinaus antizipieren und einmal getroffene Entscheidungen revidieren, sofern diese nicht innerhalb des Planabstandes liegen. Die wichtigsten Voraussetzungen, um diese Eigenschaft realisieren zu können, sind zuverlässige Prognosen. Sollten diese für den Planungshorizont nicht ausreichend genug sein, müssen bereits umgesetzte Entscheidungen ggf. unter hohen Kosten revidiert werden.[147]
- **Begrenzte Vorausschau:** Die Wirkung von Entscheidungen kann auf Grund des begrenzten Planungshorizontes nicht immer auf spätere Perioden abgeschätzt werden, da nicht genügend Informationen vorliegen. Der Gesamtplanungszeitraum wird daher künstlich abgeschlossen, da Informationen, die zu weit in der Zukunft liegen, mit einer zu großen Unsicherheit behaftet sind.[148]

[145] Quelle: in Anlehnung an Scholl (2001), S. 34.
[146] Vgl. Scholl (2001), S. 139.
[147] Vgl. Scholl (2001), S. 139.
[148] Vgl. ebd., S. 139f.

- **Reaktionsmöglichkeiten:** Durch die Möglichkeit bereits getroffene Entscheidungen auf Basis neuer Informationen neu zu evaluieren und ggf. zu revidieren, sofern die Entscheidungen noch nicht realisiert sind, gilt die rollierende Planung als zulässigkeitsrobuster als etwa die Anschlussplanung. Eine günstige Beurteilung der Optimalitäts- und Ergebnisrobustheit hängt stark vom gewählten Planabstand und der Genauigkeit der Prognosen ab. Durch die Möglichkeit auf negative Entwicklung reagieren zu können, ist jedoch in den meisten Fällen davon auszugehen, dass sich das Ergebnis positiv entwickelt und dadurch die Ergebnisrobustheit gut ist.[149]
- **Planungsnervosität:** Sollten die Entscheidungen zu oft revidiert werden, wenn etwa der Planabstand D gering ist oder die Annahmen des Planungshorizontes schlecht waren, kann dies zu einer schlechten Planungsrobustheit und großer Planungsnervosität führen.[150]
- **Informationsbedarf:** Die rollierende Planung hat einen erheblichen Informationsbedarf und setzt zuverlässige Prognosen voraus. Die Zuverlässigkeit nimmt mit dem Prognosezeitraum ab und die Kosten für zuverlässige Informationen nehmen zu. Somit kann der positive Effekt der anpassenden Planung durch die Prognosekosten aufgehoben werden. Es ist somit wichtig, ein Augenmerk auf die Wahl der Planreichweite und des Prognoseverfahrens zu legen.[151]

Die Planungsnervosität gehört zu den negativsten Seiten der rollierenden Planung. Sie entsteht durch Planabweichungen auf Grund der Modifikationen von Entscheidungen von einem Planungsschritt auf den nächsten. Die Nervosität kann durch eine gewichtete Summe der Gesamtveränderungen eines Plans abgebildet werden.

$$j = 1 \dots m\,, i = 2 \dots n \qquad \delta_{ij} = \sum_{\tau=1}^{T-D} w_{j\tau} \cdot \left| x_{i,j,h_i+\tau} - x_{i-1,j,h_i+\tau} \right| \tag{3.22}$$

Formel (**3.22** zeigt die Planabweichung zwischen den Schritten i und $i-1$ der geplanten Menge für Produkt j. Dabei wird die gewichtete Summe δ_{ij} der absoluten Differenzen berechnet. Die Anzahl der Produkte wird durch m angegeben und n ist die Anzahl der Planungsschritte. Als letzte Periode vor Beginn des Planungsschrittes $i = 1, \dots, n$ wird $h_i = (i-1) \cdot D$ verwendet. Die beiden Terme der absoluten Differenz ist die in Schritt i geplante Produktionsmenge von Produkt $j = 1, \dots, m$ für Periode t ($x_{i,j,t}$). Bei $w_{j\tau}$ handelt es sich um einen Gewichtungsfaktor für die Planabweichung bei Produkt j zwischen einem Planungsschritt i und dem vorhergehenden $i-1$. Bezogen wird diese auf die Periode $h_i + \tau$ mit $\tau = 1, \dots, T-D$ und dient dazu, um Ergebnisse späterer Planungsläufe geringer zu gewichten. Mit der gewichteten Summe können nun verschiedene Maßnahmen ergriffen werden, um die Planabweichungen zu verringern. Es ist möglich, für jedes Produkt einen Maximalwert für die Summe anzugeben. Weiters kann δ_{ij} mit Strafkosten belegt und in die Zielfunktion eingebunden werden. Falls sie direkt mittels Kostensätzen berechnet wird,

[149] Vgl. Scholl (2001), S. 140.
[150] Vgl. Scholl (2001), S. 140.
[151] Vgl. ebd., S. 140.

kann sie auch als Gesamtsumme $\sum_j \delta_{ij}$ für die Kosten der Planungsnervosität angenommen werden. Zusätzlich ist es möglich, nach der Minimierung der Gesamtsumme zu streben und daraus ein eigenes Ziel zu definieren. Der Planungshorizont ist eine, Größe die das Ergebnis der Planung maßgeblich beeinflusst. Es muss angestrebt werden, ihn so groß zu wählen, dass er alle Auswirkungen von Entscheidungen in der Zukunft möglichst vollständig unter der Voraussetzung berücksichtigt, dass die prognostizierten Werte hinreichend genau sind und die Kosten für die Erhebung der Daten noch in einem wirtschaftlichen Rahmen liegen. Eine Möglichkeit der Ermittlung des idealen Planungshorizontes bedient sich der Tatsache, dass die Unsicherheit der Informationen mit wachsendem Zeithorizont immer größer wird. Somit können seriös nur mehr Intervalle von Werten in der Zukunft angegeben werden. Die Überlegung in der Wahl des Planungshorizontes besteht darin ihn so festzulegen, dass eine eindeutige Reihung der Alternativen noch möglich ist. Das ist so lange der Fall, wie sich die Intervalle nicht überlappen. Eine wesentliche Einschränkung der Anwendbarkeit dieses Konzeptes besteht darin, dass sich die Alternativen wesentlich voneinander unterscheiden müssen, damit eine solche Reihung sinnvoll ist.[152]

In der Praxis wird es jedoch nötig sein, den Planabstand sowie den Planungshorizont anwendungsspezifisch zu wählen. Meist ergeben sich die Werte auch aus der Notwendigkeit heraus, dass auf den einzelnen Ebenen die Intervalle aufeinander sinnvoll abgestimmt werden müssen.[153]

Mit wachsender Unsicherheit kann auch empfohlen werden, die Perioden gegen Ende des Planungshorizontes zu aggregieren. Damit verringert sich die Unsicherheit ebenfalls durch eine Reduzierung der Periodenanzahl und der entsprechenden Interdependenzen. Die Nervosität steigt gerade, wenn sich Planungshorizonte überlappen und die Planabstände kurz sind. Der Planabstand soll also möglichst groß gewählt werden, ohne jedoch die Ergebnis-, Optimalitäts- oder Zulässigkeitsrobustheit zu negativ zu beeinflussen. Planabstände größer als eins verringern die Nervosität erheblich und beeinflussen die Qualität des Ergebnisses nur gering[154].

3.4.2 Robuste stochastische Bedarfsermittlung

Im Unterschied zur klassischen deterministischen Bedarfsermittlung wird bei der robusten stochastischen Bedarfsermittlung die Unsicherheit durch verrauschte, unvollständige und fehlerhafte Daten berücksichtigt.

Das folgende Modell versucht eine optimale Bedarfsermittlung mit Produktionsplan und Mitarbeiteranzahl für die mittelfristige Planung zu erstellen. Dabei sollten die Produktions-, Personal- und Bestandskosten unter der Berücksichtigung verschiedener wirtschaftlichen Szenarien minimiert werden. Das Modell baut auf der robusten Optimierung auf und verwendet zwei Regelsätze. Strukturregeln, welche mittels linearer Programmierung aufgestellt werden und deren Eingangsvariablen frei von Störungen sind. Steuerregeln sind unterstützender Natur und werden durch verrauschte Daten gestört. Um das System zu

[152] Vgl. Scholl (2001), S. 142ff.
[153] Vgl. Gebhard (2009), S. 19.
[154] Vgl. Scholl (2001), S. 144f.

vervollständigen, werden zwei Mengen von Variablen eingeführt. Dies sind die Designvariablen, welche nicht abgeändert werden können, und die Steuerungsvariablen, welche abänderbar sind, sobald der Unsicherheit unterworfene Daten beobachtet wurden.[155]

Formel 3.23 ist die grundlegende Zielfunktion der klassischen Optimierung. Die Designvariablen werden durch den Vektor x und die Steuerungsvariablen durch den Vektor y dargestellt.

$$\min c^T x + d^T y \tag{3.23}$$

$$Ax = b \tag{3.24}$$

$$Bx + Cy = e \tag{3.25}$$

$$x, y \geq 0 \tag{3.26}$$

Nebenbedingung (**3.24** sind die Strukturregeln, welche frei von Unsicherheit sind, und Nebenbedingung (**3.25** stellt die Steuerregeln dar, deren Koeffizienten durch Unsicherheit verrauscht sind. Die Nichtnegativitätsbedingung (**3.26** soll sicher stellen, dass sich die Ergebnisse im realen Raum befinden. Das robuste Modell bezieht nun verschiedene Szenarien s aus einer Gesamtmenge $\Omega = \{1,2,\dots,S\}$ ein. Wie bereits erwähnt, sind die Steuerregeln jene, die auf Unsicherheiten und Störungen reagieren, um den Plan dadurch zu adaptieren. Daher werden die Koeffizienten den Szenarien s zugeordnet $\{d_s, B_s, C_s, e_s\}$. Jedes Szenario hat eine Eintrittswahrscheinlichkeit p_s, wobei für das gesamte Modell $\sum_{s=1}^{S} p_s = 1$ gilt. Das Modell wird auf die Optimalitäts- und Zulässigkeitsrobustheit hin optimiert. Zwischen diesen beiden Kriterien wird es sehr wahrscheinlich zu einem Zielkonflikt kommen, da es kaum möglich ist, absolute Optimalitäts- und Zulässigkeitsrobustheit für alles Szenarien eines Plans zu erreichen. Das robuste Planungsmodell wird so formuliert, um diesen Zielkonflikt zu messen[156].

Für das robuste Modell wird der Vektor mit den Steuerungsvariablen von den Szenarien abhängen müssen (y_s). Zusätzlich wird ein Vektor δ_s für die Messung der Nicht-Zulässigkeit des Modells eingeführt. Somit kann das robuste Optimierungsmodell aufgestellt werden.

$$\min \sigma(x, y_1, y_2, \dots, y_S) + \omega\rho(\delta_1, \delta_2, \dots, \delta_S) \tag{3.27}$$

$$Ax = b \tag{3.28}$$

$$\forall\, s \in \Omega \qquad B_s x + C_s y_s + \delta_s = e_s \tag{3.29}$$

[155] Vgl. Leung/Wu (2004), S. 504.
[156] Vgl. Leung/Wu (2004), S. 504f.

$$\forall\, s \in \Omega \qquad x, y_s \geq 0 \tag{3.30}$$

Der erste Term der Zielfunktion 3.27 ist wie bei der stochastischen linearen Programmierung üblich der Erwartungswert $\sigma(\cdot) = \sum_{s\in\Omega} p_s \xi_s$. Er misst die Optimalitätsrobustheit. Die klassische Zielfunktion 3.23, nur angepasst an die Szenariobetrachtung, befindet sich hinter dem Term $\xi_s = c^T x + d_s^T y_s$. Der zweite Term von 3.27, $\rho(\cdot)$ bestraft Verletzungen der Steuerungsregeln, wenn nicht zulässige Lösungen unter einem Szenario ermittelt wurden und misst die Zulässigkeitsrobustheit. Mit Hilfe des Gewichtungsfaktors ω kann der Zielkonflikt zwischen den beiden Robustheitskriterien dargestellt werden. Wäre er null, so müsste nur $\sigma(\cdot)$ minimiert werden und auf Grund des Fehlens der Steuerungsregeln bestünde die Möglichkeit, dass die Lösung unzulässig ist. Wählt man den Faktor sehr groß, dominiert der Term der Zulässigkeitsrobustheit und die Kosten würden sehr stark ansteigen.[157]

Robustes Modell für mehrere Szenarien

Es kann nun das Modell unter Berücksichtigung mehrerer Szenarien aufgestellt werden, wobei die Unsicherheit von ausgelagerten Fertigungskosten, die Kosten für die Schwankung des Mitarbeiterstandes und vor allem der Kundenbedarf berücksichtigt wird. Tabelle 3 gibt eine Aufstellung und Erklärung über die berücksichtigten Variablen in der Kostenermittlung.[158]

[157] Vgl. Leung/Wu (2004), S. 505.

[158] Vgl. Leung/Wu (2004), S. 508.

Tabelle 3: Modellvariablen robuste stochastische Bedarfsermittlung[159]

Variable	Bedeutung
P_{it}	Produktionsmenge von Produkt i in Periode t bei Regelarbeitszeit [Stk].
Y_{it}	Produktionsmenge von Produkt i in Periode t bei Überstunden [Stk].
Z_{it}	Produktionsmenge von Produkt i in Periode t durch externe Fertigung [Stk].
H_t	Anzahl der eingestellten Mitarbeiter in Periode t [Manntage].
L_t	Anzahl der ausgestellten Mitarbeiter in Periode t [Manntage].
I_{it}^s	Bestand von Produkt i in Periode t bei Szenario s [Stk].
W_t	Mitarbeiterstand in Periode t [Manntage].
c_{pi}	Produktionsstückkosten (ohne Lohkosten) für Produkt i bei Regelarbeitszeit in Periode t [GE/Stk].
c_{Yi}	Produktionsstückkosten (ohne Lohkosten) für Produkt i bei Überstunden in Periode t [GE/Stk].
c_{Zi}^s	Produktionsstückkosten für Produkt i bei externer Fertigung für eine Periode bei Szenario s [GE/Stk].
c_{Wt}	Lohnkosten in Periode t [GE/Manntage].
c_{Ii}	Bestandskosten für Produkt i am Ende einer Periode für eine Periode [GE/Stk].
c_{Ht}^s	Kosten einen Mitarbeiter in Periode t einzustellen bei Szenario s [GE/Manntage].
c_{Lt}^s	Kosten einen Mitarbeiter in Periode t auszustellen bei Szenario s [GE/Manntage].

Vier Gleichungen definieren die zu optimierenden Kostenblöcke. Die Produktionskosten (PK) 3.31 enthalten die regulären Kosten, die Kosten für Fertigung mittels Überstunden und die Kosten für die externe Fertigung durch Fremdfirmen. Gleichung 3.32 bestimmt die totalen Lohnkosten (LK). Die totalen Bestandskosten (BK) von Einheiten eines Produktes im Lager werden mittels Gleichung 3.33 bestimmt. Die Kosten für die Schwankung im Mitarbeiterbestand (SK), z.B. durch Zeit- oder Kurzarbeit, sind in Form von Gleichung 3.34 wiedergegeben.[160]

$$\sum_{i=1}^{N}\sum_{t=1}^{T}(c_{pi}P_{it} + c_{Yi}Y_{it} + c_{Zi}^{s}Z_{it}) \tag{3.31}$$

$$\sum_{t=1}^{T} c_{Wt}W_t \tag{3.32}$$

[159] Quelle: in Anlehnung Leung/Wu (2004), S. 506f.

[160] Vgl. Leung/Wu (2004), S. 507.

$$\sum_{i=1}^{N}\sum_{t=1}^{T} c_{Ii} I_{it}^{s} \tag{3.33}$$

$$\sum_{t=1}^{T} (c_{Ht}^{s} H_t + c_{Lt}^{s} L_t) \tag{3.34}$$

Die Kostenblöcke werden nun in die Gleichung 3.27 eingesetzt, wodurch sich die endgültige Zielfunkton für dieses robuste Planungsmodell ergibt.[161]

$$\begin{aligned} &\min \sum_{s=1}^{S} p_s (PK + LK + BK + SK) \\ &+\lambda \sum_{s=1}^{S} p_s \left[(PK + LK + BK + SK) - \sum_{s=1}^{S} (PK + LK + BK + SK) + 2\theta_s \right] \\ &+\omega \sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{t=1}^{T} p_s \varepsilon_{it}^{s} \end{aligned} \tag{3.35}$$

Die ersten beiden Terme von 3.35 messen die Optimalitätsrobustheit, wobei der erste Term der Erwartungswert und der zweite Term die absolute Gesamtkostenabweichung bezüglich des Erwartungswertes ist. Die Variable λ bestimmt die Sensitivität gegenüber Unsicherheit in den Daten in allen Szenarien. Je größer λ ist, desto unsensibler ist die Lösung gegenüber Unsicherheit. Der dritte Term misst die Zulässigkeitsrobustheit in Bezug auf die Ungültigkeit ergebend aus der Nebenbedingung 3.36 und der Eintrittswahrscheinlichkeit p_s für Szenario s. Diese Nebenbedingung berechnet den prognostizierten Nachfragebedarf von Produkt i in Periode t bei Szenario s. Die entsprechende Unterdeckung des Nachfragebedarf ist ε_{it}^{s}.[162]

$$P_{it} + Y_{it} + Z_{it} + I_{it-1}^{s} - I_{it}^{s} + \varepsilon_{it}^{s} = D_{it}^{s} \tag{3.36}$$

$$(PK + LK + BK + SK) - \sum_{s=1}^{S} p_s (PK + LK + BK + SK) + \theta_s \geq 0 \tag{3.37}$$

$$P_{it}, Y_{it}, Z_{it}, W_t, H_t, L_t, I_{it}^{s}, \varepsilon_{it}^{s}, \theta_s \geq 0 \tag{3.38}$$

Für die Nebenbedingungen 3.36, 3.37 und 3.38 gilt $i = 1, \dots, N$, $t = 1, \dots, T$, und $s = 1, \dots, S$. Bei θ_s handelt es sich um eine szenarioabhängige nicht negative Ausgleichsvariable, die unter Einhaltung von Nebenbedingung 3.37 dimensioniert werden muss und eine Erleichterung der Lösung der Zielfunktion bringt.[163]

[161] Vgl. Leung/Wu (2004), S. 508.
[162] Vgl. Leung/Wu (2004), S. 508.
[163] Vgl. ebd., S. 505f.

Fallbeispiel

Das Modell wird in einem einfachen Beispiel mit zwei Produkten und über acht Perioden angewandt. Es werden vier wirtschaftliche Szenarien - *Boom, Gut, Ausreichend und Schlecht* - mit entsprechenden Eintrittswahrscheinlichkeiten p_s angenommen. Für jedes dieser Szenarien wird für jede Steuerungsvariable ein Wert angegeben und für jede Periode wird für jedes Produkt und jedes Szenario eine Bedarfsprognose erstellt.

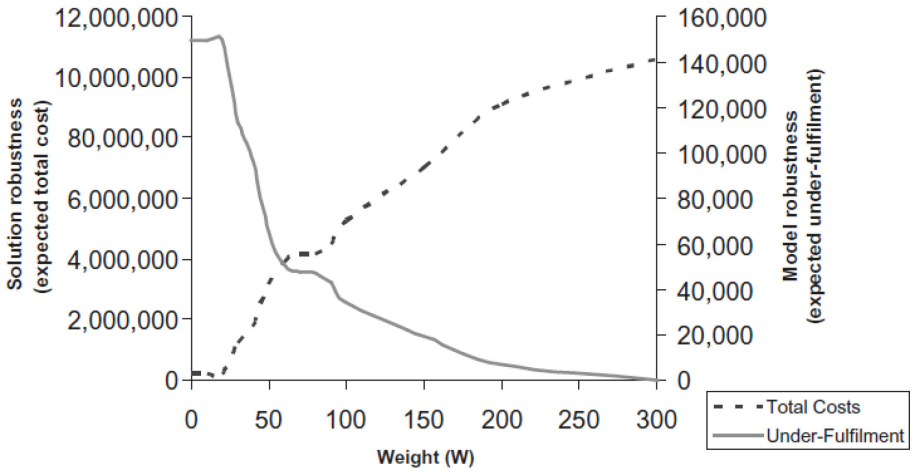


Abbildung 24: Optimalitätsrobustheit vs. Zulässigkeitsrobustheit[164]

Die Variable λ blieb in der Simulation unverändert, ω wurde hingegeben verändert, um den Zielkonflikt zwischen Optimalitätsrobustheit und Zulässigkeitsrobustheit abzubilden und einen idealen Produktionsplan zu ermitteln.[165] Abbildung 24 veranschaulicht diesen Zielkonflikt. Mit steigender Optimalitätsrobustheit (solution robustness) fällt die Zulässigkeitsrobustheit (model robustness).

3.4.3 Robuste hierarchische Produktionsplanung

Im folgenden Abschnitt wird die Anwendung von robusten Mechanismen auf die hierarchische Produktionsplanung erarbeitet. Der Umfang beschränkt sich jedoch auf die mittelfristige oder taktische Bedarfsplanung. Weitere Aspekte werden nur im Rahmen eines groben Überblicks behandelt. Der gewählte Ansatz geht das Problem in vier Schritten an. Wobei er sich an die Prinzipien der hierarchischen Planung hält und diese um Aspekte der robusten Planung erweitert.

Hierarchisierung und Dekomposition

Schritt eins ist die Zerlegung des Gesamtplanungsproblems in lösbare Teilprobleme und die Hierarchisierung selbiger. Die Hierarchieebenen werden klassisch in strategisch, taktisch und operativ gegliedert, wobei diese Arbeit den Fokus auf die taktische Ebene mit einer Fristigkeit von einem halben Jahr bis zwei Jahre legt.[166]

[164] Quelle: Leung/Wu (2004), S. 512.
[165] Vgl. ebd., S. 509ff.
[166] Vgl. Gebhard (2009), S. 69ff.

Modellauswahl und Strategiefestlegung

Der zweite Schritt behandelt die Auswahl des Planungsmodells und die Festlegung der Planungsstrategien für die Hierarchieebenen. Als Modell wird AGGRPLAN von Günther und Tempelmeier[167] verwendet.

Tabelle 4: Variablen für AGGRPLAN[168]

Variable	Bedeutung
$\boldsymbol{C^{max}}$	Verfügbare Normalkapazität Periode
$\boldsymbol{L_{gt}}$	Lagerbestand von Produktfamilie g am Ende von Periode t.
L_g^0	Anfangslagerbestand von Produktfamilie g.
U_t	Genutzte Zusatzkapazität an Periode t.
U^{max}	Maximale Zusatzkapazität pro Periode.
X_{gt}	Produktionsmenge von Produktfamilie g in Periode t.
a_g	Produktionskoeffizient von Produktfamilie g.
$\boldsymbol{d_{gt}}$	Nachfragemenge von Produktfamilie g in Periode t.
l_g	Lagerkostensatz für Produktfamilie g pro Mengeneinheit und Periode.
u	Kosten für eine Einheit Zusatzkapazität.

$$\min \sum_{g \in G} \sum_{t=1}^{T} l_g \cdot L_{gt} + \sum_{t=1}^{T} u \cdot U_t \tag{3.39}$$

$$\forall (g \in G; t \in T) \qquad L_{g,t-1} + X_{gt} - L_{gt} = d_{gt} \tag{3.40}$$

$$\forall t \in T \qquad \sum_{g \in G} a_g \cdot X_{gt} \leq C^{max} + U_t \tag{3.41}$$

$$\forall t \in T \qquad U_t \leq U^{max} \tag{3.42}$$

$$\forall g \in G \qquad L_{g,0} = L_g^0 \tag{3.43}$$

$$\forall (g \in G; t \in T) \qquad U_t, L_{gt}, X_{gt} \geq 0 \tag{3.44}$$

Dabei soll die Zielfunktion 3.39, bestehend aus den Lagerkosten und den Kosten für die genutzte Zusatzkapazität, minimiert werden. Die Minimierung erfolgt unter den

[167] Vgl. Günther/Tempelmeier (2012)
[168] Vgl. Gebhard (2009), S. XXIII ff.

Nebenbedingungen 3.40 - 3.44. Diese wären die Lagerbilanzgleichung 3.40 und die Gleichung für Kapazitätsrestriktionen 3.41, die sicherstellt, dass die maximale Kapazität verwendet und ggf. erweitert wird. Nebenbedingung 3.42 stellt die Obergrenze für die Zusatzkapazität dar. Gleichung 3.43 legt die Anfangsbestände für Produktfamilien fest und 3.44 sind die obligaten Nichtnegativitätsbedingungen.[169]

Zur Beurteilung des Modells wird das Erwartungswert-Regret-Kriterium verwendet. Der Detaillierungsgrad der Ebenen richtet sich nach deren Fristigkeit. Die mittelfristige Ebene basiert auf der Bildung von Produktfamilien, die kurzfristige operative Ebene auf einzelnen Produkten. Weiters werden im zweiten Schritt die unsicheren Parameter und die Szenarien ermittelt. Der einzige unsichere Parameter ist im betrachteten Vorgehen die Nachfragemenge. Mit dieser Festlegung können die Szenarien ermittelt werden. Die Planung wird im rollierenden Verfahren durchgeführt. Die Vorteile der laufenden Aktualisierung der Daten überwiegen den Nachteile der Planungsnervosität. Um die Planungsnervosität gering zu halten, wird empfohlen die verschiedenen Zeitabstände aufeinander abzustimmen. Da zwischen den Ebenen Kopplungen bestehen, müssen die Periodenlänge und die eingefrorenen Planungshorizonte sinnvoll zusammen hängen. Der Planabstand der taktischen Ebene sollte somit wenigstens so groß sein wie jener der operativen Ebene.[170]

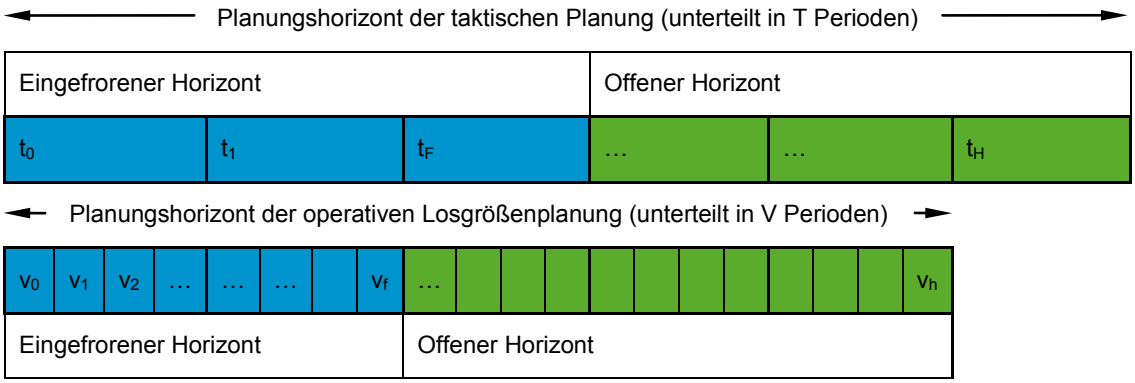


Abbildung 25: Abgestimmte Planungshorizonte[171]

Abbildung 25 veranschaulicht das Zusammenspiel der einzelnen Periodenlängen über die Hierarchieebenen hinweg. Die Zeitabstände der taktischen Ebene werden oft als Makroperioden, die der operativen Ebene als Mikroperioden bezeichnet.

Aggregation der Daten und Kopplung der Ebenen

Im Rahmen des dritten Schrittes wird die Koordination der Modelle und Ebenen vorgenommen. Ein zentraler Punkt in dieser Phase der Planung ist die Aggregation bzw. die Disaggregation der Elemente. Die Aggregation der Elemente soll im Rahmen der robusten hierarchischen Produktionsplanung so erfolgen, dass Fehler und Abweichungen durch die Aggregation möglichst ausgeschlossen werden. Ein weiterer zentraler Punkt ist die Kopplung der Hierarchieebenen. Hier stellt sich die Frage über den Zeitpunkt des Austausches von Informationen zwischen den Ebenen. Im Falle der robusten

[169] Vgl. Günther/Tempelmeier (2012) zitiert nach Gebhard (2009), S. 72.
[170] Vgl. Gebhard (2009), S. 75f.
[171] Quelle: in Anlehnung an Gebhard (2009), S. 76.

hierarchischen Produktionsplanung ist es nötig, spezielle Gegebenheiten zu beachten. So müssen Vorgabewerte von übergeordneten Ebenen mittels szenariounabhängiger Entscheidungsvariablen ermittelt werden. Andere Werte, welche auf die untergeordneten Ebenen keinen Einfluss haben, können auch durch szenarioabhängige Entscheidungsvariablen modelliert werden.[172]

In dieser Phase wird auch die Planungsstabilität maßgeblich beeinflusst. Diese sollte möglichst groß sein, da einmal gemachte Vorgaben aus der taktischen Ebene in der operativen Ebene nur mehr schwer rückgängig gemacht werden können. Zusätzlich birgt eine häufige Änderung auch die Gefahr der Planungsnervosität. Durch die Beschränkung der Änderungen von Entscheidungsvariablen in den aufeinanderfolgenden Läufen der rollierenden Planung können stabile Pläne erzielt werden. Die Beschränkung erfolgt durch die Vorgabe von Grenzwerten für die Änderungen oder durch die Einführung von Strafkosten, anhand derer die Änderungen bewertet werden. Die Strafkosten werden in die Gesamtzielfunktion integriert. Die Herangehensweise über Strafkosten hat den Vorteil gegenüber jener der Grenzwerte, dass sie flexibler ist. Sobald ein Grenzwert nicht eingehalten wird, ist das Modell mathematisch nicht mehr korrekt lösbar. Bei den Strafkosten verschlechtert sich das Ergebnis nur bleibt in Summe jedoch gültig.[173]

Systemanwendung

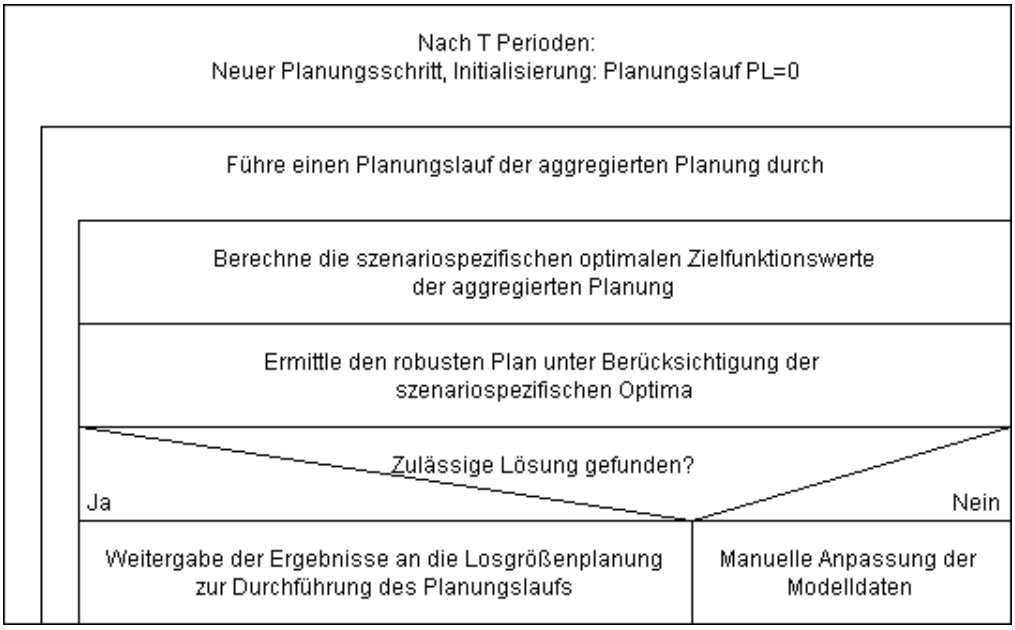


Abbildung 26: Struktogramm des taktischen Planungsablaufes[174]

Im betrachteten Beispiel wird die Planungsstabilität durch die Einführung von Strafkosten gewährleistet. Diese fallen an, wenn die Werte des vorherigen Planungslaufs von jenen des aktuellen abweichen. Folgend wird das mathematische Modell aufgestellt, wobei jeweils gilt: $\forall s \in S;\ \forall g \in G$. Tabelle 14 im Anhang gibt eine Übersicht über die Variablen des

[172] Vgl. Gebhard (2009), S. 77ff.
[173] Vgl. Gebhard (2009), S. 79ff.
[174] Quelle: in Anlehnung an Gebhard (2009), S. 82.

Gleichungssystems. Es wird nur ein Auszug ausgewählter Gleichungen gezeigt und beschrieben.[175]

$$\min \frac{1}{|S|} \cdot \sum_{s=1}^{S} \frac{ZF_s^+}{zfopt_s} \tag{3.45}$$

$$ZF_s - ZF_s^+ = zfopt_s \tag{3.46}$$

$$ZF_s = \left[\begin{array}{l} \sum_{g=1}^{G} \sum_{t=t_0}^{t_H} \left(l_g \cdot L_{gt}\right) + \sum_{g=1}^{G} \sum_{t=t_F}^{t_H} \left(l_g \cdot \left(L_{sgt}^+ - L_{sgt}^-\right)\right) \\ + \sum_{t=t_0}^{t_H} (u \cdot U_t) + \sum_{g=1}^{G} \sum_{t=t_0}^{t_F} \left(cl_{gt}^+ \cdot L_{gt}^+ + cl_{gt}^- \cdot L_{gt}^-\right) \\ + \sum_{t=t_0}^{t_F} (cu_t^+ \cdot U_t^+ + cu_t^- \cdot U_t^-) + \sum_{t=t_F}^{t_H} \left(u \cdot (U_{st}^+ - U_{st}^-)\right) \end{array} \right] \tag{3.47}$$

$$t = t_F + 1, \dots, t_H \qquad d_{sgt} = \begin{bmatrix} \left(L_{g,t-1} + L_{sg,t-1}^+ - L_{sg,t-1}^-\right) \\ -\left(L_{gt} + L_{sgt}^+ - L_{sgt}^-\right) + X_{sgt} \end{bmatrix} \tag{3.48}$$

$$t = t_0, \dots, t_F \qquad \sum_{g=1}^{G} a_g \cdot X_{sgt} \leq c^{alt} \cdot (C^{max} + U_t) \tag{3.49}$$

$$t = t_F, \dots, t_H \qquad \sum_{g=1}^{G} a_g \cdot X_{sgt} \leq c^{alt} \cdot (C^{max} + U_t + U_{st}^+ - U_{st}^-) \tag{3.50}$$

$$t = t_0, \dots, t_F \qquad U_t - U_t^+ + U_t^- = U_t^{alt} \tag{3.51}$$

$$t = t_F, \dots, t_H \qquad U_t + U_{st}^+ \leq U^{max} \tag{3.52}$$

Das Modell ist zeitinvariant. t_0 ist die erste Periode des taktischen Plans, t_F ist die letzte Periode des gefrorenen Horizontes und t_H ist die letzte Periode des Planungshorizontes. Gleichung 3.45 stellt die Zielfunktion dar und minimiert die Summe der relativen Abweichungen der szenarioabhängigen Zielfunktionswerte von den szenariospezifischen Optima. Diese Abweichung wird in Gleichung 3.46 berechnet. Die Kosten des Grundplans mit den szenariospezifischen Anpassungen setzten sich aus den Lagerhaltungskosten und den Kosten für die Nutzung der Zusatzkapazität sowie aus den Strafkosten für die Planabweichungen zusammen und werden mit Gleichung 3.47 ermittelt. Die Lagerbilanzgleichung 3.48 beinhaltet die szenarioabhängigen Anpassungen der Lagerbestände außerhalb des Vorgabezeitraumes. Die Gesamtkapazität verringert sich durch Rüstvorgänge in der operativen Planung. Diesem Umstand wird in Gleichung 3.49

[175] Vgl. Gebhard (2009), S. 83.

Rechnung getragen. Der Abschlagfaktor c^{alt} wird nach einem Planungsdurchgang auf der operativen Ebene aktualisiert und an die übergeordnete taktische Ebene zurück gemeldet.[176]

Weitere Anpassungen der Kapazitätsrestriktionen werden in 3.50 vorgenommen und szenariospezifische Überstunden berücksichtigt. Die Aktualisierung der geplanten Überstunden gegenüber des vorhergehenden Planungslaufes wird in 3.51 vorgenommen. Die maximale Zusatzkapazität wird in Gleichung 3.52 sichergestellt. Der letzte Teil des vierten Schrittes ist die laufende Anpassung der Daten und Parameter. Es empfiehlt sich, das gesamte System an Vergangenheitsdaten zu testen und ggf. die Einstellungen zu anzupassen. Im Laufe der Planung ist es ratsam, in regelmäßigen Abständen die verwendeten Parameter zu überprüfen und bei Bedarf die Datenbasis anzupassen. Die Qualität der Szenarien und der Aggregation kann anhand von Prognose- und Aggregationsfehlern festgestellt werden.[177]

3.5 Fazit der robusten Planung

Die robuste Planung versucht für verschieden Szenarien einen allgemeingültigen Plan zu entwickeln. Dieser Plan entspricht zwar nicht dem Kostenoptimum, es entfallen jedoch diverse Neuplanungsvorgänge. Die deterministische Planung würde hingegen für jedes mögliche Szenario ein Optimum suchen. Der Planungsaufwand und die Effekte durch oftmaliges Umplanen können Kostenvorteile des punktuellen Optimums zunichtemachen.

Abbildung 27 zeigt dieses Konzept anhand der Kosten der Versorgungskette und der Unsicherheit. Während die deterministische Planung das Optimum sucht und geringere Kosten erzielt, versucht die robuste Planung eine Lösung über eine größere Reichweite des Szenarioraumes.

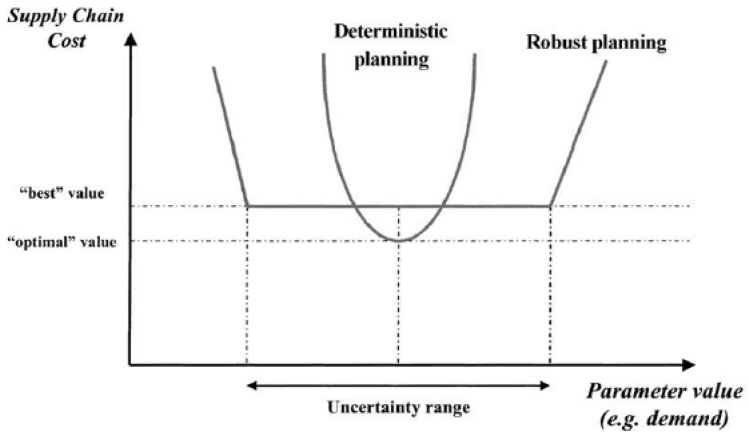


Abbildung 27: Konzept der robusten Planung[178]

[176] Vgl. Gebhard (2009), S. 83ff.
[177] Vgl. Gebhard (2009), S. 84ff.
[178] Quelle: Van Landeghem/Vanmaele (2002), S. 774.

Eine zentrale Rolle bei der robusten Planung spielt die Unsicherheit der Daten und daher die Erstellung von zuverlässigen Prognosen. Je zuverlässiger die Prognosen sind, desto einfacher ist es für einen Planer ein geeignetes Modell zu erstellen und die richtigen Szenarien zu wählen. Somit setzen sich die Vorteile der robusten Planung noch stärker gegen die der deterministischen Planung durch.

Ein weiterer Vorteil ist, dass durch die Erweiterung der mathematischen Modelle von bereits bekannten Planungsansätzen die robuste Planung integriert werden kann. Dies ist bei der rollierenden Planung sowie bei der Bedarfsermittlung der Fall. Es kann das Konzept auf eine Kombination dieser beiden Ansätze, der hierarchischen Produktionsplanung, ausgedehnt werden. Ein zentraler Punkt in den vorgestellten Planungsansätzen, sowie in der Kapazitätsgrobplanung, ist die Aggregation der Daten. Je besser diese erfolgt, desto besser ist die Robustheit des Plans. Das folgende Kapitel beschäftigt sich mit Aggregationsmöglichkeiten und deren Anwendung.

4 Datenaggregation

Im Rahmen der taktischen Grobplanung ist es nötig, mit Daten in aggregierter Form, zu arbeiten. Zum einen würde die Planung mit allen einzelnen Planungsobjekten zu viel Zeit in Anspruch nehmen und dem Konzept der Grobplanung widersprechen, zum anderen liegen, über den Planungszeitraum betrachtet, die Planungsobjekte nicht in der erforderlichen Informationsgranularität vor, um sie für sich stehend in den Planungsprozess einzubeziehen.[179]

In welcher Form und aufgrund welcher Attribute die Planungsobjekte zu Gruppen aggregiert werden ist vom Planungsziel abhängig. Im Rahmen dieser Arbeit wird die Datenaggregation für die Bildung von Produktfamilien verwendet. Mithilfe dieser Produktfamilien wird die Kapazitätsgrobplanung durchgeführt. Die Aggregationsattribute sind entweder produktspezifischer Natur, wie geometrische Maße und Werkstoffeigenschaften, oder wertstromspezifischer Natur wie die Anlagenfolge der Produktion.

Folgend werden zwei Möglichkeiten zur Datenaggregation allgemein betrachtet und Anwendungsbeispiele mit Produktionsplanungshintergrund gegeben. Zu Beginn wird die Fuzzy Mengenlehre erörtert. Diese eignet sich gut für Probleme, bei denen die Abgrenzung der Aggregationskriterien nicht eindeutig festlegbar ist.[180] Als zweite Möglichkeit werden die multivariaten Ansätze durchgenommen. Der Vorteil besteht in der einfachen Implementierung und des breiten Einsatzfeldes.

4.1 Fuzzy Mengenlehre

Die Fuzzy Mengenlehre kommt zur Anwendung, wenn Aggregationsobjekte keine klar definierte Abgrenzung ihrer Zugehörigkeit haben. Beispiele wären geometrische Spezifikationen wie „erheblich größer/kleiner als…“ oder ähnliche verbale Beschreibungen von Eigenschaften, die nicht immer eindeutig einem scharfen Wert zuordenbar sind. In vielen Situationen, speziell beim Übergang von einer manuellen Tätigkeit auf einen automatisierten Ablauf, ist es nötig das angesammelte Wissen in Form der Erfahrung von Mitarbeitern in ein für ein Programm verwertbares Format zu bringen. Da diese verbalen Beschreibungen meist nicht scharf abgegrenzt sind, sondern eine Vielzahl von gleitenden Übergängen von Zugehörigkeiten, s.g. Memberships, haben, eignet sich die Fuzzy-Mengenlehre speziell um diese Situation abzubilden.[181]

Während in der klassischen Mengenlehre Objekte zu einer Menge gehören und zu den restlichen Mengen nicht, ist es in der Fuzzy-Mengenlehre möglich, dass Objekte zu verschiedenen Mengen gleichzeitig gehören. Der Grad der Zugehörigkeit wird durch $f(x)$ abgebildet. X ist eine Ansammlung von Objekten mit einem generischen Element X, damit gilt $X = \{x\}$. Die Fuzzymenge A in X wird durch die Menge von Zugehörigkeitsgraden der Punkte $x \in A$, der s.g. Zugehörigkeitsfunktion, charakterisiert. Die wichtigste Eigenschaft

[179] Vgl. Steven (1994), S. 64f.
[180] Vgl. Sangwan/Kodali (2004), S. 292f.
[181] Vgl. Zadeh (1965), S. 338f.

der Fuzzy-Mengenlehre wird in 4.1 formalisiert. Sie besagt, dass die Zugehörigkeit eines Punktes x zu einer Menge A im Intervall $[0,1]$ liegt.[182]

$$0 \leq f_A(x) \leq 1 \tag{4.1}$$

Je nähre an eins, desto größer ist die Zugehörigkeit des Punktes zur Menge. Die Menge A kann somit als eine Menge von geordneten Paaren, welche die Zugehörigkeitsfunktion abbildet, wie in 4.2 zu sehen ist, betrachtet werden.

$$x \in X \qquad A = \{(x, f_a(x))\} \tag{4.2}$$

Eine weitere wichtige Definition ist der Betrag der Menge A, wie sie in 4.3 gegeben wird. Dieser ist als die Summe über alle Zugehörigkeitsfunktionen, der der Menge angehörigen Punkte x definiert.[183]

$$|A| = \sum_{x \in A} f_A(x) \tag{4.3}$$

Die parametrische Funktion 4.4 der Zugehörigkeitsfunktion kann graphisch dargestellt werden, indem die Zugehörigkeit über die betrachtete Größe aufgetragen wird. Die entstehende Kurve kann je nach den Funktionswerten unterschiedliche Formen annehmen. Eine der gängigsten Formen ist eine Dreieckskurve, welche ab hier als trianguläre Zugehörigkeitsfunktion bezeichnet wird und beispielhaft in Abbildung 28 zu sehen ist. P_2 wird als Maximumstelle der Dichtefunktion oder auch als aussichtsreichster Wert angenommen. P_1 ist der pessimistischste Wert und P_3 der optimistischste.[184]

$$f(x) = \begin{cases} 0 & wenn \quad x \leq P_1 \vee x \geq P_3 \\ \dfrac{x - P_1}{P_2 - P_1} & wenn \quad P_1 \leq x \leq P_2 \\ \dfrac{P_3 - x}{P_2 - P_3} & wenn \quad P_2 \leq x \leq P_3 \end{cases} \tag{4.4}$$

[182] Vgl. Zadeh (1965), S. 339.
[183] Vgl. Klüver et al. (2012), S. 161.
[184] Vgl. Arnold (2013), S. 20f.

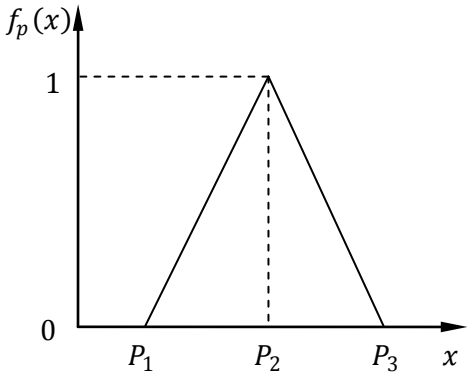


Abbildung 28: Trianguläre Zugehörigkeitsfunktion[185]

Die Beschreibung weiterer Definitionen und Rechenoperationen der Fuzzy-Mengenlehre sind für die folgenden Verfahren der Aggregation nicht nötig und würden zusätzlich den Rahmen der Arbeit sprengen. Bei Interesse wird auf die einschlägigen Quellen der Fußnoten, zu finden im Literaturverzeichnis, verwiesen.

4.1.1 Fuzzy Produktfamilienbildung mit AHP

Die Produktfamilienbildung ist ein zentrales Thema in der Kapazitätsgrobplanung und die Fuzzy Mengenlehre ist ein äußerst geeignetes Einsatzgebiet. Produkt-, oder Teilefamilien können auf Grund verschiedenster Merkmale gebildet werden. Bei technischen Merkmalen ist eine zu scharfe Abgrenzung nicht immer zielführend. Die geometrischen Maße zweier Produkte können sich z.B. um nur einige hundertstel Millimeter unterscheiden, durch eine scharfe Grenze kann es jedoch sein, dass die beiden verschiedenen Gruppen zugeordnet werden. In einer solch klassischen Aggregation gehören Produkte nur in eine Gruppe und nicht in alle anderen. Bei der Fuzzy Mengenlehre können Produkte graduell in einem Intervall von $[0,1]$ zu mehreren Gruppen gehören.[186]

Analytic Hierarchy Process

Die Methode des Analytic Hierarchy Process (AHP) ist ein Verfahren zur Lösung von Entscheidungsproblemen. Jedes größere Problem kann in Subprobleme untergliedert werden. Diese Dekomposition wird durch AHP formalisiert. Die Entscheidungsprobleme sind Aufgabenstellungen mit mehreren Kriterien. AHP löst diese Komplexität durch einen paarweisen Vergleich zweier Handlungsalternativen. Das Ergebnis eines Vergleiches a_{ij} gibt an, wie stark der Entscheider die Alternative i der Alternative j vorzieht. Beurteilt wird das Ausmaß der Präferenz anhand von zuvor festgelegten Beurteilungskriterien. Via der Hierarchisierung wird durch das Vergleichsergebnis auch festgelegt, welche Bedeutung den beiden Kriterien i, j in Bezug auf das direkt übergeordnete Kriterium zugemessen wird. Die Beurteilungsskala geht von eins bis neun. Wobei eins eine Gleichwertigkeit der Alternativen i und j anzeigt. Drei und fünf stehen für eine etwas bzw. deutlich höhere Präferenz von i gegenüber j und sieben und neun für eine viel und sehr viel höhere Präferenz. Die

[185] Quelle: in Anlehnung an Arnold (2013), S. 21.
[186] Vgl. Sangwan/Kodali (2004), S. 293.

restlichen Zahlen sind Zwischenwerte. Will man eine Präferenz von j gegenüber von i ausdrücken, gibt man die Reziprokwerte an.[187]

Einer der Hauptkritikpunkte an AHP ist die Tatsache, dass die Methode nur Anwendung bei klar abgegrenzten Informationen findet. Diese Abgrenzung ist bei der Wiedergabe von Beurteilungen und Erfahrungen mittels natürlicher Sprache nicht immer eindeutig zu ziehen. Um dieser Unsicherheit der Grenze zu begegnen wird versucht AHP mit der Fuzzy Mengenlehre zu kombinieren.[188]

Anfangsbedingungen

Im behandelten Ansatz gibt es N Produkte welche zu C Familien zusammengefasst werden sollen. Der Zugehörigkeitsgrad $f_a(x)$ wird zu f_{ij} um die Zugehörigkeit und somit die Ähnlichkeit des j-ten Produktes zur i-ten Familie anzuzeigen. Formel 4.1 wird daher dem Problem entsprechend zu 4.5 erweitert.

$$\forall\, i,j \qquad 0 \leq f_{ij} \leq 1 \tag{4.5}$$

Es ergibt sich eine s.g. Klassifikationsmatrix A (Abbildung 29) mit den Zeilen welche die Gruppen darstellen und den Spalten, die die Produkte abbilden. Die Zeileneinträge sind die Zugehörigkeitsgrade laut 4.5.

$$A = \begin{array}{c} \\ G_1 \\ G_2 \\ \vdots \\ G_C \end{array} \begin{array}{c} \begin{array}{cccc} X_1 & X_2 & \dots & X_N \end{array} \\ \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1N} \\ u_{21} & u_{22} & \dots & u_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ u_{C1} & u_{C2} & \dots & u_{CN} \end{bmatrix} \end{array}$$

Abbildung 29: Klassifikationsmatrix[189]

Die Klassifikationsmatrix A hat zwei zentrale Eigenschaften. 4.6 ist eine spezielle Abänderung von 4.3 und besagt, dass die Summe jeder Zeile eins sein muss. Die Zugehörigkeitsfunktion einer jeden Produktfamilie ist folglich auf eins normiert. Da jedes Produkt wenigstens einer Familie zugeordnet werden muss, ist es erforderlich, dass wie durch 4.7 ausgedrückt, jede Spalte wenigstens einen Eintrag größer als null hat.[190]

$$\forall\, j \qquad \sum_{i=1}^{C} f_{ij} = 1 \tag{4.6}$$

$$\forall\, i \qquad \sum_{j=1}^{N} f_{ij} > 0 \tag{4.7}$$

[187] Vgl. Peters/Zelewski (2004), S. 298.
[188] Vgl. Sun (2010), S. 7746.
[189] Quelle: in Anlehnung an Sangwan/Kodali (2004), S. 293.
[190] Vgl. Sangwan/Kodali (2004), S. 293.

Modellumsetzung

Der Ansatz verwendet Designattribute der Produkte. Um Skalierungsprobleme zu vermeiden, werden relative Ähnlichkeitszugehörigkeiten der Produktattribute berechnet. Die Attribute werden mit einer Gewichtung versehen, die mittels AHP bestimmt wird. Dadurch wird eine Konsistenz der Gewichtungen gewährleistet. Der Planer weißt jedem Attribut eine Zahl zwischen 1 und 9 zu um die Wichtigkeit im Vergleich zu einem weiteren Attribut anzugeben. Der vorgeschlagene Algorithmus befolgt elf Schritte.[191]

1. **Eingabe:**
   Die Anzahl der Produkte und Attribute, sowie deren Werte muss bekannt gegeben werden.
2. **Zugehörigkeitsgrad 4.8:**
   Für jedes Attribut wird der Zugehörigkeitsgrad zwischen den zwei Produkten bestimmt.

$$f_{1-2}^{a} = 1 - \frac{(x_1 - y_2)}{(x - y)_{max}^{a}} \tag{4.8}$$

   x und y sind hierbei jeweils die Werte des Attributes a für Produkt 1 und 2.

3. **Zugehörigkeitsfunktion Attribute:**
   Bestimmung der Zugehörigkeitsfunktion für die Attribute.
4. **Gewichtungsfaktoren:**
   Mittels AHP sind die Gewichtungsfaktoren für jedes Attribut zu ermitteln.
5. **Zugehörigkeitsfunktion Gewichtungsfaktoren:**
   Festlegung der Zugehörigkeitsfunktionen für die Gewichtungsfaktoren.
6. **Entscheidungslogik:**
   Festlegung der „wenn-dann"-Bedingungen für die Entscheidungen welche das Handeln des Planers wiedergeben sollen.
7. **Zugehörigkeitsfunktion der Ähnlichkeit f_R^0 4.9:**
   Diese ergibt sich für jede Entscheidung aus dem Minimum einer Zugehörigkeitsfunktion einer jeden Eingabe.

$$f_R^g = \min\left\{f_{Wert\ Eingabe\ 1}^{g}, \dots, f_{Wert\ Eingbae\ 2}^{g}\right\} \tag{4.9}$$

8. **Ähnlichkeitskoeffizient R_0 4.10:**
   Bestimmung der Koeffizienten für die Ähnlichkeitsmatrix. R_0 ist die Ähnlichkeit eines Produktpaars. i bildet eine Regel ab, R ist die nummerische Bewertung und f_R^g entspricht dem minimalen Zugehörigkeitsgrad für die Regel.

$$R_0 = \frac{\sum_i f_R^g \times R}{\sum_i f_R^g} \tag{4.10}$$

9. **Maximale Ähnlichkeit:**
   Bestimme den maximalen Wert in der Ähnlichkeitsmatrix.

[191] Vgl. Sangwan/Kodali (2004), S. 294f.

10. **Vereinige Produkte, bzw. Produktgruppen:**
    Jene Produkte bzw. Produktgruppen (z.B. p', q') mit der maximalen Ähnlichkeit werden zu einer neuen Gruppe v vereinigt. Danach muss die Ähnlichkeitsmatrix aktualisiert werden, indem die Ähnlichkeitskoeffizienten zwischen der neuen Gruppe v und allen anderen Gruppen g oder Produkten bestimmt werden, wie in 4.11 formalisiert ist.

$$p \in v, q \in g \qquad S_{vg} = max\{S_{pq}\} \tag{4.11}$$

11. **Abschluss:**
    Wiederhole die Schritte 9 und 10 solange, bis alle Produkte einer Gruppe zugeteilt wurden.

Schlussbetrachtung

Das Modell wird mittels eines Beispiels bewertet, indem mehrere Produkte mit drei Attributen zu Gruppen aggregiert werden. Es werden drei Fälle durchgerechnet, die sich durch Gewichtungen der Attribute unterscheiden. Dabei zeigt sich, dass die Gewichtungen einen Einfluss auf die Gruppierungsreihenfolge haben.[192]

Die Vorteile der Vorgehensweise ergeben sich hauptsächlich aus den Eigenschaften und Vorteilen der Fuzzy Mengenlehre und von AHP. Es werden die einzelnen Produkte nicht von Beginn an in fest gelegte Gruppen gezwungen. Den Attributen können unterschiedliche Gewichtungen zugeordnet werden. Diese Zuordnung wird durch AHP konsistent durchgeführt. Skalierungsprobleme werden durch AHP vermieden. Darunter sind die möglichen, extremen Unterschiede bei z.B. geometrischen Maßen zu verstehen. Eine Längenangabe von einer Menge von Produkten kann von mehreren Millimetern zu einigen Metern reichen. Die Anzahl der Produktfamilien kann durch Änderungen von Ähnlichkeitsgrenzen beeinflusst werden.[193]

4.1.2 Produktionsplanung mit unsicherem Bedarf

Das folgende Modell ist ein Ansatz, welcher die Bedarfsunsicherheit der Produktionsplanung entlang der Versorgungskette mithilfe der Fuzzy-Mengenlehre abbilden will. Die Möglichkeiten der Fuzzytheorie eignen sich sehr gut, um Unsicherheiten aller Art abzubilden. Diese Eigenschaft liegt bereits im Wort Fuzzy, welches sich mit unscharf, undeutlich oder verschwommen übersetzen lässt.

Der Bedarf in jeder Stufe der Versorgungskette ist eine der häufigsten Quellen der Unsicherheit und soll mittels Fuzzy Mengenlehre in die Produktionsplanung integriert werden. Ausgangspunkt für die Bedarfsunsicherheit bildet in dem Modell eine Informationslücke über die Bedarfsdaten. Das Grundgerüst des Modells bildet ein gemischt-ganzzahliges Programm. Der betrachtete Zeithorizont beträgt ein bis zwei Jahre und ist daher ein taktisches Grobplanungsproblem. Es wird angestrebt die begrenzten Kapazitäten des Unternehmens optimal zuzuweisen, um den Kundenbedarf zufrieden zustellen. In der mathematischen Beschreibung werden nur die Zielfunktion mit den für

[192] Vgl. Sangwan/Kodali (2004), S. 294ff.
[193] Vgl. Sangwan/Kodali (2004), S. 301.

die Thematik dieser Arbeit wichtigen Nebenbedingungen erörtert. Dazu zählen die Nebenbedingungen, welche sich mit der Produktfamilienbildung beschäftigen, die begrenzte Kapazität abbilden und in diesem Fall jene die Fuzzyzahlen beinhalten.[194]

Mathematisches Modell

Wie für ein gemischt-ganzzahliges Programm üblich muss eine Zielfunktion unter der Berücksichtigung von mehreren Nebenbedingungen minimiert oder maximiert werden. Im Falle der Zielfunktion 4.12 wird eine Minimierung durchgeführt.

Dabei werden die Produktionskosten, die Kosten für Rohmaterialien, jene Kosten die für den Verbrauch von Zwischenprodukten anfallen, Strafkosten für Bestand und Unterversorgung, Transportkosten und die Fixkosten der Produktfamilien berücksichtigt.[195] Tabelle 5 gibt eine Auflistung der Bedeutung der Variablen der Zielfunktion.

Tabelle 5: Zielfunktionsvariablen Fuzzybedarf[196]

Variable	Bedeutung
P_{ijst}	Produktionsmenge von Produkt $i \in I \backslash I^{RM}$ auf Ressource j in Fabrik s in Periode t.
C_{ist}	Verbrauch an Rohmaterial oder Zwischenprodukt $i \in I \backslash I^{FP}$ in Fabrik s in Periode t.
I_{ist}	Bestand von Produkt $i \in I \backslash I^{RM}$ in Fabrik s am Ende von Periode t.
S_{isct}	Versorgung mit Fertigprodukten $i \in I^{FP}$ von Fabrik s für Kunden c in Periode t.
$\sigma_{is\acute{s}t}$	Fluss von Zwischenprodukten $i \in I^{IP}$ in Fabrik s in Periode t.
$I^{\wedge}_{ist}$	Abweichung unter Sicherheitsbestand für Produkt $i \in I$ in Fabrik s in Periode t.
I^{-}_{ict}	Fehlmenge von Fertigprodukt $i \in I^{FP}$ für Kunden c in Periode t.
Y_{fjst}	Binäre Variable welche anzeigt ob Produkt i auf Ressource j in Fabrik s in Periode t produziert wird.
v_{ijs}	Variable Kosten für die Produktion einer Einheit von Produkt i auf Ressource j in Fabrik s.
p_{is}	Preis von Rohmaterial $i \in I^{RM}$ in Fabrik s.
h_{ist}	Bestandskosten einer Einheit von Produkt i in Fabrik s in Periode t.
t_{sc}	Transportkosten einer Einheit der Produktes von Fabrik s zu Kunden c.
$t_{ss'}$	Transportkosten einer Einheit der Produktes von Fabrik s zu Fabrik s'.
ζ_{is}	Strafe für Unterdeckung des Sicherheitsbestandes von Produkt i in Fabrik s.
μ_{ic}	Erlös pro Einheit von Produkt i beim Verkauf an Kunden c.
f_{fjs}	Fixkosten für die Produktion von Familie f auf Ressource j in Fabrik s.

[194] Vgl. Mula et al. (2010), S. 136f.
[195] Vgl. Mula et al. (2010), S. 137.
[196] Quelle: in Anlehnung an Mula et al. (2010), S. 137.

$$\min \sum_{i,j,s,t} v_{ijs} P_{ijst} + \sum_{i,s,t} p_{is} C_{ist} + \sum_{i,s,t} h_{ist} I_{ist} + \sum_{i,s,c,t} t_{sc} S_{isct} + \sum_{i,s,s',t} t_{ss'} \sigma_{iss't} + \sum_{i,s,t} \zeta_{is} \hat{I}_{ist} + \sum_{i,c,t} \mu_{ic} I^{-}_{ict} + \sum_{f,j,s,t} f_{fjs} Y_{fjst} \tag{4.12}$$

Aus dem umfassenden Angebot von Nebenbedingungen werden u.a. jene mit Fuzzyzahlen exemplarisch erwähnt. Nebenbedingung 4.13 setzt ein Produkt mit seiner Produktfamilie in Beziehung, 4.14 beschreibt die Kapazitätsbeschränkungen der Produktion. Die weiteren drei Nebenbedingungen 4.15 bis 4.17 bedienen sich einer triangulären Fuzzyzahl um die Bedarfsunsicherheit zu modellieren. Gleichung 4.15 besagt, dass sich Fehlmengen in der Versorgung von einer Periode in die nächste auswirken. Nebenbedingung 4.16 erlaubt es, dass Bedarfe vergangener Perioden in der aktuellen Periode erzeugt werden. Die letzte Nebenbedingung 4.17 stellt sicher, dass der Bedarfsrückstand immer geringer ist als der gesamte kumulierte Bedarf der laufenden Periode.[197]

Tabelle 6: Nebenbedingungsvariablen Fuzzybedarf[198]

Variable	Bedeutung
$\boldsymbol{FRL_{fjst}}$	Produktionszeit von Familie f auf Ressource j in Fabrik s in Periode t.
$\boldsymbol{RL_{ijst}}$	Produktionszeit von Produkt $i \in I \backslash I^{RM}$ auf Ressource j in Fabrik s in Periode t.
$\boldsymbol{k_{if}}$	Binärer Parameter welcher anzeigt ob Produkt i zu Familie f gehört.
$\boldsymbol{H_{jst}}$	Verfügbare Produktionszeit auf Ressource j in Fabrik s in Periode t.
$\boldsymbol{\tilde{d}_{ict}}$	Fuzzy Bedarf an Fertigprodukt i für Kunden c in Periode t.

$$\forall \begin{pmatrix} f \in F, j \in J, \\ s \in S, t \in T \end{pmatrix} \quad FRL_{fjst} = \sum_{k_{if}=1} RL_{ijst} \tag{4.13}$$

$$\forall \begin{pmatrix} j \in J, \\ s \in S, t \in T \end{pmatrix} \quad \sum_{f} FRL_{fjst} \leq H_{jst} \tag{4.14}$$

$$\forall (i \in I^{FP}, t \in T) \quad I^{-}_{ict} \geq I^{-}_{ic(t-1)} + \tilde{d}_{ict} - \sum_{c} S_{isct} \tag{4.15}$$

$$\forall (i \in I, t \in T) \quad \sum_{s,t \leq t} S_{isct} \leq \sum_{t \leq t} \tilde{d}_{ict} \tag{4.16}$$

$$\forall (i \in I, c \in C) \quad I^{-}_{ict} \leq \sum_{t \leq t} \tilde{d}_{ict} \tag{4.17}$$

[197] Vgl. Mula et al. (2010), S. 137f.

[198] Quelle: in Anlehnung an Mula et al. (2010), S. 137.

Fuzzy Modell

Das Modell wird im Rahmen der Supply Chain Produktionsplanung verwendet. Mithilfe triangulärer Zugehörigkeitsfunktionen werden der Informationsmangel und die Unsicherheit im Bedarf abgebildet. Es wird ein fuzzy-lineares Programm mit scharfen - also nicht fuzzy - Zielbedingung und Fuzzy-Nebenbedingungen in ein reines lineares Programm mit scharfen Parametern transformiert. Die Gleichungen 4.18 bis 4.20 bilden das noch nicht transformierte Ausgangsmodell entsprechend der allgemeinen Struktur eines linearen Programms, wie in 3.1 und 3.2 beschrieben. Die Zielfunktion 4.18 setzt sich aus der j-ten Entscheidungsvaribalen x_j mit dem scharfen Koeffizienten c_j zusammen. Die beiden Nebenbedingungen 4.19 und 4.20 bauen sich aus dem scharfen Koeffizienten a_{ij} für die i-te Nebenbedingung der j-ten Entscheidungsvariabeln und der triangulären Zugehörigkeitsfunktion $\tilde{b}_i$ auf. $\tilde{b}_i$ besteht aus den drei Fuzzyzahlen b_{i1}, b_{i2} und b_{i3}. Die parametrische Darstellung ist 4.4 und die graphische Visualisierung Abbildung 28 zu entnehmen.

$$\min \sum_{j=1}^{n} c_j x_j \tag{4.18}$$

$$i = 1,2, \dots, m_1 \qquad \sum_{j=1}^{n} a_{ij} x_j \geq \tilde{b}_i \tag{4.19}$$

$$i = m_1 + 1, \dots, m_2 \qquad \sum_{j=1}^{n} a_{ij} x_j \leq \tilde{b}_i \tag{4.20}$$

Zur Lösung des Fuzzy-Problems wird die Zielfunktion 4.18 unter den transformierten Nebenbedingungen 4.21 und 4.22 minimiert. Der Koeffizient α ist ein Schwellenwert der parametrisch festgelegt werden muss. Er kann einen Wert im Intervall [0,1] annehmen. Das Ergebnis ist eine Fuzzy Menge. Es obliegt der Entscheidung des Planers, welches Wertepaar (α, z), wobei z die Zielfunktion ist, er als optimal ansieht unter dem ein Ergebnis mit scharfen Zahlen zu erhalten ist. Für einen Wert von $\alpha = 1$ liegt ein gemischt ganzzahliges Programm vor.[199]

$$i = 1, \dots, m \qquad \sum_{j=1}^{n} a_{ij} x_j \geq (1 - \alpha) b_{i1} + \alpha b_{i2} \tag{4.21}$$

$$i = 1, \dots, m \qquad \sum_{j=1}^{n} a_{ij} x_j \leq (1 - \alpha) b_{i3} + \alpha b_{i2} \tag{4.22}$$

[199] Vgl. Mula et al. (2010), S. 138f.

Übertragen auf das mathematische Modell von oben, werden die Nebenbedingungen 4.15 bis 4.17 in die folgenden 4.23 bis 4.25 transformiert. Dabei wird 4.23 in eine Form gebracht, damit auf der rechten Seite der Ungleichung die Fuzzykoeffizienten stehen.[200]

$$\forall(i \in I^{FP}, t \in T) \qquad I_{ict}^{-} - I_{ic(t-1)}^{-} + \sum_{c} S_{isct} \geq (1-\alpha)d_{ict1} + \alpha d_{ict2} \tag{4.23}$$

$$\forall(i \in I, t \in T) \qquad \sum_{s,\acute{t} \leq t} S_{isc\acute{t}} \leq \sum_{\acute{t} \leq t} \left((1-\alpha)d_{ic\acute{t}3} + \alpha d_{ic\acute{t}2}\right) \tag{4.24}$$

$$\forall(i \in I, c \in C) \qquad I_{ict}^{-} \leq \sum_{\acute{t} \leq t} \left((1-\alpha)d_{ic\acute{t}3} + \alpha d_{ic\acute{t}2}\right) \tag{4.25}$$

Modellbewertung

Das Verfahren wurde an einem Beispiel der chemischen Industrie getestet. Dabei wurden zwei Standorte mit unterschiedlichem Produktportfolio einbezogen. Die Produktion des zweiten Standortes hing dabei zum Teil von den Produkten des ersten Standortes ab. Der Planungshorizont betrug zwölf Monate. Der unsichere Bedarf wird in Form einer triangulären Zugehörigkeitsfunktion berücksichtigt. Entsprechend Abbildung 28 wird P_2 der wahrscheinlichste Wert des Bedarfs mit dem Monatsschnitt angenommen. Der pessimistischste Wert P_1 ist 80% von P_2 und der optimistischste Wert P_3 wird mit 120% von P_2 angegeben. Weiters wird angenommen, dass der Bedarf für Periode t am Beginn dieser Periode bekannt ist. Somit ergeben sich die analogen Bezeichnungen für P_1, P_2 und P_3 folglich mit d_{ict1}, d_{ict2} und d_{ict3}. Der Schwellenwert α wird in 0,1 Inkrementen erhöht. Als Ergebnis liefert der Vergleich, dass der Fuzzyansatz (FMILP) im Vergleich zum reinen linearen Programm (MILP) zwar nur ein Servicelevel von 98% liefert, das MILP hingegen 100%, jedoch mit einem höheren Bestand. Die Planungsnervosität ist beim FMILP gleich oder geringer als beim MILP und die Gesamtkosten sind bei jedem Wert für α geringer als das Ergebnis des MILP. Speziell die geringeren Gesamtkosten sind auf die Einbindung der Unsicherheit im Bedarf zurückzuführen, da Bestände gezielt gehalten werden können. Eventuell hohe Bestandskosten und bei Unterversorgung folgende Produktionsausfälle können vermieden werden. Die Kapazitätsauslastung ist je nach α Wert unterschiedlich, jedoch in den meisten Fällen höher als beim MILP. Das wird durch die unterschiedlichen Entscheidungen der Planer erklärt, die andere, meist bessere Entscheidungen treffen falls sie die Bedarfsunsicherheit in ihre Planungsmodelle integrieren können.[201]

[200] Vgl. Mula et al. (2010), S. 139.

[201] Vgl. Mula et al. (2010), S. 139ff.

4.2 Mulitvariate Methoden

Multivariate Analyseverfahren sind statistische Methoden, welche eine Menge von Objekten anhand mehrdimensionaler Merkmalsausprägungen analysieren. Es werden jeweils zwei Objekte in Form eines paarweisen Vergleichs hinlänglich der verschiedenen Merkmalsausprägungen analysiert. Dadurch unterscheiden sie sich von den s.g. uni- und bivariaten Verfahren, welche nur ein oder zwei Merkmal(e) im paarweisen Vergleich berücksichtigen.[202]

Generell können die multivariaten Methoden in strukturprüfende und strukturentdeckende Verfahren unterteilt werden. Die Strukturprüfenden dienen der Überprüfung von Zusammenhängen von Merkmalsausprägungen. Es wird die Abhängigkeit einer s.g. abhängigen Variablen von einer oder mehreren unabhängigen Variablen geprüft. Bekannte Verfahren sind die Regressionsanalyse, die Varianz- und Diskriminanzanalyse oder die logistische Regression. Die Strukturentdeckenden haben das Ziel Zusammenhänge von Variablen oder zwischen Objekten zu finden. Bekannte Verfahren sind die Faktorenanalyse, die Clusteranalyse oder auch neuronale Netze.[203] In dieser Arbeit kommt die Clusteranalyse zum Einsatz und wird folglich allgemein und dann anhand eines Beispiels vorgestellt.

4.2.1 Clusterverfahren

Das Ziel der Clusterverfahren ist es eine Gesamtheit von Objekten $O = \{O_1 \dots O_m\}$ eindeutig Gruppen $C_1 \dots C_p$ zuzuordnen. Wobei die Objekte einer Gruppe möglichst homogen, die Gruppen untereinander möglichst heterogen sein sollen. Generell lassen sich die Clusterverfahren wie in Abbildung 30 ersichtlich, in vier Unterkategorien gliedern. Es werden weiter die hierarchischen Verfahren und hier die agglomerativen beschrieben. Bei diesen wird von der feinsten Partition ausgegangen. Es liegen also alle Objekte der Menge einzeln als eigenständige Cluster vor. Mit jedem Gruppierungsschritt werden die jeweils ähnlichsten Objekte und im Laufe des Verfahrens Cluster vereinigt. Divisive Verfahren starten mit der gröbsten Partition, wenn alle Objekte in einem Cluster vorliegen, und teilen bei jedem Schritt einen Cluster in zwei kleinere Cluster auf.[204]

[202] Vgl. Bohnen/Deuse (2000), S. 20.
[203] Vgl. Backhaus et al. (2011), S. 13f.
[204] Vgl. Handl (2010), S. 374.

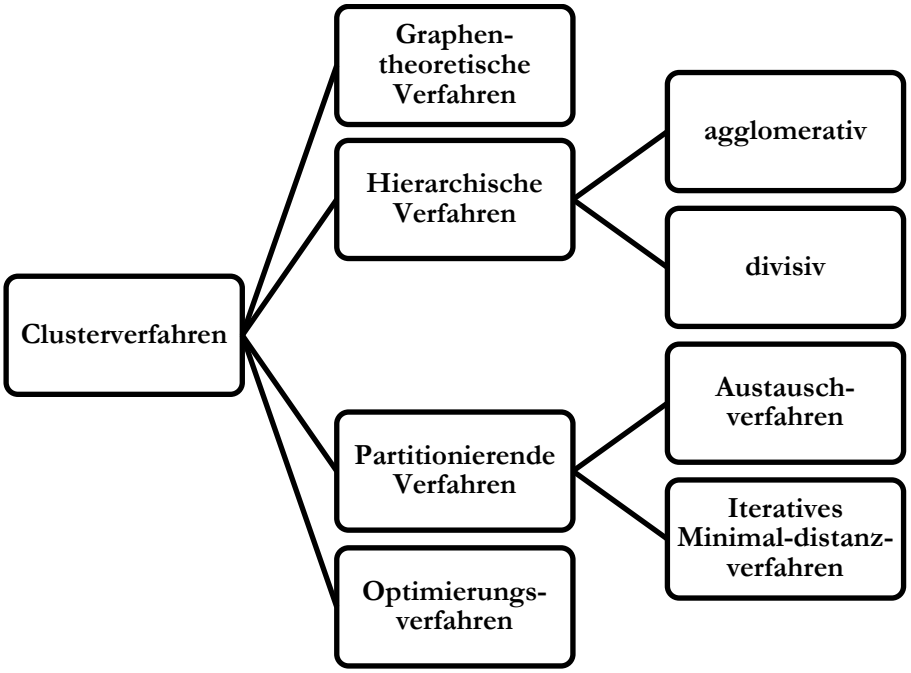


Abbildung 30: Unterteilung des Clusterverfahren[205]

Die Ähnlichkeit, bzw. Unähnlichkeit von Objekten oder Clustern werden mittels entsprechender Proximitätsmaße bestimmt. Welches Maß zum Einsatz kommt hängt sehr stark von der Skalierung der Merkmalsausprägung ab.[206]

Skalierungen

Die Art der Messskala einer Merkmalsausprägung gibt bekannt, wie eine Variable gemessen wurde. Es gibt vier Arten dieser Skalierung.

- Nominalskala
- Ordinalskala
- Intervallskala
- Ratioskala

Die Nominalskala teilt die Daten in Gruppen ein. Beispiele sind das Geschlecht, Farben, Religions- oder Staatszugehörigkeit. Eine spezielle Form ist die Binärskala, wo eine Eigenschaft vorhanden ist oder nicht vorhanden ist. Metrische Rechenoperationen wie Addition oder Subtraktion sind mit diesen Skalen nicht möglich. Die Ordinalskala stellt die Objekte in eine Rangfolge. Also das Produkt A besser ist als Produkt B und dieses wiederum besser als Produkt C. Es kann jedoch keine Auskunft über die Distanz und daher das Ausmaß des Besser seins gegeben werden. Die Intervallskalen lassen eine Quantifizierung des Ausmaßes, welches in der Ordinalskala fehlt, zu. Ein Beispiel sind Temperaturskalen. Es fehlt jedoch ein definierter Nullpunkt. Die Ratioskala hat einen definierten Nullpunkt. Es existiert eine Reihenfolge und eine Quantifizierungsmöglichkeit

[205] Quelle: in Anlehnung an Backhaus et al. (2011), S. 418.
[206] Vgl. Backhaus et al. (2011), S. 399.

des Ausmaßes des Abstandes zweier Bewertungen. Ein Beispiel sind Längen- oder Gewichtsangaben.[207]

Proximitätsmaße

Besteht Klarheit über die Skalierung der Daten, kann eine geeignete Auswahl der Ähnlichkeitsbestimmung der Daten erfolgen. Dabei kann genau genommen die Ähnlichkeit aber auch die Unähnlichkeit der Daten bestimmt werden. Distanzmaße für die Unähnlichkeit werden verwendet, wenn der absolute Abstand zweier Objekte von Interesse ist. Ähnlichkeitsmaße kommen zur Anwendung, wenn der Lauf zweier Profile verglichen werden soll. Die Darstellung der Distanz oder Ähnlichkeit erfolgt in Form einer symmetrischen Matrix, der Distanzmatrix.[208] Tabelle 7 zeigt eine, nach Skalen geteilte, Übersicht von Proximitätsmaßen.

Tabelle 7: Ausgewählte Proximitätsmaße[209]

	Skalenniveau der Merkmalsausprägungen		
	ratio (metrisch/Intervall)	**nominal (Häufigkeiten)**	**binär (0/1)**
Ähnlichkeitsmaß	• Cosinus von Vektoren • Q-Korrelationskoeffizient		• Würfelmaß • Jaccard-Koeffizient • M-Koeffizient • RR-Koeffizeint
Distanzmaß	• Euklidische Metrik • Block Metrik • Tschebycheff Metrik • Malahanobis Metrik	• Chi-Quadrat-Maß • Phi-Quadrat-Maß	• Varianz • Größendifferenz • Lance-Williams-Maß • Binäres Euklidisches Distanzmaß

Während die angeführten Distanzmaße ihre Anwendung bei der Ratio- und Intervalskala finden, kommen die Ähnlichkeitsmaße bei Nominal- und Ordinalskalen zu Anwendung. Die Distanzmaße bestimmen die Unähnlichkeit zweier Objekte anhand geometrischer Überlegungen. Betrachtet man eine ebene Fläche, mit Punkten als zu gruppierende Objekte, so wären die kartesischen Koordinaten die beiden Merkmalsausprägungen. Die Block Metrik würde die Summe der beiden orthogonalen Abstände als Distanz der beiden Punkte verwenden, während die euklidische Metrik die Diagonale zwischen den Punkten heranzieht. Welches Maß ein Planer für die Beurteilung verwendet hängt vom Skalenniveau und der Anwendung ab. [210]

Clusterverfahren

Wenn die Startdistanzen oder Ähnlichkeiten der Objekte mittels des geeigneten Proximitätsmaßes bestimmt wurden, erfolgt die Bildung der Cluster. Dafür werden die

[207] Vgl. Backhaus et al. (2011), S. 10f.
[208] Vgl. Handl (2010), S. 83f.
[209] Quelle: in Anlehnung an Backhaus et al. (2011), S. 401.
[210] Vgl. Backhaus et al. (2011), S. 402ff.

verschiedenen Clusterverfahren verwendet. Diese bestimmen wie die neuen Distanzen zwischen des durch den aktuellen Gruppierungsschrittes gebildeten Clusters und den anderen Objekten, oder Cluster, berechnet werden. Eine weitere Berechnung durch die Proximitätsmaße wird nicht mehr vorgenommen. Tabelle 8 gibt eine Übersicht von ausgewählten Clusterverfahren. Die Berechnung erfolgt allgemein laut Formel (**4.26)**, wobei die Koeffizienten A, B, E und G für die einzelnen Verfahren Tabelle 8 zu entnehmen sind. Die Variablen $D(R,P)$, $D(R,Q)$ und $D(P,Q)$ sind jeweils die Distanzen zwischen den Clustern R und P, R und Q und folglich P und Q. Die Variablen NR, NP und NQ aus Tabelle 8 sind die Anzahl der Objekte in Cluster R, P bzw. Q.[211]

$$D(R;P+Q) = A \cdot D(R,P) + B \cdot D(R,Q) + E \cdot D(P,Q) + G \cdot |D(R,P) - D(R,Q)| \tag{4.26}$$

Tabelle 8: Clusterverfahren mit Berechnung[212]

Cluster-verfahren	Konstanten lt. 4.26 A	B	C	G
Single Linkage	0,5	0,5	0	-0,5
Complete Linkage	0,5	0,5	0	0,5
Ward	$\frac{NR+NP}{NR+NP+NQ}$	$\frac{NR+NQ}{NR+NP+NQ}$	$-\frac{NR}{NR+NP+NQ}$	0
Average Linkage (ungewichtet)	0,5	0,5	0	0
Average Linkage (gewichtet)	$\frac{NP}{NP+NQ}$	$\frac{NQ}{NR+NP+NQ}$	0	0

Der Clustervorgang kann gestoppt werden, wenn die gewünschte Anzahl von Gruppen erreicht wurde, oder die Distanz zwischen den Objekten innerhalb der Cluster zu stark zunimmt, also die Homogenität zu stark abnimmt.[213]

4.2.2 Gruppenbildung mittels Clusterung und genetischer Algorithmen

Die Anwendung von Clusteralgorithmen setzt voraus, dass die Merkmalsausprägungen oder Gruppierungskriterien, die zur Gruppenbildung herangezogen werden, bekannt sind. Die Auswahl der Kriterien erfolgt meist durch Erfahrung des Planers. Sollte diese Erfahrung jedoch fehlen, sich die Produktstruktur häufig ändert oder es gewünscht ist die

[211] Vgl. Backhaus et al. (2011), S. 420ff.
[212] Quelle: in Anlehnung an Backhaus et al. (2011), S. 422.
[213] Vgl. Handl (2010), S. 407ff.

Erfahrungswerte zu überprüfen, dann ist es nötig eine Alternative zu finden, die diese Entscheidung dem Menschen abnimmt. Eine Möglichkeit die versucht die intuitiven Entscheidungen von Menschen nachzubilden sind evolutionäre Algorithmen. In Folge werden daraus die genetischen Algorithmen vorgestellt.

Genetische Algorithmen

Genetische Algorithmen versuchen die treibenden Prozesse der Evolution als Computerprogram auf verschiedene Probleme umzulegen. Diese treibenden Prozesse sind Selektion, Kreuzung und Mutation. Durch die Ermittlung einer Startpopulation von Individuen erhält man einen Paarungspool. Die Kreuzung von Individuen des Paarungspools erzeugt neue Individuen. Diese werden auf ihre Fitness geprüft und die besten durch die Selektion in einen neuen Paarungspool übernommen. Der Vorgang der Kreuzung, Bewertung und Selektion wiederholt sich. In zufälligen Abständen werden die Individuen des Paarungspools vor der weiteren Kreuzung mittels Mutation leicht verändert, um die genetische Diversität zu simulieren und dadurch ggf. bessere Lösungen zu erzeugen. Wenn sich die Fitness von einer Populationen im Vergleich zu Vorgängerpopulation nach mehreren Durchläufen nicht mehr oder nur mehr sehr gering verbessert wurde eine maximal fitte Population gefunden. Diese oder das beste Individuum daraus ist die gesuchte Lösung des Problems.[214]

Im folgenden Modell wird die fitnessproportionale Selektion angewandt. Dabei kommen fitte Individuen öfter in den Paarungspool als weniger fitte. Die Gesamtfitness einer Population wird in die Teilfitnessblöcke der einzelnen Individuen unterteilt. Nun wird eine zufällige Zahl zwischen null und der Gesamtfitness erzeugt. Beginnend mit dem Fittesten Individuum werden die Fitnesswerte kumuliert. Jenes Individuum welches mit seinem Beitrag die Zufallszahl überschreitet kommt in den Paarungspool. Der Vorgang wird solange wiederholt, bis die gewünscht Zahl von Individuen im Paarungspool ist. Durch die absteigende Sortierung der Individuen wird sicher gestellt, dass fittere öfters in den Paarungspool kommen. Die Gesamtfitness kann auch auf eins skaliert werden.[215]

Als Kreuzungsverfahren wird eine n-Punkt Kreuzung angewandt. Dabei werden zufällig n Punkte in der Bitstruktur gewählt. Die Information zwischen diesen Punkten wird unter den beiden Elternteilen ausgetauscht und es entstehen zwei neue Individuen für den Paarungspool der nächsten Generation. Abbildung 31 zeigt ein Beispiel einer 2-Punkt Kreuzung. Die Informationen zwischen den beiden Kreuzungspunkten KP1 und KP2 werden dabei vertauscht. Diese einfache Form der n-Punkt Kreuzung ist jedoch nur bei einer binären Kodierung möglich.[216]

[214] Vgl. Scheithauer (2008), S. 138f.
[215] Vgl. Weicker (2007), S. 71f.
[216] Vgl.ebd., S. 83ff.

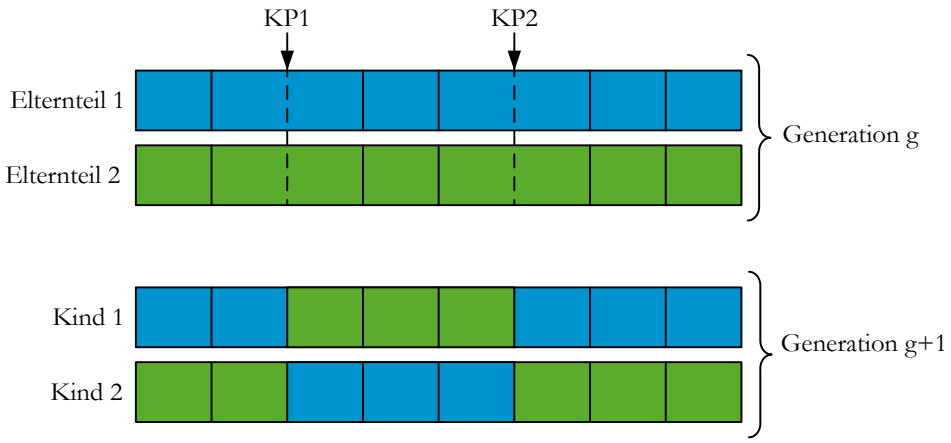


Abbildung 31: 2-Punkt Kreuzung[217]

Das gewählte Mutationsverfahren ist eine binäre Mutation. Dabei wird mit einer Mutationswahrscheinlichkeit P_m das Bitmuster invertiert. Aus allen Einsen werden Nullen und aus allen Nullen werden Einsen. Eine Mutation erfolgt, um zu verhindern, dass sich der Algorithmus in ein lokales Optimum festfährt. Es besteht jedoch die Gefahr, dass zu häufige Mutationen den Genpool so oft stören, dass ein erfolgreicher Weg zu einem globalen Optimum erschwert oder verhindert wird. Daher soll P_m nicht zu hoch angesetzt werden.[218]

Modellumsetzung

Zur Umsetzung der Clusterung zur Produktfamilienbildung und der genetischen Algorithmen müssen allgemeine Terminologien auf die speziellen Verhältnisse umgelegt werden. Tabelle 9 gibt eine Übersicht der Ausdrücke der Gruppentechnologie umgelegt auf die Evolutionstheorie.

Tabelle 9: Terminologien Evolution vs. Gruppentechnologie[219]

Evolutionstheorie	Produktfamilienbildung
Individuum	Teil, Produkt
Spezies	Produktfamilie
Gene	Produkteigenschaften
Überlebenseigenschaften	Unterscheidungsmerkmale der Produktfamilien
Überlebenserfolg	Erfolg in der Familienbildung welcher die Homogenität erhöht.

[217] Quelle: Eigene Darstellung
[218] Vgl. Weicker (2007), S. 58ff.
[219] Quelle: in Anlehnung an Lee-Post (2000), S. 798.

Die Produkteigenschaften werden in Form eines Codierungsschemas abgebildet. Vom Planer werden alle infrage kommenden Eigenschaften festgelegt und in eine Codierungsstruktur gebracht. Werden exemplarisch die Form, allgemeine Eigenschaften, Größe, Genauigkeit und das Material verwendet, so besteht die Struktur aus fünf Stellen. Diese werden als Bitmuster dargestellt, wobei 1 bedeutet, dass die Eigenschaft berücksichtigt wird und 0, dass die Eigenschaft nicht berücksichtigt wird. Ein Bitmuster wie ‚00110' würde folglich bedeuten, dass die Größe und die Genauigkeit als Gruppierungsattribute berücksichtigt werden.

Die Fitnessevaluation beruht auf der Ähnlichkeit der Produkte in der Familie bezogen auf deren gewählte Eigenschaften. Formel 4.27 zeigt die Berechnung der Ähnlichkeit S_{ij} zwischen zwei Produkten i und j. S_{ijk} ist die Ähnlichkeit von Attribut k bei Produkt i und j und K ist die Anzahl der betrachteten Eigenschaften, welche der Anzahl der Einsen im Bitmuster entspricht.[220]

$$S_{ij} = \frac{\sum_{k=1}^{K} S_{ijk}}{K} \tag{4.27}$$

Algorithmus

1. Initialisiere den Generationenzähler $g = 0$.
2. Initialisiere die Startpopulation $P(g)$. Erzeuge dazu eine Anzahl S von zufällig erstellten Bitmustern, ab hier als Indiviuum bekannt.
3. Bewerte $P(g)$.
    a. Für jedes Individuum erzeuge die Produktfamilien mittels eines Clusterverfahrens.
    b. Berechne die Fitness $f(s)$ eines jeden Individuums.
    c. Summier $f(s)$ zu einer Gesamtfitness $f_g(s)$ der Population.
    d. Sortiere die Individuen absteigend nach $f(s)$.
4. Erzeuge eine neue Population $P(g+1)$.
    a. Erstelle den Paarungspool durch eine fitnessproportionale Selektion.
    b. Erzeuge neue Individuen durch eine 2-Punkt Kreuzung.
    c. Mit P_m wende eine binäre Mutation auf alle neuen Individuen an.
5. Setzte $P(g) = P(g+1)$
    a. Wiederhole die Schritte 3 bis 5 solange bis die Abbruchkriterien erfüllt wurden.

Der Algorithmus startet mit der Erstellung einer Initialpopulation $P(g)$. Diese ist S groß und besteht aus zufällig erstellten Bitmustern. Für jedes Bitmuster wird mittels Average-Linkage eine Anzahl von N Produktfamilien gebildet. Es wird die Fitness $f(s)$ berechnet und alle Fitnesswerte werden zu einem Gesamtwert $f_g(s)$ der Population addiert. Die Werte $f(s)$ werden absteigend sortiert.

Mittels Selektion wird ein Paarungspool erstellt. Die fittesten Individuen werden dafür öfters gewählt. Aus dem Pool werden zufällig zwei Individuen gewählt und mittels

[220] Vgl. Lee-Post (2000), S. 799ff.

Kreuzung neue Individuen für die neue Population erzeugt. Mit der vorab festgelegten Wahrscheinlichkeit P_m werden alle Stellen des Bitmusters im Rahmen der Mutation geändert. Die Selektion und Kreuzung ist zu Ende, sobald die neue Population die idente Anzahl von Bitmustern wie die Initialpopulation hat. Der Vorgang der Produktfamilienbildung, Fitnessbewertung, Selektion, Kreuzung und Mutation werden solange wiederholt, bis die Abbruchkriterien erfüllt wurden. Diese sind entweder eine vorgegebene Anzahl von Programmdurchläufen, auch Generationen genannt, oder ein Schwellwert der Fitnessverbesserung der unterschritten wird.[221]

Modellbewertung

Das Modell wurde bei einem Produzenten von Energieübertragungsausrüstung angewandt. Der Produzent hatte acht Grundprodukte in 104 verschiedenen Ausführungen in seinem Portfolio. Diese unterschieden sich in zehn bis fünfzehn Eigenschaften. Die Auswertung des Modells ergab, dass die Homogenität der Produktfamilien bis zu 99% betrug. Durch die Berechnung von allen möglichen Kombinationen von Teileeigenschaften wurde eine schlechteste Lösung erstellt. Die Lösung des genetischen Ansatzes mittels Clusterung war im Schnitt um 33% besser als dieser schlechteste Fall. Da der Aufwand mit der Anzahl der Produkteigenschaften exponentiell steigt, ist die Rechenzeit zur Ermittlung aller und somit der schlechtesten oder besten Lösung, für die meisten industriellen Anwendungen zu lange.[222]

Es bleibt festzuhalten, dass die Clustermethoden mit genetischen Algorithmen eine geeignete Lösung für die Produktfamiliengenerierung bildet. Die ermittelten Gruppen sind sehr homogen und es entfällt mit der Auswahl der Gruppierungseigenschaften eine erhebliche, manuelle Arbeit, die vielfach nur mittels viel Erfahrung durchgeführt werden kann. Durch den evolutionären Gedanken führt das Modell schnell zu einer sehr guten Lösung. Die Optimalität könnte nur mittels einer Auswertung aller Kombinationen festgestellt werden.

Das folgende Kapitel beschreibt ein Verfahren zur robusten Produktfamilienbildung anhand eines Fallbeispiels. Es kommt ein Ansatz mittels Clusterverfahren in Kombination mit genetischen Algorithmen, ähnlich dessen von LEE-POST, zur Anwendung. Die Beurteilung der Fitness und der Robustheit wurde der Situation angepasst.

[221] Vgl. Lee-Post (2000), S. 798f.
[222] Vgl. ebd., S. 804ff.

5 Robuste Produktfamilienbildung anhand eines Fallbeispiels

Im folgenden Kapitel sollen die theoretisch erarbeiteten Gebiete der vorhergehenden Kapitel verwendet werden, um einen Algorithmus zu entwickeln, welcher mit wenig Zutun des Anwenders robuste Produktfamilien bildet. Aus jeder Produktfamilie wird ein sogenannter Planartikel gewählt, welcher die Eigenschaften der Produkte in der Familie möglichst gut abbildet. Mit diesem Artikel wird stellvertretend die Kapazitätsbedarfsermittlung durchgeführt. Als Datengrundlage werden die Auftrags- und Arbeitsplandaten der AMAG rolling GmbH, ab hier als AMAG geführt, herangezogen. Die relevanten Daten und Informationen über die Abläufe der AMAG wurden bei Besprechungen mit in diesen Feldern tätigen Mitarbeiten gesammelt. Dabei sind Herr Christian Grubmüller, Herr Hans Langgartner, Herr Heinz Hopfgartner und Herr Christoph Lobe zu erwähnen.

Zu den Kernkompetenzen der AMAG zählen Flexibilität, ein hoher Qualitätsstandard und die technologische Kompetenz in der Entwicklung von hochfesten Legierungen und dem Walzplattieren von Glanz- und Luftfahrprodukten. Am Standort in Ranshofen werden aus allen Legierungsfamilien mehr als 100 verschiedene Legierungen produziert. Durch einen massiven Ausbau der Produktionsstätte bis zum Jahr 2014 wird die Kapazität um das ca. eineinhalbfache erhöht. [223] Diese Punkte waren ausschlaggebend diese Arbeit zu verfassen.

Das Kapitel gliedert sich in drei Unterkapitel, wobei sich das erste mit dem derzeitigen Ablauf der Planung beschäftig, das zweite die Entwicklung der Methodik der robusten Produktfamilienbildung behandelt und das letzte genauer auf die zentralen Punkte eingeht.

5.1 Stand des Planungsablaufes

Die Kapazitätsgrobplanung der AMAG gliedert sich in die Bereiche Datenerhebung, Produktfamilienbildung, Bedarfsermittlung und Terminierung. Die eingesetzten Programme sind MS Access mit einer ODBC Verbindung zu einem SQL-Server, MS Excel und MATLAB.

Die Auftragsdaten, sowie die Daten des Arbeitsplans, sind zentral in einer SQL-Datenbank gespeichert. Für eine benutzerfreundlichere Handhabung werden diese mittels MS Access abgerufen und in MS Excel exportiert, wo die weitere Planung erfolgt.

MATLAB wird zur Produktfamilienbildung eingesetzt. Es dient zu numerischen Berechnungen, zur Datenanalyse und vielseitigen Visualisierung der Ergebnisse. Ein großer Umfang mathematischer Funktionen ist enthalten, welche durch Zusatzpakete erweitert werden können. Das erlaubt eine schnelle Lösungsfindung, welche mit herkömmlichen Programmiersprachen, wie Java, C/C++ oder einfachen Tabellenkalkulationsprogrammen, nicht möglich ist. Diese Programme können jedoch als Ein- oder Ausgabemedien verwendet werden. Das Einsatzgebiet geht von der Modellerstellung und Simulation über

[223] Vgl. AMAG (2014)

die Signalverarbeitung und Regelungstechnik hin zur Bild- und Videoverarbeitung. Eingesetzt wird es in der Forschung, Lehre und Industrie.[224]

5.1.1 Stand der Kapazitätsgrobplanung

Zentraler Drehpunkt der Kapazitätsgrobplanung ist eine umfangreiche Exceldatei, welche alle benötigten Daten zusammenfasst. Sie beinhaltet die Auftrags- und Arbeitsplandaten, die Absatzplanung des laufenden Jahres und der kommenden fünf Jahre, den Vormaterialbedarf für die Gießerei sowie die aggregierten Mengen der Produktfamilien und Belegungszeiten der Anlagen.

Neben den Produktfamilien bilden die strategischen Geschäftseinheiten SGEs eine weitere zentrale Ebene der Grobplanung. Sie hat die größte Aggregationsdichte, da hier alle Werte der Produktfamilien nochmals zusammengefasst werden. SGEs sind Teilbereiche, welche innerhalb des Unternehmens unabhängig von anderen agieren können. Sie repräsentieren eigenständige Marktaufgaben, sollen sich von der Konkurrenz abgrenzen und in sich homogen und zu anderen SGEs heterogen sein. Es wird ein eigenes Rechnungswesen für die SGEs verwendet, damit ihr Erfolg individuell den Aufgaben angepasst, beurteilt und mit anderen verglichen werden kann. Die Führung einer SGE verfügt in einem gewissen Rahmen über Vollmachten, die ihr die Möglichkeit geben, das Agieren auf dem Markt eigenständig zu gestalten. Dadurch sollen die Anforderungen der Kunden oder Kundengruppen zufriedenstellend erfüllt werden.[225]

In der AMAG wurden zur Zeit der Erstellung der Arbeit 23 SGEs geführt, wobei die geplanten SGEs, die durch den Ausbau des Standortes hinzu kommen, nicht eingerechnet wurden. Der Artikelstamm der Auftragsdaten enthält mehrere tausend individuelle Artikel, welche jeweils eindeutig einer SGE zugeordnet sind. Diese Artikel werden anhand von zehn bis fünfzehn Merkmalen zu Produktfamilien zusammengefasst, wobei jede dieser Produktfamilien durch den Planartikel in der weiteren Planung repräsentiert wird. Der Planartikel vereint das gesamte Bestellgewicht aller Artikel der Produktfamilie auf sich. In der Planung wird die Annahme getroffen, dass er entsprechend oft produziert wird, um das Bestellgewicht zu erreichen. Somit belastet er, stellvertretend für die anderen Artikel, die Kapazität der involvierten Produktionsanlagen. Die verwendete Kapazitätsgröße sind die zur Verfügung stehenden Planstunden der Anlage. Die Werte der Fertigtonnen des Liefergewichts und der dafür benötigten Planstunden werden für jede Anlage zusammengefasst. Daraus ergibt sich die Anlagenauslastung für das aktuelle Planjahr und mittels der hochgerechneten Prognosewerte für die folgenden fünf Jahre. Die Prognosewerte ergeben sich aus den Bedarfsprognosen. Die zukünftige Bedarfsermittlung ist eine Kombination aus deterministischen und heuristischen Verfahren, wie in Kapitel 2.1.1 erwähnt wird. Für jede Anlage kann eine Aufschlüsselung nach SGE und weiter nach Planartikel erfolgen sowie eine Untergliederung in die produzierten Fertigtonnen und die verbrauchten Planstunden.

[224] Vgl. MathWorks (2014)
[225] Vgl. Syska (2006), S. 140f.

Arbeitsplan

Der Arbeitsplan enthält für jeden Artikel die produktionsrelevanten Werte. Er besteht aus den Arbeitsvorgängen (AVO), die der Reihe nach abgearbeitet werden, um einen Artikel herzustellen. Ein AVO hat eine fortlaufende Nummer, eine Anlage, dargestellt durch die entsprechende Anlagennummer, auf der der Produktionsschritt ausgeführt wird und weitere technische Spezifikationen. Zu diesen zählen die Arbeitszeit je Los (ARBLO) und das Plangewicht, mit dem der Auftrag den AVO startet. In der Regel verringert sich das Gewicht durch die Abarbeitung des Arbeitsplans, zum Beispiel nach der Durchführung von Fräse- und Schneidearbeiten. Im Falle von Walzvorgängen bleibt das Gewicht meist gleich und bei Plattierungsvorgängen kann es sich zwischenzeitlich erhöhen. Die Zeitangaben erfolgen in Stunden. Die erwähnten Anlagen sind nicht nur physischer Natur wie Öfen, Sägen, Fräsen und Walzanlagen, sondern es kann sich dabei um technische Wartezeiten oder Qualitätsprüfungsvorgänge handeln, die durch eine Anlagennummer abgebildet werden. Ein Arbeitsplan endet immer mit der Anlagennummer 999, welche das Ende der Produktion darstellt. Der Planartikel, als ein physisch tatsächlich vorliegender Auftrag, hat ebenfalls einen Eintrag im Arbeitsplan. Dadurch gelingt die Einbindung des Arbeitsplans in die Kapazitätsgrobplanung, indem der Wert für die Arbeitszeit je Los entsprechend auf das kumulierte Gewicht des Planartikels hochgerechnet wird.

Berechnung der Zeit

Da in den im Arbeitsplan hinterlegten AVOs die Arbeitszeit sowie das Plangewicht je Los angegeben ist, muss die Bearbeitungszeit je Fertigtonne ($AZFto$) eines Auftrags berechnet werden. Diese kann dann für das gesamte Bestellgewicht des Planartikels hochgerechnet werden, womit die grobe Belastung der Anlage bestimmbar ist. Vorab wird die Ausbringung AB benötigt. Die Werte liefert die Produktion durch die Auswertung der Einbringungsmenge EKG und des Liefergewichts LKG und berechnet sich laut Formel 5.1. Die Daten werden von den Anlagen an die Produktionsplanung zurückgemeldet.

$$AB = \frac{LKG}{EKG} \tag{5.1}$$

Das Liefergewicht darf nicht mit dem Bestellgewicht verwechselt werden. Das eine wird dem Kunden geliefert, das andere wird vom Kunde bestellt. In der Prozessindustrie ist es üblich, dass die bestellte Menge des Kunden nicht immer genau produziert werden kann. Er muss eine gewisse Unter- oder Überlieferung akzeptieren. Bei der AMAG bewegt sich dieser Wert im einstelligen Prozentbereich der bestellten Menge. Die Bearbeitungszeit je Fertigtonne kann mittels dem Plangewicht PKG, der Arbeitszeit je Los $ARBLO$ und der Ausbringung AB laut Formel 5.2 berechnet werden.

$$AZFto = \frac{ARBLO}{AB * PKG} * 100.000 \tag{5.2}$$

Die Werte der Bearbeitungszeiten und Gewichte werden für jede Anlage zur Gesamtbelastung summiert. Für die Berechnung der Auslastung der Anlage wird die verfügbare Kapazität bestimmt. Die Grundlage der verfügbaren Kapazität bildet die Kalenderzeit eines acht Stunden Arbeitstages. Es wird individuell für jede Anlage unterschieden, ob diese im ein, zwei, drei oder vier Schichtbetrieb gefahren wird. Daraus

ergeben sich 8, 16 oder 24 Stunden pro Arbeitstag, bzw. entsprechend mehr bei vier Schichtbetrieb mit Samstag und Sonntag. Die Arbeitstage sind Montag bis Freitag ohne Feiertage. Es ergibt sich daraus ein Kapazitätsangebot je Anlage und Jahr von mehreren tausend Stunden. Für die Berechnung der Auslastung wird diese Gesamtkapazität mittels der Verfügbarkeit in Prozent nochmals verringert. Es wird somit die in Kapitel 2.1 erwähnte TEEP verwendet. Mit der Berechnung und graphischen Darstellung der Auslastung ist die Planung abgeschlossen.

5.1.2 Stand der Produktfamilienbildung

Die Produktfamilienbildung erfolgt für jede SGE. Der Vorgang der Artikelaggregation zu den Familien sowie die Ermittlung der Planartikel wurden mittels eines MATLAB Programms bereits automatisiert und verwendet hierarchische agglomerative Clustermethoden. Die Gruppierungskriterien werden vom Anwender vorgegeben. Wenn die Artikelstruktur der SGE zu komplex wird, werden die Artikel mittels eines Vorfilterkriteriums in quasi Sub-SGEs geteilt. Jede dieser Sub-SGEs kann eigene Clusterkriterien haben. Durch das strikte Verlangen von Homogenität bezogen auf ausgewählte Kriterien wird sichergestellt, dass eine Gruppierung auf Grund bestimmter Fertigungsströme erfolgt. Auch diese Vorgabe kann für jede Sub-SGE verschieden sein. Die SGEs werden weiter zur größeren strategischen Geschäftsfeldern SGFs zusammengefasst, wobei die gröbste Unterscheidung zwischen Handel und OEM Ware ist.

Tabelle 10 zeigt einen Ausschnitt der SGEs und ihrer Zugehörigkeit zu den SGFs mit den für die Produktfamilienbildung wichtigen Vorfilterkriterien, Clusterkriterien sowie den Trennungskriterien für den Fertigungsstrom.

Tabelle 10: SGEs mit Gruppierungskriterien[226]

SGE	Vorfilterung	Clusterung	Fertigungsstrom
Handel – vergütete Produkte			
VB		Dicke, Breite	Dicke = 6,35mm
VN		Dicke, Breite	
Handel – naturharte Produkte			
WA		Dicke, Breite, Werkstoff, Vorplanmaterial	
WN		Dicke, Breite	
OEM – Ski, Kathoden, Spezialprodukte			
WC		Dicke, Breite	
WZ		Dicke, Breite, Vorplanmaterial	
OEM – Luftfahrt			
VL	Form = 120 Bänder Form = 140/141 Bleche Form = 142 Bleche gereckt	Dicke, Breite, Zustand Dicke, Breite, Grundwerkstoff Dicke, Breite	 Dicke = 2,03mm & 3,5mm Dicke = 3,5mm
VM		Dicke, Breite	Dicke = 25,4mm
OEM – Lot Automobil			
WD		Dicke, Breite, Oberfläche	
VD		Dicke, Breite	
OEM – Glanzqualität			
WT		Dicke, Zustand, Qualität, Oberfläche	Vorplanmaterial
WO		Dicke, Breite	Vorplanmaterial

Die SGE VL zeigt exemplarisch die Komplexität des Vorganges. Alle Produkte der SGE werden vor der eigentlichen Clusterung anhand der Form in drei Sub-SGEs geteilt. Jede Sub-SGE wird eigens zu Produktfamilien gruppiert, wobei für die gereckten Bleche die Dicke und die Breite des Produktes als Clusterkriterien herangezogen werden. Da es jedoch nötig ist, dass keine Produkte mit einer Dicke größer 3,5mm mit jenen der Dicke kleiner 3,5mm gemischt werden, ist sicherzustellen, diese Grenze einzuhalten. Eine solche strikte

[226] Quelle: Eigene Darstellung

Trennung deutet auf einen unterschiedlichen Fertigungsstrom hin. Dies zeigt eine der Schwächen des aktuellen Produktfamilienbildungsprozesses auf. Es werden mittels aufwendiger Kombination von Filterkriterien Fertigungsfamilien gebildet, um das eigentliche Ziel von Ablauffamilien zu erhalten. Die Unterscheidung wurde in Kapitel 2.2.2 getroffen.

Über eine Konfigurationsdatei werden diese Werte dem MATLAB Programm bekannt gegeben. Weiters werden Ober- und Untergrenzen für die Anzahl der Produktfamilien je SGE und die Ober- und Untergrenzen des Gesamtgewichtes je Produktfamilie je SGE in die Konfigurationsdatei eingetragen.

FamSize	17	37				
Vorfilterkriterium	2	1311				
Constraints	8000	0,2	17			
Gruppierungskriterien (Spaltennummern)	3	4	17			
LKG_Grenze	16000	17500	1000			
Nominale Kriterien	0	0	1			
Gewichtungen	1	0,001	10			
TH1 (17)	3	3	24,9			
TH2 (18)						
TH3 (19)						
Gruppierungskriterien (Spaltennummern)	3	4	17			
LKG_Grenze	10000	10000	9000			
Nominale Kriterien	0	0	1			
Gewichtungen	1	0,001	10			
TH1 (17)	3	3	29,9			
TH2 (18)						
TH3 (19)						

Abbildung 32: Alte Konfigurationsdatei[227]

Abbildung 32 zeigt exemplarisch eine alte Konfigurationsdatei für eine SGE mit den unzähligen Einträgen ohne genauer darauf einzugehen. Sie veranschaulicht den verständlichen Wunsch der Aufgabenstellung nach einer Vereinfachung der Konfigurationsdatei.

Wenn die Produktfamilien vorliegen, wird der Planartikel auf Grund der Häufigkeit der einzelnen Ausprägungen der Clusterkriterien und des Bestellgewichtes ausgewählt. Ein Planartikel wird mit folgenden Attributen dargestellt:

SGE – Werkstoff – Dicke x Breite x Länge – Zustand – Qualität – Oberfläche – Materialnummer

Für Werkstoff, Dicke, Breite und Länge, den Zustand, die Qualität und die Oberfläche wird der Variationskoeffizient laut 5.3 berechnet. Dabei ist s die Standardabweichung und $E(X)$ der Erwartungswert. Das Attribut mit dem geringsten Koeffizienten wird zuerst gewählt.[228]

$$VarK(X) = \frac{s}{E(X)} \tag{5.3}$$

Jener Wert des Attributes des Artikel mit dem meisten Liefergewicht ist der Wert für den Planartikel. Es wird eine Teilmenge der Produktfamilie gebildet, in der nur Produkte mit dem gewählten Wert des Attributes vorhanden sind. Der Variationskoeffizient wird

[227] Quelle: Eigene Darstellung

[228] Vgl. Bartsch (2001), S. 575.

nochmal berechnet. Der Vorgang wiederholt sich so lange, bis alle Attribute bestimmt wurden. Am Ende liegt der Planartikel vor. Mit den Werten muss jene Auftragsnummer gesucht werden, welche den aktuellsten Arbeitsplan hat. Diese Auftragsnummer und die Summe des Bestellgewichtes der Produktfamilie werden dem Planartikel zugewiesen. Damit ist die Planartikelsuche beendet. Sie wird für jede Produktfamilie wiederholt und am Ende werden die Artikel sortiert nach Produktfamilien und die Planartikel nach SGEs in einer Exceldatei ausgegeben. Diese bildet einen wichtigen Teil der Kapazitätsbedarfsermittlung und somit der Kapazitätsgrobplanung.

Die Aufnahme des Ist-Zustandes bildet die Grundlage für die Erarbeitung diverser Alternativen, um die Aufgabenstellung einer robusten Produktfamilienbildung zu lösen. Das folgende Unterkapitel stellt diese Entscheidungsfindung dar.

5.2 Alternativen und Algorithmusablauf

Ziel ist es, die Produktfamilienbildung im Unternehmen zu verbessern und unter Robustheitskriterien erfolgen zu lassen. Die aufwendige Konfigurationsdatei muss einfacher gestaltet und die Clusterkriterien automatisch ermittelt werden. Der Programmcode soll weiterhin in MATLAB geschrieben werden, auf dem alten aufbauen und als Ein- und Ausgabemedium weiterhin Exceldateien verwenden.

5.2.1 Entscheidungsfindung und Lösungsalternativen

Über die gefundenen Quellen zeichneten sich drei potentielle Entwicklungsrichtungen ab. Die bereits verwendeten Clustervorgänge in Kombination mit anderen Methoden wie genetische Algorithmen, Verfahren basierend auf der Fuzzy-Logik und Fuzzy-Mengenlehre und Multiagenten-Ansätze. Für die drei Alternativen wurden die Vor- und Nachteile erarbeitet, gegenübergestellt und nach Absprache mit dem Betreuerteam aus Industrie und Universität eine Entscheidung getroffen.

Clustermethoden mit genetischen Algorithmen

Diese Methodik wurde bereits in Kapitel 4.2 ausführlich beschrieben. Die Bewertung der Robustheit der Produktfamilie passiert indirekt über die Homogenität, zum Beispiel das Kapazitätsprofil der Produkte in der Familie. Für das Kapazitätsprofil sollte zusätzlich die Anlagenfolge und somit der Arbeitsplan in das Datengerüst aufgenommen werden. Damit würde die komplexe Gestaltung der Konfigurationsdatei wegfallen, da eine Trennung und Vorfilterung für die Fertigungsströme nicht mehr nötig sein sollte. Die möglichen Clusterkriterien werden binär codiert, womit eine automatische Ermittlung möglich wäre. Die Fitnessbewertung der Lösungen erfolgt über die Homogenität der Attribute oder über den Gleichlauf des Kapazitätsprofils. Eine Herausforderung wäre die Rechenzeit, da der Arbeitsplan die Datenmenge um den Faktor zehn bis fünfzehn erhöhen würde, sowie die Implementierung der Fitness- bzw. Robustheitsbewertung.

Fuzzy Programmierung

Prinzipiell wäre sie aufgrund der unklar definierten Natur des Produktfamilienbildungsvorgangs geeignet, da sie bei der verbalen Beschreibung von Problemen eingesetzt wird. Die Beschreibung der Clusterkriterien, der

Fertigungsstromtrennung und die weiteren Werte der Konfigurationsdatei sind ein Beispiel dafür, die Beschreibungen in einer für ein Programm verständlichen Weise abzubilden. Die Aufträge würden mit unterschiedlichen Zugehörigkeiten mehreren Familien zugeteilt. Die Attribute wären durch Gewichtungsfaktoren skalierbar und auf Konsistenz z.B. mittels AHP überprüfbar. Das Expertenwissen würde über die „wenn-dann"-Bedingungen integriert. Die größten Herausforderungen wären die Definition der Zugehörigkeitsfunktionen, die Rechenzeit und die Erstellung der „wenn-dann"-Regeln.

Multiagenten-Ansätze

Das Einsatzgebiet ist hauptsächlich der Informationsgewinn aus großen Datenmengen. Jedes Produkt müsste als eigenständiger Agent implementiert werden. Diese suchen sich aufgrund ihrer Attribute, wodurch sich die Cluster bilden. Bei einer Überschreitung der Grenze des Bestellgewichts müssten sie geteilt werden.[229] Die größte Herausforderung ist die grundlegende Implementierung in MATLAB, da hierfür zwangsläufig objektorientierte Programmierung eingesetzt werden müsste, wozu MATLAB aufgrund des Konzeptes seiner Programmiersprache nur sehr umständlich, wenn überhaupt, geeignet ist.

Tabelle 11: Bewertung Verfahrensauswahl[230]

Kriterien	Ausprägung				Punkte HAC	Fuzzy	MAA
Baut auf bestehender Lösung auf	nicht (0)	gering (1)	mittel (2)	sehr (3)	3	1	1
Ist in MATLAB realisierbar	nein (0)	kaum (1)	gut (2)	sehr gut (3)	3	2	1
Es existiert facheinschlägige Literatur	keine (0)	wenig (1)	einige (2)	viel (3)	2	2	3
Es gibt bereits Erfahrung des Entwicklers	keine (0)	wenig (1)	einige (2)	viel (3)	3	2	1
Führt in vorgegebener Zeit zur Lösung	nein (0)	eher nein (1)	eher ja (2)	ja (3)	3	2	1
Deckt die Anforderungspalette am besten ab	nein (0)	kaum (1)	einige (2)	voll (3)	3	2	1
	Summe				17	11	8

Tabelle 11 ist eine Quantifizierung der Verfahrenswahl. Jedes der kurz vorgestellten Verfahren wurde auf Grund von sechs Kriterien in vier Ausprägungen (0-3) bewertet. Kriterium eins soll das Verfahren bewerten, ob es mit dem Eingesetzten der AMAG kombinierbar ist, bzw. darauf aufbaut. Verfahren der hierarchischen agglomerativen

[229] Vgl. Ogston et al. (2003), S. 789f.
[230] Quelle: Eigene Darstellung

Clusterung (HAC) erfüllen dieses Kriterium sehr, da solche Clusterverfahren bereits zum Einsatz kommen und ggf. mit anderen Methoden erweitert würden. Die Fuzzy Mengenlehre bzw. Multiagenten-Ansätze (MAA) erfüllen diese Voraussetzung nur gering, da sie ein völlig neues Konzept darstellen würden. Die HAC wäre nur noch als kleiner Teil unterstützend tätig.

Die Realisierbarkeit mittels MATLAB ist das zweite Kriterium. Während die HAC und die Fuzzy Mengenlehre sehr gut bzw. gut mittels MATLAB realisierbar wären ist es im Falle der MAA kaum der Fall, da ein objektorientierter Zugang nötig wäre. Die Eignung der Fuzzy Mengenlehre basiert auf der Existenz von zukaufbaren Paketen für die Fuzzy Logik.

Für alle drei Verfahren existiert ausreichend facheinschlägige Literatur, was Kriterium drei ist. Kriterium vier hängt mit diesem und Kriterium fünf zusammen. Während es ausreichend Erfahrung bei HAC, genetischen Algorithmen sowie der Fuzzy-Logik gibt, beschränkt sich das Wissen bei MAA auf einfache Ameisenalgorithmen.

Da das Programm noch in der aktuellen Planungsphase eingesetzt werden soll, war eine Einschätzung der zeitnahen Realisierbarkeit nötig. Diese wurde mit Kriterium fünf vorgenommen. Auf Grund der vorherigen Kriterien ist die Wahrscheinlichkeit sehr hoch, dass eine Lösung innerhalb der zwei Monate Entwicklungszeit mit HAC gefunden wird. Während bei den anderen beiden Verfahren diese Sicherheit nicht in diesem Ausmaß existiert.

Das letzte Kriterium soll prüfen, wie sehr die Anforderungen durch die Verfahren erfüllbar sind. Diese sind, wie bereits erwähnt, der Aufbau auf der vorhandenen Lösung, die Verwendung von MATLAB, die automatische Ermittlung der Clusterkriterien mit der gleichzeitigen Vereinfachung der Konfigurationsdatei und die robuste Bildung der Produktfamilien. HAC in Kombination mit genetischen Algorithmen erfüllt dieses Kriterium voll, die Fuzzy Mengenlehre mit gewissen Abschlägen, was vor allem der Wiederverwendung der existierenden Lösung zuschulden ist. Die MAA erfüllen das Kriterium kaum. Aufgrund der Auswertung fiel die Entscheidung auf die hierarchische agglomerative Clusterung mit genetischen Algorithmen.

5.2.2 Algorithmusablauf

Abbildung 33 zeigt den groben Ablauf des Programms zur robusten Produktfamilienbildung mit automatischer Clusterattributermittlung. Die genaue Beschreibung der Programmteile ist dem Anhang zu entnehmen. Die Verweise auf die entsprechenden Stellen im Anhang werden in Klammern im Fließtext angegeben.

Abbildung 33: Ablaufdiagramm des Programms[231]

[231] Quelle: Eigene Darstellung

Nach dem Start des Programms öffnet sich ein Menü wie in Abbildung 34 *("A. Planartikelermittlung")*. Der Planer kann entscheiden, ob er die Informationen der Konfigurationsdatei verwendet und die Clusterattribute neu bestimmen lässt *(„B. Hauptprogramm mit* Attributermittlung*")*, oder ob er die Protokolldatei einliest und die darin bestimmten Clusterattribute verwendet *(„C. Hauptprogramm ohne Attributermittlung")*. Im ersten Fall kann ein Programmdurchlauf, je nach Anzahl der gewählten SGEs und der Datenmenge, mehrere Stunden dauern. Im anderen Fall beträgt die Laufzeit maximal 35 Minuten.

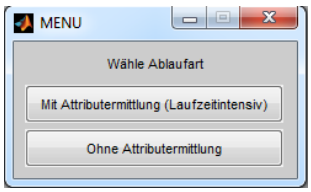


Abbildung 34: Auswahlmenü[232]

Nach dieser Auswahl öffnet sich ein Standardfenster von Windows zur Bekanntgabe der benötigten Datenbasis und der Speicherpfade für die Ergebnisdateien. Dem folgt eine Auswahl, wie in Abbildung 35 zu sehen, der SGEs welche man bearbeiten will.

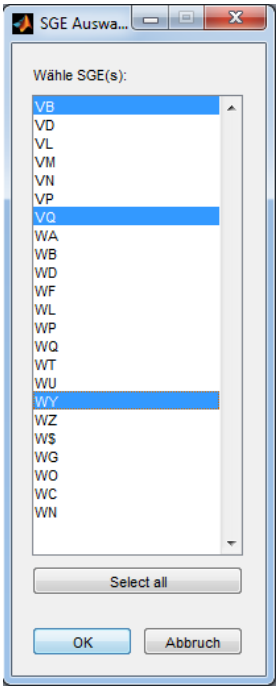


Abbildung 35: SGE Auswahlliste[233]

Unabhängig der Wahl in Abbildung 34 erfolgt nun die Ermittlung der Anlagenfolge und des Belastungsprofils sowie der Nullstellen des Polynoms *(„D. Verknüpfung Auftragsdaten und Arbeitsplandaten" , „E. Vereinfachung des Arbeitsplans" und „F. Berechnung der Varianz der Nullstellen des Belastungsprofilpolynoms")*. Für die Erstellung der gesamt gültigen Distanzmatrix wird die Distanzmatrix der Anlagenfolge berechnet *(„G. Berechnung der Distanzmatrix aufgrund*

[232] Quelle: Eigene Darstellung
[233] Quelle: Eigene Darstellung

der Anlagenfolge"), indem die Distanz zwischen den Aufträgen aufgrund der Anlagenfolge bestimmt wird *(„H. Berechnung der Distanz zwischen zwei Anlagen" und „I. Berechnung Anzahl unterschiedlichen Anlagenfolgen")*. Im Falle des schnellen Durchlaufes ohne Attributermittlung folgt der Clustervorgang *(„L. Durchführung des Clustervorgangs")* und die Planartikelermittlung mittels Rekursion *(„Q. Ermittlung* der Planartikel*"* und *„R. Einschränkung der Auftragsauswahl für die Planartikelermittlung")*. Bei der Auswahl mit Attributermittlung wird vor der Clusterung die Ausgangspopulation erstellt. Dazu werden vorab jene Attribute gesucht, die wegen des Überhangs einer Ausprägung kaum einen Informationsgewinn zum Clustervorgang liefern *(„J. Bestimmung der Attribute, die nicht beachtet werden* sollten*")*. Danach wird die Population mit Individuen befüllt *(„K. Population auf die richtige Größe bringen")*. Nach dem Clustervorgang wird die Fitness bestimmt *(„M. Berechnung der Fitness")*. Wenn das Abbruchkriterium nicht erfüllt ist, werden die genetischen Operationen durchgeführt. Ein Abbruchkriterium ist, dass das Programm bereits eine bestimmte Anzahl von Generationen simuliert hat. Das zweite ist, dass sich die Fitness seit einigen Durchläufen nicht mehr verbessert hat.

Die Auffüllung des Paarungspools erfolgt mittels fitnessproportionaler Selektion *(„N. Durchführung der* fitnessproportionalen Selektion*")* und der Erstellung der neuen Population mittels einer 2-Punkt-Kreuzung *(„O. Durchführung einer n-Punkt Kreuzung")*. Vor dem neuen Start des Clustervorgangs wird auf die Population eine Mutation angewandt *(„P. Durchführung der genetischen Mutation")*.

Ist eines der Abbruchkriterien erfüllt, wird, wie auch im Falle des Durchlaufes mit Attributermittlung, mit der Bestimmung der Planartikel fortgefahren *(„Q. Ermittlung der Planartikel" und „R. Einschränkung der Auftragsauswahl für die Planartikelermittlung")*. Zum Abschluss werden die Ergebnisse protokolliert *(„S. Protokollieren der Ergebnisse" und „T. Ermittlung der Spaltenbezeichnung")* und mit den Ergebnissen der Planartikelsuche und Produktfamilienbildung gespeichert *(„U. Speichern des Ergebnisses")*.

Der allgemeine Ablauf des Algorithmus setzt sich aus einer Vielzahl anderer Verfahren zusammen, die nach Bedarf wiederholt werden. Diese Verfahren bilden die Anforderungen des Unternehmens und die theoretischen Methoden ab, und werden im folgenden Unterkapitel genauer beschrieben.

5.3 Verfahrensbeschreibung

Zur Lösung mittels Clusterverfahren und genetischer Algorithmen wurde der Ansatz von LEE-POST[234] als Orientierung verwendet. Das Verfahren wurde bereits in Kapitel 4.2.2 beschrieben. Vorab musste die Datenbasis aufbereitet werden.

[234] Vgl. Lee-Post (2000)

5.3.1 Datenvorbereitung

Zur automatischen Wahl der Gruppierungskriterien musste eine Liste von infrage kommenden Kriterien erstellt werden. Diese Liste wurde mit Hilfe von erfahrenen Planern erarbeitet. Dazu wurden 52 Anlagen, von Sägen, Fräsen und Walzaggregaten über Scheren, Stanzen und Öfen bis zu Plattieranlagen, Prüfeinrichtungen und Verpackungsmaschinen bewertet. Die Bewertung erfolgte anhand einer Liste von Kriterien und deren Einfluss auf die Durchlaufzeit bei diesen Anlagen. Die Bewertungsskala enthielt die Werte:

- keinen Einfluss (0)
- geringer Einfluss (1)
- mittlerer Einfluss (2)
- starker Einfluss (als Hauptkriterium) (3)

Die gewählten Kriterien orientierten sich an der vorhandenen Lösung und wurden noch ergänzt. Diese Liste enthielt die geometrischen Maße des Auftrages mit Dicke, Breite, Länge und Rondendurchmesser. Materialeigenschaften wie der verwendete Grundwerkstoff, der prozentuale Anteil der Plattierung an der Gesamtdicke und der daraus resultierende Werkstoff. Weiters wurde der Zustand, die Qualität, die Oberfläche und die Form in die Bewertung einbezogen, sowie die Sondervorschrift, die spezielle Anweisungen für die Bearbeitung enthält. Das genaue Ergebnis der Auswertung ist Tabelle 13 im Anhang zu entnehmen.

Die Dicke des Produktes spielte auf vielen Anlagen eine Rolle, von allen Kriterien sogar am häufigsten die Gewichtigste. Eine weitere wichtige Erkenntnis ist, dass die Plattierung eine sehr untergeordnete Priorität hat. Auf nur einer Anlage hatte sie starken Einfluss, das jedoch neben drei weiteren Attributen mit dieser Ausprägung. Daher wurde sie nicht weiter berücksichtigt, da jedes Attribut weniger, die Laufzeit verringert. Die Sondervorschrift, als neu hinzugenommenes Kriterium, wurde durchgehend als einflussreich eingestuft. Auf zwei Anlagen sogar als einziges mit starkem Einfluss. Da dies zwei hoch ausgelastete Anlagen sind, wurde die Sondervorschrift in die weitere Betrachtung aufgenommen. Der Rondendurchmesser wurde auf nur zwei Anlagen als Kriterium mit starkem Einfluss gewertet, ansonsten immer ohne Einfluss angegeben. Trotzdem verblieb er in der Liste der Clusterkriterien. Somit ergeben sich als potentielle Clusterkriterien

Werkstoff - Grundwerkstoff - Dicke - Breite - Länge - Zustand - Qualität - Oberfläche - Form - Sondervorschrift - Rondendurchmesser - Vorplanmaterial.

Das Vorplanmaterial wurde in die Expertenbewertung nicht einbezogen, da es sich um ein Hilfskonstrukt handelt, welches bereits als Fertigungsstromtrennungskriterium in den Artikelstamm eingepflegt wird.

Analgen und Arbeitsplan

Da ein Ziel der Kapazitätsgrobplanung in der AMAG auch die Verringerung von teils erheblichen Rüstzeiten ist, ist es sinnvoll die Anlagen in die weitere Betrachtung einzubeziehen. Solche Zeiten können bei einem Walzenwechsel mehrere Stunden betragen. Anstatt von Fertigungsfamilien sollen Ablauffamilien gebildet werden (siehe Kapitel 2.2.2). Es bietet sich an, die Analgenfolge zur Fitnessbewertung heranzuziehen. Je einheitlicher die Anlagenfolge der Produkte ist, desto besser wurde die Produktfamilie gebildet. Weiters

kann die Anlagenfolge im Falle der AMAG als Robustheitskriterium eingesetzt werden. Die Bedarfsunsicherheit innerhalb des Planungshorizontes von einem Jahr drückt sich hauptsächlich dadurch aus, dass die Bestellungen kurzfristig in kleinen Details wie der Dicke, der Legierungsgruppe - also dem Werkstoff, oder Änderungen im Zustand aus. Der Gesamtbedarf einer SGE bleibt somit gleich. Es verschieben sich die Quantitäten innerhalb der SGE, wodurch sich die Anlagenfolge ändern kann. Bleibt die Anlagenfolge gleich, so soll sich die Fitness nicht ändern. Kommt es zu Abweichungen in der Anlagenfolge, so besteht die Möglichkeit, dass diese bereits existiert und somit sich eine Produktfamilie auf deren Grundlage gebildet hat. Es würde lediglich zu einer Verschiebung des Bestellgewichtes zwischen den Planartikeln kommen, an der Zahl der Produktfamilien ändert sich nichts. Diese Unveränderlichkeit im Ergebnis, trotz anderer Anfangsbedingungen, ist der Kern der Robustheit. Die Anlagenfolge muss folglich in die Liste der Clusterkriterien aufgenommen werden.

Werkstoff - Grundwerkstoff - Dicke - Breite - Länge - Zustand - Qualität - Oberfläche - Form - Sondervorschrift - Rondendurchmesser - Vorplanmaterial - Anlagenfolge.

Der Arbeitsplan wird in die Datenmenge aufgenommen, um die Anlagenfolge und ein Belastungsprofil zu erhalten. Im Sinne der Laufzeitreduktion wurde eine eigene Datenbankabfrage in Auftrag gegeben, die nur die benötigten Daten enthält. Dazu zählen die kombinierte Auftragsnummer, bestehend aus der Kommissionsnummer und der Positionsnummer, die Anlagennummer, die AVO-Nummer, die Bearbeitungszeit je Los, das Plangewicht und der Termin der Einpflegung ins System. Eine eigene Abfrage für die Auftragsdaten wurde aufgegeben, um die Datenbasis weiter zu verringern. Bisher wurden 52 Datenfelder aus der Datenbank ausgespielt, obwohl ein großer Teil dafür nicht nötig ist. Die neue Abfrage enthält nur 19 Datenfelder und ist auszugsweise im Anhang als Abbildung 43 zu finden.

Die codierten Attribute sind in der Datenbank ursprünglich alphanummerisch hinterlegt. Da MATLAB jedoch mit Zahlen unkomplizierter und effizienter arbeiten kann, werden jene Datenfelder mit alphanummerischen Einträgen zu eindeutigen Zahlen codiert. Dies geschieht mittels der Datenbankabfrage.

Eine weitere effizienzsteigernde Maßnahme ist die Reduktion der Anlagen. Erste Tests zeigten größere Inhomogenitäten in den Belastungsprofilen und Anlagenfolgen, als ursprünglich erwartet. Dies war nach eingehender Analyse darauf zurückzuführen, dass scheinbar idente Aufträge unterschiedliche Fertigungsströme aufwiesen. Der Grund lag in den meisten Fällen in technisch gleichwertige Anlagen, die jedoch unterschiedliche Anlagennummern hatten. Solche Fälle sind bei den Öfen, den Beschichtungsanlagen und den Richtmaschinen zu finden. Da die Planung die Kapazitäten dieser identen Anlagen auch zusammenfasst und mit der Gesamtkapazität, als eine Anlage betrachtet, wurde in der Bildung der Anlagenfolge und des Belastungsprofils ebenfalls so vorgegangen. Gleiche Anlagen an gemeinsamen Standorten wurden unter einer Anlagennummer aggregiert. Weiters hatten diverse Wartezeiten, Zwischenlagerungsvorgänge und nicht zeitkritische Prüf- und Qualitätssicherungsmaßnahmen ebenfalls Anlagennummern und wurden im Arbeitsplan geführt. Die Datensätze solcher Anlagennummern wurden entfernt. Dadurch sanken die Variation und die Unsicherheit in der Planung und stieg somit die Robustheit des Belastungsprofils. Der Vorgang der Bereinigung des Arbeitsplans ist in der

Codedokumentation genau beschrieben und im Anhang unter *„E. Vereinfachung des Arbeitsplans“* zu finden.

5.3.2 Genetischer Algorithmus

Die automatisierte Wahl der Clusterkriterien, sowie die Bewertung der Robustheit der Lösung, erfolgt durch genetische Algorithmen. Die Erstellung des Paarungspools geschieht mittels fitnessproportionaler Selektion, die Kreuzung via eines 2-Punkt Verfahrens und die Mutation ist eine Abänderung des binären Ansatzes aus Kapitel 4.2.2. Zu Beginn wird eine Anfangspopulation erstellt. Wie bereits erwähnt, existieren 13 Clusterkriterien. Ein Individuum besteht somit aus einem binären Feld mit 13 Stellen. Wobei eine Null bedeutet, dass das Attribut an der Stelle nicht gewählt wird und eine Eins, dass es gewählt wird. Theoretisch existieren $2^{13} = 8192$ Möglichkeiten ein Individuum zu bilden. Um diese Anzahl einzuschränken, werden die Attribute vorab auf ihre Relevanz hin überprüft.

Dazu wird bestimmt, ob es in der SGE einen Attributwert gibt, welcher überproportional vorhanden ist. Für jedes Attribut wird für jede Ausprägung das Bestellgewicht bestimmt, welches es auf sich vereint. Existiert eine Ausprägung, welche zum Beispiel 98% des gesamten Gewichtes ausmacht, so ist der Informationsgewinn einer Clusterung anhand dieses Attributes gering, da Produktfamilien, die nur die anderen Ausprägungen enthalten wegen Gewichtsuntergrenzen abgelehnt würden. Eine fixe Annahme der Ablehnungsgrenze über alle SGEs ist schwer, nachdem jede unterschiedliche Gewichtsuntergrenzen für die Produktfamilien besitzt. Somit wird als Ablehnungskriterium ALK die doppelte Gewichtsuntergrenze UG verwendet. Existiert eine Attributausprägung, die mehr Gewicht auf sich vereint, wird das Attribut abgelehnt. Formel 5.4 zeigt das Ablehnungskriterium und das Vorgehen wird anhand eines fiktiven Beispiels erörtert, wobei absolute und keine relativen Werte verwendet werden.

$$ALK = 100\% - 2 * UG[\%] \tag{5.4}$$

Das Gewicht der SGE beträgt 15.000kg und es werden die Attribute Werkstoff und Dicke verwendet. Der Werkstoff hat die Ausprägungen 1500, 1750, 5000. Die Dicke hat die Ausprägungen 1,5mm, 3,2mm und 8,7mm. Die Gewichtsuntergrenze, die jede Produktfamilie mindestens haben muss, ist 2% des Gesamtgewichtes des SGE, in diesem Fall 300kg. Das Ablehnungskriterium liegt folglich bei 14.400kg. Jedes Attribut wird nun für sich betrachtet. Produkte mit dem Werkstoff 1500 vereinen 5.000kg auf sich, jene mit 1750 kommen auf 7.800kg und die Gruppe mit Werkstoff 5000 wiegt 2.200kg. Keine Ausprägung des Attributes Werkstoff vereint mehr Gewicht auf sich, als das Ablehnungskriterium erlauben würde. Somit ist es für eine Clusterung aussagekräftig. Die Dicke vereinigt für 1,5mm 300kg, für 3,2mm 14.500kg und für 8,7mm 200kg jeweils auf sich. Der Wert 3,2mm verstößt gegen das Ablehnungskriterium, da er überproportional viel wiegt. Produktfamilien mit nur den anderen Ausprägungen würden abgelehnt. Somit wird die gesamte Dicke nicht bei der Clusterung berücksichtigt. Die Umsetzung im Algorithmus ist im Anhang unter *„J. Bestimmung der Attribute, die nicht beachtet werden sollten“* zu finden.

$$\begin{matrix} Attribute \\ Generation \\ Population \end{matrix} \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\ 2 & 2 & 2 & 2 & 2 & 3 & 5 & 10 & 20 & 40 & 80 & 120 & 180 \\ 4 & 4 & 10 & 20 & 20 & 20 & 20 & 20 & 20 & 20 & 20 & 20 & 20 \end{bmatrix}$$

Abbildung 36: Populations-Generations-Matrix[235]

Sobald alle Attribute festgestellt wurden, wird eine Ausgangspopulation, mit einer der Anzahl der Attribute entsprechenden Größe, erstellt. Das Vorgehen ist dem Anhang unter *„B. Hauptprogramm mit Attributermittlung - Initialisierung der Ausgangspopulation"* zu entnehmen. Als Grundlage dient die Matrix von Abbildung 36. Die erste Zeile enthält die Anzahl der relevanten Attribute, Zeile zwei gibt die dafür vorgesehene Anzahl der Generationen vor und in der letzten Zeile wird die Populationsgröße festgelegt. Der Clustervorgang wird aufgrund des Bitmusters der Individuen der Population durchgeführt. Die Lösungsbewertung erfolgt anhand des Belastungsprofils. Gültige Lösungen werden entsprechend der Fitness in den Paarungspool aufgenommen und miteinander gekreuzt. Die Fitnessbewertung ist im Anhang unter *„M. Berechnung der Fitness"*, die Selektion unter *„N. Durchführung der* fitnessproportionalen Selektion*"* zu finden. Ungültige Lösungen sind jene die Produktfamilien beinhalten, welche die durch die Konfigurationsdatei vorgegebenen Gewichtsunter- und Obergrenzen nicht einhalten. Die jeweils beste Lösung wird als solche gespeichert. Wenn der neue Paarungspool durch Kreuzung *(„O. Durchführung einer n-Punkt Kreuzung")* befüllt ist, wird sichergestellt, dass die ausgeschlossenen Attribute nicht verwendet werden, indem diese Spalten mit null überschrieben werden. Mit einer vorgegebenen Mutationswahrscheinlichkeit werden zwei Stellen der gewählten Individuen invertiert. Siehe hierzu im Anhang *„P. Durchführung* der genetischen Mutation*"*. Danach starten die Vorgänge der Clusterung, Fitnessbewertung, Selektion, Kreuzung und Mutation neu. Der Ablauf ist abgeschlossen, wenn sich die Fitness für drei Durchgänge im Vergleich mit der besten Lösung nicht mehr ändert oder eine vorgegebene Anzahl von Generationen bereits simuliert wurde.

Fitness- und Robustheitsbewertung

Für die Bewertung wird ein Belastungsprofil erstellt. Es handelt sich um eine Matrix mit zwei Zeilen und einer Spaltenanzahl, die den betrachteten Anlagen entspricht. Die erste Zeile enthält die Anlagennummer, die zweite Zeile die Arbeitszeit je Los ($ARBLO$). Da die Arbeitszeit in der Regel proportional zum Gewicht steigt, ist es nicht nötig die $ARBLO$ in die Arbeitszeit je Fertigtonne umzurechnen. Abbildung 37 zeigt eine solche Matrix mit fiktiven Werten. Zeile eins entspricht dem Clusterkriterium der Anlagenfolge.

$$\begin{matrix} Anlage \\ ARBLO \end{matrix} \begin{bmatrix} 5 & 12 & 370 & 605 & 275 & 30 \\ 0{,}6 & 1{,}02 & 0{,}2 & 6{,}56 & 2{,}01 & 8{,}36 \end{bmatrix}$$

Abbildung 37: Matrix Belastungsprofil[236]

Jede Spalte der Matrix wird als Koordinate verstanden und in ein kartesisches Koordinatensystem eingetragen, wobei die Anlagennummer an der Abszisse und die Arbeitszeiten an der Ordinate aufgetragen werden. Durch die Punkte wird ein Polynom gefittet mit (Anzahl Spalten-1)-ten Grades. Von diesem Polynom werden die positiven

[235] Quelle: Eigene Darstellung

[236] Quelle: Eigene Darstellung

reellen Nullstellen bis zu x = 999 bestimmt. Obwohl es keine negativen Punkte und keine über x = 999 gibt, kommt es durch Ein- und Ausschwingvorgänge zu Nulldurchstößen außerhalb dieser Werte. Es handelt sich um Runges-Phänomen. Von diesen Nullstellen wird die Verteilung mittels Varianz bestimmt. Je ähnlicher die Verteilung der Produkte innerhalb einer Produktfamilie ist, desto homogener ist der Fertigungsstrom und desto besser ist folglich die Fitness der Lösung und die Robustheit der Produktfamilie.

Es muss angemerkt werden, dass das Polynom ein reines Hilfskonstrukt darstellt. Aus den Werten auf der Kurve sind keine weiteren Schlüsse zu ziehen. Negative Ordinatenwerte machen physikalisch keinen Sinn und andere Abszissenwerte als jene der Matrix sind aufgrund der fehlenden Anlagen nicht existent. Es geht bei dieser Methode rein um ein leicht implementierbares Verfahren zur Bewertung des Profils.

Abbildung 38 zeigt ein Beispiel einer gut gebildeten Produktfamilie am Ende eines Clustervorgangs. Aufgrund von Skalierungseffekten durch die Polynomfittung sind Unterschiede zwischen den Punkten der Matrix und den Nullstellen auf der Abszisse nicht mehr zu erkennen. Die Abbildung zeigt jedoch sehr gut das Kurvenfeld, welches sich durch die Belastungsprofile der Produkte in der Produktfamilie ergibt. Die dicke blaue Line zeigt jenen Auftrag, welcher als Planartikel für diese Produktfamilie gewählt wurde. Die Bildung des Planartikels wird noch erörtert.

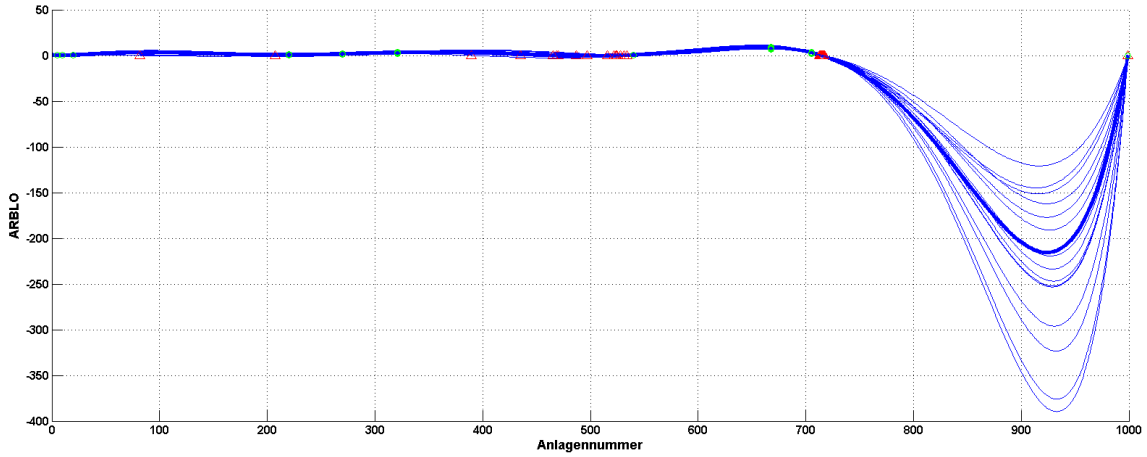


Abbildung 38: Kurvenfeld Produktfamilie mit Planartikel[237]

Würde nun ein Auftrag z.B. in der Menge oder in einem Attribut wie der Dicke geändert, würde sich der Arbeitsplan zwar im Wert der *ARBLO* ändern. Dies wäre in der Amplitude der Kurve, in diesem Fall sehr gut im letzten Ausschlag, zu sehen. In Summe würde der Auftrag jedoch weiterhin in der Produktfamilie bleiben. Sie würde sich nicht ändern, was im Sinne der Robustheit ist.

5.3.3 Clustervorgang

Der Clustervorgang verwendet zwei verschiedene Distanzmaße und zwei verschiedene Gruppierungsalgorithmen. Das genaue Vorgehen, mit Beschreibung, ist im Anhang im Kapitel *„L. Durchführung des Clustervorgangs"* zu finden. Die Distanzmaße sind die euklidische Distanz und die Mahalanobis-Distanz. Die Gruppierungsalgorithmen sind das

[237] Quelle: Eigene Darstellung

Wardverfahren und Single-Linkage. Vor der Berechnung der Distanzmatrix werden die gewählten Clusterattribute geteilt. Es werden jene mit einer Ratioskala und einer nominalen Skala unterschieden. Die Anlagenfolge wird für sich behandelt, da sie keiner Skala zugeordnet werden kann. Es werden die beiden Anlagenfolgen der vergleichenden Aufträge Position für Position miteinander verglichen. Das Ergebnis wird invertiert somit wird Gleichheit mit einer Null und Ungleichheit mit einer Eins angezeigt. Danach wird die Summe der Einsen gebildet und durch die Anzahl der verschiedenen Anlagenfolgen der SGE dividiert. Somit erhält man das Distanzmaß.

Formel 5.5 zeigt die Berechnung der Werte für die Distanzmatrix der Anlagenfolge. d_{ijk} ist die Distanz zwischen Auftrag i und Auftrag j gemessen am Attribut k, hier die Anlagenfolge. a_{ik} und a_{jk} sind die Anlagenfolge jeweils von Auftrag i und j. $\delta(a_{ik}, a_{jk})$ steht für die Positionen der Anlagenfolge, die nicht ident sind, somit die Distanz zwischen den Aufträgen. L_k ist die Anzahl der verschiedenen Anlagenfolgen in der SGE. Es handelt sich um ein der Problemstellung angepasstes Distanzmaß von LEE-POST.[238]

$$d_{ijk} = \frac{\delta(a_{ik}, a_{jk})}{L_k} \tag{5.5}$$

Mit den ermittelten Werten wird eine eigene Distanzmatrix für die Anlagenfolge aufgebaut. Somit kann es bis zu drei Distanzmatrizen geben, eine für die Attribute der Ratioskala, eine für jene der nominalen Skala und die Distanzmatrix der Anlagenfolge. Diese werden durch arithmetische Mittelung, Position für Position, zu einer Gesamtdistanzmatrix kombiniert. Mit dieser wird die Gruppierung anhand der Clusteralgorithmen durchgeführt. Die gebildeten Produktfamilien werden mittels ihrer Fitness bewertet und der Clustervorgang wird so oft wiederholt, wie es der genetische Algorithmus verlangt. Die Erstellung der Distanzmatrix der Anlagenfolge, die Berechnung der Distanz nach 5.5 und die Bestimmung der Anzahl der Anlagenfolge sind detailliert im Anhang unter den Kapiteln *„G. Berechnung der Distanzmatrix aufgrund der Anlagenfolge“*, *„H. Berechnung der Distanz zwischen zwei Anlagen“* und *„I. Berechnung Anzahl unterschiedlichen Anlagenfolgen“* zu finden.

Planartikelbestimmung

Aus der Produktfamilie, welche durch die Clusterung hervorgeht, wird ein Planartikel gewählt, der möglichst repräsentativ ist. Dazu wird eine Teilmenge der Familie gebildet, die in den Attributausprägungen völlig homogen ist. Da die Fitness, oder Güte einer Produktfamilie, anhand des Belastungsprofils berechnet wird, können je Attribut verschiedene Ausprägungen vorhanden sein. Ziel ist es eine Menge von Aufträgen in der Produktfamilie zu finden, die je Attribut völlig homogen ist. Dies erfolgt über ein rekursives Programm, *(Anhang: „R. Einschränkung der Auftragsauswahl für die Planartikelermittlung“)* welches schrittweise die Auswahl verringert. Es wird jene Attributausprägung gesucht, die am meisten Gewicht auf sich vereint. Mit diesem Wert wird eine Teilmenge von Aufträgen gebildet, welche nur mehr diese enthält. Das Attribut, welche die Ausprägung enthält, ist somit homogen und wird nicht mehr betrachtet. Der Vorgang wiederholt sich solange, bis alle Attribute homogen sind. Am Ende existiert eine Menge von Aufträgen, die völlig ident sind, mit Ausnahme der Auftragsnummer und ggf.

[238] Vgl. Lee-Post (2000), S. 800.

des Bestellgewichts. Es besteht auch die Möglichkeit, dass nur ein Auftrag in dieser Menge ist. Somit wäre die weitere Vorgangsweise trivial, da dieser der Planartikel ist. Sollten am Ende der Rekursion mehrere Artikel vorhanden sein, wird jener Artikel bestimmt, der am geringsten von der Nullstellenstreuung der Produktfamilie abweicht. Gibt es auch hier mehrere, so wird jener mit dem meisten Bestellgewicht gewählt. Eine noch detailliertere Beschreibung ist dem Anhang unter *„Q. Ermittlung der Planartikel"* zu entnehmen.

Protokollieren und Speichern

Mit der Bestimmung der Planartikel ist die zentrale Aufgabe des Programms abgeschlossen. Die Planartikel und die Aufträge, sortiert nach der Zugehörigkeit zu den SGEs und den Planartikeln, werden in eine Exceldatei gespeichert. Die genaue Beschreibung des Vorganges ist in Kapitel *„U. Speichern des Ergebnisses"* im Anhang zu finden. Das erste Tabellenblatt enthält die Planartikel, alle weiteren Tabellenblätter sind für die Aufträge einer SGE. Die Planartikel haben eine laufende Nummer, welche den zugehörigen Aufträgen ebenfalls zugeordnet werden. Damit kann ein Konnex zwischen Planartikel und den Aufträgen der Produktfamilie hergestellt werden. Abbildung 39 zeigt einen exemplarischen Ausschnitt aus dem Tabellenblatt der Planartikel. Abbildung 44 im Anhang zeigt einen Auszug aus der Datei der Produktfamilien mit den Spaltenbeschriftungen.

SGE	Wkst	Dicke	Breite	Länge	Zustand	Qualität	Oberfläche	Anteil	BKG	MatNr	AuftragsNr	FamNr
VB	2112	5	1000	2000	61	2	10	3,34	60000	93013395	8713161	1
VB	2148	3,17	1524	1981,2	61	2	10	17,62	316597,6	93021704	6425661	2

Abbildung 39: Ausschnitt Planartikeldatei[239]

In einer weitern Exceldatei wird das Protokoll des Programmdurchlaufs gespeichert *(„S. Protokollieren der Ergebnisse")*. Diese Datei enthält für jede SGE einen Datensatz. Er beinhaltet Informationen wie das Distanzmaß, das Clusterverfahren sowie die Anzahl der Produktfamilien und das Gewicht der leichtesten und schwersten als Unter- und Obergrenze. Die verwendeten Clusterattribute werden binär codiert dargestellt. Abbildung 42 im Anhang zeigt einen Auszug dieser Datei.

5.4 Fazit

Die Aufgabenstellung kann als erfüllt betrachtet werden. Eine robuste Produktfamilienbildung und eine automatische Auswahl der Clusterkriterien ist machbar. Diese beiden Konzepte wurden in einem Computerprogramm umgesetzt, welches die vorhandenen Daten nutzt und das Ergebnis in einer kompatiblen Form ausgibt.

Somit erleichtert sich die Wartung der Konfigurationsdatei, was zu einer Steigerung der Benutzerfreundlichkeit des Programms führt. In Abbildung 40 ist die neue Konfigurationsdatei zur gleichen SGE aus Abbildung 32 zu sehen. Die Vereinfachung ist klar ersichtlich. Einzig die erste und, in abgeänderter Form, die dritte Zeile sind von der alten Datei geblieben. Es müssen noch eine Untergrenze und eine Obergrenze für die Produktfamilienanzahl einer SGE angegeben werden. Für jede Produktfamilie gilt die Ober- und Untergrenze des Gewichtes in der zweiten Zeile. Dieses Ergebnis der Arbeit ist eine wesentliche Erleichterung für den Planer.

[239] Quelle: Eigene Darstellung

Produktfamilienanzahl UG|OG	25	40
Produktfamiliengroße UG in %|OG in % vom Gesamtgewicht der SGE	0,3	14

Abbildung 40: Neue Konfigurationsdatei[240]

Betrachtet man das gesamte Protokoll als Ergebnis des Clustervorgangs, fällt auf, dass der Rondendurchmesser, welcher als eines der Hauptkriterien auf zwei Anlagen betrachtet wurde, in keiner SGE als Gruppierungskriterium Verwendung fand. Somit kann die Empfehlung ausgesprochen werden, diesen aus der Liste zu entfernen, da es zu einer Verbesserung der Laufzeit führt.

Eine weitere Auffälligkeit ergibt sich, wenn man die Auswertung des Protokolls mit der Beurteilung der Attribute durch die Experten vergleicht. Der Werkstoff ist in beiden Fällen das wichtigste Attribut. Vom Algorithmus wird er 15 mal als eines der Clusterkriterien verwendet. Die Anlagenfolge, welche nicht bewertet wurde, wurde am zweit häufigsten gewählt. Dicke und Breite befinden sich ebenfalls bei beiden Beurteilungen im Spitzenfeld. Die Form wurde im Gegenzug bei der Expertenbewertung hinter dem Rondendurchmesser als das unwichtigste Kriterium bewertet, während der Algorithmus die Form mit den Attributen Länge und Zustand mit sieben mal gleich oft wählt. Die anderen Attribute befinden sich mit leichten Abweichungen in einer ähnlichen Reihenfolge zueinander. Mit zehnmaliger Wahl liegt das Vorplanmaterial im oberen Viertel. Das ist wenig verwunderlich, da es die Fertigungsströme markiert.

Ein großer Unterschied ergibt sich im Vergleich der alten Konfigurationsdatei mit den Ergebnissen des Protokolls. In der alten Datei wurde die Dicke bei jeder SGE und die Breite bei allen bis auf zwei als Gruppierungskriterium gewählt. Das alte Programm verwendet somit diese beiden Kriterien, neben anderen, immer zur Gruppenbildung. Der neue Algorithmus wählt die Dicke nur für 13 der 23 SGEs und die Breite elf mal. Oftmals wird die Breite in Fällen gewählt in denen die Dicke nicht ausgesucht wird. Dieser Umstand ist überraschend, da zu Beginn angenommen wurde, dass die Dicke in einem Walzwerk fast automatisch als Kriterium gewählt werden würde.

Die Länge wird vom neuen Algorithmus sieben mal gewählt, während sie für den alten nie vorgegeben wurde. Ähnlich extrem ist die Situation bei der Form, wo das Verhältnis 7:1 ist. Tabelle 12 gibt einen Überblick über die Reihung der Attribute. Dabei wird die Reihung auf Grund der Anzahl der Auswahl vorgenommen. Für den neuen Algorithmus, die alte Konfigurationsdatei und die Expertenbewertung ergeben sich drei Reihungen. Diese werden in Tabelle 12 gegenübergestellt. Da die Anlagenfolge ausschließlich beim neuen Algorithmus berücksichtigt wird, kommt sie in der Auswertung nicht vor.

[240] Quelle: Eigene Darstellung

Kriterien die in einem der drei Fälle nicht vorkommen wurden mit 0 bewertet. Gute Übereinstimmungen sind grün gekennzeichnet, während eklatante Unterschiede in der Reihung orange markiert sind. Der Vergleich wird jeweils mit dem neuen Algorithmus vorgenommen.

Tabelle 12: Vergleich der Kriterienreihung[241]

Kriterium	Algorithmus neu	Konfigurationsdatei alt	Expertenbewertung
Werkstoff	1	4	1
Dicke	2	1	3
Breite	3	2	2
Länge	5	10	4
Zustand	5	5	6
Qualität	10	6	7
Oberfläche	8	6	5
Form	5	8	11
Grundwerkstoff	11	8	9
Vorplanmaterial	4	3	0
Sondervorschrift	8	0	8
Rondendurchmesser	12	0	10

Die Summe der gebildeten Produktfamilien beträgt im Testfall, welcher aus realen Auftrags- und Arbeitsplandaten besteht, 454. Diese Größenordnung entspricht den Ergebnissen des alten Algorithmus und jenen, die noch vor diesem in manueller Arbeit ermittelt wurden. Unterschiede existieren in den einzelnen SGEs. In Fällen wie der SGE WN ist die Anzahl der Produktfamilien mit fünf ident und bei manchen sehr ähnlich wie bei WP mit 26 zu 24. Starke Unterschiede existieren bei VD mit 26 zu 14 im Vergleich von neu zu alt. Die Spannweite der leichtesten und schwersten Produktfamilie in einer SGE ist bei beiden Algorithmen vergleichbar.

[241] Quelle: Eigene Darstellung

6 Zusammenfassung und Ausblick

Die Arbeit zeigt die Komplexität und Wichtigkeit der Kapazitätsgrobplanung. Als Schnittstelle zwischen der strategischen und der operativen Ebene steht sie an zentraler Stelle der Planung. Die Entscheidung über die Umsetzbarkeit des Produktionsprogrammplans wird hier getroffen. Auf Grund des Planungszeitraumes besteht noch die Möglichkeit, Engpässe zu erkennen, zu beheben und die Liefertreue sicherzustellen. Sollte die Grobplanung auf längere Sicht nicht mehr in der Lage sein, das geplante Produktionsprogramm umzusetzen, ist das ein Zeichen für die strategische Ebene tiefgreifende Entscheidungen zu treffen, um die Produktion sicherzustellen. Weiters werden die ersten Weichen für die operative Ablaufplanung gestellt. Nimmt die Grobplanung eine falsche Einschätzung der Machbarkeit vor, kann es zum Zeitpunkt der Ablaufplanung zu erheblichen Schwierigkeiten wie Produktion- und Lieferverzögerungen kommen.

Durch den volatilen Absatzmarkt und die hohen Anforderungen an die Flexibilität wird es immer schwerer, stabile mittel- und langfristige Pläne zu erstellen. Häufige Umplanungsvorgänge verringern das Vertrauen und die Planungsqualität und erhöhen die Unsicherheit. Es ist nötig, um diesen Effekten entgegen zu wirken, robuste Ansätze in die Planung zu integrieren. Dies erfolgt durch die Erstellung und Bewertung unterschiedlicher Szenarien. Wird ein Plan so erstellt, dass er unter einer großen Bandbreite von Szenarien seine Gültigkeit erhält, gilt er als robust. Diese Planungsansätze können in bekannte Planungsmethoden wie der rollierenden Planung oder der hierarchischen Planung eingebunden werden.

Die Datenaggregation gilt als eine der zentralen Aufgaben der Kapazitätsgrobplanung. Je präziser und schneller sie erfolgt, desto flexibler ist die Planung. Sie ist eine einfache Art, um Robustheit in einem System zu erzeugen. Wird eine aggregierte Gruppe als eine Black Box betrachtet, ohne die Teile darin zu kennen, so dringen kleine Änderungen in der Gruppe nicht nach außen. Werden Aufträge einer Produktfamilie geändert, oder storniert, so sind die Auswirkungen auf die gesamte Familie gering. Wird mit den aggregierten Werten der Gruppe geplant, beeinflussen kleine Änderungen das Ergebnis nicht auf die Weise, dass das Ergebnis unbefriedigend wäre und somit der Plan geändert werden müsste. Die Literatur ist in diesem Bereich inkonsistent. In einigen Fällen wird Aggregation als eine mögliche Quelle der Unsicherheit bezeichnet, wie in Kapitel 3.1 erwähnt, und anderen als eine potentielle Lösung. Der Widerspruch kann aufgelöst werden, wenn man den Fokus der Aggregation betrachtet. Eine Aggregation von Planungsannahmen über mehrere Planungsperioden kann die Informationsqualität derart verschlechtern, dass Unsicherheit entsteht. Ein adäquater Planungszeitraum, der für jede Problemstellung individuell gewählt werden muss, verbessert das Ergebnis. Eine Aggregation von Objekten innerhalb einer Periode ist nicht problematisch, solange mit der Summe der Aggregation geplant wird. Darauf beruht das Konzept der Gruppierung von Produkten zu Produktfamilien. Es bleibt somit festzuhalten, dass in der Literatur hin und wieder eine unsaubere Abgrenzung der Problemfelder vorgenommen wird.

Im Praxisteil der Arbeit wird gezeigt, dass die theoretisch erarbeiteten Themen im Rahmen der Aufgabe der Problemabgrenzung verwendet werden können, um Produktfamilien robust zu bilden. Weiters ist es gelungen, die Wahl der Clusterkriterien automatisch vorzunehmen, wodurch sich der Prozess der Produktfamilienbildung vereinfacht.

6.1 Ergebnisanalyse anhand der Forschungsfragen

Aufgrund der Ergebnisse der theoretischen Ausarbeitung der Themen und der praktischen Anwendung an einem Fallbeispiel, können die Forschungsfragen aus Kapitel 1.1 beantworten werden.

Ist es möglich, die für die Kapazitätsgrobplanung nötige Datenaggregation unter dem Gesichtspunkt der Robustheit vorzunehmen?

Ein zentraler Punkt der Kapazitätsgrobplanung ist die Datenaggregation. Durch die Zusammenfassung von Objekten zu Gruppen reagieren diese Gruppen auf Änderungen, die einzelne oder wenige Objekte betreffen, wenig bis nicht. Somit stellt die Aggregation für sich eine wichtige Maßnahme zur Robustheit von Planungsprozessen dar.

Wird diese Aggregation unter Robustheitskriterien vorgenommen, verstärkt sich dieser Effekt. Im speziellen Fall des Grobplanungsprozesses der AMAG konnte gezeigt werden, dass die Aggregation mithilfe des Belastungsprofils robust vorgenommen werden kann. Je einfacher das Belastungsprofil gehalten wird, desto einfacher ist es, die Robustheit zu bewerten. Es macht Sinn, nicht benötigte Anlagen von der Betrachtung auszuschließen und Anlagen, wenn möglich, zu Gruppen zusammenzufassen. Wobei wiederholt das Konzept der Aggregation angewandt wird, um Robustheit zu erlangen. Abbildung 41 zeigt einen Vergleich von zwei Belastungsprofilen. Das obere Profil gehört zu einer Produktfamilie, welche mit dem alten Algorithmus gebildet wurde, das untere Profil zu einer, welche mit dem neuen Algorithmus gruppiert wurde. Die dicke blaue Kurve entspricht jeweils dem Planartikel. Das obere Belastungsprofil ist uneinheitlich und beinhaltet somit eine Vielzahl von Anlagenfolgen. Die Robustheit ist nicht gegeben. Das Belastungsprofil darunter ist einheitlich, was auf eine homogene Anlagenfolge der Aufträge und somit hohe Robustheit der Produktfamilie hindeutet. Die Frage kann positiv beantwortet werden.

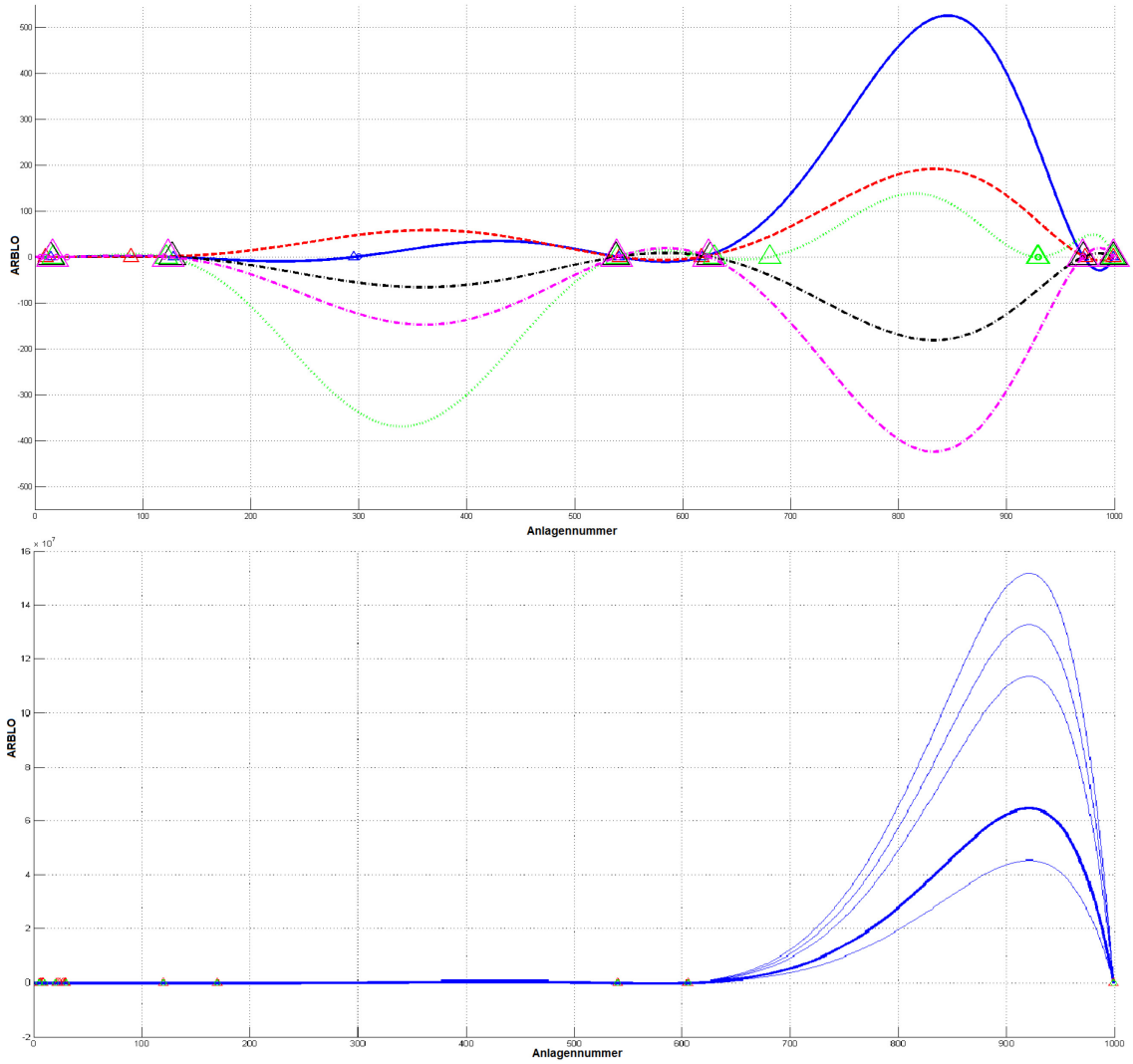


Abbildung 41: Belastungsprofil - alte vs. neue Produktfamilie[242]

Kann mithilfe des Robustheitsmaßes die Auswahl der Aggregationskriterien erfolgen?

Neben dem Distanzmaß und dem Gruppierungsverfahren beeinflussen die Aggregationskriterien das Ergebnis des Clustervorgangs maßgeblich. Bewertet man das Ergebnis des Clustervorganges mit dem Robustheitsmaß, werden Ergebnisse mit guter Robustheit bevorzugt. Diese Ergebnisse kommen durch die Wahl der Kriterien zustande. Somit können die Aggregationskriterien unter Zuhilfenahme des Robustheitsmaßes gewählt werden.

[242] Quelle: Eigene Darstellung

6.2 Weiterer Forschungsbedarf

Im Rahmen der Erstellung der Arbeit haben sich zusätzliche Fragen ergeben, die nicht mehr berücksichtigt werden konnten und einen zusätzlichen Forschungsbedarf bedingen.

Ist es möglich, die Fuzzy Mengenlehre über die gesamte Planungskette einzusetzen, um damit die Unsicherheiten in unterschiedlichen Bereichen zu behandeln? Die vorgestellten Modelle beschäftigen sich mit der Unsicherheit auf taktischer Ebene. Es stellt sich die Frage, ob solche Unsicherheiten ihre Quellen schon früher, z.B. auf strategischer Ebene wie der Rohstoffbeschaffung, oder auf der operativen Ebene durch den Ausfall von Anlagen oder der Erkrankung von Mitarbeitern, haben.

Weiters kann untersucht werden, inwieweit alternative Cluster- oder Informationsbeschaffungsmethoden eingesetzt werden können? Im Rahmen von Big Data und Data-Mining sind die Multiagentenverfahren solche. Da sich die erfasste Datenmenge in einem Unternehmen stetig erhöht, könnte es möglich sein, aus diesen Informationen Wissen zu gewinnen, welches die Planung verbessern würde. Softwareagenten können mit unterschiedlichen Tätigkeiten beauftragt werden und untereinander Informationen austauschen.[243] Ist es somit möglich, ein Modell der Planung zu erstellen, welches auf Veränderungen reagiert oder für Simulationen von Szenarien verwendet werden kann?

Die Thematik der Elastizität der Planung oder Supply Chain wurde in dieser Arbeit ausschließlich in Form einer Abgrenzung zur Robustheit behandelt. Es sollte jedoch untersucht werden, wie dieser alternative Ansatz integriert werden könnte und welche Methoden es in diesem Bereich gibt.

Speziell auf den praktischen Teil der Arbeit bezogen, stellt sich die Frage, wie das System reagiert, falls man nur mehr die Anlagenfolge für die Produktfamilienbildung verwenden würde, wenn also von einer Mischung aus Fertigungs- und Ablauffamilien zu reinen Ablauffamilien übergegangen werden würde. Dazu müssten über einen längeren Zeitraum die beiden Möglichkeiten parallel geführt und die Ergebnisse ausgewertet werden.

Weiters stellt sich in diesem Zusammenhang die Frage, ob es möglich ist, die Konfigurationsdatei nochmals zu vereinfachen, indem man die Vorgabe der Ober- und Untergrenze der Anzahl der Produktfamilien weglassen kann? Oder ob es sogar möglich ist, auf die Konfigurationsdatei vollständig zu verzichten?

[243] Vgl. Park/Oh (2007), S. 779.

Literaturverzeichnis

AMAG (2014): Kernkompetenzen @ AMAG - Austria Metall AG. .Abgerufen September 3, 2014, von http://www.amag.at/Kernkompetenzen.rollingmission.0.html

Arnold, C. (2013): Entwicklung fuzzybasierter Leitkomponenten für das Klimamanagement in der präventiven Konservierung. Wiesbaden: Springer Vieweg, 1. Auflage.

Arnold, D.; Isermann, H.; et al. (Hrsg.) (2008): Handbuch Logistik. Berlin: Springer Verlag, 3. Auflage.

Backhaus, K.; Erichson, B.; et al. (2011): Multivariate Analysemethoden; Eine anwendungsorientierte Einführung. Berlin: Springer Verlag, 13. Auflage.

Bartsch, H.-J. (2001): Taschenbuch mathematischer Formeln. München: Carl Hanser, 19. Auflage.

Berndt, R.; Cansier, A. (2007): Produktion und Absatz. Berlin: Springer Verlag, 2. erweiterte Auflage.

Biedermann, H. (2008): Anlagenmanagement. (Oberhofer, A.F.,Hrsg.), Erfolgspotentiale für Unternehmer und Führungskräfte. Köln: TÜV Media GmbH, 2. Auflage.

Bohnen, F.; Deuse, J. (2000): Entwicklung einer systematischen Vorgehensweise zur Produktnivellierung der varaintenreichen Kleinserienfertigung, Schlussbericht No. 15865 N/1. Dortmund: Technische Universität Dortmund, S. 123.

Bushuev, M. (2013): Convex optimisation for aggregate production planning. In: International Journal of Production Research, Vol. 52, Nr. 4, S. 1050–1058.

Chhaochhria, P.; Graves, S.C. (2013): A forecast-driven tactical planning model for a serial manufacturing system. In: International Journal of Production Research, Vol. 51, Nr. 23-24, S. 6860–6879.

Confalonieri, R.; Bregaglio, S.; et al. (2010): A proposal of an indicator for quantifying model robustness based on the relationship between variability of errors and of explored conditions. In: Ecological Modelling, Vol. 221, Nr. 6, S. 960–964.

Daniel, V.; Guide, R.; et al. (1997): Rough-cut capacity planning for remanufacturing firms. In: Production Planning & Control, Vol. 8, Nr. 3, S. 237–244.

Dyckhoff, H. (2006): Produktionstheorie; Grundzüge industrieller Produktionswirtschaft. Berlin: Springer Gabler, 5. Auflage.

Englberger, J.; Herrmann, F. (2013): Simulationsbasierte Ermittlung von Kapazitätsbelastungsfaktoren zur Produktionsprogrammplanung. In: Simulation in Produktion und Logistik, S. 631–640.

Erlach, K. (2010): Wertstromdesign; Der Weg zur schlanken Fabrik. Berlin: Springer, 2. Auflage.

Galbraith, J.R. (1973): Designing Complex Organizations. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1. Auflage.

Gebhard, M. (2009): Hierarchische Produktionsplanung bei Unsicherheit. Produktion und Logistik. Wiesbaden: Gabler, 1. Auflage.

Gebhard, M.; Kuhn, H. (2007): Robuste hierarchische Produktionsplanung mit Bedarfsszenarien. In: Otto, A.; Obermaier, R. (Hrsg.), Logistikmanagement. Wiesbaden: Deutscher Universitäts-Verlag, 1. Auflage., S. 161–184.

Geng, N.; Jiang, Z. (2009): A review on strategic capacity planning for the semiconductor manufacturing industry. In: International Journal of Production Research, Vol. 47, Nr. 13, S. 3639–3655.

GoogleScholar (2014): Google Scholar. .Abgerufen August 17, 2014, von http://scholar.google.at/

Gudehus, T. (2010): Logistik; Grundlagen - Strategien - Anwendungen. Berlin: Springer Verlag, 4. aktualisierte Auflage.

Günther, H.-O.; Tempelmeier, H. (2012): Produktion und Logistik. Berlin: Springer, 9. Auflage.

Gupta, A.; Maranas, C.D. (2003): Managing demand uncertainty in supply chain planning. In: 2nd Pan American Workshop in Process Systems Engineering, Vol. 27, Nr. 8–9, S. 1219–1227.

Gutenberg, E. (1983): Grundlagen der Betriebswirtschaftslehre. Berlin: Springer Verlag, 24. Auflage., Bd. Erster: Die Produktion.

Handl, A. (2010): Multivariate Analysemethoden; Theorie und Praxis multivariater Verfahren unter besonderer Berücksichtigung von S-PLUS. Statistik und ihre Anwendungen. Berlin: Springer Verlag, 2. Auflage.

Hans, E.W.; Herroelen, W.; et al. (2007): A hierarchical approach to multi-project planning under uncertainty. In: Omega, Vol. 35, Nr. 5, S. 563–577.

Herrmann, F. (2011): Operative Planung in IT-Systemen für die Produktionsplanung und -steuerung. IT-Management und -Anwendung. W: Vieweg+Teubner, 1. Auflage.

Ho, C. (1989): Evaluating the impact of operating environments on MRP system nervousness. In: International Journal of Production Research, Vol. 27, S. 1115–11135.

Ho, C.-J. (1989): Evaluating the impact of operating environments on MRP system nervousness. In: International Journal of Production Research, Vol. 27, Nr. 7, S. 1115–1135.

Jodlbauer, H. (2008): Produktionsoptimierung; Wertschaffende sowie kundenorientierte Planung und Steuerung. Wien: Springer Verlag, 2. Auflage.

Kaipia, R.; Korhonen, H.; et al. (2006): Planning nervousness in an demand supply network: an empirical study. In: International Journal of Logistics Management, Vol. 17, Nr. 1, S. 95–113.

Kistner, K.-P.; Steven, M. (2001): Produktionsplanung. Heidelberg: Physica-Verlag, 3. Auflage.

Klaus Erlach (2010): Wertstromdesign. Springer.

Klüver, C.; Klüver, J.; et al. (2012): Modellierung komplexer Prozesse durch naturanaloge Verfahren; Soft Computing und verwandte Techniken. Wiesbaden: Springer Vieweg, 2. Auflage.

Koop, A.; Moock, H. (2008): Lineare Optimierung; Eine anwendungsorientierte Einführung in Operations Research. Berlin: Springer Verlag, 1. Auflage.

Van Landeghem, H.; Vanmaele, H. (2002): Robust planning: a new paradigm for demand chain planning. In: Journal of Operations Management, Vol. 20, Nr. 6, S. 769–783.

Lanza, G.; Stoll, J.; et al. (2013): Measuring Global Production Effectiveness. In: Procedia {CIRP}, Vol. 7, Nr. 0, S. 31 – 36.

Laus, H.; Gillenkirch, R.M.; et al. (2012): Entscheidungstheorie. Berlin: Springer Gabler, 8. Auflage.

Leung, S.C.H.; Wu, Y. (2004): A robust optimization model for stochastic aggregate production planning. In: Production Planning & Control, Vol. 15, Nr. 5, S. 502–514.

MathWorks (2014): Hauptmerkmale - MATLAB - MathWorks Deutschland. .Abgerufen August 30, 2014, von http://www.mathworks.de/products/matlab/description1.html

Mele, F.D.; Guillén, G.; et al. (2007): An agent-based approach for supply chain retrofitting under uncertainty. In: ESCAPE-15 Selected Papers from the 15th European Symposium on Computer Aided Process Engineering held in Barcelona, Spain, May 29-June 1, 2005., Vol. 31, Nr. 5–6, S. 722–735.

Muchiri, P.; Pintelon, L. (2008): Performance measurement using overall equipment effectiveness (OEE): literature review and practical application discussion. In: International Journal of Production Research, Vol. 46, Nr. 13, S. 3517–3535.

Mula, J.; Peidro, D.; et al. (2010): The effectiveness of a fuzzy mathematical programming approach for supply chain production planning with fuzzy demand. In: Integrating the Global Supply Chain, Vol. 128, Nr. 1, S. 136–143.

Mula, J.; Poler, R.; et al. (2006): Models for production planning under uncertainty: A review. In: International Journal of Production Economics, Vol. 103, Nr. 1, S. 271 – 285.

Nakajima, S. (1988): Introduction to TPM: Total Productive Maintenance. Preventative Maintenance Series. Productivity Press, 1. Auflage.

Neumann, K. (1996): Produktions- und Operations- Management. Berlin: Springer, 1. Auflage.

Ogston, E.; Overeinder, B.; et al. (2003): A Method for Decentralized Clustering in Large Multi-Agent Systems. Proceedings of the Second International Joint Conference on

Autonomous Agent and Multi Agent Systems. Gehalten auf der Autonomous Agent and Multi Agent Systems, Melbourne: ACM, S. 789–796.

Park, J.-E.; Oh, K.-W. (2007,November): Multi-Agent Systems for Intelligent Clustering. World Academy of Science, Engineering and Technology.

Peters, M.L.; Zelewski, S. (2004): Möglichkeiten und Grenzen des „Analytic Hierarchy Process“ (AHP) als Verfahren zur Wirtschaftlichkeitsanalyse. In: Zeitschrift für Planung & Unternehmenssteuerung, Vol. 15, S. 295–324.

Lee-Post, A. (2000): Part family identification using a simple genetic algorithm. In: International Journal of Product Research, Vol. 38, Nr. 4, S. 793–810.

Rommelfanger, H.; Eickemeier, S.H. (2002): Entscheidungstheorie: Klassische Konzepte und Fuzzy-Erweiterungen. Berlin: Springer Verlag, 1. Auflage.

Sangwan, K.S.; Kodali, R. (2004): Fuzzy part family formation for cellular manufacturing systems. In: Production Planning & Control, Vol. 15, Nr. 3, S. 292–302.

Scheithauer, G. (2008): Zuschnitt- und Packungsoptimierung; Problemstellungen, Modellierungstechniken, Lösungsmethoden. Wiesbaden: Vieweg+Teubner, 1. Auflage.

Schneeweiß, C. (2002): Einführung in die Produktionswirtschaft. Berlin: Springer Verlag, 8. Auflage.

Scholl, A. (2001): Robuste Planung und Optimierung. Heidelberg: Physica-Verlag, 1. Auflage.

Schönsleben, P. (2007): Integrales Logistikmanagement; Operations und Supply Chain Management in umfassenden Wertschöpfungsnetzwerken. Berlin: Springer Verlag, 5. Auflage.

Schuh, G. (Hrsg.) (2006): Produktionsplanung und -steuerung; Grundlagen, Gestaltung und Konzepte. Berlin: Springer, 3. Auflage.

Science Direct (2014): ScienceDirect.com. .Abgerufen August 5, 2014, von http://www.sciencedirect.com/

Spiegler, V.L.M.; Naim, M.M.; et al. (2012): A control engineering approach to the assessment of supply chain resilience. In: International Journal of Production Research, Vol. 50, Nr. 21, S. 6162–6187.

Springerlink (2014): Springer - Home. .Abgerufen August 16, 2014, von http://link.springer.com/

Steffen C., E.; Tobias, W.; et al. (2013): Modell zur bedarfsgerechten Kapazitätsplanung. In: ZWF, Vol. 108, Nr. 9, S. 634–638.

Steven, M. (1994): Hierarchische Produktionsplanung. Heidelberg: Physica, 2. Auflage.

Steven, M. (2012): BWL für Ingenieure. München: Oldenbourg Wissenschaftsverlag, 4. Auflage.

Sun, C.-C. (2010): A performance evaluation model by intergrating fuzzy AHP und fuzzy TOPSIS methods. In: Expert Systems with Applications, Vol. 37, S. 7745–7754.

Syska, A. (2006): Produktionsmanagement; Das A-Z wichtiger Methoden und Konzepte für die Produktion von heute. Wiesbaden: Gabler, 1. Auflage.

TandF (2014): Taylor & Francis Online. .Abgerufen August 4, 2014, von http://www.tandfonline.com/

Volling, T.; Matzke, A.; et al. (2013): Planning of capacities and orders in build-to-order automobile production: A review. In: European Journal of Operational Research, Vol. 224, Nr. 2, S. 240–260.

Vollmuth, H.J. (2008): Controlling - Instrumente von A-Z. München: Rudolf Haufe Verlag, 7. Auflage.

Wannenwetsch, H. (2010): Integrierte Materialwirtschaft und Logistik; Beschaffung, Logistik, Materialwitschaft und Produktion. Berlin: Springer, 4. Auflage.

Weicker, K. (2007): Evolutionäre Algorithmen. Leitfäden der Informatik. Wiesbaden: Teubner, 2. Auflage.

Wieland, A.; Wallenburg, C.M. (2013): The influence of relational competencies on supply chain resilience: a realtional view. In: International Journal of Physical Distribution & Logistics Management, Vol. 43, Nr. 4, S. 300–320.

Zadeh, L.A. (1965): Fuzzy Sets. In: Information an Control, Vol. 8, S. 338–353.

Zimmermann, H.-J. (2008): Operations Research: Methoden und Modelle. Wiesbaden: Vieweg & Sohn, 2. Auflage.

Zsifkovits, H.E. (2013): Logistik. Grundwissen der Ökonomik. Konstanz: UVK, 1. Auflage.

Anhang

Dokumentation Programmcode

A. Planartikelermittlung

- Autor: Robert Bernerstätter
- Dateiname: Planartikelermittlung_Front.m
- Datum: 10.09.2013
- Version: 3.00
- Beschreibung: Das Programm ermittelt die Planartikel für die Kapazitätsgrobplanung. Für den Ablauf muss nur dieses Programm gestartet werden. Über ein einfaches GUI können Einstellungen vorgenommen werden. Die einzelnen Funktionen und Programmteile sind in den entsprechenden Dateien an den jeweiligen Stellen kommentiert. Nach Auswahl der Ablaufart werden Aufträge in Produktfamilien verdichtet. Die Planartikel werden gesucht und die jeweiligen Aufträge werden diesen zugeordnet. Das Ergebnis wird in einer Exceldatei gespeichert. Zu Beginn können zwei Arten des Programmablaufes ausgewählt werden. Die Erste ermittelt für jede SGE die Attribute auf Grund welcher die Produktfamilien gebildet werden. Hierfür wird ein genetischer Algorithmus angewandt. Diese Methode erfordert jedoch erhebliche Rechenzeit. Es muss mit einer Laufzeit von mehreren Stunden gerechnet werden. Die zweite Methode verwendet die Ergebnisse früherer Durchläufe und ermittelt die Attribute nicht. Daher ist die Laufzeit erheblich kürzer, als im Falle von Methode eins und beträgt einige Minuten bis zu einer halben Stunde. Das Programm benötigt als Informationsgrundlage die Auftragsdatenbank in einer vordefinierten Formatierung sowie die dazu gehörigen Arbeitsplandaten, ebenfalls in einer festgelegten Formatierung.

Festlegung der Grundeinstellungen

Es werden die mathematischen Verfahren sowie die Spaltenüberschriften für die Ausgabe, definiert.

```
clear all;
close all;
metrics = {'euclidean';'mahalanobis'};
algorithms = {'single';'ward'};
captionFamilien = {'SGE'; 'Wkst'; 'Dicke'; 'Breite'; 'Länge'; 'Zustand';...
    'Qualität'; 'Oberfläche'; 'BKG'; 'MatNr'; 'AuftragsNr'; 'PPTERM';...
    'Form'; 'GrundWkst'; 'VorPlanMat'; 'SV';'RoDm';'FamNr'};
captionPlanartikel = {'SGE'; 'Wkst'; 'Dicke'; 'Breite'; 'Länge';...
    'Zustand';'Qualität'; 'Oberfläche'; 'Anteil'; 'BKG'; 'MatNr';...
    'AuftragsNr'; 'FamNr'};
```

Popupmenü

Hier erfolgt die Auswahl für die beiden Modi zur Planartikelermittlung. Bei *Methode 1* `'Mit Attributermittlung (Laufzeitintensiv)'` werden die Clusterattribute mittels eines genetischen Algorithmus ermittelt. Auf Grund dieser Vorgehensweise, muss, je nach Datenmenge, mit einer Laufzeit von 3-6 Stunden gerechnet werden. Bei *Methode 2* `'Ohne Attributermittlung'` werden Vorgabewerte für die Clusterkriterien verwendet. Die Laufzeit bewegt sich im zweistelligen Minutenbereich.

```
aufwand = menu('Wähle Ablaufart',...
    'Mit Attributermittlung (Laufzeitintensiv)','Ohne Attributermittlung');
```

Festlegung der Dateipfade

Es werden die Dateien für die Eingabe und für die Ausgabe festgelegt. Die Auswahl erfolgt über ein Standardauswahlmenü von Windows. Als Erstes wird die Datei der Auftragsdaten angegeben. Es folgen die Dateien mit den Arbeitsplandaten und dem Speicherort für das Ergebnis. Beim Ergebnis handelt sich um die Datei mit den Planartikeln und Produktfamilien. Die letzte Datei, die gewählt werden muss, ist jene des Protokolls.

```
[dateiAuftrag, pfadAuftrag] = uigetfile({'*.xlsx'; 'xls'},...
    'Wähle Auftragsdaten','C:\Users\Robert\Documents\Conf_NoDrop\');
leseAuftrag = strcat(pfadAuftrag, dateiAuftrag);
[dateiArbeitsplan, pfadArbeitsplan] = uigetfile({'*.xlsx'; 'xls'},...
    'Wähle Arbeitsplandaten','C:\Users\Robert\Documents\Conf_NoDrop\');
leseArbeitsplan = strcat(pfadArbeitsplan, dateiArbeitsplan);
[dateiErgebnis, pfadErgebnis] = uiputfile('*.xlsx',...
    'Ergebnis speichern als...','C:\Users\Robert\Documents\Conf_NoDrop\');
schreibeErgebnis = strcat(pfadErgebnis, dateiErgebnis);
[dateiProtokoll, pfadProtokoll] = uigetfile('*.xlsx',...
    'Wähle Protokolldatei',...
    'C:\Users\Robert\Dropbox\Uni\1_Masterarbeit\EXCEL\Protokolle\');
leseProtokoll = strcat(pfadProtokoll,dateiProtokoll);
```

Methode 1

Hier muss zusätzlich die Konfigurationsdatei bekannt gegeben werden. Diese enthält alle wichtigen Daten um die Attribute eigenständig zu ermitteln. Die Konfigurationsdatei muss ebenfalls eine vorgegebene Formatierung einhalten. Nach dieser Festlegung startet neben der Ermittlung der Planartikel, die Laufzeitmessung. Sobald der Vorgang abgeschlossen wurde, wird das Protokoll sowie das Ergebnis der Planartikel und der Produktfamilien in Excelform gespeichert.

```
if(aufwand == 1)
    [dateiKonfig, pfadKonfig] = uigetfile({'*.xlsx'; 'xls'},...
        'Wähle Konfigurationsdatei',...
        'C:\Users\Robert\Dropbox\Uni\1_Masterarbeit\');
    leseKonfig = strcat(pfadKonfig, dateiKonfig);
    tic;
    [Planartikel, Produktfamilien, Protokolldatei, captionProtokoll]...
        = Hauptprogramm_langsam(leseAuftrag,leseKonfig, leseArbeitsplan,...
        leseProtokoll, metrics, algorithms);
    xlswrite(leseProtokoll,[captionProtokoll;Protokolldatei],'Protokoll');
    speichereErgebnis(captionPlanartikel,Planartikel, captionFamilien, ...
        Produktfamilien, schreibeErgebnis);
```

Methode 2

Im Unterschied zu Methode 1 muss keine Konfigurationsdatei angegeben werden, da alle wichtigen Informationen aus der Protokolldatei gewonnen werden können. Alles Weitere ist wie bei *Methode 1*. Wird keine Methode gewählt, bricht das Programm ab.

```
elseif(aufwand == 2)
    tic
    [Planartikel, Produktfamilien, Protokolldatei, captionProtokoll]...
        = Hauptprogramm_schnell(leseAuftrag, ...
        leseArbeitsplan, leseProtokoll);
    xlswrite(leseProtokoll,[captionProtokoll;Protokolldatei],'Protokoll');
    speichereErgebnis(captionPlanartikel, Planartikel, captionFamilien, ...
        Produktfamilien, schreibeErgebnis);
else
    fprintf('Programm abgebrochen!\n\n');
end
```

Laufzeitermittlung

Die Laufzeitmessung wird gestoppt und die Laufzeit in einer Dialogbox in Stunden, Minuten und Sekunden ausgegeben. Nach Bestätigung der Dialogbox ist das Programm beendet.

```
laufzeit = toc;
stunde = floor(laufzeit/3600);
minute = floor(mod(laufzeit,3600)/60);
sekunde = floor(mod(mod(laufzeit,3600),60));
zeit = [num2str(stunde),'h,',' ',num2str(minute),'min,',' ',...
    num2str(sekunde),'sek'];
lz = 'Laufzeit:';
txt = [lz,' ',zeit];
waitfor(msgbox(txt,'FERTIG!','warn'));
```

B. Hauptprogramm mit Attributermittlung

- **Autor**: Robert Bernerstätter
- **Dateiname**: Hauptprogramm_langsam.m
- **Datum**: 23.09.2013
- **Version**: 2.00
- **Beschreibung**: Das Programm ruft die benötigten Funktionen für die Planartikelsuche und Attributermittlung auf. Zu Beginn werden die Auftragsdaten mit den Arbeitsplandaten verknüpft. Im Anschluss wird die Konfigurationsdatei eingelesen. Danach startet mit einer For-Schleife über alles SGEs die Ermittlung der Planartikel. Auf Grund dieses Ergebnisses wird die Fitness der Lösung bestimmt. Mittels mehrerer Durchläufe werden verschiedene Lösungen generiert. Die Lösungen unterscheiden sich durch die heran gezogenen Attribute in den Auftragsdaten. Jene Lösung mit der besten Fitness wird gewählt und die heran gezogenen Attribute werden für diese SGE als verwendet gespeichert. Am Ende wird das Gesamtergebnis in einer Protokolldatei abgelegt. Diese Datei dient als Datengrundlage für den *Programmablauf ohne Attributermittlung*.

Funktionsaufruf

`Übergabeparameter`:

- **leseAuftrag**: Pfad der Auftragsdatenbank.
- **leseKonfig**: Pfad der Konfigurationsdatei.
- **leseArbeitsplan**: Pfad der Arbeitsplandatei.
- **leseProtokoll**: Pfad der Protokolldatei.
- **metrics**: Mathematische Verfahren zur Ermittlung der Distanzmatrix.
- **algorithms**: Mathematische Verfahren zur Clusterung.

`Rückgabeparameter`:

- **Planartikel**: Eine `n` x `m` Matrix mit den Planartikeln. Jede Zeile `n` enthält einen Planartikel. Die Spalten `m` enthalten die technischen Spezifikationen, die Auftrags- und Materialnummer, sowie eine laufende Nummerierung.
- **Produktfamilien**: Ein `n` x `1` Cellarray mit den Aufträgen für die Planartikel. Jede Zeile `n` enthält eine Zelle mit einer `j` x `k` Matrix. Jede Zeile `j` enthält einen Auftrag der SGE. Die Spalten `k` enthalten die technischen Spezifikationen, die Auftrags- und Materialnummer. In der letzten Spalte befindet sich eine Nummer, die den Auftrag seinem Planartikel eindeutig zuordnet (siehe Beschreibung des Rückgabeparameters **Planartikel**.
- **Protokolldatei**: Daten welche in die Protokolldatei geschrieben werden. Es handelt sich um eine `n` x `m` Matrix. Jede Zeile `n` entspricht einer SGE. Die Spalten `m` enthalten alle wichtigen Informationen über das Ergebnis des Programms in der Reihenfolge der Spaltenüberschriften (siehe *captionProtokoll*).
- **captionProtokoll**: Spaltenüberschriften für die Protokolldatei.

```
function [Planartikel, Produktfamilien, Protokolldatei, captionProtokoll] =...
    Hauptprogramm_langsam(leseAuftrag, leseKonfig, leseArbeitsplan,...
    leseProtokoll, metrics, algorithms)
```

Datenvor und -aufbereitung

Zuerst werden die Daten des Protokolls eingelesen, um diese ggf. anzupassen oder zu erweitern. Alle SGEs welche in der Konfigurationsdatei stehen, können vom Benutzer im Anschluss aus einer Liste zur Bearbeitung ausgewählt werden. Es folgt die Verknüpfung der Auftragsdaten mit den jeweiligen Arbeitsplänen. Etwaige ungültige Einträge werden entfernt, um einen Programmabsturz zu verhindern. Weiters werden Zusatzinformationen betreffend des Datentyps und der Gewichtung der Attribute eingelesen. Als letzter vorbereitender Schritt, werden diverse Variablen initialisiert.

```
[~,~,protokoll_alt] = xlsread(leseProtokoll,'Protokoll','A2:S50');
[~, sges] = xlsfinfo(leseKonfig);
[auswahl,~] = listdlg('ListSize',[160 350],'PromptString',...
    'Wähle SGE(s):','SelectionMode','multiple','Name','SGE Auswahl',...
    'CancelString','Abbruch','ListString',sges(1,1:size(sges,2)-2));

[ClusterMatrix, AuftragRaw, AuftragNum] = ...
    auftragsdatenArbeitsplanMerger(leseAuftrag, leseArbeitsplan);

nanind = find(isnan(AuftragNum(:,22)));
AuftragNum(nanind,:) = [];
AuftragRaw(nanind,:) = [];
ClusterMatrix(nanind,:) = [];
[attributinfoNum, attributinfoTxt, ~] =...
    xlsread(leseKonfig, size(sges ,2) - 1, 'B1:P3');
Planartikel{999,13} = [];
famcountergesamt = 1;
fitvektor_protokoll{size(sges,2)-2,7} = [];
planartikelcounter = 1;
Produktfamilien = cell(size(auswahl,2),2);
```

Die Möglichkeitsmatrix gibt, abhängig von der Anzahl der relevanten Attribute in der SGE, an, wieviele Individuen eine Population hat und wieviele Generationen durchlaufen werden sollen. Es gibt 2^n Möglichkeiten Attribute zur Produktfamilienverdichtung auszuwählen. `n` ist dabei die Anzahl der Attribute. Die Möglichkeitsmatrix ist eine `13` x `3` Matrix. Die erste Spalte gibt die Anzahl der relevanten Attribute an, die zweite Spalte die Größe der Population und die dritte Spalte die Anzahl der Generationen. Damit wird das Programm flexibel auf die Komplexität des Problems eingestellt.

```
MoeglichkeitsMatrix = [1 2 4;2 2 4;3 2 10;4 2 20;5 2 20;6 3 20;7 5 20;...
    8 10 20;9 20 20;10 40 20;11 80 20;12 120 20;13 180 20];
```

Programmdurchlauf

Im folgenden Programmteil werden die Aufträge zu den Produktfamilien verdichtet und die Lösung wird anhand einer Fitnessfunktion bewertet. Durch weitere Durchläufe mit

unterschiedlichen Clusterkonfigurationen wird versucht bessere Lösungen zu generieren. Die beste Lösung, welche nach diversen Abbruchkriterien akzeptiert wird, dient zur Ermittlung der Planartikel. Zusätzlich dazu werden die verwendeten Attribute in der besten Lösung ermittelt und in Form eines Protokolls zur weiteren Verwendung dem Benutzer zur Verfügung gestellt. Für jede SGE, welche aus der oben erschienen Liste ausgewählt wurde, wird die For-Schleife einmal durchlaufen. Zu Beginn werden diverse Variablen reinitialisiert, um diese im Anschluss mit den Werten aus der Konfigurationsdatei neu zu beschreiben.

```
for(auswahlcounter = 1:1:size(auswahl,2))
    sgecount = auswahl(1,auswahlcounter);
    woSGE = sprintf('Gerade bei %s', sges{sgecount});
    disp(woSGE);
    uGFehler = 0;
    oGFehler = 0;
    ProdFams = [];
    clear konfig;
    konfig = xlsread(leseKonfig, sgecount, 'B1:C2');
    anzClustUG = konfig(1, 1);
    anzClustOG = konfig(1, 2);
    gewichtUGRel = konfig(2, 1)/100;
    gewichtOGRel = konfig(2, 2)/100;
    bestSgeFitness = inf('double');
    clustAnz = [anzClustUG:1:anzClustOG];
    indexSGE = find(strcmp(cell2mat(AuftragRaw(:,1)), sges(1,sgecount)));
    verbesserungsabstand = 0;
```

Initialisierung der Ausgangspopulation

Vor dem Start des Clustervorgangs wird die Ausgangspopulation für jede durchlaufende SGE erstellt. Die If-Bedingung prüft vorab, ob für die SGE Einträge in der Auftragsdatenbank existieren. Sollte das nicht der Fall sein, wird der Schleifendurchlauf für diese SGE beendet. Wenn Datensätze für die SGE existieren, werden diese gewichtet. Die nummerischen Daten werden aus Performancegründen extrahiert. Die Distanzmatrix der Anlagenfolge wird mittels einer eigenen Funktion berechnet, da die Standardmethoden von MATLAB ungeeignet sind. Um die Populationsgröße und die Generationenanzahl zu ermitteln, werden die relevanten Attribute bestimmt. Mit der Anzahl der relevanten Attribute wird aus der Möglichkeitsmatrix die Populationsgröße bestimmt. Mittels eines einfachen Zufallsgenerators wird die Ausgangspopulation erstellt.

```
    if(indexSGE>0)

        ClusterMatrixSGE = ClusterMatrix(indexSGE,:);
        ClusterMatrixSGE(:,3) =...
            num2cell(cell2mat(ClusterMatrixSGE(:,3))./1000);
        ClusterMatrixSGE(:,4) =...
            num2cell(cell2mat(ClusterMatrixSGE(:,4))./1000);
        RawMatrixSGE = AuftragRaw(indexSGE,:);
        NumMatrixSGE = AuftragNum(indexSGE,:);
        DistanzMatrixAnlFolge =...
            getDistMatAnlagenfolge(RawMatrixSGE(:,20),1).*10;
        bkgSge = sum(NumMatrixSGE(:,9));
        ignoreidx = toIgnore(ClusterMatrixSGE, NumMatrixSGE(:,9),...
            bkgSge, gewichtUGRel);

        if(isequal(cell2mat(ClusterMatrixSGE(:,1)),...
                cell2mat(ClusterMatrixSGE(:,9))))
            ignoreidx = unique(sort([ignoreidx 9]));
        end;

        relevanteAttribute = size(ClusterMatrixSGE,2)-size(ignoreidx,2);
        MoeglichkeitsVektor = MoeglichkeitsMatrix...
            (MoeglichkeitsMatrix(:,1)==relevanteAttribute,:);
        generations = MoeglichkeitsVektor(1,2);
        populationSize_soll = MoeglichkeitsVektor(1,3);
        Population = randi([0,1],populationSize_soll,...
            size(ClusterMatrixSGE,2));
        Population(:,ignoreidx) = 0;
        Population(sum(Population,2)==0,:)=[];
        Population = unique(Population,'rows');
        populationSize_haben = size(Population,1);
        fitvektor_sge = [];

        if(populationSize_haben<populationSize_soll)
            Population = refill(Population,...
                abs(populationSize_haben-populationSize_soll),0);
            Population(:,ignoreidx) = 0;
            Population(sum(Population,2)==0,:)=[];
        elseif(populationSize_haben>populationSize_soll)
            Population = refill(Population,...
                abs(populationSize_haben-populationSize_soll),1);
            Population(:,ignoreidx) = 0;
            Population(sum(Population,2)==0,:)=[];
        end;
```

Clusterung

Die Anzahl der Schleifendurchläufe ergibt sich aus der Generationenanzahl in der Möglichkeitsmatrix. Nach dem vollständigen Durchlauf der Schleife, wird die beste ermittelte Lösung verwendet. Ein vorzeitiger Abbruch der Schleife erfolgt, falls in drei hintereinander folgenden Durchläufen keine Verbesserung des Ergebnisses erfolgte.

```
        for(gencount = 1:1:generations)

            fitvektor_gen = cell(populationSize_soll,7);
            woGen = sprintf('Generation %d', gencount);
            disp(woGen);
```

Jede Population in einer Generation repräsentiert eine andere Kombination der Attributauswahl zur Produktfamilienverdichtung. Für jede dieser Populationen wird die For-Schleife durchlaufen und die Verdichtung durchgeführt. Für die Lösung wird die Fitness bestimmt und mit der bis dahin Besten verglichen. Sollte sich eine bessere Fitness ergeben, wird die dazu gehörige Lösung als bis dahin beste gespeichert. Um die Verdichtung mathematisch korrekt durchzuführen, müssen die Attribute und folglich die korrespondierenden Werte in nominale und metrische unterschieden werden. Dazu werden sie in jeweils zwei eigene Matrizen gespeichert. Nach dem Clustervorgang wird die Fitness der Lösung ermittelt und für diese Population gespeichert. Nachdem für jede Population die beste Fitness ermittelt wurde, wird aus diesen Werten die beste Fitness der Generation bestimmt. Danach erfolgt ein Vergleich mit der bisher besten Fitness der SGE. Ist die Fitness der neuen Generation besser, als die bisherige beste Fitness der SGE, wird die neue Fitness der Generation als beste Fitness der SGE gespeichert und der Zähler für die Verbesserung der Fitness auf *0* gesetzt. Hat sich die Fitness in der Generation nicht verbessert, so wird der Zähler (*verbesserungsabstand*) um *1* erhöht. Die Suche wird abgebrochen, wenn nach drei Generationen keine Verbesserung gefunden wurde.

```
for(popcount = 1:1:size(Population,1))
    clusterindex = find(Population(popcount,:)==1);
    GenMatrix = ClusterMatrixSGE(:,clusterindex);
    attributVektor = attributinfoNum(:,clusterindex);
    nomIndx = find(attributVektor(1,:)==1);
    metrIndx = attributVektor(1,:)==0;
    AnlFolgebenoetogt = 0;

    if(~isempty(find(clusterindex==13, 1)))
        nomIndx(:,size(nomIndx,2))=[];
        AnlFolgebenoetogt = 1;
    end;

    GenMatrixNom = cell2mat(GenMatrix(:,nomIndx));
    GenMatrixMet = cell2mat(GenMatrix(:,metrIndx));

    if(AnlFolgebenoetogt ==1)
        [famind] = clusterung(GenMatrixMet, GenMatrixNom,...
            metrics, algorithms, clustAnz, DistanzMatrixAnlFolge);
    else
        [famind] = clusterung(GenMatrixMet, GenMatrixNom,...
            metrics, algorithms, clustAnz);
    end;

    [fitvektor, uGFehler, oGFehler] = fitness(NumMatrixSGE, ...
        famind, gewichtUGRel, gewichtOGRel, bkgSge,...
        clusterindex, uGFehler, oGFehler);
    fitvektor_gen(popcount,1) = sges(sgecount);
    fitvektor_gen(popcount,[2:7]) = fitvektor;
    Population(popcount,14) = fitvektor{1,1};
end;

bestGenFitness = min(cell2mat(fitvektor_gen(:,2)));
sgefitnessindex =...
    find(cell2mat(fitvektor_gen(:,2)) == bestGenFitness);
fitvektor_sge = ...
    [fitvektor_sge;fitvektor_gen(sgefitnessindex(1,1),:)];

if(bestGenFitness<bestSgeFitness)
    bestSgeFitness = bestGenFitness;
    verbesserungsabstand = 0;
else
    verbesserungsabstand = verbesserungsabstand+1;
end;

if(verbesserungsabstand == 3)
    break;
end;
```

Erstellung einer neuen Population

Es werden zwei Fälle unterschieden:

1. In der Population wurde wenigstens ein Individuum mit einer gültigen Lösung (Fitness < Unendlich) gefunden.
2. In der Population gibt es kein Individuum mit einer gültigen Lösung.

Im ersten Fall wird die neue Population mittels fitnessproportionaler Selektion gewählt. Die Kreuzung erfolg durch ein n-Punkt Verfahren. Auf die neue Population wird eine 9 prozentige Mutationswahrscheinlichkeit an zwei Stellen angewandt. Sollten sich idente Individuen ergeben, werden diese entfernt und der Pool wird durch zufällig erstellte Individuen aufgefüllt.

Im zweiten Fall produzierte die Populationen keine gültige Lösung. Daher kann keine zur Paarung herangezogen werden. Der neue Pool für die nächste Generation wird durch zufällig erstellte Individuen aufgefüllt.

```
            if(~isempty(find(Population(:,14)==0, 1)))
                break;
            end
            if(~isempty(find(Population(:,14)~=Inf, 1)))
                Population = fitnessproportional(Population);
                MutationMatrix = find(rand(1,size(Population,1))>=0.91);
                MutationMatrix([2,3],[1:size(MutationMatrix,2)]) = ...
                    sort(randi(size(ClusterMatrixSGE,2),2,...
                    size(MutationMatrix,2)));
                Population(MutationMatrix(1,:),:) = mutation(Population...
                    (MutationMatrix(1,:),:), MutationMatrix([2,3],:));
                Population(:,ignoreidx) = 0;
                Population(sum(Population,2)==0,:)=[];
                Population = unique(Population,'rows');
                populationSize_haben = size(Population,1);

                if(populationSize_haben<populationSize_soll)
                    Population = refill(Population,...
                        abs(populationSize_haben-populationSize_soll),0);
                    Population(:,ignoreidx) = 0;
                    Population(sum(Population,2)==0,:)=[];
                elseif(populationSize_haben>populationSize_soll)
                    Population = refill(Population,...
                        abs(populationSize_haben-populationSize_soll),1);
                    Population(:,ignoreidx) = 0;
                    Population(sum(Population,2)==0,:)=[];
                end;

            else
                Population = randi([0,1],populationSize_soll,...
                    size(ClusterMatrixSGE,2));
                Population(:,ignoreidx) = 0;
                Population(sum(Population,2)==0,:)=[];
                Population = unique(Population,'rows');
                populationSize_haben = size(Population,1);

                if(populationSize_haben<populationSize_soll)
                    Population = refill(Population,...
                        abs(populationSize_haben-populationSize_soll),0);
                    Population(sum(Population,2)==0,:)=[];
                elseif(populationSize_haben>populationSize_soll)
                    Population = refill(Population,...
                        abs(populationSize_haben-populationSize_soll),1);
                    Population(sum(Population,2)==0,:)=[];
                end;
            end;

        end;
```

Planartikelfindung

Es werden zwei Fälle unterschieden:

1. Es existiert wenigstens eine gültige Lösung.
2. Es existiert keine gültige Lösung.

Im ersten Fall - eine gültige Lösung existiert - wird die beste Lösung gespeichert (*fitvektor_sge*). Mit den Informationen wird für jede Produktfamilie ein Planartikel aus den Aufträgen, die dieser repräsentieren soll, ermittelt. Der Planartikel, sowie die dazugehörigen Artikel, werden in zwei Matrizen gespeichert. Diese beiden Matrizen werden am Ende des Programms als Ergebnis in eine Exceldatei geschrieben.

Im zweiten Fall - keine gültige Lösung existiert - wird eine Standardlösung generiert. Dafür wird nur die Anlagenfolge als Attribut herangezogen. Der Clustervorgang wird mittels Ward-Verfahren durchgeführt. Die Planartikelanzahl ergibt sich aus Ober- und Untergrenze für die Clusteranzahl aus der Konfigurationsdatei. Der Benutzer wird über die Tatsache, dass keine Lösung gefunden werden konnte, informiert. Er kann somit die Daten in der Konfigurationsdatei ändern und das Programm nur für diese SGE nochmals laufen lassen.

```
        if(~isempty(find(cell2mat(fitvektor_sge(:,2))~=Inf, 1)))
            fitvektor_sge = sortrows(fitvektor_sge,2);
            fitvektor_protokoll(sgecount,:) = fitvektor_sge(1,:);
            famvektor = unique(fitvektor_sge{1,6});
            RawMatrixSGE(:,23) = num2cell(fitvektor_sge{1,6});
            clear Planartikel_sge;
            Planartikel_sge = cell(size(famvektor,1),13);

            for(famcounter=1:1:size(famvektor,1))
                [PlanArt, ProdFam] = planartikelFinder...
                    (RawMatrixSGE(cell2mat(RawMatrixSGE(:,23))==...
                    famcounter,:),fitvektor_sge{1,7}, bkgSge);
                Planartikel_sge(famcounter,:) = PlanArt;
                Planartikel_sge{famcounter,13} = planartikelcounter;
                ProdFam(:,18) = num2cell(planartikelcounter);
                ProdFams = [ProdFams;ProdFam];
                planartikelcounter = planartikelcounter+1;
            end;

            Planartikel([famcountergesamt:famcountergesamt+...
                size(famvektor,1)-1],:) = Planartikel_sge;
            famcountergesamt = famcountergesamt+size(famvektor,1);
        else
            fprintf('Für die SGE %s konnte kein Ergebnis gefunden werden.\nEs wird eine
Standardlösung generiert, die nicht den Einstellungen der Konfigurationsdatei
entspricht.\n', sges{sgecount});
            if(uGFehler>oGFehler)
                fprintf('Verkleinern Sie den Wert der Gewichtsuntergrenze oder die
Obergrenze der Anzahl der Produktfamilien.\n\n');
            elseif(oGFehler>uGFehler)
                fprintf('Vergrößern Sie den Wert der Gewichtsobergrenze oder die
Obergrenze der Anzahl der Produktfamilien.\n\n');
            else
                fprintf('Es wird empfohlen die Einstellungen in der Konfigurationsdatei
```

```
weniger restriktiv zu machen.\n\n');
            end;

            Z = linkage(DistanzMatrixAnlFolge,'ward');
            famind = cluster(Z,'maxclust',...
                round(anzClustUG+((anzClustOG-anzClustUG)/2)));
            famvektor = unique(famind);
            fitvektor_protokoll(sgecount,1) = sges(sgecount);
            fitvektor_protokoll{sgecount,2} = inf('double');
            fitvektor_protokoll{sgecount,3} = ...
                round(anzClustUG+((anzClustOG-anzClustUG)/2));
            fitvektor_protokoll{sgecount,4} = 'ward';
            fitvektor_protokoll{sgecount,5} = 'genetisch';
            fitvektor_protokoll{sgecount,6} = famind;
            fitvektor_protokoll(sgecount,7) = num2cell(13);
            RawMatrixSGE(:,23) = num2cell(famind);
            clear Planartikel_sge;
            Planartikel_sge = cell(size(famvektor,1),13);

            for(famcounter=1:1:size(famvektor,1))
                [PlanArt, ProdFam] = planartikelFinder...
                    (RawMatrixSGE(cell2mat(RawMatrixSGE(:,23))==...
                    famcounter,:),13, bkgSge);
                Planartikel_sge(famcounter,:) = PlanArt;
                Planartikel_sge{famcounter,13} = planartikelcounter;
                ProdFam(:,18) = num2cell(planartikelcounter);
                ProdFams = [ProdFams;ProdFam];
                planartikelcounter = planartikelcounter+1;
            end;

            Planartikel([famcountergesamt:famcountergesamt+...
                size(famvektor,1)-1],:) = Planartikel_sge;
            famcountergesamt = famcountergesamt+size(famvektor,1);
        end;
```

```
    end;
    Produktfamilien(auswahlcounter,1) = sges(sgecount);
    Produktfamilien{auswahlcounter,2} = ProdFams;
end;
```

Erstellung des Protokolls

Für alle gewählten SGEs werden die wichtigsten Daten (mathematische Verfahren, Planartikelanzahl, Ober- und Untergrenzen, usw.) in einer Protokollmatrix (*Protokolldatei*) abgelegt. Diese Matrix wird in einer Exceldatei gespeichert.

```
if(~cellfun(@isempty,Planartikel(1,1)));
    [Protokolldatei, captionProtokoll, Planartikel] =  protokolliere...
        (Planartikel, fitvektor_protokoll, attributinfoTxt, protokoll_alt);
end;
```

```
end
```

C. Hauptprogramm ohne Attributermittlung

- **Autor**: Robert Bernerstätter
- **Dateiname**: Hauptprogram_schnell.m
- **Datum**: 23.09.2013
- **Version**: 2.00
- **Beschreibung**: Das Programm ruft die benötigten Funktionen für die Planartikelsuche auf. Es werden die Daten der Protokolldatei eingelesen und die Auftragsdaten mit dem Arbeitsplan verknüpft. Es folgt die Verdichtung zu den Produktfamilien aufgrund der Daten aus der Protokolldatei. Für jede Produktfamilie wird ein Planartikel ermittelt, der die Aufträge in der Familie möglichst gut repräsentiert. Am Ende erfolgt die Aktualisierung der Protokolldatei mit den neu gewonnen Daten.

Funktionsaufruf

Übergabeparameter:

- **leseAuftrag**: Pfad der Auftragsdatenbank.
- **leseArbeitsplan**: Pfad der Arbeitsplandatei.
- **leseProtokoll**: Pfad der Protokolldatei.

Rückgabeparameter:

- **Planartikel**: Eine `n` x `m` Matrix mit den Planartikeln. Jede Zeile `n` enthält einen Planartikel. Die Spalten `m` enthalten die technischen Spezifikationen, die Auftrags- und Materialnummer, sowie eine laufende Nummerierung.
- **Produktfamilien**: Ein `n` x `1` Cellarray mit den Aufträgen für die Planartikel. Jede Zeile `n` enthält eine Zelle mit einer `j` x `k` Matrix. Jede Zeile `j` enthält einen Auftrag der SGE. Die Spalten `k` enthalten die technischen Spezifikationen, die Auftrags- und Materialnummer. In der letzten Spalte befindet sich eine Nummer, die den Auftrag seinem Planartikel eindeutig zuordnet (siehe Beschreibung des Rückgabeparameters **Planartikel** .
- **Protokolldatei**: Daten welche in die Protokolldatei geschrieben werden. Es handelt sich um eine `n` x `m` Matrix. Jede Zeile `n` entspricht einer SGE. Die Spalten `m` enthalten alle wichtigen Informationen über das Ergebnis des Programms in der Reihenfolge der Spaltenüberschriften (siehe *captionProtokoll*).
- **captionProtokoll**: Spaltenüberschriften für die Protokolldatei.

```
function [Planartikel, Produktfamilien, Protokolldatei, captionProtokoll] = ...
    Hauptprogramm_schnell(leseAuftrag,  leseArbeitsplan, leseProtokoll)
```

Datenvor- und aufbereitung

Zuerst werden die Daten des Protokolls eingelesen, um diese ggf. anzupassen oder zu erweitern. Es folgt die Verknüpfung der Auftragsdaten mit den jeweiligen Arbeitsplänen. Etwaige ungültige Einträge werden entfernt, um einen Programmabsturz zu verhindern.

Weiters werden Zusatzinforamtionen betreffend des Datentyps und der Gewichtung der Attribute eingelesen. Als letzter vorbereitender Schritt, werden diverse Variablen initialisiert.

```
[protNum,protTxt,protokoll_alt] = xlsread(leseProtokoll,'Protokoll','A2:S50');
sges = protTxt(:,1);
[auswahl,~] = listdlg('ListSize',[160 350],'PromptString',...
    'Wähle SGE(s): ','SelectionMode','multiple',...
    'Name','SGE Auswahl','CancelString','Abbruch','ListString',sges);

[ClusterMatrix, AuftragRaw, AuftragNum] = ...
    auftragsdatenArbeitsplanMerger(leseAuftrag, leseArbeitsplan);

nanind = find(isnan(AuftragNum(:,22)));
AuftragNum(nanind,:) = [];
AuftragRaw(nanind,:) = [];
ClusterMatrix(nanind,:) = [];
[attributinfoNum, attributinfoTxt, ~] = ...
    xlsread(leseProtokoll, 'Attributinformationen', 'B1:P3');
Planartikel{999,13} = [];
famcountergesamt = 1;
fitvektor_protokoll{size(auswahl,2),7} = [];
planartikelcounter = 1;
Produktfamilien = cell(size(auswahl,2),2);
```

Programmdurchlauf

Für jede SGE, die aus der Liste ausgewählt wurde, wird der folgende Programmteil einmal durch gelaufen. Zu Beginn werden die Daten aus der Protokolldatei ausgelesen. Es handelt sich hierbei um Extremwerte aus vorherigen Durchläufen. Daher werden die Spannweiten der Unter- und Obergrenzen bei Bestellgewicht und Clusteranzahl erhöht, um einer eventuellen Veränderung in der Zusammensetzung des Auftragsmixes Rechnung zu tragen.

```
for(auswahlcounter = 1:1:size(auswahl,2));
```

```
    uGFehler = 0;
    oGFehler = 0;
    sgecount = auswahl(1,auswahlcounter);
    ProdFams = [];
    wo = sprintf('Gerade bei %s', sges{sgecount});
    disp(wo);
    clear konfig;
    konfig = protNum(sgecount,[1,2,3]);

    anzClustUG = konfig(1, 1)-3;
    if(anzClustUG<=0)
        anzClustUG = 2;
    end;

    anzClustOG = konfig(1, 1)+3;
    gewichtUGRel = (konfig(1,2)*0.8)/100;
    gewichtOGRel = (konfig(1,3)+2)/100;
    metrics = protTxt(sgecount,2);
    algorithms = protTxt(sgecount,3);
    clustAnz = [anzClustUG:1:anzClustOG];
    indexSGE = find(strcmp(cell2mat(AuftragRaw(:,1)), sges(sgecount,1)));
    fitvektor_sge = [];
```

Datenextraktion

Falls für die gewählte SGE Datensätze in den Auftragsdaten vorhanden sind, werden diese extrahiert und in eigene Matrizen geschrieben. Die Distanzmatrix für die Anlagenfolge wird mittels eines eigenen Verfahrens ermittelt. Die zu verwendeten Attribute, sowie die Datentypen, werden aus der Protokolldatei gelesen.

```
    if(indexSGE>0)

        ClusterMatrixSGE = ClusterMatrix(indexSGE,:);
        ClusterMatrixSGE(:,3) = ...
            num2cell(cell2mat(ClusterMatrixSGE(:,3))./1000);
        ClusterMatrixSGE(:,4) = ...
            num2cell(cell2mat(ClusterMatrixSGE(:,4))./1000);
        RawMatrixSGE = AuftragRaw(indexSGE,:);
        NumMatrixSGE = AuftragNum(indexSGE,:);
        DistanzMatrixAnlFolge = ...
            getDistMatAnlagenfolge(RawMatrixSGE(:,20),1).*10;
        bkgSge = sum(NumMatrixSGE(:,9));
        Population = protNum(sgecount,[4:16]);
        clusterindex = find(Population(1,:)==1);
        GenMatrix = ClusterMatrixSGE(:,clusterindex);
        attributVektor = attributinfoNum(:,clusterindex);
        nomIndx = find(attributVektor(1,:)==1);
        metrIndx = attributVektor(1,:)==0;
        AnlFolgebenoetogt = 0;

        if(~isempty(find(clusterindex==13, 1)))
            nomIndx(:,size(nomIndx,2))=[];
            AnlFolgebenoetogt = 1;
        end;

        GenMatrixNom = cell2mat(GenMatrix(:,nomIndx));
        GenMatrixMet = cell2mat(GenMatrix(:,metrIndx));
```

Clustervorgang und Ergebnisbeurteilung

Wenn die Anlagenfolge ein Clusterattribut ist, wird die Verdichtung zu den Produktfamilien durchgeführt. Anschließen wird für jede Lösung (der Unterschied liegt in der Anzahl der Produktfamilien) die Fitness bestimmt.

```
if(AnlFolgebenoetogt ==1)
    [famind] = clusterung(GenMatrixMet, GenMatrixNom,...
        metrics, algorithms, clustAnz, DistanzMatrixAnlFolge);
else
    [famind] = clusterung(GenMatrixMet, GenMatrixNom, ...
        metrics, algorithms, clustAnz);
end;

[fitvektor, uGFehler, oGFehler] = fitness(NumMatrixSGE, famind, ...
    gewichtUGRel, gewichtOGRel, bkgSge, ...
    clusterindex, uGFehler, oGFehler);
fitvektor_gen(1,1) = sges(sgecount);
fitvektor_gen(1,[2:7]) = fitvektor;
Population(1,14) = fitvektor{1,1};
bestGenFitness = min(cell2mat(fitvektor_gen(:,2)));
sgefitnessindex = ...
    find(cell2mat(fitvektor_gen(:,2)) == bestGenFitness);
fitvektor_sge = ...
    [fitvektor_sge;fitvektor_gen(sgefitnessindex(1,1),:)];
```

Planartikelfindung

Es werden zwei Fälle unterschieden:

1. Es existiert wenigstens eine gültige Lösung.
2. Es existiert keine gültige Lösung.

Im ersten Fall - eine gültige Lösung existiert - wird die beste Lösung gespeichert (*fitvektor_sge*). Mit den Informationen wird für jede Produktfamilie ein Planartikel aus den Aufträgen, die dieser repräsentieren soll, ermittelt. Der Planartikel, sowie die dazugehörigen Artikel, werden in zwei Matrizen gespeichert. Diese beiden Matrizen werden am Ende des Programms als Ergebnis in eine Exceldatei geschrieben.

Im zweiten Fall - keine gültige Lösung existiert - wird eine Standardlösung generiert. Dafür wird nur die Anlagenfolge als Attribut herangezogen. Der Clustervorgang wird mittels Ward-Verfahren durchgeführt. Die Planartikelanzahl ergibt sich aus Ober- und Untergrenze für die Clusteranzahl aus der Konfigurationsdatei. Der Benutzer wird über die Tatsache, dass keine Lösung gefunden werden konnte, informiert. Er kann somit die Daten in der Konfigurationsdatei ändern und das Programm nur für diese SGE nochmals laufen lassen.

```
if(~isempty(find(cell2mat(fitvektor_sge(:,2))~=Inf, 1)))
    fitvektor_sge = sortrows(fitvektor_sge,2);
    fitvektor_protokoll(sgecount,:) = fitvektor_sge(1,:);
    famvektor = unique(fitvektor_sge{1,6});
    RawMatrixSGE(:,23) = num2cell(fitvektor_sge{1,6});
    clear Planartikel_sge;
```

```
            Planartikel_sge = cell(size(famvektor,1),13);

            for(famcounter=1:1:size(famvektor,1))
                [PlanArt, ProdFam] = planartikelFinder(RawMatrixSGE...
                    (cell2mat(RawMatrixSGE(:,23))==famcounter,:),...
                    fitvektor_sge{1,7}, bkgSge);
                Planartikel_sge(famcounter,:) = PlanArt;
                Planartikel_sge{famcounter,13} = planartikelcounter;
                ProdFam(:,18) = num2cell(planartikelcounter);
                ProdFams = [ProdFams;ProdFam];
                planartikelcounter = planartikelcounter+1;
            end;

            Planartikel([famcountergesamt:famcountergesamt+...
                size(famvektor,1)-1],:) = Planartikel_sge;
            famcountergesamt = famcountergesamt+size(famvektor,1);
        else
            fprintf('Für die SGE %s konnte kein Ergebnis gefunden werden.\nEs wird eine
Standardlösung generiert, die nicht den Einstellungen der Konfigurationsdatei
entspricht.\n', sges{sgecount});
            if(uGFehler>oGFehler)
                fprintf('Verkleinern Sie den Wert der Gewichtsuntergrenze oder die
Obergrenze der Anzahl der Produktfamilien.\n\n');
            elseif(oGFehler>uGFehler)
                fprintf('Vergrößern Sie den Wert der Gewichtsobergrenze oder die
Obergrenze der Anzahl der Produktfamilien.\n\n');
            else
                fprintf('Es wird empfohlen die Einstellungen in der Konfigurationsdatei
weniger restriktiv zu machen.\n\n');
            end

            Z = linkage(DistanzMatrixAnlFolge,'ward');
            famind = cluster(Z,'maxclust',...
                round(anzClustUG+((anzClustOG-anzClustUG)/2)));
            famvektor = unique(famind);
            fitvektor_protokoll(sgecount,1) = sges(sgecount);
            fitvektor_protokoll{sgecount,2} = inf('double');
            fitvektor_protokoll{sgecount,3} = ...
                round(anzClustUG+((anzClustOG-anzClustUG)/2));
            fitvektor_protokoll{sgecount,4} = 'ward';
            fitvektor_protokoll{sgecount,5} = 'genetisch';
            fitvektor_protokoll{sgecount,6} = famind;
            fitvektor_protokoll(sgecount,7) = num2cell(13);
            RawMatrixSGE(:,23) = num2cell(famind);
            clear Planartikel_sge;
            Planartikel_sge = cell(size(famvektor,1),13);

            for(famcounter=1:1:size(famvektor,1))
                [PlanArt, ProdFam] = planartikelFinder...
                    (RawMatrixSGE(cell2mat(RawMatrixSGE(:,23))==...
                    famcounter,:),13, bkgSge);
                Planartikel_sge(famcounter,:) = PlanArt;
                Planartikel_sge{famcounter,13} = planartikelcounter;
                ProdFam(:,18) = num2cell(planartikelcounter);
                ProdFams = [ProdFams;ProdFam];
                planartikelcounter = planartikelcounter+1;
            end;
```

```
            Planartikel([famcountergesamt:famcountergesamt+...
                size(famvektor,1)-1],:) = Planartikel_sge;
            famcountergesamt = famcountergesamt+size(famvektor,1);

        end;
        Produktfamilien(auswahlcounter,1) = sges(sgecount);
        Produktfamilien{auswahlcounter,2} = ProdFams;

    end;

end;
```

Erstellung des Protokolls

Für alle gewählten SGEs werden die wichtigsten Daten (mathematische Verfahren, Planartikelanzahl, Ober- und Untergrenzen, usw.) in einer Protokollmatrix (*Protokolldatei*) abgelegt. Diese Matrix wird in einer Exceldatei gespeichert.

```
if(~cellfun(@isempty,Planartikel(1,1)));
    [Protokolldatei, captionProtokoll, Planartikel] =  protokolliere...
        (Planartikel, fitvektor_protokoll, attributinfoTxt, protokoll_alt);
end;

end
```

D. Verknüpfung Auftragsdaten und Arbeitsplandaten

- **Autor**: Robert Bernerstätter
- **Dateiname**: auftragsdaten_arbeitsplan_merger.m
- **Datum**: 23.09.2013
- **Version**: 2.00
- **Beschreibung**: Das Programm stellt die Verbindung zwischen den Auftragsdaten und den Arbeitsplandaten her. Für jeden Auftrag wird mittels der Auftragsnummer (Kommissionsnummer + Positionsnummer) der korrespondierende Arbeitsplan gesucht. Aus dem Arbeitsplan wird die Arbeitsvorgangsfolge bestimmt und als Attribut für den Auftrag angelegt. Dieses Attribut dient als Clusterkriterium. Zudem wird ein so genanntes Belastungsprofil erstellt. Dabei handelt es sich um eine n x 2 Matrix.

Funktionsaufruf

Übergabeparameter:

- **pfad_Auftragsdaten**: Pfad für die Auftragsdatenbank.
- **pfad_Arbeitsplandaten**: Pfad für die Arbeitsplandatei.

Rückgabeparameter:

- **ClusterMatrix**: Eine `n` x `m` Matrix mit den Clusterattributen. Jede Zeile `n` enthält einen Auftrag, jede Spalte `m` den korrespondierenden Wert für ein Attribut.
- **AUFRaw**: Eine `n` x `m` Matrix mit den Rohdaten der Auftragsdatenbank. Jede Zeile `n` einthält einen Auftrag, jede Spalte `m` den korrespondierenden Wert eines Attributes. Diese Attribute sind neben jenen der Clustermatrix, auch nicht für den Clustervorgang relevante Daten. Dazu zählen das Belastungsprofil, Auftrags- und Materialnummer oder Bestellgewicht.
- **AUFNum**: Eine `n` x `m` Matrix mit den nummerischen Daten der Auftragsdatenbank. Jede Zeile `n` enthält einen Auftrag, jede Spalte `m` den korrespondierenden Wert des Attributes. Die Matrix ist mit der *AUFRaw* insofern ident, als dass sie alle als nummerischen Daten enthält. Spalten die Textdaten, Vektoren oder Matrizen enthalten, sind als *nan* (not a number) beschrieben. Dadurch wird gewährleistet, dass z.B. Spalte 4 der *AUFRaw* die gleichen Daten besitzt wie Spalte 4 der *AUFNum*, vorausgesetzt sie sind nummerisch.

```
function [ClusterMatrix, AUFRaw, AUFNum] = auftragsdaten_arbeitsplan_merger...
    (pfad_Auftragsdaten, pfad_Arbeitsplandaten)
```

Datenaufbereitung

Die Nummerischen- und Rohdaten der Auftragsdaten und Arbeitsplandaten werden eingelesen und vereinfacht. Die Funktion *vereinfacheAnlagen* reduziert die Komplexität der Datenbasis, indem nicht benötigte Anlagen oder Arbeitsvorgänge entfernt werden. Andere Anlagen, wie zum Beispiel Öfen können zu einer Anlagennummer zusammen gefasst werden. Für die spätere Bewertung der Fitness ist diese Art Vereinfachung eine erhebliche Vereinfachung. Arbeitsplandaten ist eine `n` x `m` Matrix. Die Anzahl der Zeilen `n` entspricht der Anzahl an Aufträgen, die Anzahl der Spalten `m` entspricht der theoretischen Anzahl von Arbeitsvorgängen. Die Zelleneinträge enthalten die Arbeitszeiten je los für einen Auftrag bei dem Arbeitsvorgang, dessen Nummer der Spaltennummer entspricht.

```
warning('off','all');

[AUFNum, ~, AUFRaw] = xlsread(pfad_Auftragsdaten,'A2:S50000');
[ARBNum, ~, ARBRaw] = xlsread(pfad_Arbeitsplandaten,'A2:G500000');

AUFRaw = AUFRaw([1:size(AUFNum,1)],:);
ARBRaw = ARBRaw([1:size(ARBNum,1)],:);
ARBRaw(isnan(cell2mat(ARBRaw(:,4))),:) = [];

ARBRaw = vereinfacheAnlagen(ARBRaw);
Arbeitsplandaten = ones(size(AUFRaw,1),999);
Arbeitsplandaten(:) = -1;
RowsAUFRaw = size(AUFRaw,1);
ColsAUFRaw = size(AUFRaw,2);
ARBRaw_Auftragsnummern = cell2mat(ARBRaw(:,1));
```

Sucher der Anlagenfolge und Erstellung des Belastungsprofils

Für jeden Auftrag (*searcher* enthält die Auftragsnummer), wird der Arbeitsplan gesucht. Sollten Anlagen oder Arbeitsvorgänge mit der gleichen Nummer mehrmals direkt hintereinander folge, so werden diese in der Anlagenfolge nur einmal berücksichtigt. Im Belastungsprofil werden die Arbeitszeiten je Los aufsummiert. Diese Vorgehensweise reduziert weiters die Komplexität für den späteren Clustervorgang und die Fitnessbewertung.

```
for(i = 1:1:RowsAUFRaw)
    searcher = AUFRaw{i,11};
    delind = find(ARBRaw_Auftragsnummern == searcher);
    if(~isempty(delind))
        RF = [];
        A = ARBRaw(delind,:);
        %ARBRaw(delind,:) = [];
        %ARBRaw = ARBRaw(~ismember(1:size(ARBRaw,1),delind),:);
        %if(isempty(find(cell2mat(A(:,3))==950, 1)))
        if(cell2mat(A(1,3))~=950 || strcmp(AUFRaw{i,1},'WC') ||
strcmp(AUFRaw{i,1},'WG'))
         %   disp('D hats nix');
            B=[];
            for(j = 1:1:size(A,1))
                try
                    if(A{j,3}==A{j-1,3})
                        A{j,5} = A{j,5} + A{j-1,5};
                        B=[B,j-1];
                    end;
                catch err

                end;
            end;
            Arbeitsplandaten(i,cell2mat(A(:,3))) = Arbeitsplandaten(i,cell2mat(A(:,3)))
+ cell2mat(A(:,5)')+1;
            A(B,:)=[];
            RF = {A(:,8)'};
            AUFRaw(i,ColsAUFRaw+1) = RF;

            RF = cell2mat(A(:,3));
            Zeiten = Arbeitsplandaten(i,RF)';
            Profil = [RF Zeiten];
            AUFRaw{i,ColsAUFRaw+2} = Profil;
       % else
        %    disp('Da hats was');
        end;
    end
end
```

Herstellung der Datenkonsitenz

Einträge für die kein Arbeitsplan gefunden wurde, werden gelöscht und stehen für den Clustervorgang nicht zu Verfügung. Das geschieht daher, da durch das Fehlen des Arbeitsplans kein Belastungsprofil angelegt werden konnte und dadurch der Auftrag nicht in die Fitnessbewertung einbezogen werden kann. Die Fitnessbewertung erfolgt aufgrund

der Nullstellen des Polynoms, welches durch die Datenpunkte des Belastungsprofils gelegt werden kann. Spalten die keine nummerischen Daten enthalten, werden mit *nan* (not a number) beschrieben. Dadurch enthalten die nummerische und Rohdatenmatrix bei nummerischen Daten in den gleichen Spalten, die gleichen Informationen. Die *ClusterMatrix* enthält die Spalten aller relevanten Clusterattribute.

```
A = find(sum(Arbeitsplandaten,2) == -999);
AUFRaw(A,:) = [];
AUFNum(A,:) = [];
AUFRaw(:,ColsAUFRaw+3) = nstFinder(AUFRaw(:,ColsAUFRaw+2));
AUFNum(:,[1,18:21]) = nan;
AUFNum(:,[2:17]) = cell2mat(AUFRaw(:,2:17));
AUFNum(:,22) = cell2mat(AUFRaw(:,22));
ClusterMatrix = AUFRaw(:,[2 3 4 5 6 7 8 13 14 15 16 17 ColsAUFRaw+1]);
warning('on','all');

end
```

E. Vereinfachung des Arbeitsplans

- **Autor**: Robert Bernerstätter
- **Dateiname**: vereinfacheAnlagen.m
- **Datum**: 23.09.2013
- **Version**: 1.50
- **Beschreibung**: Um die Komplexität der Arbeitsplandaten zu verringern und dadurch die Qualität der Fitnessfunktion zu erhöhen, wird dieses Programm ausgeführt. Anlagen oder Arbeitsvorgänge, die für eine kapazitive Auswertung nicht benötigt werden, werden entfernt. Die in ihrer Tätigkeit identen und bei der weiteren Planung gemeinsam betrachteten Anlagen und Arbeitsvorgänge, werden unter einer Nummer zusammengefasst.

Funktionsaufruf

Übergabeparameter:

- **ARBRaw**: Die ursprüngliche Matrix mit den Rohdaten des Arbeitsplans.

Rückgabeparameter:

- **ARBRaw**: Die vereinfachte Matrix mit den Rohdaten des Arbeitsplans.

```
function [ARBRaw] = vereinfacheAnlagen(ARBRaw)
```

Anlagen entfernen

Angabe der Anlagennummern, deren Datensätze gelöscht werden sollen.

Syntaxbeschreibung für *Entferne*: `955:1:962` - Anlagen von 955 in 1er Schritten bis 962

```
Entferne = [275 500 795 800 (809:1:815) 820 830 835 840 845 852 870 900 ...
            920 921 (955:1:962) (964:1:970) (974:1:998)];
ARBRaw_Anlagennummer = cell2mat(ARBRaw(:,3));
ind = [];

for(i=1:1:size(Entferne,2))
    ind = [ind;find(ARBRaw_Anlagennummer == Entferne(1,i))];
end;
```

Anlagen ersetzen

Anlagen und Arbeitsvorgänge werden unter einer einheitlichen Nummer zusammengefasst.

Syntaxbeschreibung für *Ersetzte*: Erste Nummer durch zweite Nummer; ersetze erste Nummer durch zweite Nummer; usw. (*Ersetze 322 durch 321*)

```
Ersetze = [322 321; 323 321; 580 605; 582 605; 606 605; 607 605; 608 605;...
           611 605; 612 605; 613 605; 614 605; 542 540; 545 540; 643 644;...
           641 644; 667 666; 669 668; 670 668; 671 668; 672 668; 674 673;...
           675 673];
ARBRaw(ind,:) = [];
ARBRaw_Anlagennummer = cell2mat(ARBRaw(:,3));

for(i=1:1:size(Ersetze,1))
    ARBRaw(ARBRaw_Anlagennummer == ...
        Ersetze(i,1),3) = mat2cell(Ersetze(i,2));
end;

ARBRaw(:,8) = cellstr(num2str(cell2mat(ARBRaw(:,3))));

end
```

F. Berechnung der Varianz der Nullstellen des Belastungsprofilpolynoms

- **Autor**: Robert Bernerstätter
- **Dateiname**: nstFinder.m
- **Datum**: 23.09.2013
- **Version**: 3.00
- **Beschreibung**: Das Programm ermittelt die Nullstellen eines 'fiktiven' Polynoms, welches durch die Punkte des Belastungsprofils gelegt wird. Die Varianz der Verteilung der Nullstellen wird berechnet und als Parameter für die Fitnessbewertung zurückgegeben.

Funktionsaufruf

Übergabeparameter:

- **Belastungsprofil**: Ein `n` x `1` Cellarray. Jede Zeile `n` entspricht einem Auftrag. Jede Cell enthält das Belastungsprofil des Auftrages der Zeile. Es handelt sich dabei um eine `k` x `2` Matrix. Jede Zeile `k` enthält einen Arbeitsvorgang. Die erste Spalte gibt die Anlagen- bzw. Arbeitsvorgangsnummer an, die zweite Spalte die Arbeitszeit pro Los bei dieser Anlagen- bzw. Arbeitsvorgangsnummer.

Rückgabeparameter:

- **nstvarvector**: Ein `n` x `1` Cellarray. Jede Zeile `n` entspricht einem Auftrag. Jede Cell enthält die Varianz der Nullstellen des Polynoms, welches sich aus dem Belastungsprofil ergibt.

```
function [nstvarvector] = nstFinder(Belastungsprofil)
```

```
nstvarvector{size(Belastungsprofil,1),1} = [];
```

Berechnung

Für jeden Auftrag werden die Daten aus dem Belastungsprofil genommen. *X* beinhaltet die Anlagennummern, *Y* die dazugehörigen Zeiten pro Los. Durch diese Pseudo-Datenpunkte wird ein Polynom (*n Punkte-1*)-Grades gefittet. Für dieses Polynom werden die Nullstellen ermittelt. Die nicht-imaginären Nullstellen sind für die weitere Betrachtung relevant. Diese werden auf zwei Nachkommastellen gerundet. Nullstellen kleiner 0 und größer 1000 werden ebenfalls ignoriert, da es sich dabei um Nullstellen handelt, die sich aus einem Ein-, bzw. Ausschwingvorgang ergeben (Runges Phänomen).

```
for(i = 1:1:size(Belastungsprofil,1))
    X = Belastungsprofil{i,:}(:,1);
    Y = Belastungsprofil{i,:}(:,2);
    pf = polyfit(X,Y,size(X,1)-1);
    try
        nst = roots(pf);
    catch err
        %disp('Fehler');
    end;
    nst = roundn(nst(imag(nst)==0),-2);
    nst = nst(nst>=0 & nst<=1000);
    nstvar = var(nst);
    nstvarvector{i,1} = nstvar;
end;
```

```
end
```

G. Berechnung der Distanzmatrix aufgrund der Anlagenfolge

- **Autor**: Robert Bernerstätter
- **Dateiname**: getDistMatAnlagenfolge.m
- **Datum**: 23.09.2013
- **Version**: 1.46
- **Beschreibung**: Das Programm berechnet die Distanzmatrix für das Attribut der Anlagenfolge. Da dieses Attribut für jede Stelle mehrere Einträge hat, kann es nicht in die Standardermittung der Distanzmatrix mittels der integrierten *MATLAB*-Funktionen mit einbezogen werden. Für die Berechnung wird eine leicht adaptierte Methode nach `'Anita Lee-Post: Part family identification using a simple genetic algorithm (2000)'` verwendet.

Funktionsaufruf

Übergabeparameter:

- **Anlagenfolge**: Ein Vektor der Länge `n`, wobei `n` die Anzahl der Aufträge ist. Jeder Eintrag ist ein Cellarray mit der Anlagenfolge.
- **flagAnzahl**: Setzt ein Flag ob für die Berechnung die Gesamtanzahl der verschiedenen Anlagenfolgen benötigt wird.

Rückgabeparameter:

- **DistMatAnlage**: Die symmetrische und quadratische `n` x `n` Distanzmatrix für die Anlagenfolge. Jede Zeile und Spalte `n` steht für einen Auftrag. Eingetragen wird die Unähnlichkeit zwischen den Anlagenfolgen der Zeile und Spalte.

```
function [DistMatAnlage] = getDistMatAnlagenfolge(Anlagenfolge, flagAnzahl)
```

Datenvorbereitung

Es wird eine Rohmatrix mit nur Nullen erstellt. Das geschieht aus Performancegründen. Sollte die Anzahl der Anlagenfolge benötigt werden, wird diese mit der Funktion *getAnzAnlagenFolge* bestimmt.

```
DistMatAnlage = zeros(size(Anlagenfolge,1));

if(flagAnzahl == 1)
    [anzFolgen] = getAnzAnlagenFolge(Anlagenfolge);
end;
```

Erstellung der Distanzmatrix

Für jeden Auftrag (erste For-Schleife) wird jeweils die Unähnlichkeit zu allen anderen Aufträgen (zweite For-Schleife) bestimmt. Unterschieden werden zwei Fälle unterschieden.

1. Die Anzahl der Anlagenfolgen wird benötigt.
2. Die Anzahl der Anlagenfolgen wird **nicht** benötigt.

```
for(i=1:1:size(Anlagenfolge,1)-1)
    for(j=i+1:1:size(Anlagenfolge,1))
```

1.Fall - Anlagenfolge benötigt

```
        if(flagAnzahl==1)
            DistMatAnlage(i,j) = 1-calcDistAnlagenfolge...
                (Anlagenfolge([i,j],:),anzFolgen);
```

2.Fall - Anlagenfolge nicht benötigt

```
        else
            DistMatAnlage(i,j) = 1-calcDistAnlagenfolge...
                (Anlagenfolge([i,j],:));
        end;
        DistMatAnlage(j,i) = DistMatAnlage(i,j);
    end;
end;
```

```
end
```

H. Berechnung der Distanz zwischen zwei Anlagen

- **Autor**: Robert Bernerstätter
- **Dateiname**: calcDistAnlagenfolge.m
- **Datum**: 23.09.2013
- **Version**: 1.50
- **Beschreibung**: Das Programm berechnet den Wert für die Distanzmatrix zwischen zwei Anlagenfolgen. Dafür werden die jeweiligen Positionen der beiden Anlagenfolgen verglichen und die Übereinstimmungen gezählt. Das Ergebnis wird von *1* subtrahiert, um von der Ähnlichkeit zur Unähnlichkeit (Distanz) zu kommen. Die Division durch die Anzahl der Anlagenfolge standardisiert das Ergebnis.

Funktionsaufruf

Übergabeparameter:

- **Anlagenfolge**: Cellarray mit zwei Anlagenfolgen zwischen denen die Distanz berechnet werden soll.
- **anzFolgen**: Anzahl der unterschiedlichen Anlagenfolgen

Rückgabeparameter:

- **DistAnlagenfolge**: Die Distanz zwischen den beiden Anlagenfolgen des Übergabeparameters *Anlagenfolge*.

```
function [DistAnlagenfolge] = calcDistAnlagenfolge(Anlagenfolge, anzFolgen)
```

Initialisierung

Die beiden Anlagenfolgen werden separiert und die Länge bestimmt.

```
Anlage1 = Anlagenfolge{1,1};
Anlage2 = Anlagenfolge{2,1};
sizeAnl1 = size(Anlage1,2);
sizeAnl2 = size(Anlage2,2);
```

Berechnung

Die Distanz wird ermittelt indem auf jeder Position der ersten Anlagenfolge überprüft wird, ob auf der gleichen Position der zweiten Anlagenfolge die gleiche Nummer für den Arbeitsvorgang steht. Die Summe der Übereinstimmungen wird gezählt und von 1 subtrahiert um die Unähnlichkeit zu erhalten. Er werden zwei Fälle bei der Berechnung unterschieden:

1. Die Anzahl der Anlagenfolgen wird **nicht** berücksichtigt.
2. Die Anzahl der Anlagenfolgen wird berücksichtig.

In beiden Fällen werden drei mögliche Szenarien unterschieden:

1. Die beiden Anlagenfolgen sind gleich lang.
2. Anlagenfolge 2 ist länger als Anlagenfolge 1.
3. Anlagenfolge 1 ist länger als Anlagenfolge 2.

Im ersten Fall ist keine weitere Aktion nötig. Die Distanz kann sofort berechnet werden. Im zweiten Fall muss Anlagenfolge 1 *verlängert* werden. Die fehlenden Einträge werden mit **000** aufgefüllt. Somit kann wieder jede Position von Anlagenfolge 2 mit der korrespondierenden von Anlagenfolge 1 verglichen werden. Im dritten Fall verhält sich die Vorgehensweise analog zu jener wenn Anlagenfolge 2 länger ist als Anlagenfolge 1.

Fall 1 - Ohne Anzahl

```
if(nargin==1)
```

Fall 1.1 - Anlagenfolgen gleich lang

```
    if(sizeAnl1 == sizeAnl2)
        DistAnlagenfolge = 1-(sum(~strcmp(Anlage1,Anlage2))/sizeAnl1);
```

Fall 1.2 - Anlagenfolge 2 länger

```
    elseif(sizeAnl1 < sizeAnl2)
        Anlage1(1,sizeAnl1+1:sizeAnl2) = {000};
        DistAnlagenfolge = 1-(sum(~strcmp(Anlage1,Anlage2))/sizeAnl2);
```

Fall 1.3 - Anlagenfolge 1 länger

```
    else
        Anlage2(1,sizeAnl2+1:sizeAnl1) = {000};
        DistAnlagenfolge = 1-(sum(~strcmp(Anlage1,Anlage2))/sizeAnl1);
    end;
end;
```

Fall 2 - Mit Anzahl

```
if(nargin==2)
```

Fall 2.1 - Anlagenfolgen gleich lang

```
    if(sizeAnl1 == sizeAnl2)
        DistAnlagenfolge = 1-(sum(~strcmp(Anlage1,Anlage2))/anzFolgen);
```

Fall 2.2 - Anlagenfolge 2 länger

```
    elseif(sizeAnl1 < sizeAnl2)
        Anlage1(1,sizeAnl1+1:sizeAnl2) = {000};
        DistAnlagenfolge = 1-(sum(~strcmp(Anlage1,Anlage2))/anzFolgen);
```

Fall 2.3 - Anlagenfolge 1 länger

```
    else
        Anlage2(1,sizeAnl2+1:sizeAnl1) = {000};
        DistAnlagenfolge = 1-(sum(~strcmp(Anlage1,Anlage2))/anzFolgen);
    end    ;
end;
```

```
end
```

I. Berechnung Anzahl unterschiedlichen Anlagenfolgen

- **Autor**: Robert Bernerstätter
- **Dateiname**: getAnzAnlagenFolge.m
- **Datum**: 23.09.2013
- **Version**: 3.05
- **Beschreibung**: Das Programm bestimmt die Anzahl der unterschiedlichen Anlagenfolgen in einem Arbeitsplan. Kommt eine Anlagenfolge mehrmals vor, so wird diese nur einmal gezählt.

Funktionsaufruf

Übergabeparameter:

- **Anlagenfolge**: Ein `n` x `1` Cellarray. Jede Zeile `n` entspricht einem Auftrag. Jede Cell enthält die Anlagenfolge des Auftrages der Zeile und ist als Spaltenvektor formatiert. Jede Spalte enthält eine Arbeitsvorgangsnummer oder eine Anlagennummer.

Rückgabeparameter:

- **anzAnl**: Ist die Anzahl der verschiedenen Arbeitsvorgangs- oder Anlagennummern.
- **markerVektor**: Ist ein Zeilenvektor mit `n` Zeilen, wobei `n` der Anzahl von Aufträgen entspricht. Er markiert gleiche Aufträge mit der gleichen Zahl und stellt somit die Beziehung zwischen Aufträgen her.

```
function [anzAnl, markerVektor] = getAnzAnlagenFolge(Anlagenfolge)
```

Anzahlsuche

Es wird eine Matrix erzeugt die in jeder Zeile die Anlagenfolge enthält. Jede Spalte enthält eine Arbeitsvorgangs-, bzw. Anlagennummer. Sollte eine Zeile weniger Spalten benötigen, werden die folgenden Spalten mit `0` befüllt. Nach der For-Schleife wird die Matrix auf die längste Zeile gekürzt. Es werden die eindeutigen Anlagenfolgen in eine Matrix extrahiert und die Anzahl der Zeilen bestimmt. Das ergibt die Anzahl der Anlagenfolgen.

```
RDM = zeros(size(Anlagenfolge,1));
maximum = 0;

for(i=1:1:size(Anlagenfolge,1))
    laenge = size(Anlagenfolge{i,:},2);
    maximum = max(maximum,laenge);
    RDM(i,[1:laenge]) = str2double(Anlagenfolge{i,:});
end;

RDMAnlagenfolge = RDM(:,[1:maximum]);
anzAnl = size(unique(RDMAnlagenfolge,'rows'),1);
```

Identifikation zusammengehöriger Zeilen

Sollte es beim Funktionsaufruf gewünscht werden, kann ein Vektor zurückgegeben werden, welcher Zeilen mit gleichen Folgen mit gleichen Nummern markiert.

```
if(nargout == 2)
    [~, ~, markerVektor] = unique(RDMAnlagenfolge,'rows');
end;

end
```

J. Bestimmung der Attribute, die nicht beachtet werden sollten

- **Autor**: Robert Bernerstätter
- **Dateiname**: toIgnore.m
- **Datum**: 23.09.2013
- **Version**: 2.00
- **Beschreibung**: Das Programm bestimmt die Attribute, bzw. deren Index in der Matrix, die für den Clustervorgang nicht verwendet werden sollen. Dafür wird ein Grenzwert für das Bestellgewicht berechnet, der von keinem Wert eines Attributes überschritten werden darf. Sollte das der Fall sein, wird das Attribut für den Clustervorgang nicht herangezogen. Der Informationsgewinn wäre zu gering und die Trennung durch den Überhang nur eines Wertes nicht zielführend.

Funktionsaufruf

Übergabeparameter:

- **Matrix**: Ein `n` x `13` Cellarray. Jede der `n` Zeilen enthält einen Auftrag. In jede der `13` Spalten sind die Werte für ein Clusterattribut eingetragen.
- **bkgVektor**: Spaltenvektor mit Anzahl-Aufträge Einträgen. Der Vektor hält das Bestellgewicht (*BKG*) des Auftrages.
- **summeBKG**: Die Summe des Bestellgewichtes der Aufträge im *bkgVektor*.
- **gewichtUG**: Relative Gewichtsuntergrenze des Bestellgewichtes für eine Produktfamilie in der SGE.

Rückgabeparameter:

- **ignoreindex**: Ein Zeilenvektor mit den Indizes der Attribute die für den Clustervorgang nicht herangezogen werden sollen.

```
function [ignoreindex] = toIgnore(Matrix, bkgVektor, summeBKG, gewichtUG)
```

Datenvorbereitung

Die 13te Spalte der Matrix enthält die Anlagenfolge. Da diese für den Vergleich nicht herangezogen werden kann, wird sie mit dem Vektor mit den Bestellgewichten überschrieben. Der Grenzwert ergibt sich aus der *gewichtUG*.

```
Matrix(:,13) = num2cell(bkgVektor);
Matrix = cell2mat(Matrix);
ignoreindex = [];
grenzwert = 1-(2*gewichtUG);
```

Suche nach den irrelevanten Attributen

Die erste For-Schleife selektiert bei jedem Durchlauf eine Spalte mit Attributen. Es wird für jedes Attribut nach den verschiedenen Werten gesucht. In der zweiten For-Schleife wird die Summe des Bestellgewichts für jeden dieser Werte, welche in der Variable *A*

gespeichert sind, ermittelt. Sollte das Gewicht dem Grenzwert des gesamten Bestellgewichtes überschreiten, wird der Index des Attributes aus der ersten For-Schleife zur Liste der zu ignorierenden Indizes hinzugefügt. Die weiteren Werte des Attributs müssen nicht mehr überprüft werden und die Schleife bricht vorzeitig ab.

Erste For-Schleife

```
for(i=1:1:size(Matrix,2)-1)
    A = unique(Matrix(:,i));
```

Zweite For-Schleife

```
    for(k=1:1:size(A,1))
        summeAttribut = sum(Matrix(Matrix(:,i)==A(k,1),13));
        if(summeAttribut/summeBKG>grenzwert)
            ignoreindex = [ignoreindex i];
            break;
        end;
    end;
end;

end
```

K. Population auf die richtige Größe bringen

- **Autor**: Robert Bernerstätter
- **Dateiname**: refill.m
- **Datum**: 23.09.2013
- **Version**: 1.00
- **Beschreibung**: Bringt die Population einer Generation auf die richtige Größe. Dazu werden entweder Individuen zufällig erzeugt und hinzugefügt, oder zufällige Individuen entfernt.

Funktionsaufruf

Übergabeparameter:

- **Population**: Eine `n` x `m` Matrix, die die Population einer Generation eines genetischen Algorithmus darstellt. Jede Zeile `n` ist ein Individuum der Population. Jede Spalte `m` ist ein Gen des Individuums und zeigt an ob ein Attribut verwendet wird (**1**) oder nicht verwendet wird (**0**).
- **difference**: Ein ganzzahliger Wert, welcher anzeigt um wieviel eine Population von der Sollgröße abweicht.
- **flag**: Ein Wert, *0* oder *1*, welcher anzeigt, ob die Abweichung aus `diffenrence` positiv oder negativ ist. *0* zeigt an, dass die Population zu klein ist und der Wert aus *difference* aufgefüllt werden muss. *1* zeigt an, dass die Population zu groß ist und der Wert aus *difference* entfernt werden muss.

Rückgabeparameter:

- **Population**: Eine `n` x `m` Matrix, welche die neue Population in der Sollgröße darstellt.

```
function [Population] = refill(Population, difference, flag)
```

Fallunterscheidung

1. Sollte die Population zu klein sein werden zufällig Individuen erzeugt (`refiller`) und der Population zugewiesen.
2. Sollte die Population zu groß sein, werden zufällig Individuen gewählt (`remover`) und aus der Population entfernt.

1.Fall - Population zu klein

```
if(flag==0)
    refiller = randi([0,1],difference, size(Population,2));
    Population = [Population; refiller];
```

2.Fall - Population zu groß

```
else
    remover = randi(size(Population,1),1,difference);
    Population(remover,:) = [];
end;
```

```
end
```

L. Durchführung des Clustervorgangs

- **Autor**: Robert Bernerstätter
- **Dateiname**: clusterung.m
- **Datum**: 23.09.2013
- **Version**: 4.10
- **Beschreibung**: Das Programm führt den Clustervorgang durch. Zu Beginn werden die Distanzmatrizen für die metrischen bzw. nominalen Attribute ermittelt. Diese werden mit jener der Anlagenfolge zu einer Distanzmatrix vereint. Mit dieser wird der Clustervorgang für verschiedene Clusteranzahlen durchgeführt.

Funktionsaufruf

Übergabeparameter:

- **MatMet**: Eine `n` x `m` Matrix mit den metrischen Clusterattributen. Jede Zeile `n` enthält einen Auftrag. Jede Spalte `m` enthält die Werte eines metrischen Clusterkriteriums.
- **MatNom**: Eine `n` x `m` Matrix mit den nominalen Clusterattributen. Jede Zeile `n` enthält einen Auftrag. Jede Spalte `m` enthält die Werte eines nominalen Clusterkriteriums.
- **metrics**: Ein `n` x `1` Cellarray mit den Verfahren zur Bildung der Distanzmatrix.
- **algorithms**: Ein `n` x `1` Cellarray mit den Verfahren zur Verdichtung zu Cluster.
- **clustAnz**: Ein `1` x `n` Zeilenvektor mit den verschiedenen Anzahlen von Clustern die gebildet werden sollen.
- **DistAnlFolge**: Ein `n` x `n` Distanzmatrix mit den Werten für die Anlagenfolge

Rückgabeparameter:

- **D**: Ein `n` x `3` Cellarray welches das Ergebnis des Clustervorgangs enthält. Jede Zeile `n` enthält ein Ergebnis eines Clustervorganges. Spalte `1` enthält das Ergebnis des Clustervorgangs. Dabei handelt es sich um eine `n` x `m` Matrix. Jede Spalte der Matrix ist ein Ergebnis mit einer Clusteranzahl aus *clustAnz*. Die Werte ordnen die Aufträge den Clustern zu. Spalte `2` enthält das angewandte Clusterverfahren für die Ergebnisse aus Spalte `1`. Spalte `3` enthält das mathematische Verfahren für die metrische Distanzmatrix. Bei rein nominalen Attributen wird *hamming* eingetragen. Sollte nur die Anlagenfolge verwendet werden wird *genetisch* eingetragen.

```
function [D] = clusterung(MatMet, MatNom, metrics, ...
    algorithms, clustAnz, DistAnlFolge)
```

Metrische Distanzmatrix

Für den Fall, dass die Variable *MatMet* Werte enthält (erste If-Bedingung), wird die Distanzmatrix für diese Werte mit unterschiedlichen Verfahren, welche in *metrics* gespeichert sind, ermittelt (erste For-Schleife). Sollte das **Mahalanobis**-Verfahren verwendet werden (zweite If-Bedingung), wird vorab geprüft, ob die Covarianzmatrix von *MatMet* invertierbar ist (dritte If-Bedingung). Sollte dies nicht der Fall sein, kann die Distanzmatrix mit diesem Verfahren nicht ermittelt werden.

```
warning('off','all');
DistMet=[];
```

Erste If-Bedingung - Metrische Werte

```
if(~isempty(MatMet))
```

Erste For-Schleife

```
    for(m=1:1:size(metrics,1))
```

Zweite If-Bedingung - Mahalanobisverfahren

```
        if(strcmp(metrics{m,1},'mahalanobis'))
            [~, flag] = chol(cov(MatMet));
```

Dritte If-Bedingung - Invertierbarkeit

```
            if(flag ~= 0)

            else
                DistMet{m,1} = squareform(pdist(MatMet,metrics{m,1}));
                DistMet{m,2} = metrics{m,1};
            end;
        else
            DistMet{m,1} = squareform(pdist(MatMet,metrics{m,1}));
            DistMet{m,2} = metrics{m,1};
        end;
    end;
    DistMet(cellfun(@isempty,DistMet(:,1)),:) = [];
end;
```

Nominale Distanzmatrix

Für den Fall, dass die Variable *MatNom* Werte enthält (vierte If-Bedingung), wird die Distanzmatrix für diese Werte mit dem *hamming* -Verfahren ermittelt.

Vierte If-Bediengung - Nominalle Werte

```
if(~isempty(MatNom))
    DistNom = [];
    DistNom = squareform(pdist(MatNom,'hamming'));
end;

LinkageMat=[];
schleifenEnde = size(DistMet,1);
```

Einheitliche Distanzmatrix

Bevor der Clustervorgang durchgeführt wird, werden die verschiedenen Distanzmatrizen zu einer einheitlichen zusammengeführt. Dabei werden sieben Fälle unterschieden.

1. Fall: Es existieren drei Distanzmatrizen. Jene mit den metrischen Attributen, jene mit den nominalen Attributen und jene mit der Anlagenfolge. In der For-Schleife wird jeder Eintrag für die metrische Distanzmatrizen durchgelaufen und mit der nominalen und jener der Anlagefolge vereint. Als mathematisches Verfahren, wird jenes der metrischen Distanzmatrix hinterlegt.
2. Fall: Es existierte **keine** Distanzmatrix für die nominalen Attribute. Ansonsten entspricht die Vorgehensweise jene vom **1.Fall**.

3. Fall: Es existiert **keine** Distanzmatrix für die metrischen Attribute. Die Distanzmatrix für die nominalen Attribute und jene der Anlagenfolge werden zu einer vereinigt. Als mathematisches Verfahren wird das *hamming*-Verfahren hinterlegt.
4. Fall: Es existiert nur die Distanzmatrix für die Anlagenfolge. Diese wird übernommen und als mathematisches Verfahren wird *genetisch* hinterlegt.
5. Fall: Es existiert **keine** Distanzmatrix für die Anlagenfolge. Ansonsten entspricht die Vorgehensweise jene vom **1.Fall**.
6. Fall: Es existieren nur Distanzmatrizen für die metrischen Attribute. Diese werden übernommen.
7. Fall: Es existiert nur eine Distanzmatrix für die nominalen Attribute. Diese wird übernommen. Als mathematisches Verfahren wird das *hamming*-Verfahren hinterlegt.

1.Fall - Alle Arten von Distanzmatrizen

```
if(nargin==6 && ~isempty(MatMet) && ~isempty(MatNom))
    LinkageMat = cell(schleifenEnde,2);
    for(i=1:1:schleifenEnde)
        LinkageMat{i,1} = (DistMet{i,1}+DistNom+DistAnlFolge)./3;
        LinkageMat{i,2} = DistMet{i,2};
    end;
```

2.Fall - Keine nominallen Attribute

```
elseif(nargin==6 && ~isempty(MatMet) && isempty(MatNom))
    LinkageMat = cell(schleifenEnde,2);
    for(i=1:1:schleifenEnde)
        LinkageMat{i,1} = (DistMet{i,1}+DistAnlFolge)./2;
        LinkageMat{i,2} = DistMet{i,2};
    end;
```

3.Fall - Keine metrischen Attribute

```
elseif(nargin==6 && isempty(MatMet) && ~isempty(MatNom))
    LinkageMat{1,1} = (DistNom+DistAnlFolge)./2;
    LinkageMat{1,2} = 'hamming';
```

4.Fall - Nur Anlagenfolge

```
elseif(nargin==6 && isempty(MatMet) && isempty(MatNom))
    LinkageMat{1,1} = DistAnlFolge;
    LinkageMat{1,2} = 'genetisch';
```

5.Fall - Keine Anlagenfolge

```
elseif(nargin==5 && ~isempty(MatMet) && ~isempty(MatNom))
    LinkageMat = cell(schleifenEnde,2);
    for(i=1:1:schleifenEnde)
        LinkageMat{i,1} = (DistMet{i,1}+DistNom)./2;
        LinkageMat{i,2} = DistMet{i,2};
    end;
```

6.Fall - Nur metrische Attribute

```
elseif(nargin==5 && ~isempty(MatMet) && isempty(MatNom))
    LinkageMat = DistMet;
```

7.Fall - Nur nominalle Attribute

```
elseif(nargin==5 && isempty(MatMet) && ~isempty(MatNom))
    LinkageMat{1,1} = DistNom;
    LinkageMat{1,2} = 'hamming';
end;
```

Verdichtung zu Clustern

Die Ergebnismatrix **D** wird initialisiert. Die Zweite For-Schleife geht jeden Clusteralgorithmus in der Variable *algorithms* durch. In der dritten For-Schleife wird der Clustervorgang für jede der vereinten Distanzmatrizen (*LinkageMat*) durchgeführt. Die erste Spalte von *D* enthält das Ergebnis des Clustervorgangs für die verschiedenen Einträge der Clusteranzahl (*clustAnz*). Die zweite Spalte enthält den verwendeten Clusteralgorithmus und die dritte Spalte das Verfahren zur Erstellung der Distanzmatrix.

```
D{size(algorithms,1)*size(LinkageMat,1),3} = [];
counter = 1;
```

Zweite For-Schleife

```
for(a=1:1:size(algorithms,1))
```

Dritte For-Schleife

```
    for(l=1:1:size(LinkageMat,1))
        Z = linkage(LinkageMat{l,1},algorithms{a,1});
        D{counter,1} = cluster(Z,'maxclust',clustAnz);
        D{counter,2} = algorithms{a,1};
        D{counter,3} = LinkageMat{l,2};
        counter = counter+1;
    end;
end;
warning('on','all');
```

```
end
```

M. Berechnung der Fitness

- **Autor**: Robert Bernerstätter
- **Dateiname**: fitness.m
- **Datum**: 23.09.2013
- **Version**: 2.00
- **Beschreibung**: Das Programm berechnet die Fitness verschiedener Lösungen des Clustervorganges und gibt das Ergebnis der besten Lösung in einem Cellarray *fitvektor* zurück. Zur Bewertung der Fitness werden die Varianzen der Belastungsprofile der einzelnen Aufträge summiert. Jene Lösung mit der kleinsten Summe wird als beste Lösung zurückgegeben.

Funktionsaufruf

Übergabeparameter:

- **RDMNum**: Eine `n` x `m` Matrix mit den numerischen Auftragsdaten. Jede Zeile `n` enthält einen Auftrag. Jede Spalte `m` die Daten eines Auftragsattributes.
- **T**: Ein `n` x `3` Cellarray. Jede Zeile `n` enthält eine Lösung mit unterschiedlicher Clusteranzahl mit einer speziellen Kombination aus den mathematischen Verfahren für die Distanzmatrixberechnung und des Clustervorgangs. Spalte 1 hält eine `i` x `j` Matrix mit verschiedenen Lösungen des Clustervorgangs. Jede Zeile `i` ist ein Auftrag. Jede Spalte `j` ist eine Lösung mit den Zuordnungsnummern, welche die Aufträge Clustern zuordnet. Spalte 2 des Cellarrays gibt das mathematische Verfahren an, welches zur Berechnung der Distanzmatrix verwendet wurde. Spalte 3 das Verfahren zur Datenclusterung.
- **UGRel**: Die Gewichtsuntergrenze für eine Produktfamilie als Anteil vom Gesamtgewicht der SGE.
- **OGRel**: Die Gewichtsobergrenze für eine Produktfamilie als Anteil vom Gesamtgewicht der SGE.
- **bkgSGE**: Das Gesamtgewicht einer SGE.
- **clusterindex**: Ein `1` x `n` Spaltenvektor, welcher die Indizes der Spalten der Clusterattribute hält.
- **UG**: Zählt wie oft die Gewichtsuntergrenze unterschritten wurde.
- **OG**: Zählt wie oft die Gewichtsobergrenze überschritten wurde.

Rückgabeparameter:

- **fitvektor**: Ein `1` x `6` Cellarray welches das Ergebnis der Fitnessbewertung speichert. Die erste Spalte enthält die Fitness der besten Lösung. Spalte Zwei die Anzahl der Produktfamilien der besten Lösung. In die dritte Spalte wird das mathematische Verfahren abgelegt, welches zur Erstellung der Distanzmatrix verwendet wurde, in der vierten Spalte jenes zur Clusterung. Die fünfte Spalte enthält den Vektor mit den Nummern, welche die Aufträge zu den Clustern zuordnet. In die sechste Spalte werden direkt die Werte des *clusterindex* geschrieben.
- **UG**: Zählt wie oft die Gewichtsuntergrenze unterschritten wurde.
- **OG**: Zählt wie oft die Gewichtsobergrenze überschritten wurde.

```
function [fitvektor, UG, OG] = fitness(RDMNum, T, UGrel, OGrel, bkgSGE,...
    clusterindex, UG, OG)
```

Datenvorbereitung

Zu Beginn wird die Variable *fitvektor* initialisiert. Die beste Fitness wird zu Beginn mit unendlich angegeben. Alle anderen Werte, außer die letzte Spalte mit dem *clusterindex* werden auf **0** gesetzt.

```
bestfit = inf('double');
fitvektor{1,1} = bestfit;
fitvektor{1,2} = 0;
fitvektor{1,3} = 0;
fitvektor{1,4} = 0;
fitvektor{1,5} = 0;
fitvektor{1,6} = clusterindex;
```

Programmdurchlauf

Es wird die Matrix mit den Lösungsvektoren in eine Variable *A* extrahiert (erste For-Schleife). Aus dieser Matrix wird der Reihe nach eine Spalte mit den Zuordnungsnummern in die letzte Spalte der Variable *RDMNum* geschrieben (zweite For-Schleife). Anhand dieser Nummerierung wird die Summe an Bestellgewicht *bkgFam* einer Produktfamilie bestimmt (dritte For-Schleife).

Erste For-Schleife

```
for(i=1:1:size(T,1))
    A = T{i,1};
```

Zweite For-Schleife - Fitnessbewertung

Am Ende der Schleife, nach dem Ende der dritten For-Schleife, wird überprüft, ob eine bessere Lösung gefunden wurde. Sollte dies der Fall sein, werden die Informationen der Lösung in den *fitvektor* gespeichert.

```
    for(j=1:1:size(A,2))
        RDMNum(:,23) = A(:,j);
        famanz = size(unique(A(:,j)),1);
        fit = 0;
```

Dritte For-Schleife - Fitnessberechnung

Nachdem die Summe des Bestellgewichtes für eine Produktfamilie berechnet wurde, werden drei Fälle unterschieden.

1. Fall: Das Liefergewicht ist zu gering. Die Lösung wird abgelehnt, indem die Variable *fit* wieder auf unendlich gesetzt wird. Der Zähler *UG* wird um *1* erhöht und die For-Schleife wird abgebrochen, da es nicht nötig ist, die anderen Produktfamilien zu überprüfen, da bereits eine die Bedingungen nicht erfüllt.
2. Fall: Das Liefergewicht ist zu hoch. Das weitere Vorgehen ist analog zum 1. Fall.
3. Fall: Das Bestellgewicht befindet sich innerhalb der Grenzen. Die Variable *fit* wird um die Summe der Auftragsvarianzen der Produktfamilie erhöht.

```
        for(k=1:1:famanz)
            bkgFam = sum(RDMNum(RDMNum(:,23)==k,9));
            if(bkgFam/bkgSGE<UGrel)
                fit = inf('double');
                UG = UG+1;
                break;
            elseif(bkgFam/bkgSGE>OGrel)
                fit = inf('double');
                OG = OG+1;
                break;
            else
                fit = fit + var(RDMNum(RDMNum(:,23)==k,22));
            end;
        end;
        fit=fit/famanz; % Berechnung der durchschnittlichen Fitness
        if(fit<bestfit)
            bestfit = fit;
            fitvektor{1,1} = bestfit;
            fitvektor{1,2} = famanz;
            fitvektor{1,3} = T{i,2};
            fitvektor{1,4} = T{i,3};
            fitvektor{1,5} = A(:,j);
        end;
    end;
end;

end
```

N. Durchführung der fitnessproportionalen Selektion

- **Autor**: Robert Bernerstätter
- **Dateiname**: fitnessproportional.m
- **Datum**: 23.09.2013
- **Version**: 2.20
- **Beschreibung**: Das Programm führt die fitnessproportionale Selektion für den Paarungspool durch. Zu Beginn werden die Individuen der Generation entsprechend deren Fitness sortiert und dann in den Paarungspool gewählt. Die neue Generation *GenX* wird durch n-Punkt Kreuzung des Paarungspools erzeugt.

Funktionsaufruf

Übergabeparameter:

- **GenW**: Eine `n` x `14` Matrix mit den Individuen einer Generation. Jede Zeile `n` enthält ein Individuum der Population der Generation. Die ersten 13 Spalten enthalten binär kodiert, ob ein Attribut verwendet wird (**1**) oder nicht verwendet wird (**0**). Spalte 14 enthält die Fitness des Individuums.

Rückgabeparameter:

- **GenX**: Eine `n` x `13` Matrix mit den Individuen der neuen Generation. Jede Zeile `n` enthält ein Individuum der neuen Generation. Jede Spalte enthält binär kodiert ob ein Attribut verwendet wird (**1**) oder nicht verwendet wird (**0**).

```
function [GenX] = fitnessproportional(GenW)
```

Datenvorbereitung

Zu Beginn werden die Individuen entsprechend der Fitness sortiert (*GenW_sorted*). Jenes Individuum mit der besten Fitness (niedrigster Wert) steht oben auf in der Liste. Alle Individuen mit einer ungültigen Lösung für die Fitness (*Unendlich*) werden entfernt und die Anzahl an gültigen Lösungen wird bestimmt (*anzInd*).

```
GenW_sorted = sortrows(GenW,14);

GenW(GenW(:,14)==Inf,:) = [];
anzInd = size(GenW,1);
```

Erstellung des Paarungspools

Er werden vier Fälle unterschieden.

1. Fall: Es existiert nur eine gültige Lösung. Diese wird in den Paarungspool übernommen, zusammen mit einer ungültigen.
2. Fall: Es existieren zwei gültige Lösungen. Beide werden in den Paarungspool übernommen.

3. Fall: Es existieren drei gültige Lösungen. Diese drei werden in den Paarungspool übernommen. Wobei die beste Lösung mit einer Ungültigen gepaart wird und die zwei weiteren gültigen Lösungen miteinander.
4. Fall: Es existieren mehr als drei gültige Lösungen. Es wird die beste mit einer ungültigen Lösung gepaart. Die weiteren Paarungen werden entsprechend der fitnessproportionalen Selektion gewählt. Dazu wird die Summe der Fitness der gültigen Lösung berechnet. Jedem gültigen Individuum wird seine relative Fitness zugeordnet. Es wird eine Zufallsvariable generiert (*selectionmarker*), welche bestimmt welche der Lösungen für den Paarungspool gewählt wird.

1. Fall - Eine gültige Lösung

```
if(anzInd == 1)
    Paarungspool = cell(1,2);
    Paarungspool{1,1} = GenW_sorted(1,[1:13]);
    Paarungspool{1,2} = GenW_sorted(size(GenW_sorted,1),[1:13]);
```

2. Fall - Zwei gültige Lösungen

```
elseif(anzInd == 2)
    Paarungspool = cell(1,2);
    Paarungspool{1,1} = GenW_sorted(1,[1:13]);
    Paarungspool{1,2} = GenW_sorted(2,[1:13]);
```

3. Fall - Drei gültige Lösungen

```
elseif(anzInd == 3)
    Paarungspool = cell(2,2);
    Paarungspool{1,1} = GenW_sorted(1,[1:13]);
    Paarungspool{1,2} = GenW_sorted(size(GenW_sorted,1),[1:13]);
    Paarungspool{2,1} = GenW_sorted(2,[1:13]);
    Paarungspool{2,2} = GenW_sorted(3,[1:13]);
```

4. Fall - Vier oder mehr gültige Lösungen

```
else
    Paarungspool = cell(ceil(size(GenW_sorted,1)/2),2);
    sumFit = sum(GenW(:,14));
    Paarungspool{1,1} = GenW_sorted(1,[1:13]);
    Paarungspool{1,2} = GenW_sorted(size(GenW,1),[1:13]);
```

Bestimmung der relativen Fitness

```
    for(i=1:1:size(GenW,1))
        GenW(i,15) = GenW(i,14)/sumFit;
    end;

    selectionmarker = rand();
    summe = 0;
```

Füllen des Paarungspools

Die Generation mit gültigen Lösungen wird der Reihe nach durch gegangen. Die Fitness der Individuen wird dabei kumuliert. Sobald die Summe den Wert des *selectionmarkers* übersteigt, wird das Individuum welches den Wert zur Summe lieferte in den Paarungspool aufgenommen und die Auswahl startet wieder von vorne der Liste mit einem neuen Wert für den *selectionmarker*.

```
    for(i=2:1:size(Paarungspool,1))
        for(m=1:1:2) % For-Schleife für Individuum 1 und Individuum 2 der Paarung
            for(k=1:1:size(GenW,1))
                summe = summe+GenW(k,15);
                if(summe >= selectionmarker)
                    Paarungspool{i,m} = GenW(k,[1:13]);
                    summe = 0;
                    selectionmarker = rand();
                    break;
                end;
            end;
        end;
    end;
end;
```

Bestimmung der neuen Generation

Der Paarungspool wird mittels n-Punkt Verfahren gekreuzt und eine neue Population für die neue Generation *GenX* wird erzeugt.

```
GenX = nPunktKreuzung(Paarungspool);
```

```
end
```

O. Durchführung einer n-Punkt Kreuzung

- **Autor**: Robert Bernerstätter
- **Dateiname**: nPunktKreuzung.m
- **Datum**: 23.09.2013
- **Version**: 1.00
- **Beschreibung**: Das Programm führt eine n-Punkt Kreuzung durch. Dazu werden zufällig Kreuzungspunkte ermittelt. Zwischen den Individuen des Kreuzungspaares werden die Sektionen, welche von den Kreuzungspunkten begrenzt werden, untereinander ausgetauscht. Die beiden neuen Individuen werden in die neue Population aufgenommen.

Funktionsaufruf

Übergabeparameter:

- **Paarungspool**: Ein `n` x `2` Cellarray mit den Individuen zur Kreuzung. Jede Zeile `n` enthält ein Kreuzungspaar. Jede der beiden Spalten enthält ein Individuum.

Rückgabeparameter:

- **PopX**: Population mit den neuen Individuen, welche sich aus dem Paarungspool ergeben.

```
function [PopX] = nPunktKreuzung(Paarungspool)
```

Datenvorbereitung

Zu Beginn werden die Kreuzungspunkte ermittelt, welche die zu tauschende Sektion begrenzen werden. Die neue Population *PopX* wird initialisiert.

```
crosspoints = sort(randi(size(Paarungspool{1,1},2),2,1));
PopX = [];
```

Programmablauf

Bei jeder Iteration der For-Schleife, werden die beiden Kreuzungspaare in eigenen Variablen (*individuum1 individuum2*) gespeichert. Aus den beiden Individuen werden jeweils die Sektionen geschnitten, welche mit dem Kreuzungspartner getauscht werden sollen (*section1 section2*). Danach werden die Sektionen in den Code des Kreuzungspartners geschrieben und die neuen Individuen in die Population aufgenommen.

```
for(i=1:1:size(Paarungspool,1))
    % Extraktion der Individuuen
    individuum1 = Paarungspool{i,1};
    individuum2 = Paarungspool{i,2};
    % Extraktion der Sektionen
    section1 = individuum1(1,[crosspoints(1,1):crosspoints(2,1)]);
    section2 = individuum2(1,[crosspoints(1,1):crosspoints(2,1)]);
    % Tausch des Codes - Erzeugung neuer Individuuen
    individuum1(1,crosspoints(1,1):crosspoints(2,1)) = section2;
    individuum2(1,crosspoints(1,1):crosspoints(2,1)) = section1;
    % Erweiterung der neuen Population
    PopX = [PopX;individuum1;individuum2];

end;
```

```
end
```

P. Durchführung der genetischen Mutation

- **Autor**: Robert Bernerstätter
- **Dateiname**: mutation.m
- **Datum**: 20.09.2013
- **Version**: 1.05
- **Beschreibung**: Das Programm führt den Mutationsschritt des genetischen Algorithmus durch. Es werden alle Individuen einer Population mit einer gewissen Mutationswahrscheinlichkeit verändert.

Funktionsaufruf

Übergabeparameter:

- **Population**: Eine `n` x `m` Matrix. Jede Zeile `n` enthält ein Individuum der Population. Jede Spalte `m` zeigt binär codiert ob ein Attribut in dem Individuum verwendet wird **1**, oder nicht verwendet wird **0**. Es handelt sich um die nicht mutierte Population.
- **Mutation**: Eine `n` x `m` Matrix. Jede Spalte `m` enthält die Indizes, welche in einem Individuum aus *Population* mutiert werden.

Rückgabeparameter:

- **Population**: Siehe *Population* in Übergabeparameter.

```
function [Population] = mutation(Population, Mutation)
```

Mutation

Die Individuen aus Population werden in der For-Schleife durchlaufen. Das Individuum aus der `i`-ten Zeile aus *Population* wird an den Indizes mutiert die in der `i`-ten Spalte in *Mutation* angegeben sind. Die Mutation erfolgt durch eine einfache Negierung (~) des Eintrages.

```
for(i=1:1:size(Population,1))
    Population(i,Mutation(:,i)) = ~Population(i,Mutation(:,i));
end;

end
```

Q. Ermittlung der Planartikel

- **Autor**: Robert Bernerstätter
- **Dateiname**: planartikelFinder.m
- **Datum**: 23.09.2013
- **Version**: 2.30
- **Beschreibung**: Das Programm ermittelt die Planartikel einer Produktfamilie. Dazu wird über ein rekursives Programm eine Teilmenge der Produktfamilie gebildet, welche als Suchraum für den Planartikel dient. Gewählt wird jener Auftrag, welcher von der Varianz der Produktfamilie am geringsten abweicht und am meisten Bestellgewicht hat.

Funktionsaufruf

Übergabeparameter:

- **Familie**: Ein `n` x `23` Cellarray welches die Aufträge einer Produktfamilie enthält. Jede Zeile `n` entspricht einem Auftrag der Produktfamilien. Jede der 23 Spalte enthält die Attributinformationen der Aufträge.
- **clusterKriterien**: Ein `1` x `n` Vektor welcher die Indizes der Clusterkriterien enthält.
- **bkgSge**: Das Bestellgewicht der SGE. Die Variable dient zur Berechnung des Gewichtanteils der Produktfamilie (des Planartikels).

Rückgabeparameter:

- **Planartikel**: Ein `1` x `13` Vektor der den Planartikel der Produktfamilie repräsentiert. Die Spalten enthalten technische Merkmale oder organisatorische Nummerierungen.
- **Familie**: Eine `n` x `18` Matrix die die Artikel der Produktfamilie enthält. Jede Zeile `n` enthält einen Auftrag und jede der Spalten ein Attribut der Auftragsdatenbank.

```
function [Planartikel, Familie] = planartikelFinder(Familie, clusterKriterien, bkgSge)
```

Datenvorbereitung

Die Indizes der Clusterkriterien in der Variable *clusterKriterien* müssen auf die Matrix *Familie* umgelegt werden. Dies geschieht mir der *UmrechnugsMatrix*. Die neuen Clusterkriterien *clusterKriterienNeu* werden in der ersten For-Schleife befüllt.

```
UmrechnungsMatrix = [1 2; 2 3; 3 4; 4 5; 5 6; 6 7; 7 8; 8 13; 9 14; 10 15; 11 16; 12 17;
13 20];
clusterKriterienNeu = zeros(size(clusterKriterien));
```

Erste For-Schleife

```
for(i=1:1:size(clusterKriterien,2))
    clusterKriterienNeu(1,i) =
UmrechnungsMatrix(clusterKriterien(1,i)==UmrechnungsMatrix(:,1),2);
end;
```

Programmablauf - Teilmengenaufbereitung

Zu Beginn wird eine Teilmenge der Produktfamilie gebildet, welche in den Ausprägungen der Clusterkriterien homogen ist. In der zweiten For-Schleife, wird für jeden Auftrag in dieser Teilmenge die Abweichung von der Varianz des Belastungsprofils, zur Gesamtvarianz der Produktfamilie bestimmt.

```
TeilMengeFamilie = planartikelRekursion(Familie, clusterKriterienNeu);
familienvarianz = var(cell2mat(Familie(:,22)));
bkgFam = sum(cell2mat(Familie(:,9)));
```

Zweite For-Schleife

```
for(i=1:1:size(TeilMengeFamilie,1))
    TeilMengeFamilie{i,25} = abs(familienvarianz - TeilMengeFamilie{i,22});
end;
```

Programmablauf - Planartikelauswahl

Jene Aufträge mit der minimalsten Abweichung der Varianz zur Gesamtvarianz werden ermittelt. Gibt es mehrere, wird jener Auftrag mit dem meisten Bestellgewicht als Planartikel gewählt. Sollte es auch hier noch mehrere geben, wird der erste Auftrag in der Liste als Planartikel verwendet.

```
Planartikelauswahl = TeilMengeFamilie(cell2mat(TeilMengeFamilie(:,25)) ==
min(cell2mat(TeilMengeFamilie(:,25))),:);

if(isempty(Planartikelauswahl))
    Planartikelauswahl = TeilMengeFamilie(1,:);
end;

[~,ind] = max(cell2mat(Planartikelauswahl(:,9)));
Planartikel = Planartikelauswahl(ind,:);
```

Formatierung

Der Vektor des Planartikels und die Matrix mit den Produktfamilien werden noch so formatiert, dass sie nur Informationen (Spalten) enthalten, die für das Endergebnis relevant sind.

```
Planartikel(:,6) = Planartikel(:,18);
Planartikel{:,9} = roundn((bkgFam/bkgSge)*100,-2);
Planartikel(:,[11,12]) = Planartikel(:,[10,11]);
Planartikel{:,10} = bkgFam;
%Planartikel(:,13) = Planartikel(:,23);
Planartikel(:,[14:25]) = [];
Familie(:,6) = Familie(:,18);
Familie(:,15) = Familie(:,19);
Familie(:,[19:23]) = [];

%nix = plottests_Planartikel(Familie(:,21),Planartikel(1,21));

end
```

R. Einschränkung der Auftragsauswahl für die Planartikelermittlung

- **Autor**: Robert Bernerstätter
- **Dateiname**: planartikelRekursion.m
- **Datum**: 23.09.2013
- **Version**: 3.22
- **Beschreibung**: Das Programm wird rekursiv mehrmals aufgerufen und verringert die Anzahl der Aufträge für die folgende Planartikelermittlung. Es wird eine Teilmenge der Aufträge, basierend auf den Clusterkriterien ermittelt. Am Ende des Vorgangs werden nur mehr Aufträge zurückgegeben, die in den Clusterkriterien homogen sind. Aus dieser Menge wird der Planartikel ermittelt.

Funktionsaufruf

Übergabeparameter:

- **Familie**: Eine `n` x `m` Matrix mit den Aufträgen der Produktfamilie, bzw. in Folge der rekursiven Aufrufe eine Teilmenge dieser. Jede Zeile `n` enthält einen Auftrag. Jede Spalte `m` enthält eines der Clusterattribute.
- **Kriterien**: Ein `1` x `n` Zeilenvektor mit den Clusterkriterien.

Rückgabeparameter:

- **TeilFamilie**: Eine `n` x `m` Matrix mit einer Teilmenge der Aufträge der Produktfamilie. Jede Zeile `n` enthält einen Auftrag und jede Spalte `m` enthält ein Clusterattribut.

```
function [TeilFamilie] = planartikelRekursion(Familie, Kriterien)
```

Abbruchbedingung der Rekursion

Die Rekursion bricht ab und in Folge dessen ist das Programm abgeschlossen, wenn der Vektor mit den Kriterien leer ist.

```
if(size(Kriterien,2)==0)
    TeilFamilie = Familie;
```

Ermittlung der Teilfamilie (Rekursionsfortschritt)

Es wird schrittweise jenes Attribut (*Kriterien*) und bei diesem wiederum jene Ausprägung gesucht, welche(s) das meiste Gewicht auf sich vereint. Mit der Ausprägung dieses Attributes, wird die Teilmenge der *Familie* gebildet. Für die weitere Suche steht somit nur mehr ein Suchraum mit Aufträgen zur Verfügung, welcher genau die eine Ausprägung des Attributes aus *Kriterien* enthält. Die weitere Suche erfolgt ohne das *Kriterium* mit einzubeziehen.

```
else
    maximum = 0;
```

Ermittlung der Maximumausprägung

Die For-Schleife durchläuft alle Attribute in *Kriterien* und bestimmt die Ausprägung mit dem meisten Gewicht. Es müssen zwei Fälle unterschieden werden.

1. Fall: Das betrachtete Kriterium ist **nicht** die Anlagenfolge.
2. Fall: Das betrachtete Kriterium ist die Anlagenfolge.

Im ersten Fall werden alle Ausprägungen des aktuell betrachteten Attributes ermittelt, dem Attribut zugewiesen und die Summe des Bestellgewichtes berechnet. Im zweiten Fall muss vorab, die Anzahl der Anlagenfolgen ermittelt werden. Zusätzlich wird ein Vektor erzeugt, welcher die Aufträge gleichen Anlagenfolgen zuordnet. Es kann somit für jede Anlagenfolge die Summe der Bestellgewichtes berechnet werden.

```
    for(i=1:1:size(Kriterien,2))
```

1.Fall - Nicht Anlagenfolge

```
        if(Kriterien(1,i)~=20)
            Auspraegung = unique(cell2mat(Familie(:,Kriterien(1,i))));
            Auspraegung(:,2) = Kriterien(1,i);
            for(j=1:1:size(Auspraegung))
                Auspraegung(j,3) =
sum(cell2mat(Familie(cell2mat(Familie(:,Kriterien(1,i)))==Auspraegung(j,1),9)));
            end;
```

2.Fall - Anlagenfolge

```
        else
            [anzFolgen, markervektor] = getAnzAnlagenFolge(Familie(:,20));
            Familie(:,24) = num2cell(markervektor); % Markierung der Aufträge mit den
Anlagenfolgen
            Auspraegung = unique(markervektor);

            Auspraegung(:,2) = Kriterien(1,i);
            for(k=1:1:anzFolgen)
                Auspraegung(k,3) =
sum(cell2mat(Familie(cell2mat(Familie(:,24))==Auspraegung(k,1),9)));
            end;
        end;
```

Bestimmung des Maximumattributes

Es wird der maximale Wert und die Position in der Matrix *Auspraegung* ermittelt.

```
        [maxAttributWert, maxAttributindex] = max(Auspraegung(:,3));
        if(maxAttributWert>maximum)
            maximumvektor = Auspraegung(maxAttributindex,:);
            maximum = maxAttributWert;
        end;
    end;
```

Reduktion auf die Teilmenge und weiterer rekursiver Aufruf

Es muss nochmals unterschieden werden, ob das aktuell betrachtete Attribut **nicht** die Anlagenfolge ist, oder ob es sich um die Anlagenfolge handelt. Die Vorgehensweise ist in beiden Fällen jedoch gleich. Es wird das Kriterium welches die maximale Ausprägung hatte aus *Kriterien* gelöscht. Dem folgt der rekursive Aufruf, wobei die neuen *Kriterien*, sowie die Teilmenge der Familie übergeben werden. Diese enthalten nur Ausprägungen des oben bestimmten Maximums.

```
    if(maximumvektor(1,2)~=20)
        Kriterien(:,Kriterien(1,:)==maximumvektor(1,2)) = [];
        TeilFamilie =
planartikelRekursion(Familie(cell2mat(Familie(:,maximumvektor(1,2)))==maximumvektor(1,1)
,:), Kriterien);
    else
        Kriterien(:,Kriterien(1,:)==maximumvektor(1,2)) = [];
        TeilFamilie =
planartikelRekursion(Familie(cell2mat(Familie(:,maximumvektor(1,2)+4))==maximumvektor(1,
1),:), Kriterien);
    end;

end;

end
```

S. Protokollieren der Ergebnisse

- **Autor**: Robert Bernerstätter
- **Dateiname**: protokolliere.m
- **Datum**: 23.09.2013
- **Version**: 2.50
- **Beschreibung**: Das Programm erstellt die Matrix, welche in die Protokolldatei geschrieben wird. SGEs die bereits in der alten Protokolldatei vorhanden sind werden überschrieben. Solche die noch nicht vorhanden sind werden unten angefügt.

Funktionsaufruf

Übergabeparameter:

- **Planartikel**: Ein `999` x `13` Cellarray welches die Planartikel aller SGEs des Programmdurchlaufs enthält. Jede Zeile `n` entspricht einem Auftrag, jede Spalte `m` einem Attribut. Die Größe ist mit *999* vorgegeben. Daher sind alle Zeilen, die keine Planartikel enthalten mit *nan* beschrieben.
- **fitvektorProtokoll**: Ein `n` x `7` Cellarray welches die Informationen über die Lösung einer SGE enthält, die in die Protokolldatei geschrieben werden müssen.
- **attributinfoTxt**: Ein `1` x `13` Cellarray mit den Spaltenüberschriften des Clusterkriterien.
- **ProtokolldateiAlt**: Ein `49` x `19` Cellarray mit den Informationen der alten Protokolldatei. Die Spalten 1 bis 6 enthalten folgende Informationen: `SGE` - `Verfahren für die Distanzmatrix` - `Verfahren für die Clusterung` - `Anzahl der Produktfamilien` - `Bestellgewicht der Produktfamilie mit dem geringsten Wert in %` vom `Gesamtgewicht der SGE` - `Bestellgewicht der Produktfamilie mit dem höchsten Wert in %` vom `Gesamtgewicht der SGE`. Die Spalten 7 bis 19 enthalten die Clusterkriterien.

Rückgabeparameter:

- **ProtokolldateiNeu**: Ein `n` x `19` Cellarray mit den Informationen für die neue Protokolldatei. Jede Zeile `n` enthält die wichtigsten Daten der Produktfamilienverdichtung für eine SGE. Die Spalten 1 bis 6 enthalten folgende Informationen: `SGE` - `Verfahren für die Distanzmatrix` - `Verfahren für die Clusterung` - `Anzahl der Produktfamilien` - `Bestellgewicht der Produktfamilie mit dem geringsten Wert in %` vom `Gesamtgewicht der SGE` - `Bestellgewicht der Produktfamilie mit dem höchsten Wert in %` vom `Gesamtgewicht der SGE`. Die Spalten 7 bis 19 enthalten die Clusterkriterien.
- **captionProtokoll**: Ein `1` x `n` Cellarray mit den Spaltenüberschriften für die Protokolldatei.
- **Planartikel**: Ein `n` x `13` Cellarray welches die Planartikel aller SGEs des Programmdurchlaufs enthält. Jede Zeile `n` entspricht einem Planartikel, jede Spalte `m` einem Attribut.

```
function [ProtokolldateiNeu, captionProtokoll, Planartikel] = protokolliere...
    (Planartikel, fitvektorProtokoll, attributinfoTxt, Protokolldatei_alt)
```

Datenvorbereitung

Die Matrix *Planartikel* enthält standardmäßig 999 Zeilen. Alle Zeilen, die keine Informationen über Planartikel enthalten werden gelöscht. Das Gleiche gilt für die Variable *fitvektorProtokoll*. Die Spaltenüberschriften für die spätere Protokolldatei (*captionProtokoll*) werden erzeugt. Ein zentraler Punkt ist die Matrix *AttributMatrix*. In ihr wird für jede SGE gespeichert, welche Clusterkriterien für die Produktfamilienverdichtung herangezogen wurden. Die Reihenfolge der Spalten entspricht jener in *attributinfoTxt*. Wird ein Attribut verwendet, wird dies mit **1** codiert, wird ein Attribut nicht verwendet wird dies mit **0** codiert.

```
Planartikel(cellfun(@isempty,Planartikel(:,1)),:) = [];
fitvektorProtokoll(cellfun(@isempty,fitvektorProtokoll(:,1)),:) = [];
[~, cols] =cellfun(@size,fitvektorProtokoll(:,7));
cols = max(cols);
Protokolldatei_neu = cell(size(fitvektorProtokoll,1),19);
captionProtokoll = ...
    {'SGE' 'Metrik' 'Algorithmus' '# Familien' 'UG BKG[%]' 'OG BKG[%]'};
%attrs(1,[1:cols]) = {'Clusterattribut'};
captionProtokoll = [captionProtokoll attributinfoTxt];
AttributMatrix = cell(size(fitvektorProtokoll,1)+1,size(attributinfoTxt,2)+1);
AttributMatrix(:) = {0};
%AttributMatrix = zeros(size(fitvektor_protokoll,1)+1,size(attributinfoTxt,2)+1);
AttributMatrix([2:1+size(fitvektorProtokoll,1)],1) = fitvektorProtokoll(:,1);
AttributMatrix(1,[2:1+size(attributinfoTxt,2)]) = attributinfoTxt;
```

Erstellen der neuen Protokolldatei

Für jeden Eintrag (Zeile) von *fitvektorProtokoll* wird eine Zeile in *Protokolldatei_neu* angelegt. Die Spalteninformationen werden entsprechend übernommen. Die Spalten 7 bis 19 sind die Einträge aus der *AttributMatrix*.

```
for(protokollcounter=1:1:size(fitvektorProtokoll,1))
   % attributbezeichnungnen = attributinfoTxt...
   %(1,fitvektorProtokoll{protokollcounter,7});
    Protokolldatei_neu(protokollcounter,[1,2,3,4]) = ...
        fitvektorProtokoll(protokollcounter,[1,5,4,3]);
    Protokolldatei_neu{protokollcounter,5} = min(cell2mat(Planartikel....
        (strcmp(cell2mat(Planartikel(:,1)),...
        fitvektorProtokoll(protokollcounter,1)),9)));
    Protokolldatei_neu{protokollcounter,6} = max(cell2mat(Planartikel...
        (strcmp(cell2mat(Planartikel(:,1)),...
        fitvektorProtokoll(protokollcounter,1)),9)));
    AttributMatrix(protokollcounter+1,...
        (fitvektorProtokoll{protokollcounter,7})+1) = num2cell(1);
    %AttributMatrix(protokollcounter+1,...
    %(fitvektor_protokoll{~protokollcounter,7})+1) = num2cell(0);
    Protokolldatei_neu(protokollcounter,[7:19]) =...
        AttributMatrix(protokollcounter+1,[2:14]);
end;
```

Zusammenführung mit der alten Protokolldatei

Für jeden Eintrag in *Protokolldatei_neu*, also für jede darin vorkommende SGE wird geprüft, ob ein Eintrag in *Protokolldatei_alt* bereits vorhanden ist. Unterschieden werden zwei Fälle.

1. Fall: Die SGE kommt in *Protokolldatei_alt* nicht vor. Hier wird der Datensatz für die SGE unten an die *Protokolldatei_alt* angefügt.
2. Fall: Die SGE kommt in *Protokolldatei_alt* vor. Hier wird der alte Datensatz mit den neuen Daten überschrieben.

Am Ende wird die Matrix *ProtokolldateiNeu* initialisiert und von Leerzeilen bereinigt.

```
%if(nargin==4)
for(unitecounter = 1:1:size(Protokolldatei_neu))
```

1. Fall - SGE existiert nicht

```
    if(isempty(find(strcmp(Protokolldatei_alt(:,1),...
            Protokolldatei_neu{unitecounter,1}), 1)))
        Protokolldatei_alt(size(Protokolldatei_alt,1)+1,:) = ...
            Protokolldatei_neu(unitecounter,:);
```

2. Fall - SGE existiert

```
    else
        Protokolldatei_alt(find(strcmp(Protokolldatei_alt(:,1),...
            Protokolldatei_neu{unitecounter,1}), 1),:) = ...
            Protokolldatei_neu(unitecounter,:);
    end;

end
```

```
ProtokolldateiNeu = Protokolldatei_alt;
ProtokolldateiNeu(cellfun(@isnan,ProtokolldateiNeu(:,7)),:) = [];
% else
%     ProtokolldateiNeu = Protokolldatei_neu;
%     ProtokolldateiNeu(cellfun(@isnan,ProtokolldateiNeu(:,7)),:) = [];
% end

end
```

T. Ermittlung der Spaltenbezeichnung

- **Autor**: Robert Bernerstätter
- **Dateiname**: getSpaltenBezeichnung.m
- **Datum**: 23.09.2013
- **Version**: 1.20
- **Beschreibung**: Das Programm sucht die Spaltenbezeichnung in Excel in Buchstabenform, einstellig oder zweistellig, für eine übergebene Spaltennummer.

Funktionsaufruf

Übergabeparameter:

- **spaltenNr**: Die Nummer der Spalte zu welcher der äquivalente Buchstabe gesucht werden soll.

Rückgabeparameter:

- **spaltenBezeichnung**: Der/die Spaltenbuchstabe/n für die übergebene Spaltennummer *spaltenNr*.

```
function spaltenBezeichnung = getSpaltenBezeichnung(spaltenNr)
```

Programmablauf

Es werden die einstelligen Buchstaben von *A-Z* gespeichert (**az**). Es sind ebenfalls die Doppelbezeichnungen (*AA-ZZ*) nötigt. Dazu werden alle zweistelligen Kombinationen von *1-26* ermittelt (**ind**) und in eine Buchstabenkombination umgewandelt. Die einstelligen und zweistelligen Buchstabenkombinationen kommen in ein gemeinsames Cellarray (**bez**). Der übergebene Index *splatenNr* entspricht dem Eintrag im Cellarray und somit dem gesuchten Buchstaben, der als *spaltenBezeichnung* zurückgegeben wird.

```
az = char(65:90)';
einstellig = cellstr(az);
n = (1:26)';
ind = allcomb(n,n);
zweistellig = cellstr(az(ind));
bez = [einstellig;zweistellig];
spaltenBezeichnung = bez{spaltenNr};

end
```

U. Speichern des Ergebnisses

- **Autor**: Robert Bernerstätter
- **Dateiname**: speichereErgebnis.m
- **Datum**: 23.09.2013
- **Version**: 1.10
- **Beschreibung**: Das Programm erstellt die Ergebnisdatei mit den Planartikel und den Produktfamilien. Die Planartikel werden in das erste Registerblatt der Exceldatei geschrieben, die Produktfamilien werden getrennt nach SGE in nachfolgende Registerblätter geschrieben.

Funktionsaufruf

Übergabeparameter:

- **captionPlanartikel**: Spaltenüberschriften für die Planartikel.
- **Planartikel**: Eine `n` x `m` Matrix mit den Planartikeln. Jede Zeile `n` enthält einen Planartikel. Die Spalten `m` enthalten die technischen Spezifikationen, die Auftrags- und Materialnummer, sowie eine laufende Nummerierung.
- **captionFamilien**: Spaltenüberschriften für die Produktfamilien.
- **Produktfamilien**: Ein `n` x `1` Cellarray mit den Aufträgen für die Planartikel. Jede Zeile `n` enthält eine Cell mit einer `k` x `m` Matrix. Jede Zeile `k` enthält einen Auftrag der SGE. Die Spalten `m` enthalten die technischen Spezifikationen, die Auftrags- und Materialnummer. In der letzten Spalte befindet sich eine Nummer, die den Auftrag seinem Planartikel eindeutig zuordnet (siehe Beschreibung des Rückgabeparameters **Planartikel** .
- **schreibeErgebnis**: Speicherpfad für die Ergebnisdatei.

Rückgabeparameter:

KEINE

```
function speichereErgebnis(captionPlanartikel,Planartikel,captionFamilien,...
    Produktfamilien, schreibeErgebnis)
```

Speichere Planartikel

Zuerst werden die Planartikel gespeichert. Um eine gute Formatierung der Daten zu garantieren wird auf die Standardbefehle für die Handhabung von Exceldateien verzichtet und mit actxserver gearbeitet. Es werden die Planartikel in die Datei geschrieben und danach die drei Standardregisterblätter von Excel (*Tabelle1- Tabelle3*) gelöscht. In der Reihenfolge der Planartikel werden im Anschluss die Produktfamilien in die Datei geschrieben. Dabei wird pro SGE ein Datenblatt verwendet und mit der Bezeichnung der SGE beschriftet. Die Aufträge werden nach deren Zugehörigkeit zu den Planartikeln sortiert.

```
warning ('off','all');
Excel = actxserver('Excel.Application');
wb = Excel.Workbooks.Add;
wb.Sheets.Add;
ws = wb.Sheets;
ws.Item(1).Name = 'Planartikel';
planartikel = get(ws, 'Item',1);
invoke(planartikel,'Activate');
Activesheet = Excel.Activesheet;
schreibePlan = [captionPlanartikel'; Planartikel];
zeilenPlan = size(schreibePlan,1);
spaltenPlan = size(schreibePlan,2);
range = strcat('A1:',getSpaltenBezeichnung(spaltenPlan),...
    num2str(zeilenPlan));
ran = Excel.Activesheet.get('Range',range);
set(ran, 'Value', schreibePlan);
```

Lösche Standardtabellenblätter

Die drei Standardregisterblätter werden gelöscht, um eine einheitliche Formatierung zu gewährleisten.

```
sheetName = 'Tabelle';
try
Excel.ActiveWorkbook.Worksheets.Item([sheetName '1']).Delete;
Excel.ActiveWorkbook.Worksheets.Item([sheetName '2']).Delete;
Excel.ActiveWorkbook.Worksheets.Item([sheetName '3']).Delete;
catch
;
end
```

Speichere Produktfamilien

Pro SGE werden die Produktfamilien in ein Registerblatt geschrieben. Das erfolgt nach dem ersten Registerblatt mit den Planartikeln.

```
spaltenFamilieNum = size(Produktfamilien{1,2},2);
spaltenFamilieStr = getSpaltenBezeichnung(spaltenFamilieNum);
for(i=1:1:size(Produktfamilien,1))
    ws = wb.Worksheets;
    ws.Add([],ws.Item(ws.Count));
    ws.Item(i+1).Name = Produktfamilien{i,1};
    sheet = get(ws, 'Item',i+1);
    invoke(sheet,'Activate');
    Activesheet = Excel.Activesheet;
    schreibeFam = [captionFamilien';Produktfamilien{i,2}];
    range = strcat('A1:',spaltenFamilieStr,num2str(size(schreibeFam,1)));
    ran = Excel.Activesheet.get('Range', range);
    set(ran, 'Value', schreibeFam);
end
```

Schließe Excelserver

Die im Hintergrund geöffneten Excelanwendungen müssen geschlossen werden, um Probleme mit anderen zu vermeiden. Die Datei wird unter dem vorab bestimmten Dateipfad gespeichert. Nach dem Schließen des Actxservers wird die Datei mit dem Ergebnis geöffnet.

```
wb.SaveAs(schreibeErgebnis);
wb.Close;
Excel.Quit;
delete(Excel);
winopen(schreibeErgebnis);
warning ('on','all');

end
```

Attributwichtigkeit auf Anlagen

Tabelle 13: Attributeinfluss auf Anlagen

Anlage	ANLNR	Werkstoff	Grundstoff	Dicke	Breite	Länge	Zustand	Qualität	Oberfl	Form	Plati(%)	SV	Rndduchm.
Blocksäge BSH	2	2	0	2	2	2	0	2	0	0	0	0	0
Blockfräse BLF	5	3	0	1	2	2	0	1	1	0	0	1	0
Warumduo WDO	10	3	3	3	2	0	0	2	0	1	3	0	0
Warmbandschopfsch.	14	0	0	3	0	1	0	0	0	0	0	0	0
Warmquarto WQ	20	3	1	3	1	0	1	1	1	0	0	1	0
Kaltquarto KQ	30	3	1	3	1	0	1	1	2	0	1	1	0
Glanzduo 1 GD1	35	3	1	3	1	0	2	1	3	0	1	1	0
Glanzduo 2 GD2	36	3	1	3	1	0	2	1	3	0	1	1	0
Glanzduo 3 GD3	37	3	1	3	1	0	2	1	3	0	1	1	0
Längsteilschere LSB	120	1	1	2	3	0	1	1	2	0	1	1	0
Längsteilschere LS1	130	1	1	1	3	0	1	1	2	0	1	1	0
Querteilschere schwer QS	170	2	1	3	1	2	1	1	2	1	1	1	0
Querteilschere leicht QL	180	2	1	2	1	2	1	1	2	1	1	1	0
Querteilschere Dickband DI	220	1	1	3	1	2	1	1	2	1	1	1	0
Querteilanlage Recker QRE	225	1	1	2	1	2	1	1	2	1	1	1	0
Plattensäge 2 PL2	270	2	1	1	2	2	1	1	1	0	0	1	0
Plattensäge 3 PL3	273	2	1	1	3	3	1	1	1	0	0	1	0
Plattensäge 4 PL4	274	2	1	1	1	2	1	1	1	0	0	1	0
Plattensäge 5 PL5	275	2	1	1	1	2	1	1	1	0	0	1	0
Streckrichtanalage STR	300	2	1	3	1	0	1	2	2	0	1	1	0
Richtmaschine 1 RI1	321	2	0	3	2	2	1	1	2	0	0	1	0
Richtmaschine 2 RI2	322	2	0	3	2	2	1	1	2	0	0	1	0
Richtmaschine 3 RI3	323	2	0	3	2	2	1	1	2	0	0	1	0
Plattenrecker 1 RE1	331	1	0	3	2	2	1	1	1	0	0	1	0
Plattenrecker 2 RE2	332	1	0	3	2	2	1	1	1	0	0	1	0
Bandbeizanalge BB	380	3	1	2	1	0	0	1	2	0	0	2	0
25T Stanze 25T	400	0	1	3	3	0	0	0	2	0	0	1	3
75T Stanze 75T	410	0	1	3	3	0	0	0	2	0	0	1	3
Plattenfräse PLF	500	2	0	3	2	3	1	0	1	2	0	0	0
Kathodenrandbesch. 1 KRB	520	0	0	1	2	1	0	0	0	0	0	3	0
Kathodenrandbesch. 2 KR2	522	0	0	1	2	1	0	0	1	0	0	3	0
Blockstosso. Gautschi BLG	540	2	2	2	1	1	0	2	0	0	1	0	0
Bundglüho. West 1 BO5	605	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. West 2 BO6	606	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. West 3 BO7	607	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. West 4 BO8	608	2	1	1	2	0	3	1	1	0	1	1	0
Banddurchzugso. 2 BD2	622	3	1	3	2	0	2	1	0	0		1	0
Banddurchzugso. 3 BD3	623	3	1	3	2	0	2	1	0	0	1	1	0
Bundglühö. Ost BOO	640	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. 1 BO1	641	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. 2 BO2	642	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. 3 BO3	643	2	1	1	2	0	3	1	1	0	1	1	0
Bundglüho. 4 BO4	644	2	1	1	2	0	3	1	1	0	1	1	0
Horizontalvergüteo. 1 HT1	705	3	0	3	3	3	0	2	0	0	0	1	0
Horizontalvergüteo. 2 HT2	706	3	0	3	3	3	0	2	0	0	0	1	0
US Prüfanlage US	805	1	0	3	2	2	0	1	0	0	0	0	0
Leitfähigkeitsprüfung LEI	806	1	0	3	2	2	0	1	0	0	0	0	0
Verp. (U.Vis)_ VV3	963	0	0	3	3	3	0	1	1	0	0	0	0
Verp.Bänder >=500 VB1	971	2	1	0	1	0	1	3	1	0	0	1	0
Verp. Bänder <500 VB2	972	2	1	0	1	0	1	3	1	0	0	1	0
Verp. Paketverp. VT1	973	2	1	0	1	1	0	2	0	0	1	1	0
Verp. Bleche West VBW	982	2	1	0	1	1	0	2	0	0	1	1	0

Protokolldatei

SGE	Metrik	Algorithmus	# Familien	UG BKG[%]	OG BKG[%]	Werkstoff	Dicke	Breite	Länge	Zustand	Qualität	Oberfläche	Form	Grundwerkst	Vorplanmater	Sondervorsch	Rondendurch	Anlagenfolge
VB	hamming	ward	7	3,34	25,69	1	0	0	0	0	0	0	0	0	1	0	0	1
VD	euclidean	ward	26	1,06	8,85	1	0	1	0	1	0	1	0	0	0	1	0	0
VL	euclidean	ward	45	0,57	5,59	1	1	1	0	1	0	0	1	1	1	1	0	1

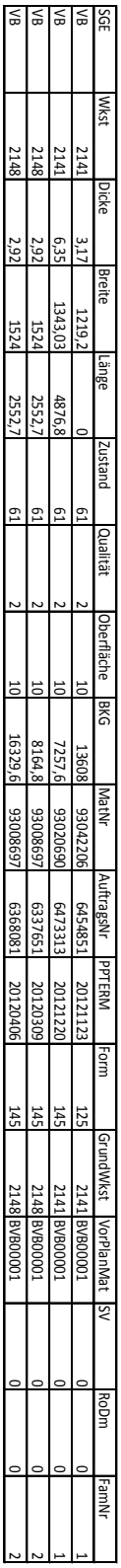
Abbildung 42: Auszug aus Protokolldatei

Datei der Auftragsdaten

SGE	Wkst	D	B	L	Zustand_Codiert	Qual	Obf1	BKG	MatNr	Auftragsnummer	Termin	Form	GWkst	VORPLAN_Codiert	SV	RnDch	Zustand	VORPLAN
VB	2112	10	1484	2600	13873	2	10	5000	93001192	6318481	20120113	145	2112	16946	0	0	61	BVB00002
VB	2112	10	1484	2040	13873	2	10	1000	93018320	6318482	20120127	145	2112	16946	0	0	61	BVB00002

Abbildung 43: Beispiel für Auftragsdatendatei

Ergebnis Produktfamilie



SGE	Wkst	Dicke	Breite	Länge	Zustand	Qualität	Oberfläche	BKG	MatNr	AuftragsNr	PPTERM	Form	GrundWkst	VorPlanMat	SV	RoDm	FamNr
VB	2141	3,17	1219,2	0	61	2	10	13608	93042206	6454851	20121123	125	2141	BVB00001	0	0	1
VB	2141	6,35	1343,03	4876,8	61	2	10	7257,6	93020690	6473313	20121220	145	2141	BVB00001	0	0	1
VB	2148	2,92	1524	2552,7	61	2	10	8164,8	93008697	6337651	20120309	145	2148	BVB00001	0	0	2
VB	2148	2,92	1524	2552,7	61	2	10	16329,6	93008697	6368081	20120406	145	2148	BVB00001	0	0	2

Abbildung 44: Auszug Planartikeldatei

Modellvariablen der aggregierten Planung

Tabelle 14: Modellvariablen der aggregierten Planung[244]

Variable	Bedeutung
a_g	Produktionskoeffizient der Produktfamilie g
c^{alt}	Kapazitätsabschlagsfaktor zur Berücksichtigung der Rüstzeiten
cl_{gt}^{+}	Strafkostensatz für positive Abweichung des Lagerbestandes von Produktfamilie g pro Mengeneinheit
cl_{gt}^{-}	Strafkostensatz für negative Abweichung des Lagerbestandes von Produktfamilie g pro Mengeneinheit
cu_t^{+}	Strafkostensatz für positive Abweichungen der Zusatzkapazität in Makroperiode t pro Kapazitätseinheit
cu_t^{-}	Strafkostensatz für negative Abweichungen der Zusatzkapazität in Makroperiode t pro Kapazitätseinheit
d_{sgt}	Nachfragemenge in Szenario s von Produktfamilie g in Periode t
l_g	Lagerkostensatz für Produktfamilie g pro Mengeneinheit und Periode
u	Kosten für eine Einheit Zusatzkapazität
$zfopt_s$	Optimaler Zielfunktionswert für Szenario s
C^{max}	Verfügbare Normalkapazität pro Periode
L_{gt}	Lagerbestand von Produktfamilie g am Ende der Periode t
L_{gt}^{+}	Erhöhung des Lagerbestands von Produktfamilie g am Ende von Periode t im Vergleich zum vorhergehenden Planungslaufs
L_{gt}^{-}	Senkung des Lagerbestands von Produktfamilie g am Ende von Periode t im Vergleich zum vorhergehenden Planungslaufs
L_{sgt}^{+}	Szenarioabhängige Erhöhung des Lagerbestandes von Produkttyp g am Ende von Periode t
L_{sgt}^{-}	Szenarioabhängige Senkung des Lagerbestandes von Produkttyp g am Ende von Periode t
U_t^{alt}	Zusatzkapazität in Periode t im vorhergegangenen Planungslauf
U^{max}	Maximale Zusatzkapazität pro Periode
U_{st}^{+}	Szenarioabhängige Erhöhung der Zusatzkapazität in Periode t
U_{st}^{-}	Szenarioabhängige Senkung der Zusatzkapazität in Periode t
U_t	Genutzte Zusatzkapazität in Periode t
U_t^{+}	Erhöhung der Zusatzkapazität in Periode t im Vergleich zum vorhergehenden Planungslauf
U_t^{-}	Senkung der Zusatzkapazität in Periode t im Vergleich zum vorhergehenden Planungslauf
X_{sgt}	Produktionsmenge in Szenario s von Produktfamilie g in Periode t.
ZF_s	Kosten des Grund- und Ergänzungsplans für Szenario s
ZF_s^{+}	Überschreitung der Kosten des Grund- und Ergänzungsplans des Optimums von Szenario s

[244] Vgl. Gebhard (2009), S. XXIIIff.