wbk
Institut für Produktionstechnik

Prädiktiv-reaktives Scheduling zur Steigerung der Robustheit in der Matrix-Produktion

Zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften

der KIT-Fakultät für Maschinenbau

Karlsruher Institut für Technologie (KIT)

genehmigte

Dissertation

von

Fabio Echsler Minguillon, M.Sc.

aus Stuttgart

Tag der mündlichen Prüfung:	15.05.2020
Hauptreferent:	Prof. Dr.-Ing. Gisela Lanza
Korreferent:	Prof. Dr.-Ing. habil. Hermann Lödding

Karlsruher Institut für Technologie

Vorwort des Herausgebers

Die schnelle und effiziente Umsetzung innovativer Technologien wird vor dem Hintergrund der Globalisierung der Wirtschaft der entscheidende Wirtschaftsfaktor für produzierende Unternehmen. Universitäten können als "Wertschöpfungspartner" einen wesentlichen Beitrag zur Wettbewerbsfähigkeit der Industrie leisten, indem sie wissenschaftliche Grundlagen sowie neue Methoden und Technologien erarbeiten und aktiv den Umsetzungsprozess in die praktische Anwendung unterstützen.

Vor diesem Hintergrund soll im Rahmen dieser Schriftenreihe über aktuelle Forschungsergebnisse des Instituts für Produktionstechnik (wbk) am Karlsruher Institut für Technologie (KIT) berichtet werden. Unsere Forschungsarbeiten beschäftigen sich sowohl mit der Leistungssteigerung von Fertigungsverfahren und zugehörigen Werkzeugmaschinen- und Handhabungstechnologien als auch mit der ganzheitlichen Betrachtung und Optimierung des gesamten Produktionssystems. Hierbei werden jeweils technologische wie auch organisatorische Aspekte betrachtet.

Prof. Dr.-Ing. Jürgen Fleischer

Prof. Dr.-Ing. Gisela Lanza

Prof. Dr.-Ing. habil. Volker Schulze

Vorwort des Verfassers

Die vorliegende Arbeit entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am wbk Institut für Produktionstechnik des Karlsruher Instituts für Technologie (KIT).

Mein besonderer Dank gilt Frau Prof. Dr.-Ing. Gisela Lanza für die Betreuung meiner Arbeit als Hauptreferentin und das mir entgegengebrachte Vertrauen in den vergangenen Jahren. Weiter danke ich Herrn Prof. Dr.-Ing. habil. Hermann Lödding für die Übernahme des Korreferats und Herrn Prof. Dr.-Ing. habil. Kai Furmans für den Prüfungsvorsitz. Bei Prof. Neil Duffie möchte ich mich für die Unterstützung und wissenschaftliche Diskussion während meines Forschungsaufenthalts an der University of Wisconsin-Madison herzlich bedanken. Dem Karlsruhe House of Young Scientists (KHYS) und dem Alumni-Club des wbk danke ich für die Förderung dieses Aufenthalts.

Bei meinen Kolleginnen und Kollegen am wbk – insbesondere im Bereich Produktionssysteme – möchte ich mich für die kollegiale Zusammenarbeit und die gute Gemeinschaft auch außerhalb der Arbeit danken. Im Zusammenhang mit der vorliegenden Dissertation gilt mein besonderer Dank Andreas Kuhnle, Constantin Hofmann und Nicole Stricker, deren hilfreiches Feedback wesentlich zum Gelingen der Arbeit beigetragen hat. Andreas Kuhnle und Benjamin Häfner möchte ich vor allem für die gute und enge Zusammenarbeit in der vergangenen eineinhalb Jahren danken. Zudem möchte ich mich bei meinen Studenten Jannik Zöllner, Philipp Spitzer, Simon Hort und Guiheng Zhang für ihren großen Einsatz und viele gewinnbringende Diskussionen bedanken.

Meinen Eltern danke ich dafür, dass Sie mir immer den Freiraum gegeben haben, meinen Interessen nachzugehen, mich stets ermutigen und mir in allen Lebenslagen zur Seite stehen.

Abschließend möchte ich dir danken, Marie. Deine unendliche Geduld und Unterstützung während der letzten Jahre hat mir die Kraft gegeben, die intensive Zeit am wbk zu meistern. Um dir alles zu sagen was ich sagen möchte, ist dieses Vorwort zu kurz. Die Arbeit ist dir gewidmet.

Karlsruhe, im Juni 2020

Fabio Echsler Minguillon

Abstract

Due to the increasing individualization of products, manufacturing companies are offering more and more variants with decreased quantities per variant. In addition, customer demand is becoming more volatile and difficult to predict. The main challenge is to economically produce a fluctuating mix of variants with fluctuating total quantities. Matrix-Production systems aiming for a production in batch size 1 decoupled from a takt are therefore a current object of research. In addition to the design of these systems, an increasingly important role is filled by production planning and control, since the material flows in such production systems are highly complex.

The state of research is characterized by a multitude of predictive-reactive methods for scheduling even in complex production systems. However, there is no approach that specifically considers robustness in predictive planning in order to enable reactive rescheduling to maintain the desired logistical performance despite unforeseen disruptions.

Therefore, a method for predictive-reactive product control of matrix-structured production systems was developed in this thesis, which allows the determination of an optimal degree of robustness in predictive robust scheduling and thus enables an optimal mix of prevention and reaction in production control. The method consists of three parts: First, in predictive robust scheduling, a schedule is generated on the basis of the production program, in which a desired extent of slip times between processing steps is then inserted. The robust schedules are then carried out in a discrete-event simulation. In the event of longer disturbances, a rescheduling corridor is determined secondly, which indicates which processing steps of which orders must be rescheduled depending on the duration of the disturbance and the underlying schedule. The rescheduling corridors are then rescheduled thirdly in reactive rescheduling and the results are transferred to the discrete event simulation for reintegration. Reactive re-scheduling uses reinforcement learning based on a decentralized Markov process to learn optimal selection strategies for orders depending on the station. The method was tested in an application for a concept of a flexible body-in-white production with a partner from the automotive industry within the BMWi-funded joint project "SmartBodySynergy".

The developed method contributes to the understanding of the concept of robustness as well as to the application possibilities and limits of reinforcement learning in production control. To the author’s knowledge, the work is the first approach to integrate robustness considerations directly into predictive-reactive scheduling approaches in order to improve the logistical performance.

Inhaltsverzeichnis

Abkürzungen

Abkürzung	Beschreibung
BMWi	Bundesministerium für Wirtschaft und Energie
CP	Constraint-Problem
CP-OPT	Constraint-basierte Optimierung
DEC-MDP	Decentralized Markov Decision Process (dezentraler Markovscher Entscheidungsprozess)
DEC-POMDP	Decentralized Partially Observable Markov Decision Process (dezentraler teilweise beobachtbarer Markovscher Entscheidungsprozess)
DPS	Distributed Policy Search (verteilte Strategie-Iteration)
FTF	Fahrerloses Transportfahrzeug
JIS	Just-In-Sequence
LP	Lineares Problem
MDP	Markov Decision Process (Markovscher Entscheidungsprozess)
MLS	Maximum Likelihood Schedule
MRCPSP	Multimode Resource-Constrained Project Scheduling Problem
NP	Nichtdeterministisch polynomiell
OEE	Overall Equipment Efficiency
OPL	Optimization Programming Language
POMDP	Partially Observable Markov Decision Process (teilweise beobachtbarer Markovscher Entscheidungsprozess)
PPS	Produktionsplanung und -steuerung
RCPSP	Resource-Constrained Project Scheduling Problem
RK	Rescheduling-Korridor
RL	Reinforcement Learning

Formelverzeichnis

Formelzeichen	Größe
A	Aktionenraum
a	Aktion
A_t	Gewählte Aktion a zum Zeitpunkt t
b_s	Pufferkapazität an Station s
c	Arbeitszentrum
C_j	Fertigstellungszeitpunkt von Auftrag j
dd_j	Fälligkeitstermin von Auftrag j
d_s	Dauer der Störung an Station s
d_s^R	Restdauer der Störung an Station s
$d_s^{B,OG}$	Belegungszeit von Station s bis zur oberen Grenze eines Rescheduling-Korridors
$d_s^{B,UG}$	Belegungszeit von Station s ab der unteren Grenze eines Rescheduling-Korridors
e	Episode des DPS
E	Epoche des DPS (Anzahl Episoden bis zur Durchführung eines Update-Schrittes)
E_{max}	Maximale Anzahl zu durchlaufender Epochen
$ed_{j,s}^{k_j}$	Erwartete Verzögerung einer eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$
G	Belohnungsfunktion
g_t	Belohnung eines Agenten zum Zeitpunkt t
$hd_{j,s}^{k_j}$	Horizontale erwartete Verzögerung einer eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$
$hs_{js}^{k_j}$	Horizontale Schlupfzeit einer eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$
$IWP_{o_j^{k_j}}$	Zeitintervall für die Wartezeit im Puffer für jeden Bearbeitungsschritt $o_j^{k_j}$
$IOWP_{m_{j,s}^{k_j}}$	Optionales Zeitintervall für jede Bearbeitungsalternative $m_{j,s}^{k_j}$ mit Größe $tb_{j,s}^{k_j}$.
$IOP_{m_{j,s}^{k_j}}$	Optionales Zeitintervall für jede Bearbeitungsalternative $m_{j,s}^{k_j}$ mit Länge $p_{j,s}^{k_j}$

$IP_{o_j^{k_j}}$	Zeitintervall für die Dauer jedes Bearbeitungsschritts $o_j^{k_j}$
$IT_{o_j^{k_j}}$	Zeitintervall für die Transportzeit für jeden Bearbeitungsschritt $o_j^{k_j}$
$IOT_{m_{j,s}^{k_j}}$	Optionales Zeitintervall für jede Bearbeitungsalternative $m_{j,s}^{k_j}$ mit Größe $t_{s,s'}$ dessen mit der Optimierung ermittelter Start und Ende angibt, wann ein Auftrag von einer Station s zu einer Station s' transportiert wird
J	Menge der n Aufträge des Produktionsprogramms mit $J = \{j_1, \dots, j_n\}$
j	Auftrag im Produktionsprogramm
JT	Menge der l Auftragstypen des Produktionsprogramms mit $JT = \{jt_1, \dots, jt_l\}$
jt_j	Auftragstyp von Auftrag j
K_j	Indexmenge der k_j Bearbeitungsschritte von Auftrag j
$L(\pi_s)$	Leistungsfähigkeit der Strategie π_s
M	Menge der Bearbeitungsalternativen aller Aufträge mit $M = \bigcup_{j\in J, k_j \in K_j} M_j^{k_j} = \{m_{js,}^1, \dots, m_{js}^{k_j}, \dots, m_{j+n-1,s}^1, \dots, m_{j+n-1}^{k_{j+n-1}}\}$
$M_j^{k_j}$	Menge der Bearbeitungsalternativen $m_{j,s}^k$ jedes Bearbeitungsschrittes o_j^k mit $M_j^{k_j} = \{m_{j,s}^1, \dots, m_{j,s}^{k_j}\}$.
$m_{j,s}^{k_j}$	Bearbeitungsalternative für den k_j-ten Bearbeitungsschritt von Auftrag j an Station s
m	Verkürzte Schreibweise für eine Bearbeitungsalternative $m_{j,s}^{k_j}$
$MTBF_s$	Mean Time Between Failures (Mittlere Zeit zwischen Störungen)
$MTTR_s$	Mean Time To Repair (Mittlere Reparaturdauer)
O	Menge der Bearbeitungsschritte aller Aufträge j mit $O = \bigcup_{j\in J} O_j = \{o_{j,}^1, \dots, o_j^{k_j}, \dots, o_{j+n-1}^1, \dots, o_{j+n-1}^{k_{j+n-1}}\}$
O_B^{OG}	Menge der Bearbeitungsschritte, die die obere Grenze eines Rescheduling-Korridors schneiden
O_B^{UG}	Menge der Bearbeitungsschritte, die die untere Grenze eines Rescheduling-Korridors schneiden
O_D	Menge der direkt von einer Störung betroffenen Bearbeitungsschritte
O_I	Menge der indirekt von einer Störung betroffenen Bearbeitungsschritte
O_j	Menge der k_j Bearbeitungsschritte $o_j^{k_j}$ jedes Auftrags mit $O_j = \{o_{j,}^1, \dots, o_j^{k_j}\}$
$o_j^{k_j}$	Bearbeitungsschritt k_j von Auftrag j
o	Verkürzte Schreibweise für einen Bearbeitungsschritt $o_j^{k_j}$

OG	Obere Grenze eines Rescheduling-Korridors
P	Übergangsgesetz
$p_{j,s}^{k_j}$	Prozesszeit zur Durchführung von $m_{j,s}^{k_j}$
$p_{j,s}^{k_j,R}$	Restliche Prozesszeit zur Durchführung von $m_{j,s}^{k_j}$ zu einem beliebigen Zeitpunkt nach Bearbeitungsbeginn
p_{krit}	Kritische Prozesszeit innerhalb eines Rescheduling-Korridors
$Q^{\pi}(z,a)$	Aktionswertfunktion eines Zustandes z bei Wahl der Aktion a in Abhängigkeit der Strategie $\pi(z)$
R	Parameter für die Robustheitsanforderung im prädiktiven robusten Scheduling
$r_{s,jt_j,jt_{j'}}$	Rüstzeit an Station s für die Umrüstung von Auftragstyp jt_j auf Auftragstyp $jt_{l'}$
rt_s	Anforderungszeit, die nach Bearbeitungsende eines Auftrags vergeht, bis ein Transportmittel an Station s verfügbar ist
rt_s^R	Restanforderungszeit, die nach Bearbeitungsende eines Auftrags noch vergeht, bis ein Transportmittel an Station s verfügbar ist
S	Menge der i Stationen bzw. Agenten des Produktionssystems mit $S = \{s_1, \dots, s_i\}$
$s_{m_{j,s}^{k_j}}$	Station bzw. Agent, an der Bearbeitungsalternative $m_{j,s}^{k_j}$ durchgeführt wird
s_i	Station bzw. Agent i des Produktionssystems
$Sched_{Präd}$	Prädiktiver Schedule
$Sched_{Rob}$	Robuster Schedule
ST	Menge der c Stationstypen des Produktionssystems mit $ST = \{st_1, \dots, st_c\}$
st_s	Stationstyp von Station s
SZ_j	Schlupfzeit von Auftrag j
SZ_{total}	Gesamte Schlupfzeit in einem Rescheduling-Korridor
T_e	Anzahl der Entscheidungssituationen in einer Episode e
t_0	Störungszeitpunkt
$tb_{j,s}^{k_j}$	Wartezeit von $m_{j,s}^{k_j}$ im Puffer von Station s
tb	Verkürzte Schreibweise für eine Wartezeit im Puffer $tb_{j,s}^{k_j}$
$td_{j,s}^{k_j}$	Gesamte erwartete Verzögerung einer eingeplanten Bearbeitungsalternative
$t_{s,s'}$	Transportzeit von Station s zu Station s'
$t_{j,s'}^R$	Verbleibende Transportzeit von Auftrag j vom aktuellen Ort zu Station s'

UG	Untere Grenze eines Rescheduling-Korridors
$V^{\pi}(z)$	Zustandswertfunktion eines Zustandes z in Abhängigkeit der Strategie $\pi(z)$
v_s	Verfügbarkeit von Station s mit $v_s \in [0,1]$
$vd_{j,s}^{k_j}$	Vertikale erwartete Verzögerung einer eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$
VI_s^S	Verfügbarkeitsintervall einer Station s im Rescheduling-Korridor
VI_j^J	Verfügbarkeitsintervall eines Auftrags j im Rescheduling-Korridor
$vs_{j,s}^{k_j}$	Vertikale Schlupfzeit einer eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$
$x_{j,s}^{k_j}$	Geplanter Startzeitpunkt von $m_{j,s}^{k_j}$
x	Verkürzte Schreibweise für einen geplanten Startzeitpunkt $x_{j,s}^{k_j}$
Z	Zustandsraum
z	Zustand
z_t	Vorliegender Zustand z zum Zeitpunkt t
α	Stationskonfiguration
β	Auftragseigenschaften bzw. Prozesscharakteristika
γ	Zielfunktion
δ	Lernrate
$\lambda_{j,s}^{k_j}$	Maximal erlaubte erwartete Verzögerung eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$
ε	Wahrscheinlichkeit zur Auswahl einer zufälligen Aktion
$\theta_{jt_j,s}^{k_j}$	Präferenz-Parameter für jeden Besuch k_j eines Auftragstyps jt an Station s zur Beschreibung der Strategie π_s
$\pi_s(z,a)$	Wahrscheinlichkeit der Wahl einer Aktion a im Zustand z bei verfolgen einer Strategie π_s
π_s	Strategie des Agenten bzw. der Station s
τ	Temperatur-Parameter der Boltzmann-Exploration mit $\tau \in [0,\infty)$
ψ_s	Sequenz von eingeplanten Bearbeitungsalternativen an einer Station s
ω	Abbruchkriterium

1 Einleitung

1.1 Ausgangssituation und Problemstellung

Das produzierende Gewerbe hat für die Wirtschaftsleistung in Deutschland nach wie vor eine enorme Bedeutung. 2017 lag der Anteil des Sektors am gesamten Bruttoinlandsprodukt bei 26 % (Statistisches Bundesamt 2018). Mit 7,3 Millionen Beschäftigten waren zudem ca. 25 % aller in der EU im produzierenden Gewerbe tätigen Personen in Deutschland angestellt (Brandmüller & Önnerfors et al. 2018). Außerdem war Deutschland im Zeitraum zwischen 2001 und 2012 das einzige westeuropäische Land, das seinen Anteil des produzierenden Gewerbes steigern konnte, während dieser in anderen Ländern um bis zu 30 % sank (Ganschar & Gerlach et al. 2013). Umso drängender sind in diesem Zusammenhang die Herausforderungen einer zunehmend unsicheren und schnelllebigen politischen und ökonomischen Umwelt, mit denen sich die Industrie derzeit konfrontiert sieht.

Der Trend zur Individualisierung von Produkten führt zu einer höheren Anzahl an Varianten bei gleichzeitig sinkenden Stückzahlen je Variante (Koren 2010). Die Produktion wird demnach vermehrt von einer individualisierten und sehr flexiblen Serienfertigung geprägt (von Heynitz & Bremicker et al. 2016; Ganschar & Gerlach et al. 2013).

Nur für 5 % der Unternehmen hat die Produktion in Losgröße 1 keine Relevanz (STAUFEN 2018). In der Automobilproduktion sind die Baureihen über Konfiguratoren individualisierbar, sodass allein für die 5er-Baureihe von BMW rechnerisch über 10^{17} unterschiedliche Fahrzeugvarianten konfigurierbar sind (Kaluza & Blecker et al. 2005). Aufgrund des vermehrten Aufkommens alternativer Antriebsstränge wird erwartet, dass die Variantenvielfalt in den nächsten Jahren weiter ansteigen wird (Bauer & Arnold et al. 2010; Greschke 2016; Karle 2018). Es ist festzuhalten, dass das Streben nach einer Massenproduktion in diesem Umfeld an Bedeutung verliert und stattdessen die Realisierung einer kostengünstigen kundenindividuellen Produktion in den Fokus rückt (BMBF 2015). Hinzu kommt die in vielen Branchen vorherrschende Verkürzung von Produktlebenszyklen, die die Produktionsdauer der angebotenen Varianten verringert

(Wallentowitz & Freialdenhoven et al. 2009). Dadurch verschärfen sich die Herausforderungen weiter[1].

Eine weitere wesentliche Herausforderung besteht in der Volatilität der Kundennachfrage, die zunehmend schwerer prognostizierbar ist (Abele & Reinhart 2011; Lanza & Nyhuis et al. 2017). Gründe für die Volatilität liegen z. B. in geändertem Kaufverhalten in Abhängigkeit von der konjunkturellen Lage, saisonal bedingten Schwankungen, der jeweiligen Phase im Produktlebenszyklus, technologischen Innovationen sowie, insbesondere bei Fahrzeugen, in Gesetzes- oder Umweltvorschriften (Grinninger 2012). So haben sich unter den 50 meistverkauften Fahrzeugmodellen in Deutschland die Zulassungszahlen im Zeitraum Januar bis Juli zwischen 2018 und 2019 von ca. -20 % bis +100 % verändert[2]. Bei Dieselfahrzeugen ergibt sich von Mitte 2018 bis Mitte 2019 eine Spannbreite der Zulassungszahlen von ca. 100 %[3], die möglicherweise auf Unsicherheit bezüglich zukünftiger gesetzlicher Rahmenbedingungen zurückgeführt werden kann. Aufgrund der Individualisierung ist Lagerhaltung zudem oft keine wirtschaftliche Option mehr (Nyhuis & Reinhart et al. 2008).

Die hohe Anzahl unterschiedlicher Produkte und der an die Kundennachfrage angepasste ständig schwankende Produktmix führen in starr verketteten Produktionssystemen zu Produktivitätsverlusten aufgrund von Taktzeitspreizungen (Greschke 2016; Schönemann & Herrmann et al. 2015). Aus diesem Grund ist die Matrix-Produktion aktueller Forschungsgegenstand (Kern & Rusitschka et al. 2015; Greschke 2016; Bochmann 2018). Im Folgenden wird unter Matrix-Produktion ein Produktionssystem verstanden, das sich dynamisch an interne und externe Einflussfaktoren wie z. B. eine Verschiebung des Variantenmix, eine Änderung des Produktionsvolumens oder eine Integration neuer Produkte anpassen kann. Aufgrund der losen Verkettung ist die Matrix-Produktion vom Takt entkoppelt und ermöglicht durch die Nutzung von fahrerlosen Transportfahrzeugen (FTF) eine Produktion in Losgröße 1. Im Rahmen von For-

[1] Wissenschaftlicher Dienst des Deutschen Bundestags (2016), *Zur Diskussion um die Verkürzung von Produktlebenszyklen*. https://www.bundestag.de/resource/blob/438002/42b9bf2ae2369fd4b8dd119d968a1380/wd-5-053-16-pdf-data.pdf [08.01.2020].

[2] Kraftfahrtbundesamt (2019), *Veränderungen der Neuzulassungen von Personenkraftwagen von Januar bis Juli 2019 nach Marken und Modellreihen in Prozent*. https://www.kba.de/DE/Statistik/Fahrzeuge/Neuzulassungen/MonatlicheNeuzulassungen/2019/201907_GImonatlich/201907_n_top50.html?nn=2162804 [08.01.2020].

[3] Statista (2019), *Anzahl der Neuzulassungen von Personenkraftwagen mit Dieselmotor in Deutschland von Juli 2017 bis Juli 2019*. https://de.statista.com/statistik/daten/studie/468652/umfrage/monatliche-pkw-neuzulassungen-in-deutschland-dieselmotor/ [19.08.2019].

schungs- und Industrieinitiativen werden derzeit Lösungsansätze verfolgt, die Realisierungen der Matrix-Produktion mit einer ähnlichen Zielsetzung darstellen. Dazu gehören z. B. die Matrix-Produktion bei der KUKA AG[4], die modulare Montage bei der Audi AG[5] oder cubeTEC bei der Daimler AG[6].

Neben der Gestaltung der Matrix-Produktion kommt der Produktionsplanung und –steuerung (PPS) aufgrund der Vielzahl von Freiheitsgraden im Materialfluss eine besondere Rolle zu (Greschke 2016). Sie ist entscheidend für deren logistische Leistungsfähigkeit und muss dafür eine Vielzahl von Anforderungen erfüllen (Bauernhansl & Hompel et al. 2014). Die kurzfristige Einhaltung kundenindividueller Liefertermine wird zunehmend als Differenzierungsmerkmal im Wettbewerb wahrgenommen (Melzer-Ridinger 2018). Die PPS muss daher in der Lage sein, diese trotz Störungen im Produktionsablauf sicherzustellen (Grinninger 2012). Zudem muss sie die gestiegene Materialflusskomplexität loser Verkettung beherrschen können (Bauernhansl & Hompel et al. 2014). Aufgrund höherer Transportaufwände nimmt zudem die Bedeutung der Routenplanung von FTF als wichtiger Bestandteil der Auftragsdurchlaufzeit zu.

Zugleich ergibt sich ein hohes Potenzial durch die fortschreitende Digitalisierung der Produktion mit einer erhöhten Echtzeitverfügbarkeit von Produktionsdaten. Damit kann auf Abweichung von Produktionsplänen durch eine Umplanung von Aufträgen reagiert werden und somit eine signifikante Steigerung der Overall Equipment Efficiency (OEE) erreicht werden. (Ganschar & Gerlach et al. 2013)

1.2 Zielsetzung

Ziel der vorliegenden Arbeit ist die Erforschung einer Methode zur Produktionssteuerung in der Matrix-Produktion, die die vorgenannten Anforderungen adressiert. Das Scheduling als Teil der Produktionssteuerung legt die zeitliche Zuordnung von Aufträgen zu Stationen fest und ist damit wesentlicher Betrachtungsgegenstand der vorliegenden Arbeit. Es soll dazu geeignet sein, eine prädiktive Feinterminierung von Aufträgen auf Basis kundenindividueller Liefertermine durchzuführen. Zudem soll die logistische Leistungsfähigkeit hinsichtlich der Einhaltung kundenindividueller Liefertermine

[4] KUKA (2016), *Matrix-Produktion. Ein Beispiel für Industrie 4.0*. https://www.kuka.com/de-de/branchen/loesungsdatenbank/2016/10/solution-systems-matrix-produktion [14.08.2019].
[5] Audi (2016), *Die modulare Montage*. https://www.audi-mediacenter.com/de/audi-techday-smart-factory-7076/die-modulare-montage-7078 [08.01.2020].
[6] Daimler (2017), *Der Rohbau der Zukunft ist flexibel*. https://media.daimler.com/marsMediaSite/de/instance/ko/Mercedes-Benz-Cars-Der-Rohbau-der-Zukunft-ist-flexibel.xhtml?oid=30023981 [14.08.2019].

auch unter Störungen durch Ausnutzung der bestehenden produktionssystemseitigen Freiheitsgrade einer Matrix-Produktion robust erreichbar sein. Schließlich soll neben der prädiktiven Terminierung auch ein reaktives Rescheduling ermöglicht werden, um echtzeitnah auf über das antizipierte Maß hinausgehende Störungen reagieren zu können. Es stellt sich in dem Zusammenhang auch die Frage, welches Maß der Störungen antizipativ im prädiktiven robusten Scheduling berücksichtigt werden kann und welche Freiheitsgrade das reaktive Rescheduling benötigt. Dabei wird folgende Hypothese aufgestellt:

Ein reaktives Rescheduling wird durch ein prädiktives robustes Scheduling befähigt.

Für den entwickelten Ansatz werden dazu die folgenden forschungsleitenden Fragestellungen für das Scheduling in der Matrix-Produktion verfolgt:

- Wie kann ein prädiktives Scheduling eine Robustheit gegenüber Störungen erlangen?
- Wie können über das antizipierte Maß hinausgehende Störungen über ein reaktives Rescheduling korrigiert werden?
- Welches Maß an Robustheit im prädiktiven Scheduling ist notwendig, um ein reaktives Rescheduling zu begünstigen?

1.3 Aufbau der Arbeit

Der Aufbau der vorliegenden Arbeit zur Erreichung der Zielsetzung wird nachfolgend anhand der Kapitelstruktur beschrieben: Die für das Verständnis der Arbeit erforderlichen Grundlagen werden in Kapitel 2 dargelegt. In Kapitel 3 wird der aktuelle Stand der Forschung hinsichtlich prädiktiv-reaktivem Scheduling, der Robustheitsbewertung im Scheduling sowie Ansätzen für das Match-Up Rescheduling und das Scheduling mit Reinforcement Learning analysiert und das Forschungsdefizit formuliert. Das Match-Up Rescheduling hat das Ziel, lediglich einen Teil des zugrundeliegenden Schedules umzuplanen, sodass dieser anschließend weiter durchgeführt werden kann. Darauf aufbauend wird in Kapitel 4 die Methode zum prädiktiv-reaktiven Scheduling entwickelt. Die Erprobung der Methode am Beispiel einer Matrix-Produktion im automobilen Karosseriebau sowie der in diesem Rahmen entwickelte Softwaredemonstrator werden in Kapitel 5 beschrieben. Die Ergebnisse der Arbeit werden in Kapitel 6 diskutiert und ein Ausblick auf weitere mögliche Forschungsrichtungen gegeben. Die Arbeit schließt mit einer Zusammenfassung in Kapitel 7.

2 Grundlagen

In diesem Kapitel werden die für das Verständnis der Arbeit erforderlichen Definitionen und Konzepte erläutert. Zunächst wird in Kapitel 2.1 eine einführende Klassifizierung von Produktionssystemen vorgestellt, um eine Einordnung der Matrix-Produktion zu ermöglichen. In Kapitel 2.2 wird anschließend auf die Produktionsplanung und -steuerung (PPS) eingegangen, die den Einsatz des Produktionssystems determiniert. Der für die Arbeit zentrale Begriff der Robustheit und insbesondere dessen Bedeutung im Kontext des Scheduling wird in Kapitel 2.3 eingeführt. Abschließend werden in Kapitel 2.4 Klassen markovscher Entscheidungsprozesse eingeführt und grundsätzliche Lösungsverfahren erklärt.

2.1 Klassifizierung von Produktionssystemen

Produktion bezeichnet die „Kombination von materiellen und immateriellen Gütern zur Herstellung […] anderer Güter“ (Dichtl & Issing 1987). Für die Durchführung dieser Kombinationsprozesse kommen von der Charakteristika der Produktion abhängige Produktionssysteme zum Einsatz (Schuh 2007). Die für die Produktionsplanung und -steuerung wesentlichen Elemente von Produktionssystemen sind die Maschinen, die im Folgenden inklusive der benötigten Peripherie für Zuführung und Weitertransport von Werkstücken einheitlich als Station bezeichnet werden, sowie das Personal..

In der industriellen Produktion wird zwischen Teilefertigung und Montage unterschieden (Eversheim 1989). Teilefertigung umfasst urformende, umformende und trennende Fertigungsverfahren, während die Montage fügende Fertigungsverfahren beinhaltet (Eversheim 1989). Historisch wurde diese Unterscheidung jedoch nicht getroffen, weshalb in der Literatur auch ohne expliziten Bezug zur Teilefertigung oft von Fertigung und Fertigungssystem gesprochen wird (Winter 2014). Nachfolgend werden die Begriffe Fertigungsarten und Fertigungskonzepte als allgemeingültig für die gesamte Produktion verwendet.

2.1.1 Fertigungsarten

Die Fertigungsart stellt ein wesentliches Merkmal der Produktion dar. Schomburg (1980) klassifiziert Fertigungsarten nach den Kriterien Auflagenhöhe (Losgröße) und Wiederholhäufigkeit (Anzahl produzierter Lose pro Jahr) der Produkte. Anhand dieser

Kriterien werden die in Abbildung 2-1 abgebildeten Fertigungsarten Einzel-, Kleinserien-, Großserien- und Massenfertigung unterschieden. Eine eindeutige Abgrenzung der Fertigungsarten existiert nicht, der Übergang ist fließend (Eversheim 1996).

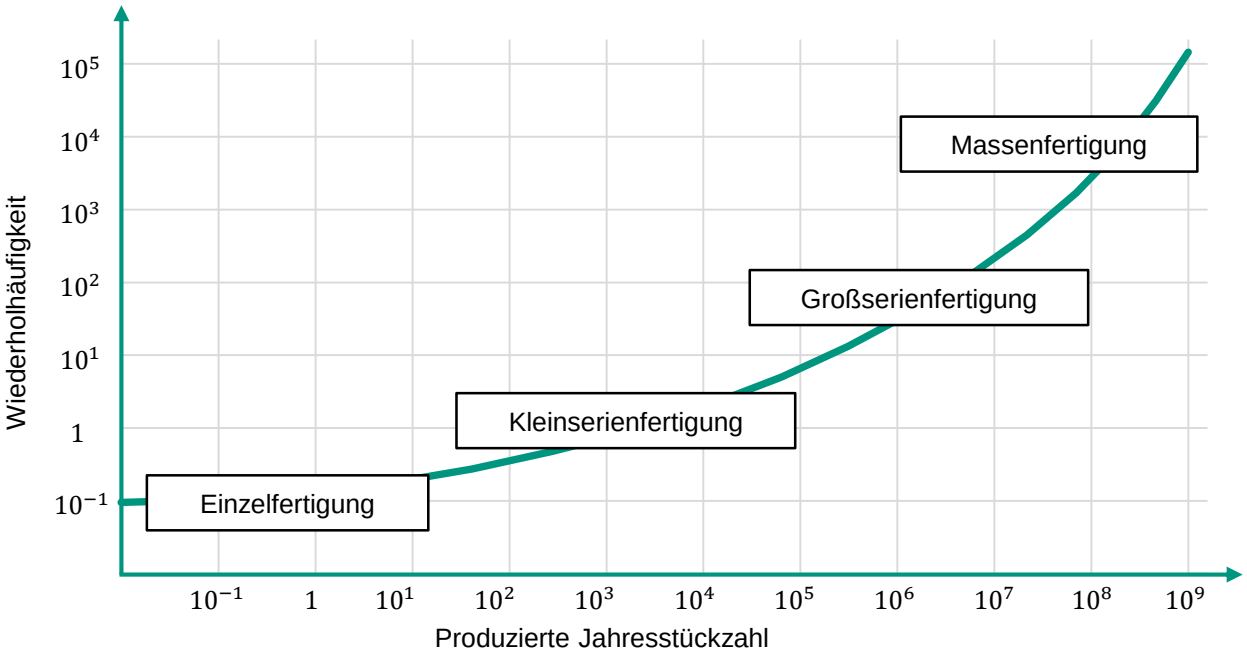


Abbildung 2-1: Fertigungsarten in Abhängigkeit von produzierter Jahresstückzahl und Wiederholhäufigkeit i. A. an (Eversheim 1996)

In der **Einzelfertigung** besteht i. d. R. ein direkter Bezug zu einem Kundenauftrag. Dieser wird einmalig ausgelöst und umfasst ein bis wenige Produkte. Die Betriebsmittel müssen dafür sehr flexibel sein. Der Anteil nicht-wertschöpfender Zeit ist durch Warte- und Rüstzeiten sehr groß. Dies führt zu einer geringen Auslastung. Typische Einzelfertiger finden sich z. B. im Bau von Großanlagen. (Eversheim 1996)

Die **Kleinserienfertigung** kommt meist für variantenreiche Produkte zum Einsatz. Es muss nicht unbedingt ein Bezug zu einem Kundenauftrag bestehen. Zum Teil wird ein Kundenentkopplungspunkt zur Verringerung der Durchlaufzeiten festgelegt. Die Produktion wird ein bis wenige Male im Jahr mit geringer Stückzahl angestoßen. Viele Hersteller von Werkzeugmaschinen sind typische Kleinserienfertiger. (Eversheim 1996)

Der **Großserien- und Massenfertigung** liegt meist ein variantenarmes Produkt zugrunde, das in großer Stückzahl produziert wird. Ein Bezug zu einem Kundenauftrag besteht nicht. Die Produktion findet anhand eines prognosebasierten Produktionsprogramms statt. Die Betriebsmittel sind meist auf die Anforderungen der Produkte spezi-

alisiert und für eine hohe und gleichmäßige Auslastung optimiert. Typische Großserienfertiger sind in der Automobilindustrie zu finden, während Massenfertiger z. B. in der Verbrauchsgüterindustrie angesiedelt sind. (Eversheim 1996)

2.1.2 Fertigungskonzepte

Das Fertigungskonzept beschreibt die physische Ausprägung des Produktionssystems. Es beinhaltet die räumliche Anordnung der Stationen, die Verteilung der Produktionsinhalte auf die Stationen, ein Materialflusskonzept für intralogistische Prozesse sowie ein Gestaltungskonzept der Stationen (Eversheim 1996; Bochmann 2018).

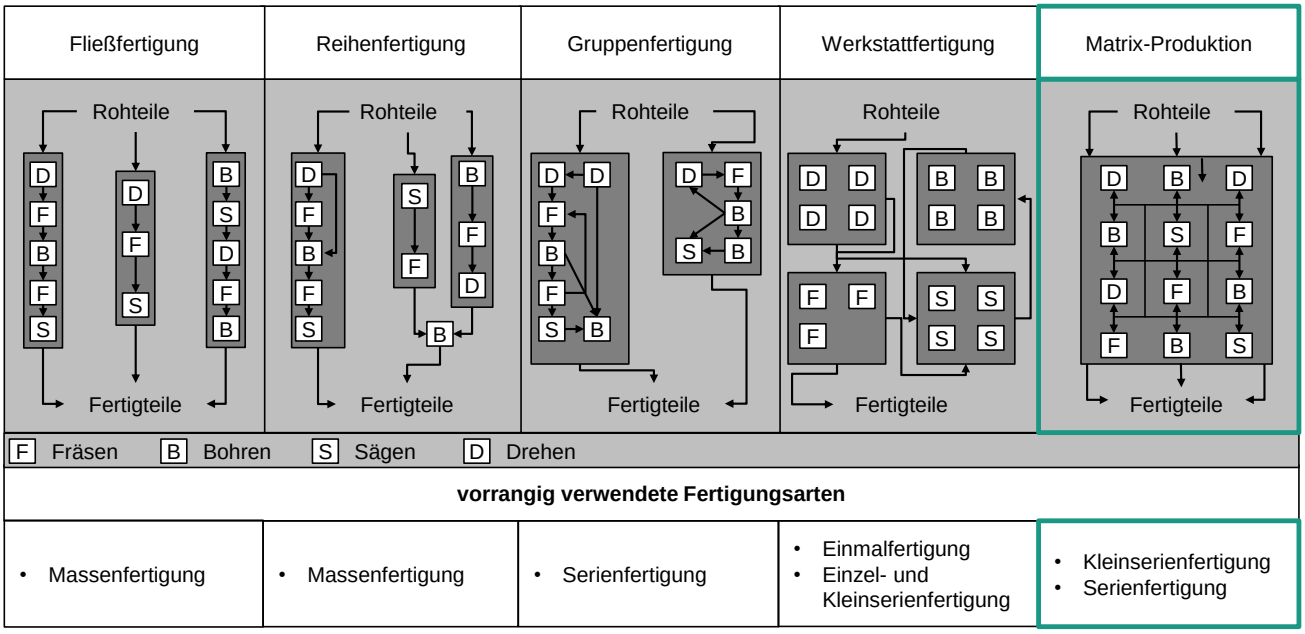


Abbildung 2-2: Fertigungskonzepte und vorrangig verwendete Fertigungsarten i. A. an (Eversheim & Schuh 1999)

Die Wahl des Fertigungskonzepts hängt vor allem von der Fertigungsart ab. In der Praxis haben sich die in Abbildung 2-2 dargestellten Fertigungskonzepte Fließ-, Reihen-, Gruppen- und Werkstattfertigung etabliert, die im Folgenden beschrieben werden. Die Matrix-Produktion wird ergänzend eingeführt.

Bei der **Fließfertigung** sind Betriebsmittel dem Materialfluss der zu fertigenden Produkte folgend angeordnet (Wannenwetsch 2014). Die Produkte werden meist im Einzelstückfluss zwischen den Stationen weitertransportiert. Der zeitliche Umfang der Bearbeitungsschritte ist gering. Um einen gleichmäßigen Materialtransport zu erreichen, sind die einzelnen Bearbeitungsstationen starr, z. B. durch Förderbänder, verkettet und durch einen Takt synchronisiert. Der Takt beschreibt die Zeit, die zur Ausführung eines

Arbeitsschrittes zur Verfügung steht (Eversheim & Schuh 1999; Weber & Kabst et al. 2018). In der Fließfertigung wird dieser durch eine Austaktung der Stationen ermittelt. Dadurch können Zwischenlager minimiert und Kapitalbindungskosten verringert werden. Zudem ergeben sich dadurch geringe Durchlaufzeiten (Wannenwetsch 2014). Damit eignet sich die Fließfertigung für die Großserien- und Massenproduktion (Bochmann 2018). Allerdings können in Fließfertigungen meist nur wenige Produktvarianten wirtschaftlich gefertigt werden. Flexibilität hinsichtlich der Bearbeitungsreihenfolge ist zudem nur durch Überspringen von Stationen gegeben. Insbesondere Störungen können sich bei Fließfertigungen schwerwiegend auswirken. Aufgrund der geringen Pufferbestände kann es außerdem zu Leerlauf bzw. Blockaden kommen (Lödding 2016).

Die **Reihenfertigung** ist durch einen gerichteten Materialfluss ohne Taktzwang sowie eine objektbezogene Zusammenfassung von Betriebsmitteln nach der Bearbeitungsreihenfolge einer Teilegruppe charakterisiert. Einzelne Fertigungsschritte können übersprungen werden, um Ablaufvarianten der Bearbeitungsfolge realisieren zu können. Dabei entsteht kein Effizienzverlust durch Leertakte (Schuh 2007; Wannenwetsch 2014; Weber & Kabst et al. 2018), es werden allerdings zusätzliche Zwischenpuffer benötigt. Die Vorteile liegen u. a. in den damit zu erzielenden relativ kurzen Durchlaufzeiten sowie einem geringen Logistikaufwand (Wannenwetsch 2014). Im Vergleich mit einer Werkstatt- oder Gruppenfertigung fällt die Reihenfolgeflexibilität jedoch wesentlich geringer aus (Bochmann 2018). Die Reihenfertigung kommt oft in variantenreichen Vormontagen, z. B. im Automobilbau, zum Einsatz.

Die **Gruppenfertigung** folgt dem Gruppenprinzip, das eine Mischform aus Verrichtungs- und Objektprinzip darstellt (Weber & Kabst et al. 2018). Ziel des Gruppenprinzips ist es, die Flexibilität hinsichtlich verschiedener Fertigungsaufgaben (Verrichtungsprinzip) und der flussorientierten Materialbewegung (Objektprinzip) zu verbinden. Dazu werden innerhalb von Funktionsgruppen Betriebsmittel zusammengefasst und wie in der Fließfertigung räumlich der Reihenfolge der durchzuführenden Bearbeitungsschritte folgend angeordnet. Die Funktionsgruppen sind so zu bilden, dass sie Teilefamilien bearbeiten können (Schuh 2007). Innerhalb der Funktionsgruppen kann der Materialfluss ungerichtet sein, um eine Reihenfolgeflexibilität zu ermöglichen. Die Gruppenfertigung erlaubt eine höhere Varianten-, Reihenfolge- und Volumenflexibilität als die Fließfertigung, liegt jedoch unterhalb der Werkstattfertigung. Volumenflexibilität wird z. B. über eine Anpassung der Mitarbeiteranzahl innerhalb einer Funktionsgruppe erreicht (Wannenwetsch 2014). Die Gruppenfertigung ist oft dann geeignet, wenn sich

Produkte aus Baugruppen zusammensetzen (Weber & Kabst et al. 2018). Sie kommt daher z. B. in der Getriebefertigung zum Einsatz (Bochmann 2018).

Die **Werkstattfertigung** folgt dem Verrichtungsprinzip, nach dem gleichartige Betriebsmittel räumlich zu Werkstätten, z. B. Dreherei, Fräserei zusammengefasst werden (Wannenwetsch 2014). Der Materialfluss zwischen den Werkstätten ist ungerichtet und abhängig von der jeweiligen Produktvariante, wobei oft ein losweiser Transport realisiert wird (Eversheim 1996). Aufgrund der räumlichen Anordnung von Betriebsmitteln ist die Werkstattfertigung flexibel hinsichtlich Produktionsvolumen und Produktvarianten mit unterschiedlichen Bearbeitungsreihenfolgen bei vergleichsweise geringen Investitionskosten (Bochmann 2018). Nachteilig sind lange Durchlaufzeiten und der Aufbau von Zwischenpuffern bei losweiser Fertigung (Wiendahl 2009). Die dadurch bedingten Liegezeiten führen zu höheren Durchlaufzeiten und Kapitalbindungskosten. Durch den ungerichteten Materialfluss erhöhen sich zudem Materialflusskomplexität und logistischer Aufwand (Wannenwetsch 2014). Werkstattfertigungen bieten sich insbesondere für die Einzel- und Kleinserienfertigung an, in denen hohe Flexibilität gefordert ist (Weber & Kabst et al. 2018). Sie sind daher oft z. B. im Prototypenbau sowie bei Sonderanfertigungen realisiert.

Die **Matrix-Produktion** stellt eine Kombination von Werkstatt- und Fließfertigung dar. Dabei wird versucht, durch das Layout einen möglichst linearen Fluss von häufig herzustellenden Produkten zu realisieren. Die Stationen sind ähnlich wie in einer Matrix räumlich in Zeilen und Spalten angeordnet, sodass sehr flexible Materialflüsse realisierbar sind (Greschke 2016). Wie in einer Werkstattfertigung existieren in der Matrix-Produktion redundante Stationen, die aber nicht zwingend nach dem Verrichtungsprinzip gruppiert sind. Zudem handelt es sich in der Regel um Mehrzweckstationen, die eine erhöhte Reihenfolgeflexibilität erlauben (Bochmann 2018; Schönemann & Herrmann et al. 2015). Der Materialfluss wird z. B. mithilfe von FTF lose verkettet. Die dadurch mögliche Entkopplung vom Takt vermeidet Produktivitätsverluste bei heterogenen Bearbeitungszeiten. Rückflüsse von Aufträgen an bereits verwendete Stationen sind ohne zusätzlichen Aufwand realisierbar. Wie bei der Werkstattfertigung ist die Durchlaufzeit von Produkten wesentlich durch die Transportstrecken zwischen Stationen und den Liegezeiten an Stationen beeinflusst; ebenfalls ist der Flächenbedarf für die Logistik aufgrund der Vielzahl möglicher Transportwege höher als in der Fließfertigung. Die Durchlaufzeit liegt aufgrund des flussorientierten Layouts unterhalb der Werkstattfertigung bei glei-

cher Flexibilität. Die PPS ist aufgrund der hohen Freiheitsgrade durch individuelle Materialflüsse je Produkt aufwändiger. Die Matrix-Produktion bietet die größten Freiheitsgrade hinsichtlich der Umplanbarkeit von Aufträgen und wird in dieser Arbeit fokussiert betrachtet.

2.2 Produktionsplanung und -steuerung (PPS)

Die Produktionsplanung und -steuerung (PPS) stellt eine wesentliche Aufgabe in Industrieunternehmen dar (Günther & Tempelmeier 2005; Schuh 2007). Im Folgenden werden in Kapitel 2.2.1 die Aufgaben und Ziele der PPS erläutert, bevor die dynamische PPS in Kapitel 2.2.2 eingeführt wird. Anschließend wird in Kapitel 2.2.3 spezieller für das Scheduling als Teil der PPS die gängige und im Rahmen der Arbeit verwendete Notation eingeführt, bevor abschließend in Kapitel 2.2.4 die wesentlichen Lösungsverfahren für das Scheduling-Problem knapp erläutert werden.

2.2.1 Aufgaben und Ziele der PPS

Die Produktionsplanung hat die Aufgabe, eine programm-, termin-, kapazitäts- und mengenbezogene Planung der zu realisierenden Produktionsprozesse darzustellen (Eversheim 2002). Die Produktionssteuerung regelt demgegenüber, wann welche Prozesse die Produktionsfaktoren (Stationen und Personal) beanspruchen (Schomburg 1980; Schuh 2007). Dabei sind die Vorgaben der Produktionsplanung sowie die logistischen Zielgrößen zu berücksichtigen. Die Produktionssteuerung ist insbesondere bei kurzfristig geänderten Auftragsmengen und -terminen (z. B. Auftragsänderungen, Eilaufträge), Störungen (z. B. Stationsausfälle), Lieferverzögerungen von Rohmaterial oder Komponenten oder entstandenem Ausschuss erforderlich, um die Planung trotz dieser Ereignisse bestmöglich zu realisieren (Wiendahl 1997).

Der Übergang zwischen Planung und Steuerung ist nicht eindeutig definiert, sondern „vollzieht sich [...] an der Stelle, an der Planvorgaben in Durchsetzungsaktivitäten übergehen“ (Corsten & Gössinger 2012). Orientiert am Zeithorizont der Aufgaben unterteilt Bischoff (1999) die PPS in drei Ebenen:

- Die **strategische Planungsebene** erstreckt sich über mehrere Jahre und legt u.a. das zu produzierende Produktspektrum fest, das mit den Mengenabschätzungen die Basis für die Planung der benötigten Kapazitäten bildet.
- Die **taktische Planungsebene** beinhaltet die Konkretisierung der Produktionsstrategie und erstreckt sich über mehrere Monate bis zu einem Jahr. Gegenstand

der mittelfristigen Produktionsplanung sind u. a. die Planung des Produktprogramms sowie die Sicherstellung der grundsätzlich benötigten Kapazitäten

- Die **operative Planungs- und Steuerungsebene** hat die Aufgabe, die vorhandenen Ressourcen kurzfristig mit einem Zeithorizont von wenigen Tagen bis Wochen effizient zur Erreichung der festgelegten Ziele einzusetzen. Dies beinhaltet das Scheduling des Produktionsprogramms auf den Betriebsmitteln unter Berücksichtigung der vorhandenen Kapazitäten und logistischen Ziele.

Eine Gestaltung des Produktionssystems wird in der vorliegenden Arbeit nicht vorgenommen. Es erfolgt eine Fokussierung auf die operative Planungs- und Steuerungsebene. Unabhängig von der Ebene verfolgt die PPS vier wesentliche Zielsetzungen: eine hohe und möglichst gleichmäßige Auslastung der Produktionskapazitäten, kurze Durchlaufzeiten, eine hohe Termintreue sowie geringe Bestände in der Produktion (Wiendahl 1997). Der Zielkonflikt, in dem die Größen zueinander stehen ist auch als Dilemma (Gutenberg 1955) bzw. Polylemma der Ablaufplanung (Hackstein 1989) bekannt. So können geringe Bestände zu einer Unterauslastung vorhandener Produktionskapazitäten führen, während eine hohe Kapazitätsauslastung zum Aufbau hoher Bestände führen kann, welche wiederum zu längeren Durchlaufzeiten führen. Während in der Vergangenheit meist eine hohe Auslastung der kapitalintensiven Produktionsanlagen das oberste Ziel war, ist dies heute zugunsten kurzer Durchlaufzeiten, hoher Termintreue und niedriger Bestände eher in den Hintergrund gerückt (Kurbel 2003).

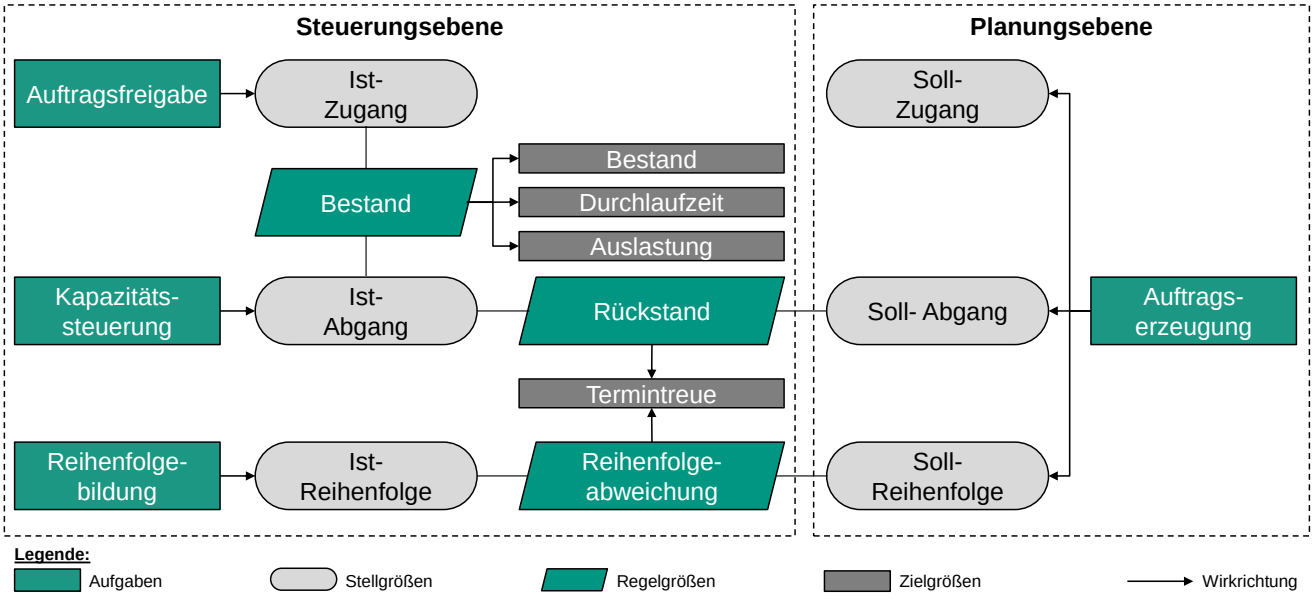


Abbildung 2-3: Modell der Fertigungssteuerung nach Lödding (2016)

Die Aufgaben der operativen Planungs- und Steuerungsebene finden sich auch im Modell der Fertigungssteuerung nach Lödding (2016) wieder (siehe Abbildung 2-3). Hierbei wird zwischen der Planungs- und der Steuerungsebene unterschieden, mit deren Aufgaben Soll-Größen vorgegeben bzw. Ist-Größen beeinflusst werden. Diese Größen wiederum ergeben Regelgrößen, die direkte Auswirkungen auf die logistischen Zielgrößen Bestand, Durchlaufzeit, Auslastung und Termintreue haben.

Die **Planungsebene** beinhaltet die **Auftragserzeugung**, in der Soll-Zugang, Soll-Abgang und Soll-Reihenfolge der Aufträge festgelegt werden. Ergebnis der Planungsebene ist ein Stationsbelegungsplan (im Folgenden: Schedule) der angibt, welcher Auftrag zu welcher Zeit an welcher Station eingeplant ist. Über den ersten bzw. letzten Bearbeitungsschritt je Auftrag wird der Soll-Zugang bzw. Soll-Abgang determiniert. Außerdem wird über die stationsindividuelle zeitliche Abfolge von Bearbeitungsschritten die Soll-Reihenfolge festgelegt.

Ein Schedule kann verzögerungsfrei, aktiv, oder semi-aktiv sein:

- In einem **verzögerungsfreien Schedule** ist keine Station unbeschäftigt, wenn Aufträge an ihr warten.
- Ein **Schedule** ist **aktiv**, wenn kein anderer Schedule gefunden werden kann, bei dem mindestens ein Bearbeitungsschritt früher beendet werden kann, ohne dass ein anderer Bearbeitungsschritt verzögert wird.
- Ein **Schedule** ist **semi-aktiv**, wenn kein Bearbeitungsschritt früher beendet werden kann, ohne die Bearbeitungsreihenfolge an einer Station zu verändern.

Verzögerungsfreie Schedules sind eine Teilmenge aktiver Schedules, die wiederum einer Teilmenge semi-aktiver Schedules sind. Für typische Zielfunktionen liegt der optimale Schedule in der Menge aktiver Schedules (Pinedo 2016). Für die zu erarbeitende Methode bedeutet dies, dass ein bewusstes Warten von Stationen trotz Aufträgen im Puffer sowohl für das Scheduling als auch das Rescheduling zur Erlangung einer hohen Lösungsgüte möglich sein soll.

Die **Steuerungsebene** beinhaltet die **Auftragsfreigabe**, die **Kapazitätssteuerung** sowie die **Reihenfolgebildung**, mit denen Ist-Zugang, Ist-Abgang und Ist-Reihenfolge der Aufträge festgelegt werden. Die Unterschiede zwischen den Soll- und Ist-Werten ergeben sich durch im Betrieb auftretende Abweichungen wie z. B. Stationsausfälle oder fehlende Mitarbeiter oder Material. Die Auftragsfreigabe legt den Zeitpunkt fest,

zu dem Aufträge freigegeben werden und bestimmt somit den Ist-Zugang zur Produktion. Die Kapazitätssteuerung legt die Höhe der Kapazitäten in der Produktion fest, indem z. B. Arbeitszeiten und Mitarbeiteranzahl angepasst werden. Dadurch wird der Ist-Abgang der Aufträge beeinflusst. Die Reihenfolgebildung schließlich determiniert die Sequenz, in der die Aufträge abgearbeitet werden und bestimmt damit die Ist-Reihenfolge. Die Differenz zwischen Ist-Zugang und Ist-Abgang stellt den Bestand dar, der sich auf die logistischen Zielgrößen Bestand, Durchlaufzeit und Auslastung auswirkt. Der Rückstand und die Reihenfolgeabweichung als Differenz zwischen Ist- und Soll-Abgang bzw. Ist- und Soll-Reihenfolge wirken auf die logistische Zielgröße Termintreue. In der Steuerungsebene wird ein zugrundeliegender Schedule aus der Planungsebene durch Rescheduling umgeplant oder ein neuer Schedule mit geringem Planungshorizont aufgebaut.

Im Rahmen der vorliegenden Arbeit werden die Aufgaben des Modells der Fertigungssteuerung mit Ausnahme der Kapazitätssteuerung betrachtet. Die Soll-Werte werden durch das Scheduling festgelegt, während die Ist-Werte durch das Rescheduling beeinflusst werden.

2.2.2 Dynamische PPS

Produktionssysteme unterliegen einer durch externe und interne Einflussfaktoren beeinflussten Dynamik, in der ein ständiger Abgleich von Soll- und Ist-Größen erforderlich ist und Anpassungsmaßnahmen kontinuierlich ergriffen werden müssen. Viele bestehende Ansätze aus der Literatur nehmen eine deterministische Umwelt an und sind für die praktische Anwendung daher ungeeignet (Suresh & Chaudhuri 1993; Ouelhadj & Petrovic 2009). Dynamische Ereignisse wie Störungen führen dort oft zu einer geringen Liefertreue (Gören & Sabuncuoglu 2008).

Die dynamische PPS hingegen umfasst diejenigen Ansätze, die dynamische Ereignisse berücksichtigen (Ouelhadj & Petrovic 2009; Al-Hinai & ElMekkawy 2011). Dynamische Ereignisse werden im Folgenden auch als Störung bezeichnet, da sie stets eine Abweichung von Ist- und Soll-Größen nach sich ziehen (Ouelhadj & Petrovic 2009). Störungen lassen sich in ressourcen- und auftragsbezogene Störungen unterteilen (Suresh & Chaudhuri 1993; Cowling & Johansson 2002; Vieira & Herrmann et al. 2003). Ressourcenbezogene Störungen sind z. B. Stationsstörungen, die Nichtverfügbarkeit von Personal oder fehlendes bzw. fehlerhaftes Material. Auftragsbezogene Störungen sind z. B. Eilaufträge, Auftragsstornierungen, geänderte Auftragsprioritäten oder geänderte

Fälligkeitstermine. Die meisten Störungen können als Stationsstörungen aufgefasst werden, da sie zu einer Verlängerung der benötigten Zeit führen und daher durch eine äquivalente Erhöhung der Prozesszeit abgebildet werden können (Abumaizar & Svestka 1997).

Eine Störung an einer Station in einem Produktionssystem wird anhand einer Verteilungsfunktion mit Mittelwerten für die mittlere Reparaturdauer ($MTTR_s$, Mean Time To Repair) und die mittlere Zeit zwischen zwei Störungen ($MTBF_s$, Mean Time Between Failures) charakterisiert. $MTTR_s$ und $MTBF_s$ ergeben sich als Quotient der Gesamtbetriebszeit bzw. Gesamtstörungsdauer und der Anzahl der Störungen im betrachteten Zeitraum. Mit den beiden Größen lässt sich die Verfügbarkeit v_s einer Station s berechnen, die den zur Produktion zur Verfügung stehenden Zeitanteil an der Gesamtzeit darstellt. Sie wird nach Formel 2-1 ermittelt. (Eberlin & Hock 2014).

$$v_s = \frac{MTBF_s}{MTBF_s + MTTR_s} \qquad 2\text{-}1$$

Des Weiteren lassen sich Störungen anhand ihrer Bestimmtheit in **deterministische** und **stochastische Störungen** unterscheiden (van Brackel 2008). Eine deterministische Störung umfasst z. B. die geplante Wartung von Betriebsmitteln (Patig & Thorhauer 2002), während stochastische Störungen z. B. Stationsstörungen oder Anpassungen von Lieferterminen umfassen.

Deterministische Störungen können in der dynamischen PPS mithilfe zusätzlicher Nebenbedingungen berücksichtigt werden (van Brackel 2008). In dieser Arbeit werden daher nur stochastische Störungen berücksichtigt, die in Lösungsverfahren nur approximativ abgebildet werden können. Der Prozess der Anpassung eines Schedules auf Störungen wird als Rescheduling bezeichnet (Bean & Birge et al. 1991; Vieira & Herrmann et al. 2003; Larsen & Pranzo 2018).

Zur Klassifikation von Ansätzen für das Scheduling und Rescheduling sowie deren Eigenschaften wird im Folgenden das in Abbildung 2-4 dargestellte Framework verwendet. Dynamische Einflüsse legen den Grad der berücksichtigten Unsicherheit fest. Der Problemtyp gibt an, ob hauptsächlich neue Schedules erzeugt oder bestehende Schedules modifiziert werden sollen. Die Strategien geben vor, mit welcher grundsätzlichen Herangehensweise der Dynamik Rechnung getragen wird. Politik und Methode beziehen sich auf spezifische Unterscheidungsmerkmale hinsichtlich der Ansätze für

das Rescheduling. Es lässt sich sagen, dass Ansätze ohne Berücksichtigung dynamischer Einflüsse stärkeren Vorausplanungscharakter besitzen, sich aber weniger reaktionsfähig verhalten. (A_Zöllner 2019; Vieira & Herrmann et al. 2003; Neuhaus 2008; Ouelhadj & Petrovic 2009; Larsen & Pranzo 2018).

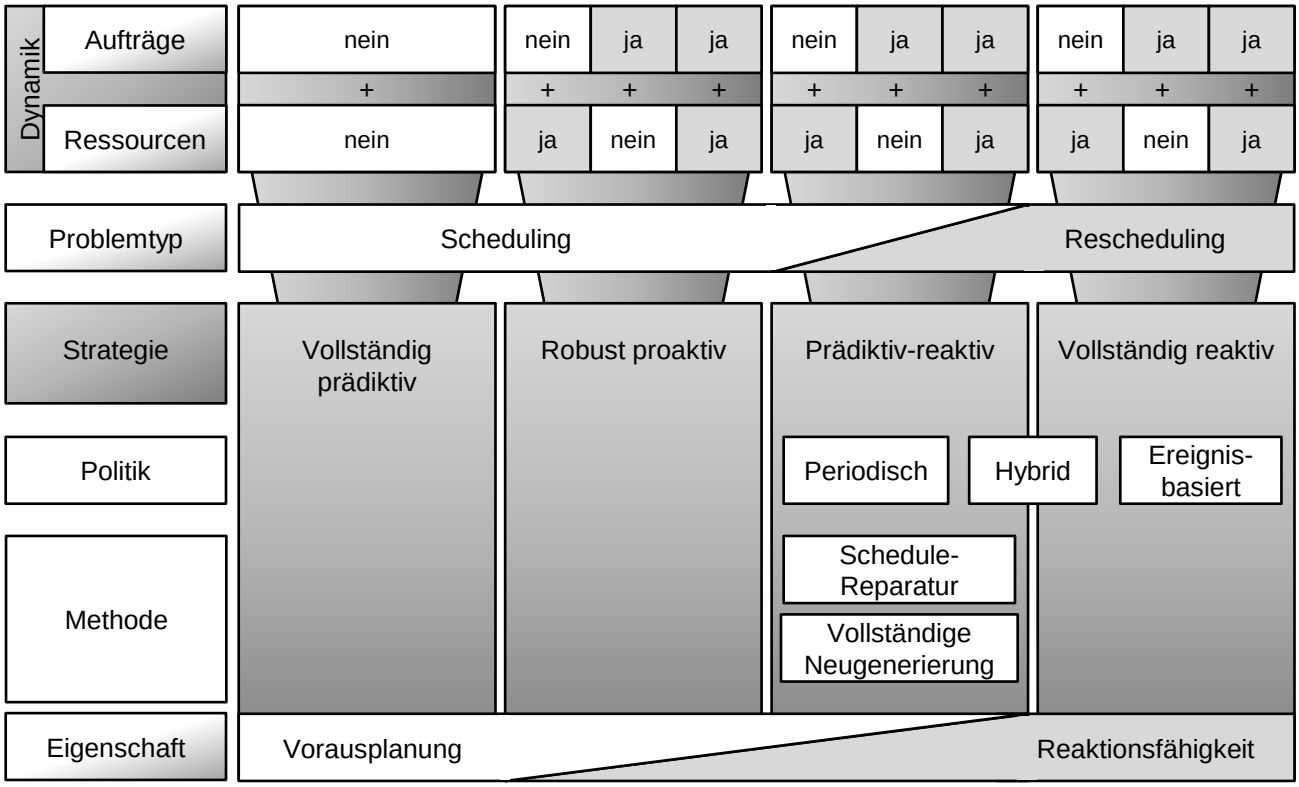


Abbildung 2-4: Framework zur Klassifizierung von Scheduling- und Rescheduling-Ansätzen i. A. an (A_Zöllner 2019; Vieira & Herrmann et al. 2003; Neuhaus 2008; Ouelhadj & Petrovic 2009; Larsen & Pranzo 2018)

Ansätze des Scheduling bzw. Rescheduling können anhand der **Berücksichtigung von auftrags- und ressourcenbezogener Dynamik** unterschieden werden. Je nach Art der berücksichtigten dynamischen Störeinflüsse und dem Problemtyp bieten sich verschiedene Strategien für das Scheduling bzw. Rescheduling an. In der Literatur wird hierbei zwischen vollständig prädiktiven, robust proaktiven, prädiktiv-reaktiven und vollständig reaktiven Verfahren differenziert (Mehta & Uzsoy 1999; Vieira & Herrmann et al. 2003; Herroelen & Leus 2004; Aytug & Lawley et al. 2005).

Eine weitere Unterscheidungsebene ergibt sich hinsichtlich der **Scheduling- bzw. Rescheduling-Strategien**:

Vollständig prädiktive Verfahren beinhalten die Erzeugung von Schedules vor Produktionsbeginn als Vorgabe für die Ablaufsteuerung. Ergebnis ist ein Schedule, der

eine deterministische Umwelt annimmt und Unsicherheiten durch Störeinflüsse vernachlässigt (Herroelen & Leus 2004).

Robust proaktive Verfahren erlauben die Erzeugung von Schedules, deren angestrebte logistische Zielerreichung trotz Dynamik erfüllbar sein soll und die damit robust gegenüber Störungen sind (Ouelhadj & Petrovic 2009). Eine wesentliche Herausforderung dieser Ansätze stellt die Bewertung der Robustheit dar (Mehta & Uzsoy 1999; Davenport & Gefflot et al. 2001). Je mehr Informationen über Störungseinflüsse vorhanden sind, desto besser kann ein robust proaktives Lösungsverfahren deren Auswirkungen antizipieren und beim Scheduling berücksichtigen (Neuhaus 2008; Claus & Herrmann et al. 2015).

Prädiktiv-reaktive Verfahren kombinieren die prädiktive Generierung Schedules mit Verfahren des Rescheduling zur Anpassung des Schedules unter dynamischen Einflüssen (Abumaizar & Svestka 1997; Sabuncuoglu & Karabuk 1999). Es bietet sich die Berücksichtigung von Robustheit in der prädiktiven Planung für die Erstellung eines robusten Schedules an, um Störungen kompensieren zu können (Ouelhadj & Petrovic 2009). Für das reaktive Rescheduling wird der jeweils vorliegende Systemzustand berücksichtigt. Dieser beinhaltet Informationen über Ort und Dauer von Störungen (Vieira & Herrmann et al. 2003). Im Rahmen dieser Arbeit wird der Systemzustand zusätzlich noch über den aktuellen Ort bereits in der Produktion befindlicher Aufträge ergänzt, also die Information, wo die Aufträge bearbeitet werden, im Puffer liegen oder zu welcher Station die Aufträge aktuell auf einem FTF unterwegs sind. Außerdem wird auf Stationen, die aktuell keinen Auftrag bearbeiten, der aktuelle Rüstzustand ausgegeben, um reihenfolgeabhängige Rüstzeiten adäquat einplanen zu können.

Bei **vollständig reaktiven Verfahren** werden keine Schedules vor Produktionsbeginn erzeugt. Entscheidungen über die Stationszuweisungen werden dynamisch getroffen (Vieira & Herrmann et al. 2003; Ouelhadj & Petrovic 2009).

Ein großer Teil der Literatur zum dynamischen Scheduling befasst sich mit Ansätzen für das prädiktiv-reaktive Scheduling. Prädiktive Ansätze berücksichtigen die Dynamik in Produktionssystemen nicht. Robust-proaktive Ansätze verringern die Leistungsfähigkeit oft so stark, dass eine Anwendung fraglich ist. Reaktive Ansätze kommen nicht zum Einsatz, wenn die Liefertreue optimiert werden soll, da diese bei dynamischem Scheduling nicht vorherbestimmbar ist (Schönemann & Herrmann et al. 2015). Die vorliegende Arbeit beschränkt sich daher auf prädiktiv-reaktive Verfahren.

Ferner kann **innerhalb des reaktiven Reschedulings** eine Untergliederung in **Rescheduling-Politik** und **Rescheduling-Methode** vorgenommen werden (Vieira & Herrmann et al. 2003).

Die **Rescheduling-Politik** legt fest, zu welchen Zeitpunkten bzw. nach welcher logischen Bedingung ein Rescheduling durchgeführt wird (Vieira & Herrmann et al. 2003). Dabei unterscheiden Vieira & Herrmann et al. (2003) die folgenden Ausprägungen:

- **Periodische Politiken** überprüfen einen Schedule in festen Zeitabständen und ermöglichen ein rollierendes Rescheduling. Sie sind geeignet, wenn eine Erfassung von Echtzeitdaten zur Bestimmung des Systemzustands nicht ohne hohen Aufwand möglich ist (Church & Uzsoy 1992). Nachteilig dabei ist die Kumulierung der Auswirkungen von Störungen bis zum Beginn einer neuen Rescheduling-Periode. Wesentliche Herausforderung ist die Ermittlung einer optimalen Rescheduling-Periode (Sabuncuoglu & Karabuk 1999; Ouelhadj & Petrovic 2009).
- **Ereignisbasierte Politiken** passen einen Schedule beim Auftreten von Ereignissen (z. B. ressourcen- oder auftragsbezogene Störungen) an (Ouelhadj & Petrovic 2009). Es kann dadurch ein Zustand des permanenten Reschedulings entstehen, welcher zu sehr hohem Rechenaufwand und organisatorischer Komplexität bei der Umsetzung führen kann (Vieira & Herrmann et al. 2003).
- **Hybride Politiken** stellen eine Mischform von periodischen und ereignisbasierten Politiken dar, indem sowohl in festen Zeitabständen als auch ereignisbasiertes Rescheduling durchgeführt wird. Letzteres betrifft z. B. größere Stationsausfälle oder Eilaufträge (Church & Uzsoy 1992).

Die **Rescheduling-Methoden** lassen sich in Schedule-Reparatur und vollständige Neugenerierung unterteilen (Sabuncuoglu & Bayız 2000; Cowling & Johansson 2002):

- Bei der **Schedule-Reparatur** wird ein zuvor erzeugter Schedule angepasst. Dies kann erfolgen, indem ein Right-Shift Rescheduling oder ein partielles Rescheduling durchgeführt wird (Sabuncuoglu & Bayız 2000). Beim Right-Shift Rescheduling werden Aufträge so lange auf der Zeitachse in die Zukunft, also nach rechts, verschoben, bis dieser zulässig ist. Beim partiellen Rescheduling wird nur der durch ein Ereignis betroffene Ausschnitt eines Schedules neu geplant, wodurch der ursprüngliche Schedule weitestgehend erhalten bleibt (Abumaizar & Svestka 1997). Ziel ist es dabei, möglichst schnell wieder zum ursprünglichen Schedule

zurückzukehren und so eine stabilere Produktion zu erhalten. Dieser Ansatz wird im Folgenden auch als **Match-Up Scheduling** bezeichnet.

- Bei der **vollständigen Neugenerierung** wird ein komplett neuer Schedule erzeugt, der alle verbleibenden Bearbeitungsschritte enthält. Der zuvor erzeugte Schedule wird somit verworfen. Für die vollständige Neugenerierung werden Verfahren des prädiktiven Scheduling verwendet. Es kann dabei zu Instabilität in der Produktion kommen, wenn der Schedule sich bei jedem Störereignis grundlegend ändert (Ouelhadj & Petrovic 2009).

In der vorliegenden Arbeit wird ein prädiktiv-reaktives Verfahren betrachtet, bei dem ereignisbasiert eine Schedule-Reparatur mittels Match-Up Scheduling durchgeführt wird. Ist dies nicht erfolgreich, wird eine vollständige Neugenerierung angestoßen.

2.2.3 Notation von Scheduling-Problemen

Für die Klassifikation von Scheduling-Problemen wird eine Notation $(\alpha|\beta|\gamma)$ zur Abbildung der relevanten Charakteristika verwendet (Graham & Lawler et al. 1979; Dempster & Lenstra et al. 1982; Herrmann & Lee et al. 1993). Dabei repräsentiert α die Stationskonfiguration, β die Auftragseigenschaften bzw. Prozesscharakteristika und γ die Zielfunktion (Pinedo 2016). Der Beschreibungsumfang dieser Parameter wurde u. a. durch Blazewicz & Ecker et al. (2007), Allahverdi & Ng et al. (2008) und Pinedo (2016) sukzessiv erweitert, um zusätzliche Fallunterscheidungen zu ermöglichen.

2.2.3.1 Stationskonfiguration α

Die Stationskonfiguration α beschreibt die generelle räumliche und funktionale Situation der Anlagen im Produktionssystem (Pinedo 2016; Bochmann 2018). Sie setzt sich aus dem Fertigungskonzept und der Anzahl vorhandener Stationen m bzw. Arbeitszentren c zusammen. Die wesentlichen Ausprägungen von α sind in Abbildung 2-5 zusammenfassend dargestellt.

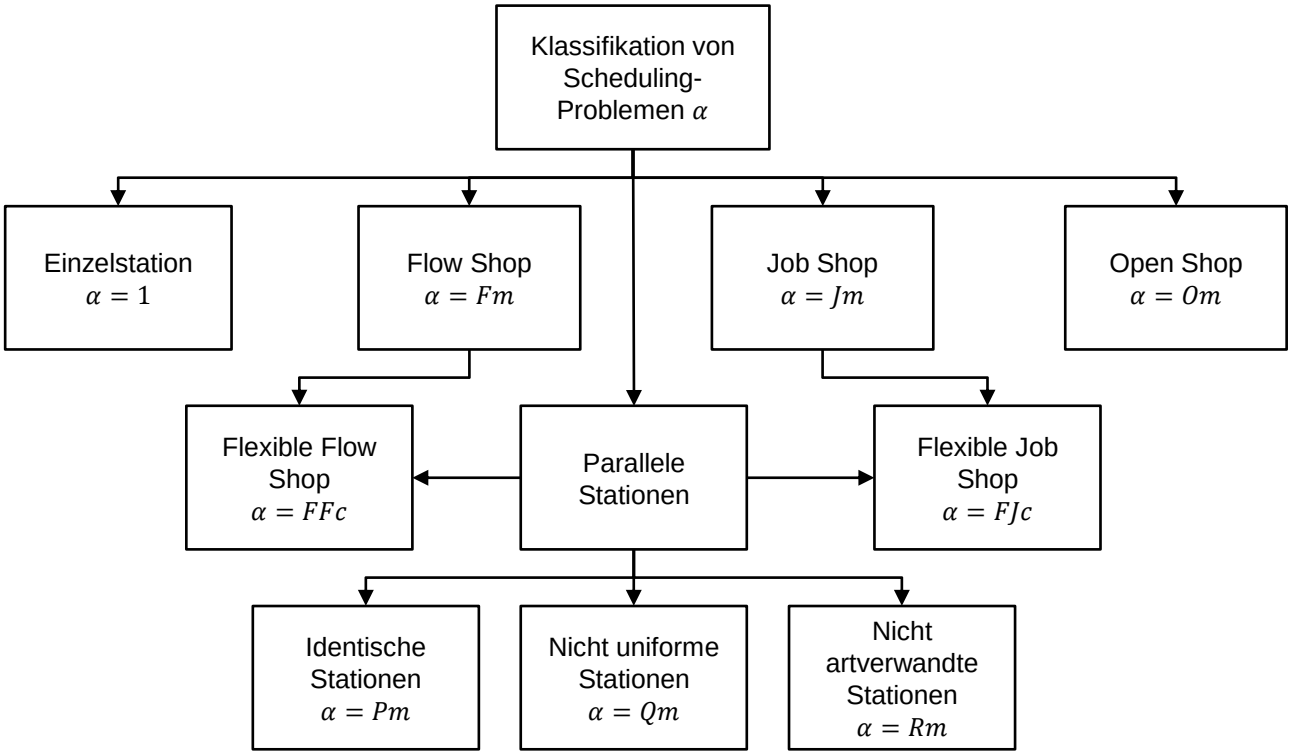


Abbildung 2-5: Klassifikation von Scheduling-Problemen anhand der Stationskonfiguration i. A. an (Pinedo 2016)

Das **Einstationen-Problem** ($\alpha = 1$) beschreibt das einfachste Scheduling-Problem. Es existiert darin entweder nur eine zu beplanende Station oder es handelt sich um eine zu beplanende Engpassstation in einer Mehrstationenumgebung.

Beim **Flow Shop Problem** ($\alpha = Fm$) sind m Stationen seriell angeordnet. Alle Aufträge durchlaufen dieselben Stationen und werden an jeder vorhandenen Station bearbeitet (Bochmann 2018). Sobald ein Auftrag an einer Station fertiggestellt wird, reiht er sich in die Warteschlange für die nächste Station ein (Pinedo 2016). Damit kann das Scheduling der in Kapitel 2.1.2 vorgestellten Fließfertigung formuliert werden.

Beim **Job Shop Problem** ($\alpha = Jm$) durchläuft jeder Auftrag das Produktionssystem entlang einer vordefinierten festen Bearbeitungsreihenfolge. Diese kann sich von Auftrag zu Auftrag unterscheiden. Das Job Shop Problem eignet sich somit zum Scheduling der in Kapitel 2.1.2 vorgestellten Werkstattfertigung, wenn für jeden Bearbeitungsschritt jeweils nur eine Station verfügbar ist (Manne 1960).

Open Shop Probleme ($\alpha = Om$) stellen einen verallgemeinerten Fall des Job Shop Problems dar, bei dem keine Reihenfolge der Bearbeitungsschritte je Auftrag vorgegeben ist. Dadurch ist der Lösungsraum im Vergleich zum Job Shop Problem deutlich vergrößert. (Bochmann 2018)

Im Scheduling-Problem **paralleler Stationen** existieren m parallele Stationen. Es handelt sich um eine einstufige Produktionsumgebung, in der jeder Auftrag genau einen Bearbeitungsschritt durchlaufen muss (Pinedo 2016). Im Fall **identischer Stationen** ($\alpha = Pm$) durchläuft ein Auftrag genau eine beliebige Station bis zu seiner Fertigstellung. Beim Einsatz **nicht uniformer Stationen** ($\alpha = Qm$) können unterschiedliche Geschwindigkeiten für Stationen definiert werden, um z. B. eine Differenzierung von Stationen unterschiedlicher Hersteller oder unterschiedlichen Alters zu ermöglichen. Eine weitere Differenzierung wird durch die Verwendung **nicht artverwandter Stationen** ($\alpha = Rm$) möglich. Dabei kann für jeden Auftrag je Station eine unterschiedliche Bearbeitungsgeschwindigkeit festgelegt werden.

Das **Flexible Flow Shop Problem** ($\alpha = FFc$) stellt eine Mischform aus parallelem Stations- und Flow Shop Problem dar. Es existieren c seriell angeordnete Arbeitszentren, die aus mehreren parallelen Stationen bestehen können. Jeder Auftrag durchläuft alle Arbeitszentren. Durch die Parallelität ergibt sich ein größerer Lösungsraum gegenüber dem Flow Shop Problem. Das Flexible Flow Shop Problem kann zum Scheduling der in Kapitel 2.1.2 vorgestellten Reihen- bzw. Gruppenfertigung verwendet werden.

Das **Flexible Job Shop Problem** ($\alpha = FJc$) stellt eine Verallgemeinerung des Job Shop Problems um parallele Stationen dar (Brucker & Schlie 1990). Jeder Auftrag besitzt eine individuelle und vordefinierte Bearbeitungsreihenfolge, für die c Arbeitszentren mit identischen und parallelen Stationen zur Verfügung stehen. Jeder Auftrag durchläuft die benötigten Arbeitszentren auf individuellem Weg und wird dort an einer der zur Verfügung stehenden Stationen bearbeitet. Es liegt eine Kombination aus Scheduling- und Routing-Problem vor, da Bearbeitungsschritte den Stationen zugeordnet (Routing) und die Reihenfolgebildung der Bearbeitungsschritte je Station (Scheduling) vorgenommen werden müssen (Brandimarte 1993; Shen & Han et al. 2017). Ferner unterscheidet die Literatur zwischen partiellen und total flexiblen Job Shop Problemen. Bei partiellen flexiblen Job Shop Problemen können einige Bearbeitungsschritte nur von einer Teilmenge der Stationen bearbeitet werden, während bei total flexiblen Job Shop Problemen jede Station jeden Bearbeitungsschritt durchführen kann (Chaudhry & Khan 2016).

Das partielle und das total flexible Job Shop Problem eignen sich somit zum Scheduling in der in Kapitel 2.1.2 vorgestellten Werkstattfertigung und der Matrix-Produktion und werden somit für den weiteren Verlauf der Arbeit gewählt.

2.2.3.2 Auftragseigenschaften bzw. Prozesscharakteristika β

Der Parameter β spezifiziert die Merkmale der Fertigung (Pinedo 2016). Blazewicz & Ecker et al. (2007) und Allahverdi & Ng et al. (2008) nennen u. a. folgende relevanten Ausprägungen, die im weiteren Verlauf der Arbeit allesamt berücksichtigt werden:

- Unterbrechbarkeit ($Prmp$): Die Bearbeitung von Aufträgen kann nach Beginn unterbrochen und zu einem späteren Zeitpunkt fortgesetzt werden. Solche Unterbrechungen werden im Folgenden als Preemption bezeichnet.
- Reihenfolgeabhängige Rüstzeiten ($s_{jj'}$): Dieser Parameter beschreibt Rüstzeiten zwischen der Bearbeitung zweier Aufträge j und j' unterschiedlichen Typs.
- Blockaden ($Block$): Bei Berücksichtigung von Puffern mit begrenzter Kapazität kann bei vollem Puffer zu Blockaden führen, die im Scheduling beachtet werden müssen.
- Rezirkulation ($Rcrc$): Dieser Parameter erlaubt es Aufträgen ohne Zwangsdurchlauf, dieselbe Station mehrfach zu besuchen.
- Stationsverfügbarkeit ($Brkdwn$): Dieser Parameter beschreibt, dass Stationsverfügbarkeiten berücksichtigt werden.

2.2.3.3 Zielfunktion γ

Der Parameter γ gibt die Zielfunktion an, nach der die Optimierung eines Schedules durchgeführt wird. Pinedo (2016) unterscheidet folgende elementare Zielfunktionen:

- Der Makespan ($\gamma = C_{max}$) beschreibt die Gesamtdurchlaufzeit eines Schedules. Er entspricht dem Maximum der Fertigstellungszeitpunkte C_j aller n Aufträge und kann mit Formel 2-2 ermittelt werden. Eine Minimierung der Makespan wird im Rescheduling angestrebt, um einen möglichst frühen Match-Up Zeitpunkt erreichen zu können.

$$C_{max} = \max\{C_1, \dots, C_n\} \qquad \text{2-2}$$

- Die Lateness ($\gamma = L_j$) eines Auftrags j gibt die Differenz zwischen Fertigstellungszeitpunkt C_j und Fälligkeitstermin dd_j an und kann mit Formel 2-3 bestimmt werden. Oft wird die maximale Lateness $L_{max} = \max\{L_1, \dots, L_n\}$ oder die gesamte Lateness $L_{tot} = \sum_{j=1}^{n} L_j$ zur Beurteilung von Schedules herangezogen. Die Lateness kann positive und negative Werte annehmen und unterliegt damit der Annahme, dass Mehrkosten durch verspätete Aufträge durch Minderkosten für früher fertiggestellte Aufträge kompensiert werden können.

$$L_j = \mathrm{C_j} - dd_j \qquad \text{2-3}$$

- Die Tardiness ($\gamma = T_j$) gibt die Verspätung des Auftrags j als Differenz zwischen Fertigstellungszeitpunkt C_j und dem Fälligkeitstermin dd_j an und wird nach Formel 2-4 ermittelt. Die Tardiness nimmt keine negativen Werte an, da nur die Verspätung von Aufträgen betrachtet wird. Sie unterliegt der Annahme, dass Mehrkosten durch Verspätungen nicht durch verfrühte Fertigstellungen von Aufträgen kompensiert werden können. Analog zur Lateness werden oft die maximale Tardiness $T_{max} = \max\{T_1, \dots, T_n\}$ oder die gesamte Tardiness $T_{tot} = \sum_{j=1}^{n} T_j$ herangezogen. Tardiness wird im weiteren Verlauf der Arbeit zur Bewertung der Lösungsgüte im prädiktiven robusten Scheduling herangezogen.

$$T_j = \max\{C_j - dd_j, 0\} \qquad \text{2-4}$$

2.2.4 Lösungsverfahren für das (Re-)Scheduling

In der Matrix-Produktion kann das zugehörige (Re-)Scheduling als flexibles Job Shop Problem formuliert werden. Das Problem kann dazu z. B. als gemischt-ganzzahliges Optimierungsproblem dargestellt werden. Schedules können dabei mit exakten oder heuristischen Verfahren erzeugt werden. Es existieren eine Vielzahl unterschiedlicher Lösungsansätze, die in Chaudhry und Khan (2016) und Çaliş und Bulkan (2015) zusammenfassend dargestellt sind.

Exakte Verfahren garantieren eine optimale Lösung. Allerdings gehört das flexible Job Shop Problem zur Klasse der NP-schweren Probleme, d. h. für den Fall $NP \neq P$ existiert kein Algorithmus, der es in polynomialer Zeit lösen kann (Garey & Johnson et al. 1976). Die Rechenzeit steigt exponentiell mit der Problemgröße an, sodass oft keine optimalen Lösungen ermittelt werden können. Bekanntester Vertreter exakter Verfahren ist der Branch&Bound-Algorithmus, der den Lösungsraum aufteilt (Branch), die resultierenden

Teilprobleme relaxiert und z. B. mithilfe des Simplex-Algorithmus löst. Ist die optimale Lösung der relaxierten Teilprobleme besser als die beste bisher bekannte Lösung, wird das Teilproblem weiter aufgeteilt und die beste Lösung aktualisiert (Bound), andernfalls wird es gelöscht. Dadurch werden bei jeder Aufteilung zusätzliche Restriktionen eingeführt, die den Lösungsraum einschränken. Im schlechtesten Fall ist eine vollständige Enumeration des Lösungsraumes notwendig. (Nickel & Stein et al. 2014).

Aufgrund der vergleichsweise hohen Rechenzeit exakter Verfahren kommen alternativ **heuristische Verfahren** zum Einsatz. Diese erlangen in sehr kurzer Zeit zulässige Lösungen auch für komplexe Probleme, können deren Optimalität aber nicht garantieren (Nickel & Stein et al. 2014). Für das Scheduling kommen dabei Metaheuristiken (z. B. Tabu- oder Nachbarschaftssuche, Ameisenalgorithmen, Schwarmalgorithmen oder Simulated Annealing) sowie evolutionäre Algorithmen zum Einsatz (Chaudhry & Khan 2016). Speziell für das Job Shop Scheduling wurde die Shifting-Bottleneck-Heuristik entwickelt (Adams & Balas et al. 1988). Dabei werden die Aufträge zunächst an jeder Station initial sequenziert und in jeder Iteration die resultierende Engpassstation („Bottleneck“) als diejenige Station identifiziert, die die maximale Verspätung aufweist. Für die verbleibenden Stationen wird ein Rescheduling durchgeführt unter Berücksichtigung des Schedules am Bottleneck. Dadurch ergibt sich in der nächsten Iteration ein neuer Bottleneck, der wiederum ein Rescheduling an den anderen Stationen auslöst. Durch das Vorgehen werden die Auswirkungen der Engpassstation auf die Makespan minimiert. Die Shifting Bottleneck Heuristik wurde im Laufe der Zeit für die Anwendung in flexiblen Job Shops weiterentwickelt (Mönch & Drießel 2005; Gao & Gen et al. 2007; Liu & Kozan 2017).

In der industriellen Praxis finden Prioritätsregeln als weitere Klasse heuristischer Verfahren häufige Anwendung (Panwalkar & Iskander 1977; Haupt 1989). Dabei werden Scheduling-Entscheidungen an jeder Station unabhängig voneinander getroffen, wobei die Prioritätsregeln der Priorisierung wartender Aufträge dienen und durch deren sequenzielle Anwendung ein Schedule ermittelt werden kann. Je nach Prioritätsregel wird entweder nur lokale Information genutzt oder auf eine erweiterte bzw. globale Informationsbasis zurückgegriffen. Die Kriterien zur Priorisierung umfassen z. B. Ankunftsreihenfolge (z. B. First In First Out, Last In First Out), die Prozesszeit oder Erweiterungen davon (z. B. Shortest Processing Time, Shortest Queue Next Operation) oder Fälligkeitstermine (z. B. Earliest Due Date). Prioritätsregeln kommen für das Scheduling in Job Shops der Halbleiterindustrie bevorzugt zum Einsatz, da die Anwendung anderer

Ansätze aufgrund der hohen Komplexität oft nicht praktikabel ist (Sarin & Varadarajan et al. 2010). Da keine Prioritätsregel strikt dominant ist, besteht die Herausforderung darin, situationsabhängig eine optimale Prioritätsregel oder deren Kombination zu identifizieren (Rajendran & Holthaus 1999).

Scheduling-Probleme können außerdem als Markovsche Entscheidungsprozesse (MDP) modelliert werden. Damit kann Unsicherheit durch Störungen berücksichtigt werden. Zur Lösung können entweder die optimierende dynamische Programmierung oder alternativ heuristische lernende Verfahren des Reinforcement Learning angewendet werden (Çaliş & Bulkan 2015; Csaji & Monostori 2006). Dynamische Programmierung iteriert ausgehend von einem Endzustand über alle möglichen vorausgehenden Zustände, um optimale Entscheidungspfade zu identifizieren. Zur Dekomposition des Problems wird das Bellman'sche Optimalitätsprinzip genutzt, nach dem sich der optimale Pfad aus jeweils optimalen Teilpfaden zusammensetzt (Bellman 1954). Der Berechnungsaufwand steigt jedoch exponentiell mit der Anzahl möglicher Zustände, sodass für größere Probleme das Reinforcement Learning eingesetzt werden kann (Busoniu & Babuska et al. 2008). Dabei wird eine optimale Strategie durch wiederholte Interaktion mit der Umwelt erlernt, allerdings ist eine Konvergenz zum Optimum in den meisten Fällen nicht garantierbar. Dennoch hat sich gezeigt, dass mit Reinforcement Learning bessere Ergebnisse als mit Prioritätsregeln erreicht werden können, weshalb im Folgenden eine Einschränkung auf diese Verfahren vorgenommen wird. Die wichtigsten Klassen der Verfahren werden näher in Kapitel 2.4 beschrieben.

2.3 Robustheit

Der Begriff der Robustheit wird in der Literatur unterschiedlich definiert. Gemeinsam ist den Definitionen, dass sie Robustheit im Allgemeinen als die Eigenschaft beschreiben, auch bei wechselnden Umweltbedingungen eine stabile Performance zu erreichen (Box & Andersen 1955; Scholl 2000; Stricker & Lanza 2014; Stricker 2016; Billaut & Moukrim et al. 2008). In der Produktion kommt der Robustheitsbegriff sowohl auf Ebene der Produktionsplanung (Stricker 2016) als auch in der Produktionssteuerung (Gören & Sabuncuoglu 2008; Jorge Leon & David Wu et al. 1994) zum Tragen.

2.3.1 Begriff der Robustheit in der Produktionssteuerung

In der Produktionssteuerung ist der Robustheitsbegriff eng mit dem Scheduling verbunden und bezeichnet analog zum allgemeinen Robustheitsbegriff die Eigenschaft eines

Schedules, seine Leistungsfähigkeit auch unter Störungen weitestgehend beizubehalten (Pinedo 2016). Robustheit ist auch für die Termin- und Kapazitätsplanung im Rahmen des Projektmanagements (Al-Fawzan & Haouari 2005; Van de Vonder, S. & Demeulemeester, E. et al. 2008; Herroelen & Leus 2004) sowie im Management von Schienen- (Yuan 2006; Caprara & Galli et al. 2010; Salido & Barber et al. 2008) und Luftverkehrsnetzen (Lan & Clarke et al. 2006; Burke & De Causmaecker et al. 2010; Yen & Birge 2006) relevant. Im Projektmanagement wird der möglichst realitätsnahen Terminierung von Projekten hohes Gewicht beigemessen, da Verzögerungen großen Einfluss auf die Profitabilität besitzen (Herroelen & Leus 2004). In Schienen- und Luftverkehrsnetzen werden Passagiere oft mithilfe einer Verkettung von Regional- und Fernverbindungen zu ihren Zielorten befördert, sodass mithilfe von Robustheit die Auswirkungen von Störungen auf nachgelagerte Verbindungen minimiert werden sollen (Burke & De Causmaecker et al. 2010). So soll ein möglichst reibungsloser Fluss von Passagieren entstehen (Caprara & Galli et al. 2010).

Im Scheduling spielen sowohl die **weitgehende Erhaltung der Leistungsfähigkeit unter Störungen** als auch die **realitätsnahe Terminierung von Aufträgen** eine zunehmende Rolle: Einerseits wird die Liefertreue als Differenzierungsmerkmal vom Wettbewerb immer wichtiger (Wiendahl 2009). Dies macht es erforderlich, im Scheduling möglichst realistische Liefertermine ermitteln zu können, die unter einer breiten Menge möglicher Umweltentwicklungen haltbar sind. Andererseits sind vor allem in der Matrix-Produktion die Auswirkungen von Störungen auf Schedules zu berücksichtigen, da aufgrund der vorherrschenden Freiheitsgrade vielfältige Handlungsoptionen zur Verbesserung der Robustheit und damit Verringerung der Auswirkungen der Störungen bestehen. Im Rahmen der vorliegenden Arbeit wird ein Schedule als robust definiert, wenn die erwartete Verzögerung je Bearbeitungsschritt minimal ist, da hierdurch die erwartete Verspätung von Aufträgen minimiert werden kann (Jamili 2016).

Mithilfe dedizierter Verfahren wird die Robustheit im Scheduling gesteigert (Claus & Herrmann et al. 2015; Gören & Sabuncuoglu 2008; Neuhaus 2008). Dazu werden Schlupfzeiten in Schedules eingefügt. Dies sind Leerzeiten zwischen Aufträgen, die nicht durch die Produktion bedingt sind. Mit ihnen können die Auswirkungen von Störungen verringert werden, da sie die Wahrscheinlichkeit verringern, aufgrund einer Störung Bearbeitungsschritte innerhalb eines Schedules verschieben zu müssen.

Zur Robustheitsmessung im Scheduling werden in der Literatur **szenariobasierte Maße** und **Ersatzmaße** unterschieden (Jorge Leon & David Wu et al. 1994). Szenariobasierte Maße bewerten die Robustheit basierend auf generierten Szenarien simulativ. Dabei wird eine Abweichung zwischen der Planung und der Realisierung je Szenario gemessen. Szenariobasierte Maße sind sehr gut geeignet, wenn die gewählten Szenarien repräsentativ für die Realität sind. In Situationen mit sehr vielen möglichen Szenarien sind unter Umständen allerdings sehr viele Szenarien nötig, um eine Aussagekraft zu erlangen. Dies geht mit einem hohen Rechenaufwand zur Bewertung aller Szenarien einher. Ersatzmaße hingegen ermitteln eine Ersatzgröße für die Robustheit, die mit dieser möglichst gut korrelieren und damit eine möglichst präzise Vorhersage über die Robustheit eines Schedules ermöglichen soll. (Jorge Leon & David Wu et al. 1994; Xiong & Xing et al. 2013; Claus & Herrmann et al. 2015; Shen & Han et al. 2017). In der vorliegenden Arbeit werden Ersatzmaße berücksichtigt, da aufgrund der Komplexität der Matrix-Produktion zu viele Szenarien nötig wären, um mit szenariobasierten Maßen eine statistisch signifikante Aussage zu erhalten.

2.3.2 Dimensionen der Robustheit

Zur Bewertung der Robustheit können unterschiedliche Robustheitskriterien verwendet werden, die im Folgenden in Anlehnung an (Scholl 2000) mit Bezug zum Scheduling beschrieben werden. Danach wird zwischen den für die vorliegende Arbeit relevanten Dimensionen Ergebnis-, Zulässigkeits- und Optimalitätsrobustheit sowie der Informations- und Planungsrobustheit unterschieden.

Von **Ergebnisrobustheit** (auch Qualitätsrobustheit) im Scheduling spricht man, wenn ein gewünschtes Anspruchsniveau der Zielerreichung (z. B. bestimmte Makespan oder maximale Terminabweichung) auch unter veränderter Umweltentwicklung erreicht werden kann (Scholl 2000; Herroelen & Leus 2004; Van de Vonder, S. & Demeulemeester, E. et al. 2008). Ein vollkommen ergebnisrobuster Schedule erreicht sein Anspruchsniveau immer, während dies bei relativ ergebnisrobusten Schedules nur für eine gewisse Schwankungsbandbreite von Umweltentwicklungen gilt (Scholl 2000).

Zulässigkeitsrobustheit (auch Lösungsrobustheit) bedeutet, dass ein Schedule trotz veränderter Umweltentwicklung weitgehend so wie geplant umsetzbar ist (Herroelen & Leus 2004; Scholl 2000; Gören & Sabuncuoglu 2008). Es besteht damit eine hohe Verwandtschaft zum Begriff der Stabilität. Die Forderung nach Zulässigkeitsrobustheit führt

zu einer wesentlichen Verschärfung gegenüber der Ergebnisrobustheit. Ein zulässigkeitsrobuster Schedule, der einem gewissen Anspruchsniveau genügt, ist gleichzeitig immer gegenüber diesem Anspruchsniveau ergebnisrobust (Scholl 2000). Ist ein Schedule für alle möglichen Umweltentwicklungen ohne Anpassung zulässig, wird er als vollkommen zulässigkeitsrobust bezeichnet. Bei geringfügigen Anpassungen hingegen wird ein Schedule als relativ zulässigkeitsrobust bezeichnet (Scholl 2000).

Von **Optimalitätsrobustheit** wird im Scheduling gesprochen, wenn ein Schedule unter allen möglichen Umweltentwicklungen die maximale Zielerreichung aufweist (Scholl 2000). In der Regel existieren solche Schedules nicht und sind daher auch nicht Ziel von Optimierungsansätzen. Stattdessen wird meist ein nahezu optimales Anspruchsniveau definiert, dass unter einer möglichst großen Bandbreite möglicher Umweltentwicklungen erreicht werden kann. Zwischen der so eingeschränkten Optimalität und der Robustheit besteht ein Trade-Off (Váncza & Monostori et al. 2011; Ben-Tal & Nemirovski 2002; Mulvey & Vanderbei et al. 1995; Kouvelis & Yu 1997; Stricker 2016). Je höher das gewählte Anspruchsniveau ist, desto geringer ist die Bandbreite möglicher Umweltentwicklungen, gegenüber denen Robustheit besteht.

Die **Informationsrobustheit** drückt aus, inwiefern ein Schedule gegenüber unvollständiger Information robust ist. Ein informationsrobuster Schedule ändert sich entsprechend bei Bekanntwerden neuer Informationen nicht (Scholl 2000). **Planungsrobustheit** schließlich bedeutet die Schaffung von Flexibilitätsvorhalten zur Plananpassung während dessen Ausführung (Scholl 2000; Bürgin 2018). Informations- und Planungsrobustheit werden im weiteren Verlauf der Arbeit nicht betrachtet.

2.3.3 Stabilität, Nervosität und Flexibilität

Im Zusammenhang mit dem Begriff der Robustheit werden im Scheduling oft die Begriffe Stabilität, Nervosität und Flexibilität verwendet, die jeweils unterschiedliche Aspekte der Robustheit herausstellen, ohne sich trennscharf von dieser abzugrenzen (Scholl 2000).

Stabilität bezieht sich auf den Unterschied zwischen geplanten und realisierten Schedules ohne explizite Berücksichtigung der Performance (Sabuncuoglu & Gören 2009). Bei einem stabilen Schedule ändern sich die geplanten Start- und Endzeiten von Bearbeitungsschritten auch bei unvorhergesehenen Störungen kaum. Ein bezüglich der Liefertreue sehr ergebnisrobuster Schedule verfügt über zeitliche Puffer, sodass diese sich erst bei sehr großen Verzögerungen verschlechtert. So kann es vorkommen,

dass der Schedule eine geringe Stabilität aufweist, gleichzeitig aber trotzdem sehr robust ist (Gören & Sabuncuoglu 2008).

Im Gegensatz dazu steht die **Nervosität**: In einem nervösen Schedule führen bereits geringfügige Störungen dazu, dass die realisierte stark von der geplanten Liefertreue abweicht (Kadipasaoglu & Sridharan 1995). Je stabiler ein Schedule ist, desto geringer ist seine Nervosität.

Flexibilität bezeichnet im Scheduling die Freiheitsgrade, die bei der Realisierung eines Schedules bestehen. Zeitliche Flexibilität bezieht sich auf die Startzeiten von Bearbeitungsschritten und ist gegeben, wenn diese sehr frei geändert werden können, ohne dass der restliche Schedule angepasst werden muss. Reihenfolgeflexibilität spiegelt wider, inwiefern Bearbeitungsschritte an Stationen getauscht werden können, ohne den verbleibenden Schedule anpassen zu müssen. Zuweisungsflexibilität schließlich gibt an, ob eine Verschiebung von Bearbeitungsschritten auf parallele Stationen möglich ist. (Billaut & Moukrim et al. 2008). Ein flexibler Schedule ist nicht automatisch robust, da es zusätzlich eines Verfahrens bedarf, das bei unvorhergesehenen Störungen die vorhandene Flexibilität zur Sicherung der Robustheit ausnutzt.

2.4 Markovsche Entscheidungsprozesse (MDP)

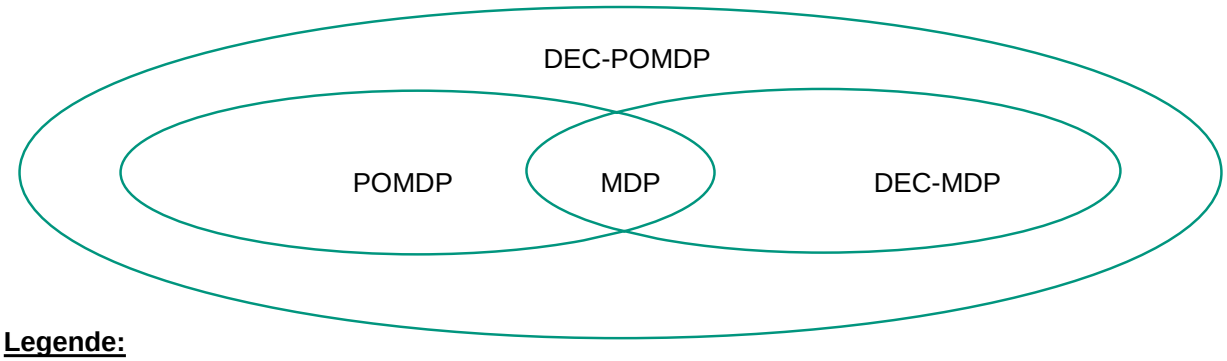
In der vorliegenden Arbeit soll auf Basis eines prädiktiv-robusten Schedules ein reaktives Rescheduling durchgeführt werden, sobald Störungen an Stationen über das erwartete Maß hinaus andauern. Das reaktive Rescheduling muss dabei in sehr kurzer Rechenzeit zu einem guten Ergebnis kommen, damit dieses in der Produktion implementiert werden kann (Schulte 1995). Dafür bietet sich die Modellierung des Reschedulings als Markovscher Entscheidungsprozess (MDP, Markov Decision Process) an. Hierbei wird das Rescheduling als sequenzielles Entscheidungsproblem dargestellt, in dem der umzuplanende Teil des Schedules durch die Aneinanderreihung lokaler Entscheidungen entsteht. Hierbei wird die Markov-Eigenschaft der Gedächtnislosigkeit ausgenutzt, die besagt, dass für eine Entscheidung nur der jeweils aktuelle Systemzustand relevant ist und die Entscheidungshistorie nicht zu berücksichtigen ist (Puterman 1994).

Daher stehen diese Verfahren für das reaktive Rescheduling im Zentrum der Betrachtungen dieser Arbeit. Vor allem dezentrale Ansätze sind hierbei von Interesse, da sie die inhärent verteilte Natur von Rescheduling-Problemen berücksichtigen, keine zent-

rale Instanz benötigen und aufgrund der kleinen Problemgröße in noch geringerer Rechenzeit gelöst und bessert skaliert werden können (Gabel 2009). Aus der Markov-Eigenschaft resultiert für lernende Verfahren der Vorteil, dass nur der aktuelle Zustand als Informationsbasis für die Entscheidungen genutzt werden muss. Im Folgenden werden die relevanten Klassen von MDP und ihre Elemente in Kapitel 2.4.1 eingeführt sowie die beiden wesentlichen Lösungsverfahren der Wert- und der Strategie-Iteration in Kapitel 2.4.2 erklärt.

2.4.1 Wesentliche Klassen Markovscher Entscheidungsprozesse

Eine Unterscheidung der Klassen von MDP kann anhand der Anzahl von Entscheidungsträgern (im Folgenden: Agenten) sowie der Beobachtbarkeit des Systemzustandes getroffen werden. Teilweise beobachtbare dezentrale Markovsche Entscheidungsprozesse (DEC-POMDP) verfügen dabei über mehrere Agenten, die parallelisiert unabhängig voneinander Entscheidungen treffen. Die teilweise Beobachtbarkeit besagt, dass die Agenten nur einen Teil des Systemzustands erhalten und ihre Entscheidungen daher ggf. auf Basis unterschiedlicher Information fällen. Dezentrale beobachtbare Markovsche Entscheidungsprozesse (DEC-MDP) unterscheiden sich davon, dass die Beobachtungen der Agenten gemeinsam den Systemzustand bestimmen. MDP und teilweise beobachtbare Markovsche Entscheidungsprozesse (POMDP) sind Spezialfälle dieser Ausprägungen mit jeweils einem einzigen Agenten. Die Zusammenhänge der Klassen sind in Abbildung 2-6 dargestellt und werden nachfolgend erläutert. (Bernstein & Givan et al. 2002)



Legende:

MDP = Markovscher Entscheidungsprozess

DEC-MDP = Dezentraler beobachtbarer Markovscher Entscheidungsprozess

POMDP = Teilweise beobachtbarer Markovscher Entscheidungsprozess

DEC-POMDP = Dezentraler teilweise beobachtbarer Markovscher Entscheidungsprozess

Abbildung 2-6: Klassen von MDP nach (Bernstein & Givan et al. 2002)

Markovsche Entscheidungsprozesse (MDP) sind sequenzielle Entscheidungsprobleme und werden nach Putermann (1994) folgendermaßen definiert: Zu einem bestimmten Zeitpunkt beobachtet ein Agent einen Systemzustand. Basierend darauf wählt er aus mehreren Aktionen eine aus und erhält dafür eine Belohnung. Außerdem wird der Systemzustand durch die gewählte Aktion mithilfe einer aktionsabhängigen Wahrscheinlichkeitsverteilung transformiert. Zu einem späteren Zeitpunkt beobachtet der Agent den Systemzustand erneut, allerdings kann sich das System nun in einem anderen Zustand befinden und es können ihm andere Aktionen zur Verfügung stehen. Dieser Vorgang wiederholt sich, bis das Ende des betrachteten Zeithorizonts erreicht ist.

Formell lässt sich ein MDP als Tupel $< Z, A, p, G >$ darstellen mit den Elementen:

- Z ist die Menge möglicher Zustände, die ein System einnehmen kann; z_t beschreibt den Zustand eines Systems zum Zeitpunkt t
- A ist die Menge möglicher Aktionen, die ein Agent wählen kann; a_t ist die zum Zeitpunkt t gewählte Aktion
- $p: Z \times A \times Z \rightarrow [0,1]$ ist die Übergangsfunktion und beschreibt die Wahrscheinlichkeit, $p(z, a, z')$ mit der die Aktion a im Zustand z dazu führt, dass das System in den Zustand z' übergeht
- $G: Z \times A \times Z \rightarrow \mathbb{R}$ ist die Belohnungsfunktion und beschreibt die Belohnung $g_t(z, a, z')$, die der Agent zum Zeitpunkt t bei der Wahl von Aktion a im Zustand z erhält, wenn das System in den Zustand z' übergeht

Die sequenzielle Natur des MDP mit den beschriebenen Elementen kann auch anhand von Abbildung 2-7 nachvollzogen werden (Sutton & Barto 2018).

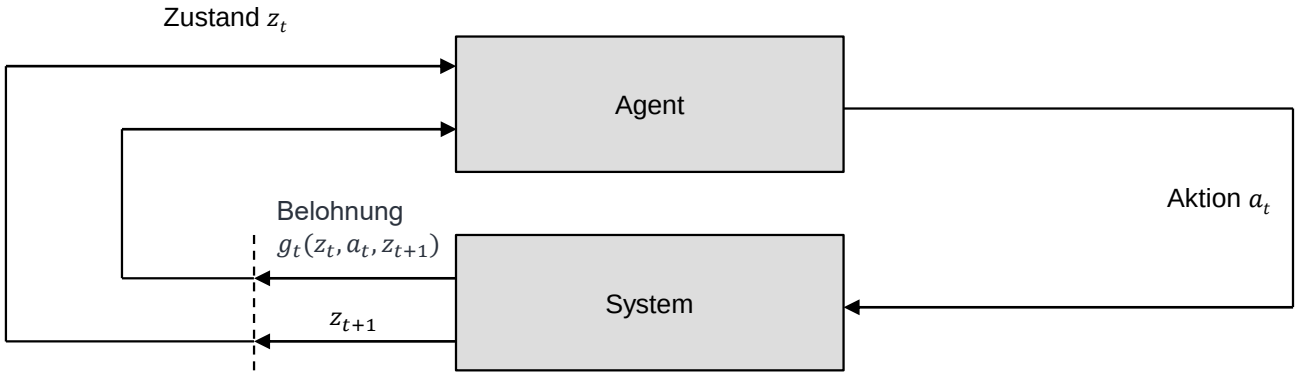


Abbildung 2-7: Interaktion eines Agenten mit dem System (Sutton & Barto 2018)

Im Gegensatz zu MDP existieren in **dezentralen beobachtbaren markovschen Entscheidungsprozessen (DEC-MDP)** mehrere Agenten, die Aktionen wählen können und damit den Systemzustand beeinflussen können. Mit DEC-MDP lassen sich so auch verteilte Entscheidungsprobleme abbilden. In der Anwendung stellt dies eine Herausforderung dar, da die Agenten Aktionen wählen können, die den Zielsystemen anderer Agenten zuwiderlaufen. Dies kann durch die Angleichung der Belohnungsfunktion aller Agenten gelöst werden.

Formell lässt sich ein DEC-MDP nach (Oliehoek & Amato 2016) als Tupel $< S, Z, A, p, G >$ darstellen mit den Elementen:

- S ist die Menge der Agenten, die Aktionen wählen können
- $A = A_1 \times \ldots \times A_{|Ag|}$ ist die Obermenge der Aktionen, die die Agenten wählen können
- Z, p, G sind wie für MDP definiert, allerdings gelten p und G jetzt für alle Aktionen $(a_1, \ldots, a_{|Ag|}) \in A$

Im weiteren Verlauf der Arbeit wird das reaktive Rescheduling mithilfe eines DEC-MDP formuliert. Die Dezentralität ist durch das Vorhandensein autonomer Stationen begründet, die selbst ihre nächsten zu bearbeitenden Aufträge wählen.

2.4.2 Reinforcement Learning als Lösungsverfahren für MDP

Sind die Elemente eines MDP definiert, ist das Modell für den Entscheidungsprozess festgelegt. Es bedarf allerdings noch eines Lösungsverfahrens zur Ermittlung einer Entscheidungsregel, mit der eine Aktion in Abhängigkeit eines beobachteten Zustandes

zur Maximierung der erhaltenen Belohnung gewählt werden kann. Diese Entscheidungsregel wird im Folgenden als Strategie bezeichnet.

Ziel des Reinforcement Learning ist es, Agenten eine optimale Strategie π über wiederholte Interaktion mit seiner Umwelt durch Versuch und Irrtum erlangen zu lassen (Sutton & Barto 2018; Russell & Norvig et al. 2016). Dieser Vorgang wird im Folgenden als Lernen bezeichnet. Verfolgt ein Agent s eine Strategie π_s, beschreibt $\pi_s(z, a)$ die Wahrscheinlichkeit, dass in Zustand z eine Aktion a gewählt wird (Formel 2-5).

$$\pi_s(z, a) = p(a|z) \tag{2-5}$$

Gewählte Aktionen führen immer zu positiven oder negativen Belohnungen und beeinflussen so die Strategie (Ertel 2016; Sutton & Barto 2018). Agenten versuchen stets, ihre kumulierte Belohnung zu maximieren. Die kumulierte Belohnung G_t kann mithilfe von Formel 2-6 als Summe aller erwarteten Belohnungen $\sum_{i=0}^{\infty} g_{t+i+1}$ diskontiert um einen Faktor $\gamma \in [0,1]$ dargestellt werden. Mit der Diskontierung lässt sich der Einfluss zukünftiger Belohnungen steuern. Für $\gamma = 1$ werden alle zukünftigen Belohnungen gleich gewichtet, mit $\gamma = 0$ ist nur die jeweils aktuelle Belohnung berücksichtigt. Diskontierungen werden für Probleme mit unendlichem Zeithorizont angewendet, da die Summe erst dann konvergiert.

$$G_t = g_{t+1} + \gamma * g_{t+2} + \gamma^2 * g_{t+3} + \cdots = \sum_{i=0}^{\infty} \gamma^i * g_{t+i+1} \tag{2-6}$$

Für die Bewertung eines Zustandes z in Abhängigkeit der Strategie $\pi(z)$ wird für viele Lösungsverfahren eine Zustandswertfunktion $V^{\pi}(z)$ (siehe Formel 2-7) eingeführt. Die Zustandswertfunktion gibt an, welchen Wert ein Zustand z unter Verwendung einer Strategie π hat. Die optimale Zustandswertfunktion $V^*(z) = \max_{\pi} V^{\pi}(z)$ besitzt für jeden Zustand z den höchsten Wert $V^{\pi}(z)$. Entsprechend ist die optimale Strategie $\pi^* = \arg\max_{\pi} V^{\pi}(z)$ diejenige Strategie, für die die Zustandswertfunktion maximal ist.

$$V^{\pi}(z) = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^i * g_{t+i+1} \,|Z_t = z\right], \forall z \in Z \tag{2-7}$$

Da mit der Zustandswertfunktion keine Aktion direkt ermittelt werden kann, wird daneben oft auch die Aktionswertfunktion (auch: Q-Funktion) $Q^{\pi}(z,a)$ (siehe Formel 2-8) bestimmt. Diese gibt an, welche Belohnung ausgehend von einem Zustand z erwartet werden kann, wenn die Aktion a gewählt und danach die Strategie $\pi(z)$ verfolgt wird. Die optimale Aktionswertfunktion $Q^{*}(z,a) = \max_{\pi} Q^{\pi}(z,a)$ besitzt für jeden Zustand z und jede Aktion a den höchsten Wert $Q^{\pi}(z,a)$. Entsprechend ist die optimale Strategie $\pi^{*} = \arg\max_{\pi} V^{\pi}(z)$ diejenige Strategie, für die die Aktionswertfunktion maximal ist.

$$Q^{\pi}(z,a) = \mathbb{E}\left[\sum_{i=0}^{\infty} \gamma^{i} * g_{t+i+1} \,|Z_t = z, A_t = a\right], \forall z \in Z; a \in A \qquad \text{2-8}$$

Das Lernen einer optimalen Strategie π kann also als Finden einer optimalen Zustandswertfunktion $V^{\pi}(z)$ oder einer Aktionswertfunktion $Q^{\pi}(z,a)$ interpretiert werden. Die Zustands- und Aktionswertfunktionen werden beim Lernen anfangs geschätzt und konvergieren durch wiederholte Übergänge zwischen Zuständen aufgrund gewählter Aktionen hin zu einem Mittelwert.

Ein intuitiver Lösungsansatz zur Erlangung optimaler Zustandswertfunktionen $V^{\pi}(z)$ oder optimaler Aktionswertfunktionen $Q^{\pi}(z,a)$ ist die vollständige Enumeration. Aufgrund der Komplexität eines MDP ist dies jedoch selbst für kleine Probleme nicht umsetzbar, da die Anzahl möglicher Strategien π exponentiell mit der Anzahl möglicher Zustände z steigt. Die Lösungsverfahren im RL sind modellfrei, d. h. sie benötigen für das Lernen kein vollständiges Abbild der Umwelt. Im Folgenden werden daher für die wesentlichen Klassen von Lösungsverfahren der Wert-Iteration und der Strategie-Iteration bekannte Vertreter modellfreier Algorithmen vorgestellt (Kaelbling & Littman et al. 1996).

2.4.2.1 Wert-Iteration am Beispiel des Q-Learning

Die Grundidee der Wert-Iteration besteht darin, eine optimale Aktionswertfunktion zu bestimmen, da diese indirekt die optimale Strategie beschreibt. Nach Formel 2-9 ist die optimale Strategie $\pi_s^{\star}$ ausgehend von einem Zustand z durch diejenige Aktionswertfunktion $Q(z,a)$ gegeben, die über alle Aktionen den höchsten Wert besitzt.

$$\pi_s^{\star} = max_{\forall a}\, Q(z,a) \qquad \text{2-9}$$

Das Q-Learning ist ein Verfahren zur Realisierung einer Wert-Iteration (Watkins & Dayan 1992). Die Q-Funktion wird dabei iterativ approximiert. Zunächst wird diese dazu zufällig oder identisch initialisiert. In jeder Iteration werden Aktionen in Zuständen gewählt und die Q-Funktion mittels Formel 2-10 aktualisiert (Ertel 2016). Dabei werden die Belohnung des aktuellen Zustands sowie die maximalen Werte der aktuellen Q-Funktion von Nachfolgerzuständen addiert. Über die Lernrate $\delta \in (0,1]$ kann eingestellt werden, wie stark die Ergebnisse der aktuellen Iteration die Q-Funktion beeinflussen. Bei kleinen Werten von δ dauert es tendenziell länger, bis die Q-Funktion das Optimum ausreichend gut approximiert. Ein sehr großes δ hingegen kann dazu führen, dass bei stochastischen Problemen keine Konvergenz erreicht wird und stattdessen ein oszillierendes Verhalten beobachtet werden kann (Sutton & Barto 2018).

$$Q_{t+1}\,(z_t, a_t) = Q_t(z_t, a_t) + \delta * [g_{t+1} + \gamma * max_{\forall a'} Q_t(z_{t+1}, a') - Q_t(z_t, a_t)] \qquad 2\text{-}10$$

Die Q-Funktion konvergiert nachweisbar, wenn der Agent in allen Zuständen Aktionen wählt, deren Wahrscheinlichkeit größer null ist (Sutton & Barto 2018). Das bedeutet, dass er keine reine Greedy-Strategie verfolgen darf, sondern mit einer gewissen Wahrscheinlichkeit $\varepsilon \in [0,1]$ eine zufällige Aktion wählen muss. Das Q-Learning terminiert, wenn die Veränderung der Q-Funktion ein Abbruchkriterium ω unterschreitet. Q-Learning ist ein modellfreies Verfahren, da die Übergangswahrscheinlichkeiten und die Belohnungsfunktion zur Berechnung nicht bekannt sein müssen.

2.4.2.2 Strategie-Iteration am Beispiel des REINFORCE Algorithmus

Die Strategie-Iteration vermeidet den Weg über eine Q-Funktion und passt stattdessen in jeder Iteration direkt die Strategie π_s an. Zum Einsatz kommt diese Lösungsverfahren oft in sehr komplexen Systemen, da der Berechnungsaufwand geringer ist (Gabel 2009). Dies ist im Falle der vorliegenden Arbeit der Fall, da eine gemeinsame Belohnungsfunktion für DEC-MDP vorliegt und die Wert-Iteration dafür nicht geeignet ist (Baird 1999; Szepesvári 2010). Der REINFORCE-Algorithmus ist ein Verfahren zur Realisierung einer Strategie-Iteration (Sutton & McAllester et al. 2000).

Zur Repräsentation der Strategie π_s wird dabei ein parametrierbarer Vektor $\overrightarrow{\theta_s}$ eingeführt. Die Skalare von $\overrightarrow{\theta_s}$ werden verwendet, um in einem Zustand die Wahrscheinlichkeit für die Wahl einer Aktion zu berechnen. Ziel der Strategie-Iteration ist es, die erwartete Leistungsfähigkeit $L(\pi_s)$ nach Formel 2-11 zu maximieren (Sutton & Barto

2018). Diese ergibt sich aus dem Erwartungswert der um γ diskontierten Belohnungen r_t. Die Leistungsfähigkeit der Strategie wird hier beispielhaft mithilfe der erzielbaren Makespan ermittelt.

$$L(\pi_s) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t * g_t | \pi_s\right] \quad \text{2-11}$$

Zu Beginn der Strategie-Iteration wird $\overrightarrow{\theta_s}$ mit null initialisiert. Damit ist die Wahrscheinlichkeit zur Wahl einer Aktion anfangs gleichverteilt. In jeder Iteration wird der Gradient von der Leistungsfähigkeit der $L(\pi_s)$ nach Formel 2-12 gebildet und $\overrightarrow{\theta_s}$ mittels Formel 2-13 aktualisiert.

$$\frac{dL(\pi_s)}{d\pi_s} = \mathbb{E}[-C_{max} * \ln \pi_s(a_{s,t}, z_{s,t}|\theta)] \quad \text{2-12}$$

Dabei wird $\overrightarrow{\theta_s}$ in Richtung des Gradienten $\frac{dL(\pi_s)}{d\pi_s}$ erhöht bzw. verringert. Der Parameter $\delta \in [0,1]$ stellt wiederum die Lernrate dar (Sutton & Barto 2018).

$$\overrightarrow{\theta_s} = \overrightarrow{\theta_s} + \delta * \frac{dL(\pi_s)}{d\pi_s} \quad \text{2-13}$$

Die Aktionen eines Agenten werden also nun in Abhängigkeit der θ gewählt. Wie die θ zur Entscheidungsfindung herangezogen werden, hängt vom verwendeten Verfahren der Strategie-Iteration ab. Zusammenfassend kann gesagt werden, dass eine Strategie-Iteration durch die kompakte Repräsentation der Strategie und die Vermeidung der Q-Funktion mit weniger Iterationen als die Werte-Iteration dieselben Ergebnisse erreichen kann. Daher wird im eigenen Ansatz auf die Strategie-Iteration zurückgegriffen.

3 Stand der Forschung

Im Folgenden wird der Stand der Forschung hinsichtlich der Problemstellung der vorliegenden Arbeit dargelegt. Dabei werden relevante Ansätze vorgestellt, ohne einen Anspruch auf Vollständigkeit zu erheben.

Zunächst werden in Kapitel 3.1 Ansätze für das prädiktiv-reaktive Scheduling erläutert und das Defizit einer fehlenden Robustheitsbetrachtung herausgearbeitet. Anschließend werden in Kapitel 3.2 Ansätze zur Robustheitsmessung im Scheduling diskutiert, die für das prädiktiv-robuste Scheduling notwendig sind. Nachfolgend werden in Kapitel 3.3 Ansätze für das Match-Up-Rescheduling vorgestellt. Diese sind erforderlich, um nach einer Störung die Informationsbasis für das Rescheduling zu ermitteln. Speziell für das Rescheduling existieren bisher kaum Ansätze, die ein Match-Up-Scheduling erlauben. Allerdings kann die Methodik in Scheduling-Ansätzen direkt auf das Match-Up-Scheduling übertragen werden. Daher werden in Kapitel 3.4 schließlich relevante Ansätze für das Scheduling vorgestellt, die auf RL basieren. Abschließend wird aus den Betrachtungen das Forschungsdefizit in Kapitel 3.5 aufgezeigt, welches mit dem in Kapitel 4 entwickelten Ansatz behoben wird.

Vorweg sei angemerkt, dass lediglich die Ansätze für das prädiktiv-reaktive Scheduling den Betrachtungsbereich der vorliegenden Arbeit vollständig adressieren, ohne dabei jedoch den Trade-Off zwischen Robustheit im prädiktiven Scheduling und Robustheit durch reaktives Rescheduling zu berücksichtigen.

3.1 Ansätze für das prädiktiv-reaktive Scheduling

Im Folgenden werden daher prädiktiv-reaktive Ansätze für das Scheduling in der Matrix-Produktion vorgestellt, da diese aufgrund ihres Aufbaus prinzipiell geeignet sind. Dazu werden folgende Bewertungskriterien verwendet:

- **Anwendbarkeit in der Matrix-Produktion:** Der Ansatz muss in der Lage sein, das Scheduling und Rescheduling für eine Matrix-Produktion mit redundanten Stationen und komplexen Materialflüssen mit Rückflüssen durchzuführen.
- **Erzeugbarkeit robuster prädiktiver Schedules**: Die prädiktive Komponente des Ansatzes soll dazu genutzt werden können, robuste prädiktive Schedules zu erzeugen. Diese sollen möglichst zulässigkeitsrobust sein, damit für das reaktive Rescheduling nutzbare Schlupfzeiten zur Verfügung stehen.

- **Berücksichtigung der Wechselwirkungen zwischen prädiktivem Scheduling und reaktivem Rescheduling:** Je nachdem, wie das prädiktive Scheduling durchgeführt wird, ist eine unterschiedliche Komplexität und Wirkungsweise des reaktiven Rescheduling zu erwarten. Der Ansatz soll diese Wechselwirkungen z. B. über eine geeignete Parametrierung berücksichtigen.

Van de Vonder behandelt das prädiktiv-reaktive Scheduling für das Resource-Constrained Project Scheduling Problem (RCPSP). Dabei werden Projekte beschränkten Ressourcen so zugewiesen, dass die Durchlaufzeit je Projekt minimiert wird. Für das prädiktive Scheduling wird eine Dekomposition des Planungsproblems durchgeführt: Zunächst wird mithilfe einer Optimierung ein nicht-robuster Schedule erzeugt, der anschließend durch das gezielte Einfügen von Schlupfzeiten basierend auf einer Heuristik zulässigkeitsrobust gemacht wird. Das reaktive Rescheduling wird ereignisbasiert bei Verzögerungen ausgelöst und mit Prioritätsregeln zur Minimierung der Auswirkung auf die Stabilität gelöst. Anhand einer experimentellen Untersuchung unterschiedlicher Kombinationen von prädiktiven und reaktiven Heuristiken wird deren jeweils sinnvoller Einsatzbereich aufgezeigt. (van de Vonder 2006)

Mahajan entwickelt eine Methode für das prädiktiv-reaktive Scheduling zur Minimierung der Makespan in einer flexiblen Fließfertigung mit parallelen Stationen, welche auf einer Kombination aus Simulation und Optimierung beruht. Im prädiktiven Scheduling wird mit einer modifizierten Shifting-Bottleneck-Heuristik (Adams & Balas et al. 1988) ein zulässiger Schedule abgeleitet, der geplante Instandhaltungsmaßnahmen, Materialverfügbarkeit und Eilaufträge berücksichtigt. Der Schedule wird danach in einer ereignisdiskreten Simulation des Produktionssystems feinabgestimmt, indem Engpässe über eine Umplanung auf parallele Stationen aufgelöst werden. Anschließend werden simulativ Störungen generiert, die ein reaktives Rescheduling erforderlich machen. Dazu wird ein Match-Up-Algorithmus mit integrierten Prioritätsregeln entwickelt, um die Abweichungen vom prädiktiven Schedule zu minimieren. (Mahajan 2007)

Duenas und Petrovic betrachten das Scheduling in einem einstufigen Produktionssystem mit parallelen Stationen, d. h. jeder Auftrag durchläuft genau einen Bearbeitungsschritt und kann dafür jede Station nutzen. Optimierungsziel ist die Minimierung der Makespan. Für das prädiktive Scheduling wird zunächst die benötigte Schlupfzeit berechnet, um den resultierenden Schedule robust zu machen. Dafür wird die Dauer der Nichtverfügbarkeit von Rohmaterialien mithilfe von Fuzzy Sets modelliert und anteilig den Prozesszeiten der Aufträge aufgeschlagen. Zur Erzeugung des prädiktiven

Schedules wird die Prioritätsregel Least Flexible Job verwendet. Für das reaktive Rescheduling werden parallel ein Left-Shift-Scheduling und vollständige Neugenerierung verwendet. Beim Left-Shift-Scheduling werden die eingeplanten Bearbeitungsschritte an einer Station vorgezogen. Dies ist z. B. dann möglich, wenn ein Bearbeitungsschritt durch eine Stationsstörung unterbrochen und aus dem Schedule entfernt wird. Übersteigt dessen geplante Prozesszeit die Dauer der Störung, so können die verbleibenden Bearbeitungsschritte vorgezogen werden. (Duenas & Petrovic 2008)

Van Brackel betrachtet eine Werkstattfertigung mit losweiser Fertigung, in der eine hohe Termintreue trotz Störungen erreicht werden soll. Dazu wird für das prädiktive Scheduling ein Optimierungsmodell mit dem Ziel der Minimierung der Produktionskosten formuliert. Dieses wird in einem Eröffnungsverfahren mithilfe der Prioritätsregel Earliest Due Date gelöst und die resultierenden Start- und Endzeitpunkte in einer Ablaufsimulation ermittelt. Als Verbesserungsverfahren wird ein Solver entwickelt, der die Loszuordnung, -reihenfolge und -zusammenfassung in begrenzter Rechenzeit anpasst. Für das reaktive Rescheduling wird bei Auftreten einer Störung ein zeitlimitbasierter Ansatz verfolgt. Dabei wird mithilfe einer Prioritätsregel zunächst ein vorläufiger zulässiger Plan erzeugt und implementiert, sowie anschließend mithilfe von Eröffnungs- und Verbesserungsheuristiken ein neuer, möglichst optimaler Plan erzeugt. Dieser ersetzt dann den vorläufigen zulässigen Plan. (van Brackel 2008)

Niehues entwickelt ein Modell zur prädiktiv-reaktiven Produktionssteuerung einer Werkstattfertigung durch eine fertigungsbegleitende Reihenfolgebildung. Dabei wir ein initialer prädiktiver Schedule durch Vorwärtsterminierung unter Berücksichtigung aller formulierten Restriktionen erzeugt. Eine Anpassung des initialen Schedules wird auf Basis einer Störungsklassifikation durchgeführt. Dabei werden Abweichungen zwischen Ist- und Soll-Verlauf sowie Abweichungen von Kapazitätsangebot und -nachfrage unterschieden. Je Störungsklasse wird eine Maßnahmenkaskade definiert, die von geringfügigen (Right-Shift-Rescheduling) bis großen Eingriffen (Reoptimierung der Auftragsreihenfolge) reicht. Dabei wird die nächstkomplexere Maßnahme ergriffen, wenn eine definierte Toleranz überschritten ist. So soll sichergestellt werden, dass der initiale Schedule nur so viel wie nötig angepasst wird. Die Reoptimierung der Auftragsreihenfolgen wird mit einem genetischen Algorithmus durchgeführt. (Niehues 2016)

Volk entwickelt einen prädiktiv-reaktiven Ansatz zur Planung des Rückbaus von Bauwerken und modelliert dies als Multimode Resource-Constrained Project Scheduling Problem (MRCPSP). Der Begriff „multimode“ beschreibt dabei, dass eine Aktivität an

unterschiedlichen Ressourcen eingeplant werden kann und es analog zur vorliegenden Arbeit nicht nur Bearbeitungsschritte, sondern auch Bearbeitungsalternativen gibt. Für die Erzeugung prädiktiver Schedules werden Szenarien generiert und optimale Rückbaustrategien abgeleitet. Die Lösungen werden anschließend auf alle Szenarien angewendet und die dazugehörigen Regret-Werte ermittelt. Die Regret-Werte geben an, wie stark sich die Lösungsgüte verschlechtert, wenn eine Lösung aus einem Szenario in einem anderen Szenario angewendet wird. Damit kann abhängig von der Risikopräferenz des Anwenders eine relativ optimalitätsrobuste Lösung vorgeschlagen werden. Dynamik entsteht durch neue Information wie z. B. Ressourcenverfügbarkeit oder geänderte Dauer von Aktivitäten. Bei einer geringfügig veränderten Informationsbasis wird ein Right-Shift-Rescheduling durchgeführt. Andernfalls wird zunächst eine lokale Suche unter den initial ermittelten Lösungen durchgeführt. Kann dort keine Lösung ausreichender Qualität identifiziert werden, werden die initialen Szenarien aktualisiert und wiederum mithilfe des Vorgehens für das prädiktive Scheduling robuste Lösungen identifiziert. (Volk 2017)

○ Nicht erfüllt
◑ Teilweise erfüllt
● Voll erfüllt

	Anwendbarkeit in der Matrix-Produktion	Erzeugbarkeit robuster prädiktiver Schedules	Wechselwirkungen zwischen prädiktivem Scheduling und reaktivem Rescheduling
van de Vonder (2006)	●	◑	◑
Mahajan (2007)	◑	○	○
Duenas & Petrovic (2008)	○	●	○
van Brackel (2009)	◑	○	○
Niehues (2016)	●	○	○
Volk (2017)	◑	●	○

Abbildung 3-1: Evaluation von Ansätzen für das prädiktiv-reaktive Scheduling

Die untersuchten Ansätze sind hinsichtlich der eingeführten Kriterien in Abbildung 3-1 bewertet und überblicksartig dargestellt. Aus der Analyse bestehender Ansätze für das

prädiktiv-reaktive Scheduling geht hervor, dass die meisten Ansätze gleich mehrere unterschiedliche Kombinationen aus prädiktiven und reaktiven Verfahren vorschlagen und deren Einfluss auf die Performance bewerten. Robustheit findet in den meisten Ansätzen keine explizite Beachtung. Eine Berücksichtigung von Wechselwirkungen zwischen prädiktivem Scheduling und reaktivem Rescheduling findet in keinem der vorgestellten Ansätze statt. Damit ist keiner der bestehenden Ansätze geeignet, um die in Kapitel 1.2 formulierte Hypothese zu überprüfen. Es ist daher ein neuer prädiktiv-reaktiver Ansatz erforderlich, der die genannten Anforderungen erfüllt. Zur Identifikation möglicher Elemente einer solchen Methodik werden im Folgenden Ansätze zur Robustheitsmessung, zum Match-Up-Rescheduling und zum Scheduling mit RL vorgestellt.

3.2 Ansätze zur Robustheitsmessung im Scheduling

In der Matrix-Produktion wird die Verfügbarkeit v_s von Stationen s durch $MTTR_s$ und $MTBF_s$ beschrieben. Die Kombinatorik von Störungen über die gesamte Matrix-Produktion hinweg folgt keiner formalisierbaren Gesetzmäßigkeit. Die hohe Anzahl möglicher Störszenarien können szenariobasierte Maße nicht adäquat abbilden (Volk 2017), weshalb die folgenden Ausführungen sich auf wesentliche Arbeiten zu Ersatzmaßen beschränken. Dabei werden folgende Bewertungskriterien zugrunde gelegt:

- **Anwendbarkeit in der Matrix-Produktion**: Das Ersatzmaß soll in der Lage sein, die Robustheit in Produktionssystemen mit redundanten Stationen und komplexen Materialflüssen mit Rückflüssen messen zu können.
- **Robustheit gegenüber auftragsindividuellen Lieferterminen**: Die Performance der Matrix-Produktion wird in dieser Arbeit anhand der mittleren Tardiness $\bar{T} = \frac{1}{n} * \sum_{j=1}^{n} T_j$ bewertet. Ersatzmaße sollen dementsprechend eine hohe Korrelation mit der mittleren Tardiness aufweisen.
- **Berücksichtigung von Störungskaskaden**: Von einer Störungskaskade spricht man, wenn eine auftretende Störung sich nicht nur auf die direkt betroffenen Bearbeitungsschritte auswirkt, sondern durch deren Verzögerung auch nachfolgende Bearbeitungsschritte beeinträchtigt (AhmadBeygi & Cohn et al. 2008; Yuan 2006). Das Robustheitsmaß soll Störungskaskaden berücksichtigen, da diese großen Einfluss auf die Liefertreue haben können.
- **Proaktives Einstellen der gewünschten Robustheit**: Das Robustheitsmaß soll genutzt werden können, um es in ein Vorgehen zur Erstellung robuster

Schedules integrieren zu können. Dabei soll über einen Parameter eine Einstellung der gewünschten Robustheit möglich sein.

Jorge Leon und David Wu definieren Ersatzmaße für das Job Shop Scheduling. Sie gehen davon aus, dass bei Störungen ein Right-Shift Rescheduling durchgeführt wird. Die Robustheit eines Schedules definieren sie als Linearkombination der Makespan und des Erwartungswerts der Verlängerung der Makespan. Die Ersatzmaße berücksichtigen die mittlere Verspätung von Bearbeitungsschritten sowie die durchschnittliche Schlupfzeit je Bearbeitungsschritt. Anhand einer Korrelationsstudie zeigen sie, dass Schlupfzeiten eine hohe Korrelation mit der Robustheit aufweisen und aufgrund ihrer einfachen Berechenbarkeit zu bevorzugen sind. (Jorge Leon & David Wu et al. 1994)

Mehta und Uzsoy definieren Robustheit anhand der Differenz zwischen geplanten und realisierten Fertigstellungszeitpunkten von Aufträgen in einem Job Shop. Optimierungszielgröße ist die Minimierung der maximalen Lateness. Es wird angenommen, dass Verteilungsfunktionen für die Störungsdauern und –abstände vorliegen, aus denen sich $MTBF_s$ und $MTTR_s$ für jede Station s ableiten lassen. Eines der vorgeschlagenen Ersatzmaße zählt die kritischen Bearbeitungsschritte je Auftrag. Ein Bearbeitungsschritt wird genau dann als kritisch bezeichnet, wenn seine erwartete Verzögerung seine Schlupfzeit übersteigt. Ein weiteres Ersatzmaß misst die Abweichung der Fertigstellungszeitpunkte aller Aufträge zwischen einem nicht-robusten Schedule und einem Schedule, in dem jeder Bearbeitungsschritt um seine erwartete Verzögerung verlängert wird. Mit einer entwickelten Heuristik erstellen die Autoren in einem zweistufigen Vorgehen einen robusten Schedule, indem zunächst ein prädiktiver Schedule erzeugt und dieser anschließend durch Schlupfzeiten ergänzt wird. (Mehta & Uzsoy 1999) Li und Wang verwenden dieselben Ersatzmaße, erweitern aber den Ansatz von Mehta und Uzsoy zur dahingehend, dass bereits existierende Schlupfzeiten im prädiktiven Schedule in die Robustheitsbewertung miteinbezogen werden. (Li & Wang 2009)

Wu und Byeon betrachten das robuste Scheduling in Job Shops unter Minimierung der gewichteten Tardiness. Mithilfe eines Branch&Bound-Algorithmus wird eine Menge unterschiedlicher Schedules für ein Szenario generiert und der robusteste ausgewählt. Das dafür verwendete Ersatzmaß stellt eine parametrierbare Linearkombination von oberer und unterer Schranke der gewichteten Tardiness dar. Der für den Anwendungsfall zu wählende Parameter wird durch dessen Variation anhand einer Korrelationsstu-

die ermittelt. Da das Ersatzmaß nur mithilfe einer großen Anzahl von Schedules operationalisiert werden kann, ist der Berechnungsaufwand hier sehr hoch. (Wu & Byeon et al. 1999)

Jensen entwickelt ein Robustheitsmaß für Job Shops, das die Robustheit als durchschnittliche Makespan der Nachbarschaft eines Schedules definiert. Eine Nachbarschaft eines Schedules wird dabei als die Menge aller Schedules definiert, die durch paarweise Vertauschungen von Bearbeitungsschritten zustande kommen. Eine Vergleichsstudie mit schlupfbasierten Robustheitsmaßen aus der Literatur zeigt, dass diese besser als das entwickelte Ersatzmaß geeignet sind, wenn ein einfaches Right-Shift-Rescheduling durchgeführt wird. Für komplexere Rescheduling-Ansätze wird das entwickelte nachbarschaftsbasierte Maß als besser eingestuft. (Jensen 2003)

Al-Fawzan und Haouari entwickeln ein Robustheitsmaß für das RCPSP, in dem die Aktivitäten Projekten entsprechen, denen Personal zugewiesen werden muss. Das Optimierungsziel ist dabei eine möglichst geringe Makespan. Die Autoren definieren Robustheit als Summe der freien Schlupfzeiten aller Projekte. Die freie Schlupfzeit eines Projektes entspricht der Zeit, die es auf der Zeitachse nach links und rechts verschoben werden kann, ohne dass ein anderes Projekt dafür verschoben werden muss. (Al-Fawzan & Haouari 2005)

Policella und Cesta betrachten ein RCPSP, in dem die Makespan minimiert werden soll. Robustheit wird mit der Flexibilität eines Schedules gleichgesetzt. Damit wird die Möglichkeit bezeichnet, den Schedule bei Bedarf mit geringem Aufwand anpassen zu können. Die Autoren greifen dazu auf das Flexibilitätsmaß von Aloulou und Portmann zurück (Aloulou & Portmann 2003). Dieses misst, wie hoch der Anteil aller Paare von Bearbeitungsschritten im Schedule ist, zwischen denen keinerlei Vorrangbeziehungen existieren. Dies folgt der Annahme, dass ein Schedule leichter angepasst werden kann, wenn einzelne Bearbeitungsschritte unabhängig von anderen verschoben werden können. Die erhaltene Aussage je Bearbeitungsschritt ist allerdings rein binär und bezieht nicht die Lage der Bearbeitungsschritte zueinander quantitativ ein. Daher schlagen die Autoren ein weiteres „Fluidität“ genanntes Ersatzmaß vor, das die Schlupfzeit zwischen Paaren von Bearbeitungsschritten ins Verhältnis zu der maximal möglichen Schlupfzeit über die Makespan des Schedules setzt. (Policella & Cesta et al. 2007)

Hazır und Haouari entwickeln und bewerten Ersatzmaße für das diskrete Zeit-/Kostenproblem im Multi-Projektmanagement. Dabei sind Ressourcen Projekten zuzuordnen,

sodass die zeitlichen Restriktionen eingehalten werden. Die entstehenden Kosten werden über einen Strafterm als weiche Nebenbedingung berücksichtigt. Eines der entwickelten Ersatzmaße summiert für jeden Bearbeitungsschritt das Produkt aus Anzahl unmittelbarer Nachfolger an derselben Station und totalem Schlupf auf. Der totale Schlupf bezeichnet das Zeitfenster zwischen frühster und spätester Ausführung von Bearbeitungsschritten an einer Station ohne Veränderung der Makespan des Schedules. (Hazır & Haouari et al. 2010) Er ist damit dem freien Puffer aus der Netzplantechnik gleichzusetzen (Küpper & Lüder et al. 2013). Damit wird der Annahme gefolgt, dass lange Ketten von Bearbeitungsschritten störungsanfälliger sind und mehr Schlupfzeiten benötigen.

Martinez betrachtet das flexible Job Shop Problem mit dem Ziel, den Makespan zu minimieren. Als robust wird ein Schedule dann bewertet, wenn seine realisierte Makespan eine möglichst geringe relative Abweichung von der geplanten Makespan hat. Um Schedules robust zu machen, wird das Konzept der Kritikalität von Stationen eingeführt. Dabei wird eine Station in einem Zeitintervall genau dann als kritisch bezeichnet, wenn die Anzahl direkt aufeinanderfolgender Bearbeitungsschritte einen Schwellwert übersteigt. Bei kritischen Stationen wird entsprechend in der betroffenen Sequenz Schlupfzeit in Höhe der erwarteten Verzögerung je Bearbeitungsschritt eingefügt. Mithilfe des Schwellwerts kann geregelt werden, wie viel Schlupfzeit in den Schedule eingebracht wird. Es wird gezeigt, dass sich gegenüber dem pauschalen Einfügen von Schlupfzeiten eine geringere Abweichung der Makespan realisieren lässt, da bestehende Schlupfzeiten im Schedule berücksichtigt werden. (Martinez 2012)

Tolio und Urgo sowie Urgo und Váncza betrachten ein stochastisches Scheduling-Problem mit einer Station, bei dem das Optimierungsziel die Minimierung der maximalen Lateness ist. Die Prozesszeiten der Bearbeitungsschritte werden als diskret verteilt angenommen. Zur Messung der Robustheit verwenden die Autoren den Value-At-Risk der die Wahrscheinlichkeit misst, mit der ein gewisser Verlust innerhalb eines definierten Zeitraums nicht überschritten wird. Ein Verlust wird als Verlängerung der Bearbeitungszeit über den Mittelwert hinaus definiert. Das zu berücksichtigende Risiko kann parametriert werden und so unterschiedliche Präferenzen abbilden. (Tolio & Urgo et al. 2011; Urgo & Váncza 2014)

Xiong und Xing betrachten die Robustheit beim Scheduling in flexiblen Job Shops unter Minimierung der Makespan. Die Autoren entwickeln ein bestehendes Ersatzmaß von Jorge Leon und Wu weiter, welches den totalen Schlupf zur Robustheitsbewertung

heran zieht. In dem neuen Ersatzmaß wird eine zusätzliche Gewichtung des totalen Schlupfes mit der relativen Auslastung der betroffenen Station eingeführt, um Schedules mit hohem totalen Schlupf an stark ausgelasteten Stationen zu bevorzugen. Ein weiteres Robustheitsmaß bezieht die Dichtefunktion der Ausfallwahrscheinlichkeit der Stationen ein, um den wahrscheinlichsten Ausfallzeitpunkt zu ermitteln und dadurch diejenigen Stellen im Schedule zu identifizieren, an dem zusätzliche Schlupfzeiten den größten erwarteten Effekt auf die Robustheit haben. (Xiong & Xing et al. 2013)

Jamili misst Robustheit als erwartete Verzögerung von Bearbeitungsschritten in einem Job Shop und definiert einen Parameter, mit dem diese durch robuste Planung auf einen maximalen Wert begrenzt werden kann. Optimierungsziel ist der Makespan des Schedules. Die beiden entwickelten Ersatzmaße benötigen eine unterschiedliche Informationsbasis hinsichtlich der Störungen von Stationen. Das erste Maß betrachtete lediglich die Verfügbarkeit von Stationen, während das zweite Maß auch die Dichtefunktion der Ausfallwahrscheinlichkeiten voraussetzt. Zur Ermittlung benötigter Schlupfzeiten werden sowohl horizontale Schlupfzeiten (entlang einer Station) als auch vertikale Schlupfzeiten (entlang eines Auftrags) erzeugt. (Jamili 2016)

Die analysierten Ersatzmaße zur Robustheitsmessung im Scheduling in Abbildung 3-2 lassen sich größtenteils auf die Matrix-Produktion übertragen. Die meisten Ersatzmaße adressieren jedoch ausschließlich den Makespan als Optimierungsziel, während Tardiness oder Lateness nur von Jorge Leon und Wu sowie von Urgo und Váncza berücksichtigt werden. Des Weiteren werden Störungskaskaden kaum betrachtet, sondern die isolierte Auswirkung von Störungen auf einzelne Bearbeitungsschritte untersucht. Das proaktive Einstellen der gewünschten Robustheit ermöglichen einige der diskutierten Ansätze, sodass ein Übertrag auf die vorliegende Arbeit möglich ist.

○ Nicht erfüllt
◑ Teilweise erfüllt
● Voll erfüllt

	Anwendbarkeit in der Matrix-Produktion	Robustheit gegenüber auftragsindividuellen Lieferterminen	Berücksichtigung von Störungskaskaden	Proaktives Einstellen der gewünschten Robustheit
Jorge Leon & David Wu et al. (1994)	◑	○	○	○
Mehta & Uzsoy (1999); Li & Wang (2009)	●	○	○	◑
Wu & Byeon et al. (1999)	◑	◑	○	○
Jensen (2003)	●	○	○	○
Al-Fawzan & Haouari (2005)	◑	○	○	○
Policella & Cesta et al. (2007)	◑	○	○	●
Hazır & Haouari et al. (2010)	◑	◑	◑	○
Martinez (2012)	●	○	◑	●
Tolio & Urgo et al. (2011); Urgo & Váncza (2014)	○	●	○	●
Xiong & Xing et al. (2013)	●	○	◑	◑
Jamili (2016)	◑	◑	●	●

Abbildung 3-2: Evaluation von Ansätzen zur Robustheitsmessung im Scheduling

3.3 Ansätze für das Match-Up-Rescheduling

Match-Up-Rescheduling bietet eine Möglichkeit, bei ressourcen- oder auftragsbezogenen Störungen nur einen Teil des zugrundeliegenden Schedules neu zu erzeugen. Ziel ist es dabei, nach einiger Zeit wieder zum zugrundeliegenden Schedule zurück zu kehren. Match-Up-Ansätze kumulieren ab Auftreten einer Störung Schlupfzeiten in Schedules die ausreichend sein sollen, um die Störung kompensieren zu können. Im Folgenden werden einige wesentliche Ansätze für das Match-Up-Rescheduling aus der Literatur vorgestellt und bezüglich folgender Bewertungskriterien diskutiert:

- **Anwendbarkeit in der Matrix-Produktion**: Der Match-Up-Ansatz soll die Freiheitsgrade der Matrix-Produktion bestmöglich ausnutzen können. Insbesondere müssen die gesammelten Schlupfzeiten parallele Stationen für die Bearbeitung der betroffenen Aufträge berücksichtigen.

- **Berücksichtigung ressourcenbezogener Störungen**: In der vorliegenden Arbeit werden Störungen an Stationen betrachtet. Es soll daher möglich sein, ressourcenbezogene Störungen berücksichtigen zu können. Auftragsbezogene Störungen, wie z. B. fehlendes Material, können als ressourcenbezogene Störung modelliert werden (Abumaizar & Svestka 1997).
- **Reintegration des Rescheduling-Ergebnisses**: Der zugrundeliegende Schedule soll sich durch das Rescheduling möglichst wenig ändern. Es soll daher möglich sein, das Ergebnis des Rescheduling möglichst nahtlos in den zugrundeliegenden Schedule zu integrieren und soweit wie möglich auf dessen weitere Modifikationen zu verzichten.

Bean und Birge schlagen ein Verfahren zum Match-Up-Rescheduling in Produktionssystemen mit parallelen Stationen und einer Produktionsstufe vor. Als zusätzliche Restriktionen werden Freigabezeitpunkte und reihenfolgeabhängige Rüstzeiten zwischen unterschiedlichen Aufträgen betrachtet. Die gewählte Rescheduling-Politik ist ereignisbasiert und abhängig von ressourcen- und auftragsbezogenen Störungen. Ziel ist die Minimierung der Tardiness. Bei Auftreten einer Störung wird durch den Anwender für jede Station ein minimaler individueller Match-Up-Zeitpunkt beliebig festgelegt. Anschließend werden die Aufträge an jeder Station mithilfe von Prioritätsregeln resequenziert. Unter den neuen Sequenzen wird je Station diejenige Sequenz mit der geringsten Tardiness gewählt. Ist mindestens an einer Station die Tardiness höher als ein benutzerdefinierter Schwellwert, wird der Match-Up-Zeitpunkt der betroffenen Station um ein Inkrement erweitert und ein erneutes Rescheduling mit Prioritätsregeln durchgeführt. (Bean & Birge et al. 1991)

Akturk und Gorgulu erweitern das Verfahren von Bean und Birge für eine Gruppenfertigung mit gerichtetem Materialfluss. Aufträge können einzelne Stationen überspringen und müssen nicht zwangsläufig alle Gruppen durchlaufen, sondern können an beliebigen Gruppen ein- und ausgeschleust werden. Das Match-Up-Verfahren wird durch eine Störung an einer Gruppe ereignisbasiert ausgelöst mit dem Ziel, die störungsbedingte Tardiness zu minimieren und einen minimalen Match-Up-Zeitpunkt zu ermitteln. Zunächst wird das Rescheduling an der gestörten Station durchgeführt, bevor vor- und nachgelagerte Stationen betrachtet werden. Diese Dekomposition ist möglich, da ohne Rezirkulation von Aufträgen und parallelen Stationen für das Rescheduling kein Routing-Problem gelöst werden muss. Beim Auftreten einer Störung wird zunächst ein Match-Up-Zeitpunkt an der gestörten Station ermittelt, indem so viel vorhandene

Schlupfzeit kumuliert wird, bis diese die Dauer der Störung übersteigt. Mithilfe eines Branch&Bound-Algorithmus werden die Aufträge anschließend so resequenziert, dass die Tardiness minimiert wird. Für die vorgelagerten Stationen wird das Rescheduling nun heuristisch durchgeführt, sodass die Fertigstellungszeitpunkte vor den Startzeitpunkten an der gestörten Station liegen. Analog wird für die nachgelagerten Stationen das Rescheduling so durchgeführt, dass die Startzeitpunkte nach den Fertigstellungszeitpunkten an der gestörten Station liegen. Aufgrund der separat gelösten Rescheduling-Probleme kann es zu Überlappungen kommen, die durch eine Verschiebung des Match-Up-Zeitpunktes iterativ behoben werden. (Akturk & Gorgulu 1999)

Mahajan entwickelt einen Match-Up-Algorithmus für flexible Fließfertigungssysteme mit parallelen Stationen und betrachtet dabei Puffer an Stationen und auftragsspezifische Fälligkeitstermine. Das Verfahren wird ereignisbasiert durch ressourcenbezogene Störungen ausgelöst. Ziel des Match-Up-Algorithmus ist die Minimierung von Abweichungen der Bearbeitungszeiten und -reihenfolgen vom zugrundeliegenden Schedule. Beim Auftreten einer Störung wird zunächst deren Auswirkung ohne Rescheduling als obere Grenze des Match-Up-Zeitpunktes ermittelt, indem ein Right-Shifting durchgeführt wird. Mithilfe von Prioritätsregeln werden danach die von einer Störung betroffenen Aufträge iterativ an parallelen Stationen eingeplant. Dabei wird ein Auftrag jeweils an der Station eingeplant, die bis zum Match-Up Zeitpunkt die höchste verfügbare Kapazität hat. Bereits eingeplante Aufträge werden nicht verschoben, sodass Überlappungen entstehen können. Die Auswirkungen der zusätzlichen Einplanungen hinsichtlich Makespan und Reihenfolgestabilität werden je Iteration simulativ bewertet. Anschließend wird diejenige Lösung implementiert, deren Auswirkungen minimal sind. Ist die eingangs ermittelte obere Schranke ausreichend, so wird kein Rescheduling auf parallelen Stationen durchgeführt. (Mahajan 2007)

Moratori und Petrovic betrachten einen flexiblen Job Shop, der auftragsbedingten Störungen durch zusätzliche Aufträge unterliegt. Ziel des entwickelten Match-Up-Verfahrens ist die Minimierung einer multikriteriellen Zielfunktion, die die Makespan sowie ein von den Autoren definiertes Stabilitätskriterium enthält. Als zusätzliche Nebenbedingungen werden Freigabezeitpunkte, Vorrangbeziehungen sowie reihenfolgeabhängige Rüstzeiten betrachtet. Es wird ein über alle Stationen einheitlicher Match-Up-Zeitpunkt bestimmt. Für die Anwendung in der Matrix-Produktion ist dies sinnvoll, da so ein einheitlicher Rescheduling-Korridor entsteht, der das Lösen des Routing- und des Scheduling-Problems erlaubt und damit die Freiheitsgrade der Matrix-Produktion bestmöglich

ausnutzt. Ähnlich wie der Ansatz von Akturk und Gorgulu werden hier Schlupfzeiten zwischen Bearbeitungsschritten kumuliert, um einen Match-Up-Zeitpunkt zu bestimmen. Ausgehend vom Beginn der Störung wird der späteste Fertigstellungszeitpunkt aller bereits in Bearbeitung befindlichen Aufträge ermittelt und daraus die untere Schranke für die Länge des Rescheduling-Korridors abgeleitet. Davon ausgehend wird die Schlupfzeit in Höhe der für den zusätzlichen Auftrag benötigten Prozesszeit auf den Stationen kumuliert. Für das Rescheduling der Bearbeitungsschritte innerhalb des Rescheduling-Korridors wird ein genetischer Algorithmus angewendet (Petrovic & Fayad et al. 2008). Da die im zugrundeliegenden Schedule eingeplanten Bearbeitungsschritte an der oberen Grenze dabei nicht berücksichtigt werden, kann es zu Überlappungen kommen, die durch ein Right-Shift Rescheduling aufgelöst werden. (Moratori & Petrovic et al. 2011)

○ Nicht erfüllt
◑ Teilweise erfüllt
● Voll erfüllt

	Anwendbarkeit in der Matrix-Produktion	Berücksichtigung ressourcenbezogener Störungen	Reintegration des Rescheduling-Ergebnisses
Bean et al. (1991)	◑	●	◑
Akturk & Gorgulu (1999)	◑	●	●
Mahajan (2007)	◑	●	◑
Moratori et al. (2012)	●	◑	◑

Abbildung 3-3: Evaluation von Ansätzen für das Match-Up-Rescheduling

Die Ansätze für das Match-Up-Rescheduling in Abbildung 3-3 lassen sich alle prinzipiell auf die Matrix-Produktion übertragen. Auch ressourcenabhängige Störungen können entweder direkt oder als zusätzliches Element integriert werden. Ebenso sehen alle Ansätze eine Reintegration des Rescheduling-Ergebnisses vor. Für die vorliegende Arbeit kann daher zu großen Teilen auf dem Stand der Technik aufgebaut werden.

3.4 Ansätze für das Scheduling mit Reinforcement Learning

Zur Lösung von MDP mit Reinforcement Learning werden ein oder mehrere Agenten benötigt, die als Entscheidungsinstanzen agieren und ein gemeinsames Ziel verfolgen. Im Fall der vorliegenden Arbeit ist dies ein möglichst schnelles Zurückkehren zum prädiktiven Schedule.

Im weiteren Verlauf der Arbeit soll ein entsprechend geeigneter Ansatz aufgegriffen und für das reaktive Rescheduling in der Matrix-Produktion hin weiterentwickelt werden. Die Ansätze werden anhand der folgenden Kriterien bewertet:

- **Anwendbarkeit in der Matrix-Produktion**: Der Ansatz soll auftragsindividuelle Materialflüsse sowie parallele Stationen berücksichtigen können, um für die Matrix-Produktion geeignet zu sein.
- **Reaktivität hinsichtlich Störungen an Stationen**: Die Agenten sollen in der Lage sein, auf immer wieder wechselnde Situationen im Rescheduling angemessen reagieren zu können.
- **Berücksichtigung von Transportzeiten**: Der Ansatz soll layoutabhängige Transportzeiten mit einbeziehen.
- **Berücksichtigung von reihenfolgeabhängigen Rüstzeiten**: Der Ansatz soll reihenfolgeabhängige Rüstzeiten beim Rescheduling berücksichtigen.
- **Berücksichtigung von Puffern**: Es soll möglich sein, den Ansatz auch bei begrenzten Puffern anwenden zu können.

Gabel und Riedmiller betrachten das Scheduling in Job Shops mit stochastischen Prozesszeiten und beschreiben das Scheduling als DEC-MDP mit mehreren Agenten, die durch die Stationen des Produktionssystems repräsentiert werden. Zur Lösung entwickeln sie ein Verfahren zur verteilten Strategie-Iteration. Die Agenten beobachten in jedem Zustand die Aufträge im Puffer sowie an vorgelagerten Stationen und wählen anhand einer Strategie den nächsten zu bearbeitenden Auftrag. Mithilfe einer verteilten Strategie-Iteration werden situationsabhängig optimale Parameter für die Auswahl von Aufträgen an jeder Station gelernt. Die Belohnungsfunktion generiert eine negative Belohnung von -1 für jeden Zeitschritt, wodurch der Makespan minimiert wird. (Gabel & Riedmiller 2012)

Martinez betrachtet das Job Shop Problem mit parallelen Stationen, welches dem flexiblen Job Shop Problem entspricht. Dabei wird angenommen, dass für jedes Arbeitszentrum mit mehreren Stationen ein gemeinsamer Puffer existiert, in dem die Aufträge

zwischengelagert werden. Zur Lösung wird eine Wert-Iteration verwendet, die jede Station als eigenen Agenten darstellt. Die Belohnung der Agenten basiert auf der Minimierung der Anzahl unbeschäftigter Stationen, um einen geringen Makespan zu erreichen. (Martinez 2012)

Qu und Wang betrachten einen Flow Shop, der von Aufträgen losweise in derselben Reihenfolge durchlaufen wird, wobei jede Produktionsstufe redundante Stationen enthalten kann. Im Scheduling werden den Stationen Aufträge und Werker in Abhängigkeit einer Qualifikationsmatrix zugewiesen. Dabei werden sowohl Puffer vor den Produktionsstufen als auch reihenfolgeabhängige Rüstzeiten an den Stationen berücksichtigt. Das Produktionssystem wird durch zwei Agenten gesteuert, die für die Sequenzierung der Aufträge an den Stationen und die Allokation von Werkern zu Stationen verantwortlich sind. Für das Anlernen wird eine Wert-Iteration verwendet. Die Belohnung der Agenten hängt von Durchsatz und der Produktivität der Werker ab. (Qu & Wang et al. 2016)

Bouazza und Sallez modellieren ein Produktionssystem als flexiblen Job Shop, in dem die Minimierung der mittleren gewichteten Wartezeit angestrebt wird. Reihenfolgeabhängige Rüstzeiten und unbeschränkte Puffer an den Stationen werden angenommen. Die Aufträge werden als Agenten modelliert. Das Routing und Scheduling wird mittels vorgegebener Prioritätsregeln vorgenommen, wobei die Wahl einer Prioritätsregel über eine Wert-Iteration ermittelt wird. Eine positive Belohnung wird generiert, wenn die mittlere gewichtete Wartezeit je Auftrag im Vergleich zur vorhergehenden Iteration verringert wird. Für das Anlernen wurde jeweils die Routing- bzw. Scheduling-Entscheidung fixiert. (Bouazza & Sallez et al. 2017)

Waschneck und Reichstaller betrachten das Scheduling in flexiblen Job Shops. Es werden dabei reihenfolgeabhängige Rüstzeiten, die Rezirkulation von Aufträgen, variierende Losgrößen sowie dynamische Verfügbarkeiten berücksichtigt und das System damit als komplexer flexibler Job Shop beschrieben. Die Routing- und Scheduling-Entscheidungen werden von dezentralen Agenten getroffen, die die Stationen repräsentieren und über tiefe neuronale Netze angelernt werden (Mnih & Kavukcuoglu et al. 2015). Die tiefen neuronalen Netze werden im ersten Schritt mithilfe von überwachtem Lernen antrainiert. Dazu werden auf Basis von Expertenwissen mit historischen Daten Zustands-Aktions-Paare gebildet. Anschließend werden sie mit einer Wert-Iteration durch Interaktion mit einer ereignisdiskreten Ablaufsimulation weiter trainiert. Obwohl ein flexibler Job Shop betrachtet wird, wird dieser durch Annahmen soweit vereinfacht, dass

eine Auswahl von Stationen an einer Produktionsstufe nicht notwendig ist. Die Reaktivität hinsichtlich Störungen wird als Möglichkeit beschrieben, ist aber nicht Teil der Auswertungen. (Waschneck & Reichstaller et al. 2018)

○ Nicht erfüllt
◑ Teilweise erfüllt
● Voll erfüllt

	Anwendbarkeit in der Matrix-Produktion	Reaktivität hinsichtlich Störungen an Stationen	Berücksichtigung von Transportzeiten	Berücksichtigung von reihenfolgeabhängigen Rüstzeiten	Berücksichtigung von beschränkten Puffern
Gabel & Riedmiller (2012)	◑	●	○	○	○
Martinez (2012)	●	●	○	○	○
Qu & Wang et al. (2016)	○	●	○	◑	●
Bouazza & Sallez et al. (2017)	◑	●	○	○	○
Waschneck & Reichstaller et al. (2018)	◑	◑	○	◑	○

Abbildung 3-4: Evaluation von Ansätzen für das Scheduling mit RL

Die Ansätze für das Scheduling mit RL in Abbildung 3-4 können bis auf eine Ausnahme entweder direkt oder mit Erweiterungen auf die Matrix-Produktion übertragen werden. Die Reaktivität auf Störungen bildet ein Kernelement in allen Ansätzen. Während reihenfolgeabhängige Rüstzeiten und beschränkte Puffer von einigen Autoren berücksichtigt werden, bezieht kein untersuchter Ansatz Transportzeiten in die Bewertung ein.

3.5 Forschungsdefizit

Bestehende Ansätze für das prädiktiv-reaktive Scheduling betonen zwar die Bedeutung prädiktiver robuster Schedules, verwenden diese aber nur punktuell. Kein untersuchter Ansatz nutzt Robustheit als Steuergröße zur Optimierung.

Im prädiktiven Scheduling wird Robustheit zur Absicherung von Schedules gegenüber Störungen verwendet. Aufgrund der Möglichkeit zum reaktiven Rescheduling sind Effizienzeinbußen zu erwarten, da mit prädiktiv vorgehaltener Robustheit lediglich auf statistisch erwartete Störungen reagiert werden kann. Es werden unter Umständen also nicht benötigte Schlupfzeiten vorgehalten, die den Nutzen der robusten Planung über-

kompensieren. Ein geeignetes Robustheitsmaß, das in flexiblen Job Shops die Robustheit hinsichtlich der auftragsspezifischen Termintreue unter Berücksichtigung von Verfügbarkeiten der Stationen ermöglicht, existiert heute nicht.

Im reaktiven Rescheduling wird Robustheit durch die Berücksichtigung des Zustandes des Produktionssystems erreicht. Dies kann zu langen Umplanungskaskaden führen, bis der ursprüngliche Produktionsplan wiederhergestellt ist. Es sind Effizienzeinbußen zu erwarten, da hier zur schnellen Entscheidungsfindung Heuristiken verwendet werden und sich während des Reschedulings potenziell größere Abweichungen zum theoretischen Optimum ergeben. Die Möglichkeiten des reaktiven Reschedulings zur Steigerung der Robustheit werden bisher nicht genutzt. Das wesentliche Defizit reaktiver Ansätze für das Rescheduling mittels RL besteht darin, dass in bisherigen Arbeiten lediglich Benchmarkprobleme gelöst wurden, die in der Matrix-Produktion relevante Nebenbedingungen wie Transportstrecken, Rüstzeiten und Pufferkapazitäten nicht abbilden können.

Es gibt bisher keine Erkenntnisse darüber, wie die Robustheit im prädiktiven Scheduling das reaktive Rescheduling beeinflusst. Die Robustheit der prädiktiven Planung ist so zu wählen, dass in der Kopplung mit dem reaktiven Rescheduling insgesamt eine hohe Robustheit erreicht wird. Dies kann nur in Abhängigkeit der stochastischen Störungen erfolgen. Ein Ansatz zur Bestimmung der optimalen Robustheit im prädiktiven Scheduling zur Minimierung der zu erwartenden Rescheduling-Aufwände existiert ebenfalls bisher nicht.

4 Methode zum prädiktiv-reaktiven Scheduling

In diesem Kapitel wird die entwickelte Methode zur prädiktiv-reaktiven Scheduling vorgestellt. Zunächst werden in Kapitel 4.1 die Anforderungen an die Methode beschrieben sowie die wesentlichen zugrundeliegenden Annahmen formuliert, bevor in Kapitel 4.2 ein Überblick über die prinzipielle Funktionsweise der entwickelten Methode gegeben wird. Anschließend werden die einzelnen Bestandteile der Methode beschrieben, beginnend bei dem Modell für das prädiktive robuste Scheduling (Kapitel 4.3), über die Ermittlung von Rescheduling-Korridoren (Kapitel 4.4) bis hin zum Modell für das reaktive Rescheduling (Kapitel 4.5). Teile der entwickelten Methode basieren auf den vom Autor angeleiteten studentischen Abschlussarbeiten (A_Neupert 2018; A_Zhang 2018; A_Zöllner 2019; A_Hort 2019).

4.1 Anforderungen und Annahmen der zu entwickelnden Methode

An das Forschungsdefizit anknüpfend ist das Ziel der vorliegenden Arbeit, eine Methode zum prädiktiv-reaktiven Scheduling zu entwickeln, die neben einer hohen Termintreue insbesondere eine hohe Robustheit gegenüber Störungen an Stationen ermöglicht. Dies soll erreicht werden, indem die Verfügbarkeit von Stationen bereits beim Scheduling berücksichtigt wird und bei signifikanten Störungen ein Rescheduling mit möglichst kurzer Rechenzeit angestoßen wird.

Die Methode soll **typische Charakteristika von Produktionssystemen und Produktionsprogrammen berücksichtigen** können, um eine möglichst breite Anwendbarkeit für unterschiedlichste Anwendungsfälle zu erlauben. Aus Sicht des Scheduling kann eine Matrix-Produktion angelehnt an Kapitel 2.2.3 als flexibler Job Shop (FJc) mit c Stationen interpretiert werden, die teilweise redundant sein können. Hinsichtlich der Struktur des Produktionssystems müssen daher die durch das Layout und die verwendeten Transportmittel determinierten Transportzeiten $t_{s,s'}$ zwischen den Stationen sowie eventuelle Anforderungszeiten für Transportmittel rt_s berücksichtigt werden. Außerdem soll die Größe der physischen Puffer b_s für Halbzeuge an den Stationen berücksichtigt werden. Der Zugriff auf die Pufferplätze erfolgt wahlfrei, es besteht also kein Zwang zur Einhaltung des FIFO-Prinzips. Mögliche Verklemmungen der Puffer sind daher zu berücksichtigen (verkürzt als $block$ dargestellt). Hinsichtlich des Produktionsprogramms sollen alternative Stationen berücksichtigt werden können. Außerdem soll eine Rezirkulation von Produkten im Produktionssystem erlaubt sein, wodurch eine

Mehrfachverwendung von Stationen für ein Produkt erlaubt wird (verkürzt als $rcrc$ dargestellt). Zusätzlich sollen deterministische reihenfolgeabhängige Rüstzeiten (verkürzt als $r_{s,j,j}$, dargestellt) berücksichtigt werden, die produkt- und stationsspezifisch definiert werden können und damit hohe Freiheitsgrade bei der Modellierung bieten. Die Verfügbarkeiten von Stationen sind bekannt und Störungen an Stationen können daher auftreten (verkürzt als $brkdwn$ dargestellt).

Das prädiktive robuste Scheduling soll zudem folgende Anforderungen erfüllen:

- Es soll ein **näherungsweise optimaler Schedule** erzeugt werden können.
- Die **Performance** des Schedules soll durch die **Liefertreue** bestimmt werden. Damit wird eine Just-In-Sequence (JIS) Produktion ermöglicht. Die Operationalisierung soll durch eine Minimierung der durchschnittlichen Verspätung (Tardiness) der Aufträge $\frac{1}{j}\sum T_j$ erfolgen (Pinedo 2016). Eine verfrühte Fertigstellung soll zugelassen werden, da sonst keine Freiheitsgrade zur Erlangung von Robustheit existieren.
- Es soll eine **zentralisierte und damit vollständige Informationsbasis** für das Scheduling verwendet werden.
- Die **Verfügbarkeit** der Stationen im Produktionssystem soll beim Scheduling **berücksichtigt** werden (Eberlin & Hock 2014).
- Das **Ausmaß des zeitlichen Schlupfes** soll **variiert** werden können, um spezifisch für jedes Produktionssystem ein optimales Verhältnis von Robustheit durch prädiktives robustes Scheduling und reaktives Rescheduling ermitteln zu können.

Damit lässt sich das vorliegende Problem wie folgt charakterisieren:

$$FJc \mid Block, Rcrc, r_{s,j,j'}, Brkdwn \mid \frac{1}{j}\sum T_j$$

Das reaktive Rescheduling soll außerdem folgende Anforderungen erfüllen:

- Es soll in **kurzer Zeit** ein Rescheduling durchgeführt werden können. Dazu soll ein heuristisches Vorgehen angewendet werden. Es wird damit der Einsicht gefolgt, dass beim Rescheduling eine zulässige Lösung mit kurzer Rechenzeit vor-

teilhaft gegenüber einer exakten Lösung mit langer Rechenzeit ist (Engell & Märkert et al. 2001; van Brackel 2008; Niehues 2016). In wie kurzer Zeit ein Rescheduling möglich ist, wird in Kapitel 4.5 ermittelt.

- Das Rescheduling soll einen **Match-Up-Ansatz** verfolgen mit dem Ziel, nur einen möglichst kleinen Ausschnitt des ursprünglichen Schedules um zu planen, sodass eine möglichst hohe Planungssicherheit hinsichtlich der Liefertreue erhalten bleibt. Gleichzeitig wird dabei die Rechenzeit beschleunigt, da die Problemgröße minimal gehalten wird.
- Die **Performance** des Rescheduling soll durch die **Makespan** des umgeplanten Teils des prädiktiven robusten Schedules bestimmt werden. Dadurch wird sichergestellt, dass das Rescheduling für einen möglichst kurzen Zeitraum durchgeführt und damit der Match-Up-Horizont minimiert wird.
- Es soll ein **dezentrales Lösungsverfahren** für das Rescheduling verwendet werden. Damit soll die Wirkungsweise einer autonomen Produktionssteuerung untersucht werden können und insbesondere der Lösungsgüte einem zentralen Scheduling vergleichend gegenübergestellt werden können.

Die der Methode zugrundeliegenden Randbedingungen und vereinfachenden Annahmen lauten:

- Die Methode kann **ausschließlich für das Scheduling und Rescheduling** genutzt werden. Andere Aufgaben der Eigenfertigungsplanung und -steuerung können damit nicht durchgeführt werden.
- Die **Kapazität** des Produktionssystems wird als **fix** angenommen. Eine **Kapazitätssteuerung** für einen Ausgleich von Produktionsrückständen wird daher nicht betrachtet.
- Es wird **von Aufträgen mit Losgröße 1** ausgegangen. Eine Losfertigung lässt sich ohne Anpassung der Methode abbilden, indem die Bearbeitungszeiten aller enthaltenen Aufträge über jede Station kumuliert werden. Die Dimension des verwendeten Transportmittels wird dabei jedoch nicht betrachtet.
- Während der Planungsperiode ist das **Produktionsprogramm fix**. Es wird davon ausgegangen, dass keine neuen Aufträge eingesteuert werden.

- Es werden **ausschließlich Störungen an Stationen** als **ressourcenbezogene Störungen** betrachtet. Wie in Kapitel 2.2.2 beschrieben, lassen sich die meisten Störungen als Störung an einer Station darstellen.
- **Flexibilität hinsichtlich der Bearbeitungsreihenfolge der Aufträge** wird nur innerhalb gleicher Stationstypen berücksichtigt. Eine gesonderte Betrachtung flexibler (Teil-)Reihenfolgen unterschiedlicher Stationstypen wird nicht durchgeführt.
- Es erfolgt **keine Berücksichtigung der Anzahl verfügbarer Transportmittel**. Es wird davon ausgegangen, dass zu jedem Zeitpunkt eine ausreichende Anzahl von Transportmitteln zur Verfügung steht.
- Es wird angenommen, dass die **Dauer einer Störung zum Eintrittszeitpunkt abgeschätzt werden kann**, der Eintrittszeitpunkt selbst aber unbekannt ist. Zudem wird angenommen, dass die Eintrittswahrscheinlichkeit einer Störung unabhängig davon ist, ob eine Station einen Auftrag bearbeitet oder nicht, d. h. **Störungen können jederzeit auftreten und sind stochastisch verteilt**.
- Beim Rescheduling sollen **keine geplanten Preemptions zugelassen** werden, d. h. dass an einer ungestörten Station keine laufende Bearbeitung unterbrochen wird, um den Auftrag einer anderen Station zuzuweisen.
- Es wird **keine Nacharbeit beim Auftreten störungsbedingter Preemptions** betrachtet, d. h. dass eine Störung an einer Station während einer laufenden Bearbeitung zu Ausschuss führt, der ausgeschleust wird. Es wird davon ausgegangen, dass der zeitliche Aufwand für das Ausschleusen kleiner als die Störungsdauer ist und daher keine separate Berücksichtigung erforderlich ist.
- Der **initiale Rüstzustand** jeder Station entspricht dem zuerst eingeplanten Auftrag. Es werden also keine Rüstaufwände vor Produktionsbeginn berücksichtigt.
- Die **Verfügbarkeit von Material, Werkzeugen und Personal** wird nicht berücksichtigt

4.2 Überblick über den Lösungsansatz

Das vorrangige Ziel beim Scheduling in der Matrix-Produktion ist die Maximierung der Termintreue. Diese soll auch dann möglichst hoch sein, wenn Störungen im Produktionssystem auftreten. Es besteht also die Anforderung, möglichst robust eine hohe Termintreue zu erreichen.

Dazu wird eine prädiktiv-reaktive Strategie gewählt, deren Aufbau und Zusammenwirken in Abbildung 4-1 dargestellt ist. Der Lösungsansatz berücksichtigt Störungen an Stationen. Da die Dynamik ausschließlich durch Störungen an Stationen zu bestimmten Zeitpunkten entsteht, wird eine ereignisbasierte Rescheduling-Politik gewählt. Mithilfe eines Match-Up-Verfahrens wird ein Rescheduling für einen begrenzten Zeithorizont durchgeführt. Ist dies nicht möglich, wird eine vollständige Neugenerierung durchgeführt.

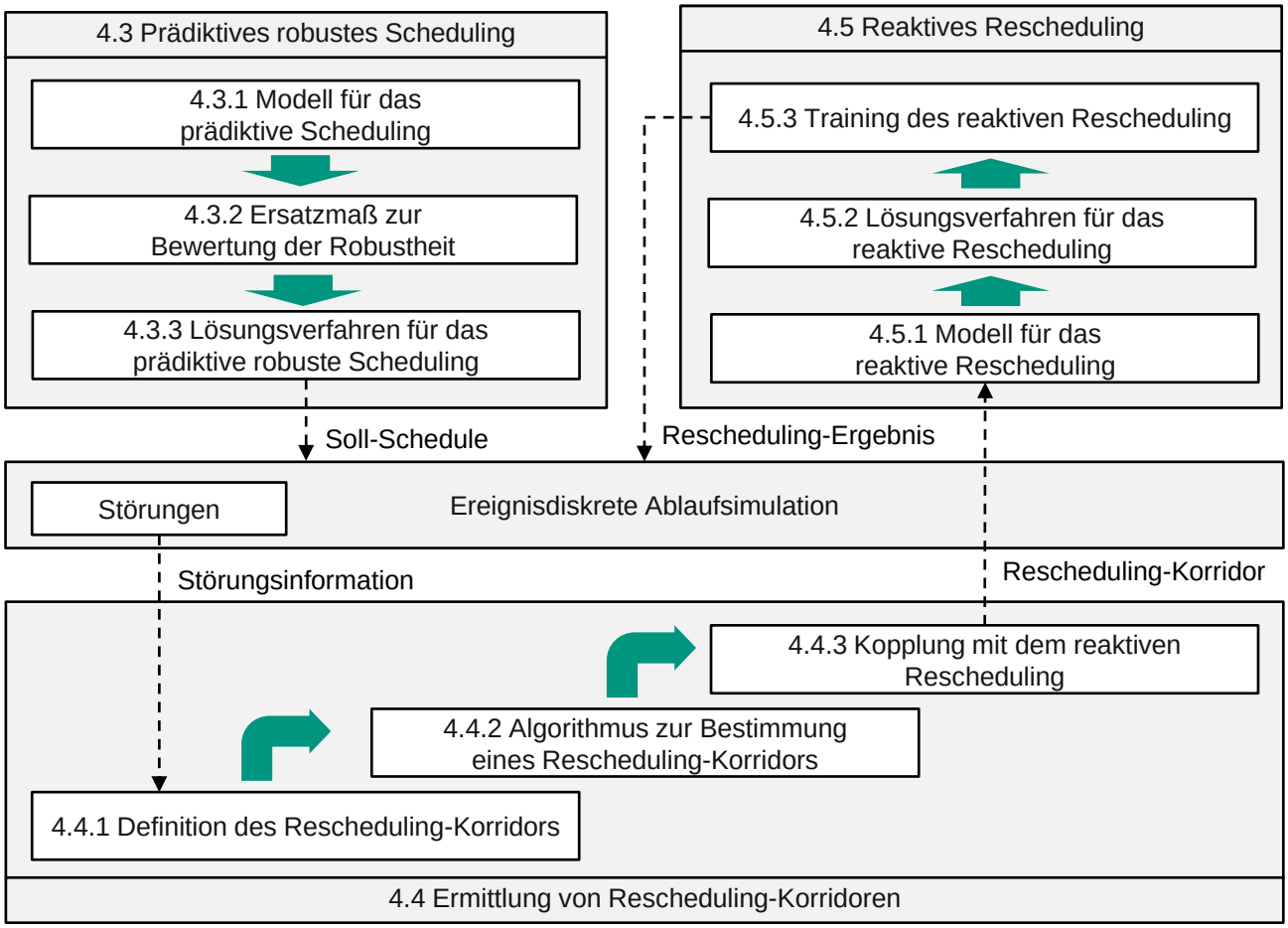


Abbildung 4-1: Überblick über den entwickelten Lösungsansatz

In der entwickelten Methode werden zur Erreichung einer hohen Robustheit zwei parallele Ansätze verfolgt:

Zum einen wird ein prädiktives robustes Scheduling durchgeführt, das die Verfügbarkeit von Stationen bereits während der Planung berücksichtigt. Dies wird erreicht, indem der Erwartungswert der Verfügbarkeitsverluste durch Einfügen zeitlichen Schlupfes zwischen Bearbeitungsschritten minimiert wird. Dadurch wird einerseits die Robustheit erhöht, andererseits kann die Liefertreue aber beeinträchtigt werden. Das dazu entwickelte Vorgehen wird in Kapitel 4.3 dargelegt.

Die Ermittlung von Rescheduling-Korridoren wird mithilfe einer Ablaufsimulation realisiert. Diese ist in der Lage, prädiktive robuste Schedules auszuführen und zufallsverteilte Störungen simulativ zu erzeugen. Beim Auftreten einer Störung muss zunächst auf Basis des robusten prädiktiven Schedules und der vorliegenden Störung die für das reaktive Rescheduling relevante Informationsbasis erzeugt werden. Dies erfolgt anhand des in Kapitel 4.4 beschriebenen Algorithmus. Zudem muss nach erfolgtem reaktivem Rescheduling der umgeplante Ausschnitt des Schedules wieder in den prädiktiven robusten Schedule integriert werden.

Im operativen Betrieb können auch Störungen auftreten, die länger als deren Erwartungswert andauern. In dem Fall wird ein reaktives Rescheduling durchgeführt, das eine möglichst schnelle Rückkehr zum prädiktiv erzeugten Schedule erlaubt. Hierfür wird ein Match-Up-Rescheduling verwendet, um möglichst nur den durch die Störung betroffenen Teil des Schedules umzuplanen. Die dazu entwickelte Methode unter Nutzung von Reinforcement Learning wird in Kapitel 4.5 beschrieben.

Im Vorgehen können anwendungsfallspezifisch unterschiedliche Grade der Robustheit im prädiktiven robusten Scheduling verwendet werden, um deren optimalen Wert zu ermitteln. Eine zu geringe Robustheitsanforderung im prädiktiven robusten Scheduling führt tendenziell zu längeren Zeiträumen für das Rescheduling, da der vorhandene Schlupf im Schedule zur Kompensation von Störungen nicht ausreicht. Aufgrund der Fokussierung auf den minimalen Makespan im Rescheduling kann sich dadurch die Termintreue verschlechtern. Umgekehrt führt eine zu hohe Robustheitsanforderung im prädiktiven robusten Scheduling tendenziell dazu, dass die geplante Fertigstellung von Aufträgen aufgrund der eingefügten Schlupfzeiten näher an ihren Liefertermin rückt. Eine später im Planungshorizont auftretende Häufung von Störungen kann dann zu

einer Gefährdung der Liefertreue führen. Der optimale Grad der Robustheit im prädiktiven robusten Scheduling hängt wesentlich vom Störungsprofil des Produktionssystems ab und wird exemplarisch im Anwendungsbeispiel des flexiblen Karosseriebaus in der Automobilindustrie in Kapitel 5.4 durchgeführt.

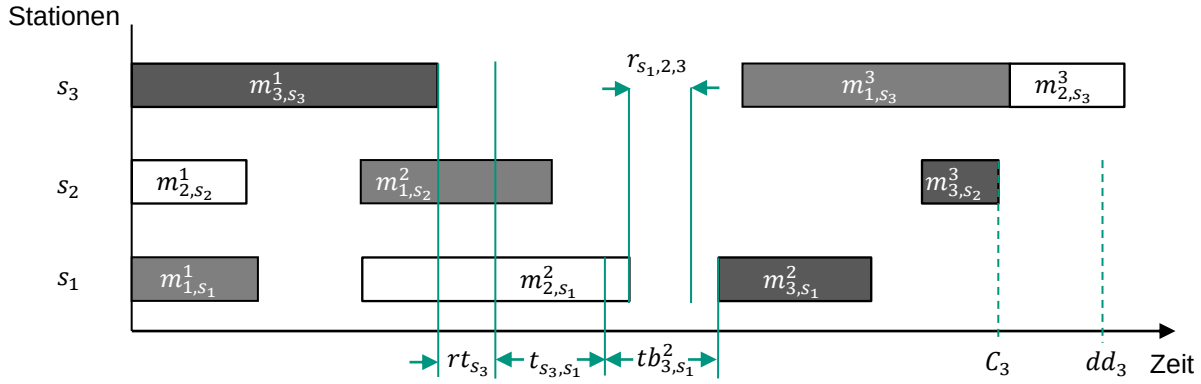
Aus Gründen der Übersichtlichkeit werden im Folgenden alle Variablen mit Bezug zu Auftrag, Station und Index der Bearbeitungsschritte ($o_j^{k_j}$, $m_{j,s}^{k_j}$, $p_{j,s}^{k_j}$, $x_{j,s}^{k_j}$ und $tb_{j,s}^{k_j}$) im Text verkürzt mit o, m, p, x und tb annotiert. Wesentlich für die weitere Arbeit ist die Unterscheidung zwischen Bearbeitungsschritten $o_j^{k_j}$ und Bearbeitungsalternativen $m_{j,s}^{k_j}$. Ein Bearbeitungsschritt beschreibt, welche Bearbeitungsschritte in welcher Reihenfolge für einen Auftrag j durchlaufen werden müssen. Bearbeitungsalternativen $m_{j,s}^{k_j}$ enthalten darüber hinaus eine Information darüber, welche Stationen s für die Durchführung eines Bearbeitungsschrittes $o_j^{k_j}$ infrage kommen. Während also jeder Bearbeitungsschritt eingeplant werden muss, gilt dies jeweils nur für genau eine der dazugehörigen Bearbeitungsalternativen. Die Unterscheidung wird getroffen, da in der Regel mehr als eine Station für die Durchführung eines Bearbeitungsschrittes verwendet werden kann. Der Startzeitpunkt einer eingeplanten Bearbeitungsalternative $m_{j,s}^{k_j}$ wird mit $x_{j,s}^{k_j}$ annotiert, während die zugehörige Prozesszeit durch $p_{j,s}^{k_j}$ beschrieben wird. Eine eventuelle Verweilzeit im Puffer der Station wird mit $tb_{j,s}^{k_j}$ annotiert.

Die wesentlichen Elemente mit zeitlichem Bezug sind beispielhaft in Abbildung 4-2 dargestellt. Die unterschiedlichen Farben stellen unterschiedliche Aufträge dar, wobei jeder einzelne Balken einen an einer Station eingeplanten Bearbeitungsschritt des jeweiligen Auftrags repräsentiert. Es wird zur Vereinfachung davon ausgegangen, dass jeder Auftrag einen eigenen Auftragstyp hat und jeder Stationstyp genau einmal existiert. Für die Erklärung ist lediglich der Durchlauf von Auftrag 3 (zuerst an s_3, nachfolgend an s_1 und abschließend an s_2) relevant.

Für den Transport von Auftrag 3 von s_3 an s_1 wird zunächst eine Anforderungszeit für Transportmittel rt_{s_3} sowie anschließend eine Transportzeit t_{s_3,s_1} benötigt. Danach verbleibt der Auftrag noch für eine Zeitdauer tb_{3,s_1}^2 im Puffer der Station, bevor der nächste Bearbeitungsschritt beginnt. Nachdem Auftrag 2 an s_1 bearbeitet wurde, kann die Station auf den nachfolgenden Auftrag 3 umgerüstet werden. Dazu fällt eine Rüstzeit in

Höhe von $r_{s_1,2,3}$ an. Dies kann parallel zur Wartezeit im Puffer tb^2_{3,s_1} durchgeführt werden. Mit dem sich ergebenden geplanten Fertigstellungszeitpunkt C_3 sowie dem extern vorgegebenen Fälligkeitstermin dd_3 lässt sich feststellen, dass Auftrag 3 keine Tardiness hat.

Es gelten zudem typische Annahmen im Scheduling, die nicht spezifisch für die vorliegende Arbeit getroffen wurden (Chaudhry & Khan 2016): Alle Stationen und Aufträge sind zu Beginn des Planungshorizontes verfügbar. Vorrangbeziehungen über Aufträge hinweg existieren nicht, d. h. es gibt keine Anforderung, bestimmte Aufträge zwingend vor anderen fertigzustellen.



Legende:

$m^{k_j}_{j,s}$	k_j-ter Bearbeitungsschritt von Auftrag j an Station s
rt_{s_3}	Anforderungszeit, die nach Bearbeitungsende eines Auftrags vergeht, bis ein Transportmittel an Station s_3 verfügbar ist
t_{s_3,s_1}	Transportzeit von Station s_3 zu Station s_1
tb^2_{3,s_1}	Wartezeit von m^2_{3,s_1} im Puffer von Station s_1
$r_{s_1,2,3}$	Rüstzeit an Station s_1 für die Umrüstung von Auftragstyp 2 auf Auftragstyp 3
C_3	Fertigstellungszeitpunkt von Auftrag 3
dd_3	Fälligkeitstermin von Auftrag 3

Abbildung 4-2: Zu berücksichtigende Elemente und Nebenbedingungen im prädiktiven robusten Scheduling

4.3 Prädiktives robustes Scheduling

Im Folgenden wird das Vorgehen für das robuste prädiktive Scheduling der Struktur aus Abbildung 4-3 folgend beschrieben.

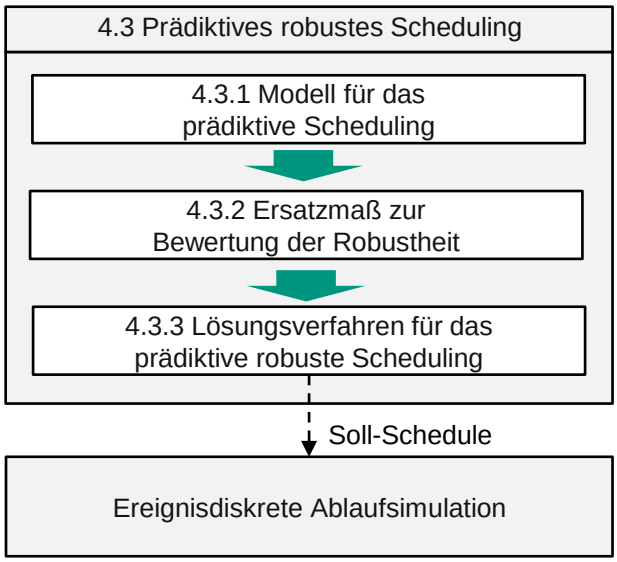


Abbildung 4-3: Struktur des Vorgehens zum prädiktiven robusten Scheduling

Für das prädiktive robuste Scheduling wird eine Dekomposition in ein prädiktives Scheduling und ein Post Processing gewählt, da die Anzahl nötiger Restriktionen zur Berücksichtigung von Robustheit in einem integrierten Modell zu hoch ist, um eine zulässige Lösung in einer Rechenzeit unterhalb des Planungshorizontes zu erlangen.

Zunächst wird daher in Kapitel 4.3.1 ein Optimierungsmodell ohne Berücksichtigung von Robustheit beschrieben. Ziel des Optimierungsmodells ist die Erzeugung eines prädiktiven Schedules, der den minimalen zeitlichen Abstand zwischen Fertigstellungszeitpunkt und Fälligkeitstermin über alle Aufträge maximiert. Damit wird erreicht, dass alle Aufträge möglichst gleichmäßig vor ihrem Fälligkeitstermin beendet werden, damit das Einfügen von Schlupfzeiten nicht direkt für einige Aufträge zu Verspätungen führt. Das Optimierungsmodell basiert auf den vom Autor angeleiteten Abschlussarbeiten (A_Neupert 2018; A_Zhang 2018).

In Kapitel 4.3.2 wird ein Ersatzmaß zur Messung der Robustheit beschrieben. Dabei werden anhand des prädiktiven Schedules und der Verfügbarkeit der Stationen die erwarteten Verzögerungen von Bearbeitungsschritten sowie deren Auswirkung auf den Schedule beschrieben sowie aufgezeigt, wie durch Einfügen von Schlupfzeiten die Auswirkungen von Störungen auf den Schedule minimiert werden können.

Abschließend wird in Kapitel 4.3.3 ein Post Processing durchgeführt, mit dem die geforderte Robustheit des Schedules nachträglich eingestellt werden kann. Ziel ist es dabei, in den Schedule Schlupfzeiten so einzufügen, dass die erwarteten Verzögerungen

auf ein definierbares Maß verringert werden. Dies kann zulasten der Termintreue gehen, daher existiert ein Trade-Off zwischen Robustheit und Termintreue. Das Vorgehen für das Post Processing basiert auf der vom Autor angeleiteten Abschlussarbeit (A_Zhang 2018).

4.3.1 Modell für das prädiktive Scheduling

Zunächst wird ein prädiktiver Schedule ohne explizite Berücksichtigung der Robustheit erzeugt und anschließend die geforderte Robustheit durch Einfügen von Schlupfzeiten in den Schedule eingebracht. Für die Modellierung des prädiktiven Scheduling bieten sich drei Vorgehensweisen an: Das Modell kann als lineares Problem (LP), als Constraint-Problem (CP) oder als eine Kombination davon formuliert werden. Numerische Untersuchungen haben gezeigt, dass die Kombination von LP und CP vorteilhaft bezüglich der erreichbaren Lösungsgüte und der benötigten Rechenzeit ist (van Hoeve & Laborie 2018). Daher wird im Folgenden das Optimierungsproblem als Kombination aus LP und CP formuliert. Zur Modellierung wird die Optimization Programming Language (OPL) verwendet[7].

4.3.1.1 Modellierung der Variablen

Das Optimierungsziel ist es, den minimalen zeitlichen Abstand zwischen Fertigstellungszeitpunkt und Fälligkeitstermin über alle Aufträge hinweg zu maximieren, um gleichmäßige Freiräume für später einzufügende Schlupfzeiten zu schaffen. Auf einen einzelnen Auftrag j bezogen kann die Schlupfzeit SZ_j mit Formel 4-1 beschrieben werden (A_Neupert 2018). SZ_j stellt damit die negierte Tardiness dar, da nur die verfrühte Fertigstellung eines Auftrags Schlupfzeiten ausweist.

$$SZ_j = max\{(dd_j - C_j), 0\} \qquad 4\text{-}1$$

Zusätzlich wird je Auftrag noch eine Gewichtung eingeführt, die den voraussichtlichen Bedarf an Schlupfzeiten berücksichtigen soll. Aufträge, die auf Stationen mit sehr hoher Verfügbarkeit eingeplant werden, haben voraussichtlich einen geringeren Bedarf an

[7] IBM (2017), *IBM ILOG CPLEX Optimization Studio OPL Language Reference Manual*., https://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.0/ilog.odms.studio.help/pdf/opl_languser.pdf [08.01.2020].

Schlupfzeiten als Aufträge, die auf Stationen mit geringer Verfügbarkeit eingeplant werden. Gleichzeitig haben Aufträge mit langen Prozesszeiten tendenziell mehr Bedarf an Schlupfzeiten als Aufträge mit kurzen Prozesszeiten. Um diesem Umstand Rechnung zu tragen, wird ein auftragsspezifischer Gewichtungsfaktor w_j eingeführt, der sich nach Formel 4-2 aus dem Produkt der Verfügbarkeiten $\sum_{\forall m_{j,s}^{k_j}} v_s$ aller entlang seiner Produktion verwendeten Stationen für alle Bearbeitungsschritte $o_{j,s}^{k_j}$ ergibt (A_Zhang 2018). Die Gewichtung entspricht daher der Verfügbarkeit einer Schaltung, bei der zwangsläufig nacheinander stattfindende Bearbeitungsalternativen seriell und die dazugehörigen identischen Stationen parallel geschaltet sind (Eberlin & Hock 2014).

$$w_j = \prod_{\forall o_{j,s}^{k_j}} \left(1 - \sum_{\forall m_{j,s}^{k_j}} (1 - v_s) \right) \qquad 4\text{-}2$$

Im Optimierungsmodell kommen die Intervallvariablen IWP, $IOWP$, IP, IOP, IT und IOT zum Einsatz, um die zeitliche Abfolge von Aktivitäten im Schedule zu charakterisieren. Ein **Intervall** ist ein Datentyp in der Constraintprogrammierung, der zur Darstellung von Aktivitäten mit Zeitbezug eingesetzt wird. Jedes Intervall wird im Rahmen der Arbeit charakterisiert über die Attribute Start, Ende und Größe (Rossi & van Beek et al. 2006). Die Größe entspricht im Scheduling der Zeitdauer einer Aktivität (z. B. Bearbeiten, Rüsten, Warten, Transportieren). Start und Ende sind meist nicht vordefiniert, sondern mithilfe des Optimierungsmodells zu ermitteln. Im vorliegenden Optimierungsmodell beschreiben IWP und $IOWP$ die Wartezeit im Puffer von Bearbeitungsschritten bzw. Bearbeitungsalternativen, IP und IOP repräsentieren deren Prozesszeit und IT sowie IOT deren summierte Anforderungs- und Transportzeiten.

Eine **Sequenz** ist ein Datentyp in der Constraintprogrammierung, der die Reihenfolge von Intervallen abbildet (Rossi & van Beek et al. 2006). Sequenzvariablen werden verwendet, um Überlappungen von Intervallen durch zusätzliche Nebenbedingungen auszuschließen. Dazu werden im Optimierungsmodell Sequenzvariablen ψ_s für alle Stationen $s \in S$ eingeführt.

4.3.1.2 Optimierungsmodell

Im Folgenden wird das mathematische Modell zur Ermittlung eines Schedules beschrieben, welches die Zielfunktion 4-3 sowie die Nebenbedingungen 4-4 bis 4-15 umfasst.

$$\max \min_{\forall j \in J} (SZ_j * w_j) \tag{4-3}$$

Die Zielfunktion 4-3 maximiert den minimalen gewichteten Schlupf je Auftrag $ST_j * w_j$. Damit wird erreicht, dass alle Aufträge möglichst gleichmäßig vor Erreichen ihres Fälligkeitstermins dd_j fertig werden. Die Gewichtung mit w_j sorgt dafür, dass der Schlupf von Aufträgen an Stationen mit geringer Verfügbarkeit geringer gewichtet wird als der Schlupf von Aufträgen an Stationen mit hoher Verfügbarkeit und damit mehr Schlupfzeiten bei tendenziell störungsanfälligeren Aufträgen vorgesehen werden.

$$\begin{gathered} alternative\left(o, all\left(m_{j,s}^{k_j} \in M_j^{k_j}\right)\right) = IOP_{m_{j,s}^{k_j}}, \\ \forall o \in O;\ j \in J;\ k_j\ \in K_j \end{gathered} \tag{4-4}$$

Die Nebenbedingung 4-4 stellt mithilfe des Operators $alternative(\cdot)$ sicher, dass für jeden Bearbeitungsschritt o jedes Auftrags genau eine dazugehörige Bearbeitungsalternative m in die Lösung aufgenommen wird. Die jeweils gültigen Bearbeitungsalternativen werden über den Operator $all(\cdot)$ aus der Menge M_j^k identifiziert.

$$\begin{gathered} noOverlap\left(\psi_s, r_{s,jt_j,jt_{j'}}\right), \\ \forall s \in S;\ j, j' \in J : j \neq j' \end{gathered} \tag{4-5}$$

Nebenbedingung 4-5 trägt über den Operator $noOverlap(\cdot)$ dafür Sorge, dass sich die Intervalle der Sequenzen ψ_s nicht mit den reihenfolgeabhängigen Rüstzeiten $r_{s,jt_j,jt_{j'}}$ überlappen.

$$\begin{gathered} endBeforeStart(IP_o, IT_o), \\ \forall j \in J;\ k_1, k_2 \in K_j;\ \forall o \in O_j;\ m_1, m_2 \in M_j^k | k_2 = k_1 + 1 \end{gathered} \tag{4-6}$$

$$\begin{gathered} endOf(IT_o) * presenceOf(m_1) * presenceOf(m_2) = \\ \left(endOf(IP_o) + rt_{s_{m_1}} + t_{s_{m_1}, s_{m_2}}\right) \\ * presenceOf(m_1) * presenceOf(m_2), \\ \forall j \in J;\ k_1, k_2 \in K_j;\ \forall o \in O_j;\ m_1, m_2 \in M_j^k | k_2 = k_1 + 1 \end{gathered} \tag{4-7}$$

$$startOf\left(IP_{o_2}\right) * presenceOf(m_1) * presenceOf(m_2) \geq \left(endOf\left(IP_{o_1}\right) + \max\left(rt_{s_{m_1}} + t_{s_{m_1},s_{m_2}}, r_{s_{m_1},jt_{m_1},jt_{m_2}}\right)\right) * presenceOf(m_1) * presenceOf(m_2), \quad \forall j \in J;\ k_1, k_2 \in K_j;\ \forall o_1, o_2 \in O_j;\ m_1, m_2 \in M_j^k | k_2 = k_1 + 1 \tag{4-8}$$

Die Nebenbedingungen 4-6 bis 4-8 sorgen für die Berücksichtigung von Anforderungszeiten und Transportzeiten im Scheduling. Betrachtet werden alle Paare von Bearbeitungsschritten und Bearbeitungsalternativen, die zum selben Auftrag gehören und zueinander Vorgänger bzw. Nachfolger darstellen.

Nebenbedingung 4-6 stellt über den Operator $endBeforeStart(\cdot)$ sicher, dass das Transportintervall IT eines Auftrags abgeschlossen ist, bevor dessen Bearbeitung beginnen kann. Die Gültigkeit für m und und aufeinanderfolgende Indizes k ist nötig, da die Nebenbedingung nur für aufeinanderfolgende Bearbeitungsalternativen desselben Auftrags gelten soll.

Nebenbedingung 4-7 legt das Ende des Transportintervall IT als die Summe aus Anforderungszeit rt und Transportzeit t fest. Dazu wird der Operator $endOf(\cdot)$ verwendet, der das Ende einer Intervallvariablen festlegt. Aufgrund der möglichen Bearbeitungsalternativen muss zudem sichergestellt werden, dass die Größe der Transportintervalle nur für die gewählten Bearbeitungsalternativen gesetzt wird. Hierzu wird der boolesche Operator $presenceOf(\cdot)$ verwendet, der den Wert 1 annimmt, wenn die entsprechende Bearbeitungsalternative gewählt wird.

Nebenbedingung 4-8 schließlich beschränkt, ab wann der nächste Bearbeitungsschritt eines Auftrags durchgeführt werden kann. Über den Operator $startOf(\cdot)$ wird der früheste Startzeitpunkt des nächsten Bearbeitungsschritts des nachfolgenden Auftrags so gesetzt, dass ausgehend vom Ende des vorhergehenden Bearbeitungsschritts über $endOf(\cdot)$ das Maximum der Summe aus Anforderungs- und Transportzeit sowie der reihenfolgeabhängigen Rüstzeit gewählt wird. Es ist damit nicht erforderlich, dass ein Auftrag direkt zum Ende seines Transports oder dem Rüsten der Station bearbeitet wird. Ein Verweilen im Puffer ist möglich.

$$startAtEnd\left(IWP_{o_2}, IT_{o_1}\right), \quad \forall o_1, o_2 \in O \tag{4-9}$$

$$endAtStart\left(IWP_{o_2}, IP_{o_2}\right), \quad \forall o_1, o_2 \in O \tag{4-10}$$

Da Bearbeitungsschritte nicht direkt nach Ende eines Transportintervalls beginnen müssen, wird ggf. der Puffer einer Station in Anspruch genommen. Das Zeitintervall der Pufferbelegung durch einen Bearbeitungsschritt wird durch das Intervall IWP beschrieben. Nebenbedingung 4-9 legt über den Operator $startAtEnd(\cdot)$ fest, dass eine Pufferbelegung IWP genau in dem Moment startet, in dem das vorangehende Transportintervall IT abgeschlossen ist. Nebenbedingung 4-10 stellt über den Operator $endAtStart(\cdot)$ sicher, dass die Pufferbelegung IWP wiederum genau in dem Moment endet, in dem das nächste Intervall eines Bearbeitungsschrittes IP beginnt.

$$alternative\left(IWP_o, all\left(o, all\left(m_{j,s}^{k_j} \in M_j^k\right)\right)\right) = IOWP_{m_{j,s}^{k_j}}, \quad \forall o \in O \tag{4-11}$$

$$alternative\left(IT_o, all\left(o, all\left(m_{j,s}^{k_j} \in M_j^k\right)\right)\right) = IOT_{m_{j,s}^{k_j}}, \quad \forall o \in O \tag{4-12}$$

Die Intervalle IWP legen die Pufferbelegung eines Bearbeitungsschrittes fest. Allerdings besteht auf der Ebene noch kein Bezug zu den gewählten Bearbeitungsalternativen und damit keine Möglichkeit, die Pufferbelegung der korrekten Station zuzuordnen. Nebenbedingung 4-11 stellt sicher, dass zu jedem Intervall IWP das korrekte Intervall $IOWP$ abhängig von der gewählten Bearbeitungsalternative beansprucht wird. Nebenbedingung 4-12 stellt ebenfalls abhängig von der gewählten Bearbeitungsalternative sicher, dass zu jedem Intervall IT das korrekte Transportintervall IOT gesetzt wird.

$$presenceOf(m) = presenceOf(IOWP_m), \quad \forall m \in M \tag{4-13}$$

$$presenceOf(m) = presenceOf(IOT_m), \quad \forall m \in M \tag{4-14}$$

Analog dazu stellen Nebenbedingungen 4-13 und 4-14 sicher, dass die Intervalle $IOWP$ bzw. IOT genau dann vorhanden sind, wenn die dazugehörigen Bearbeitungsalternativen gewählt werden.

$$\sum_{\forall m \in M} pulse(IOWP_m) \leq b_s, \quad \forall s \in S \tag{4-15}$$

Schließlich wird mit Nebenbedingung 4-15 sichergestellt, dass die Pufferkapazität b_s an keiner Station zu keinem Zeitpunkt überschritten wird. Dazu wird die Funktion $pulse(\cdot)$ verwendet, deren Wert bei jedem beginnenden bzw. endenden Intervall $IOWP$ um das Inkrement 1 erhöht bzw. verringert wird und damit den aktuellen Füllstand des Puffers widerspiegelt.

Mithilfe eines Solvers für CP-Modelle kann mit dem beschriebenen Optimierungsmodell nun ein prädiktiver Schedule $Sched_{Präd}$ erzeugt werden, der angibt, wann welcher Auftrag an welcher Station eingeplant werden soll, sodass der minimale gewichtete Schlupf maximal ist und gleichzeitig alle in Kapitel 4.2 formulierten typischen Charakteristika einer komplexen Produktion berücksichtigt sind.

4.3.2 Ersatzmaß zur Bewertung der Robustheit

Zur Bewertung der Robustheit wird zunächst ausgehend von $Sched_{Präd}$ die erwartete Verzögerung aller Bearbeitungsschritte ermittelt. Dabei sind deren Prozesszeiten und Einplanung über die Zeit sowie die Verfügbarkeit der Stationen relevant. Die Robustheit soll über einen Parameter eingestellt werden können, der angibt, um welchen Anteil die erwarteten Verzögerungen in $Sched_P$ verringert werden sollen. Um $Sched_{Präd}$ gemäß diesem Parameter robust zu machen, werden Schlupfzeiten berechnet, die nach jedem Bearbeitungsschritt eingefügt werden. Es entsteht letztlich der robuste prädiktive Schedule $Sched_{Rob}$, der über die gewünschte Robustheit verfügt.

Die erwartete Verzögerung ed einer eingeplanten Bearbeitungsalternative m kann nach Formel 4-16 berechnet werden. Bei einer Verfügbarkeit von z. B. $v_s = \frac{4}{5} = 80\ \%$ beträgt die erwartete Verzögerung damit $\left(\frac{5}{4}\right) - 1 = 25\ \%$ der Bearbeitungszeit.

$$ed_{j,s}^{k_j} = \frac{p_{j,s}^{k_j}}{v_s} - p_{j,s}^{k_j}, \forall j \in J, s \in S, k_j \in K_j \tag{4-16}$$

In Abbildung 4-4 ist ein Beispiel dargestellt, in dem drei Aufträge an drei Stationen bearbeitet werden. Die Verfügbarkeiten der Stationen sind als $v_1 = 0{,}75$, $v_2 = 0{,}5$ und

$v_3 = 1$ gegeben. Die erwarteten Verzögerungen ed je Bearbeitungsschritt sind schraffiert dargestellt. Es wird aus den Überlappungen der erwarteten Verzögerungen mit nachfolgenden Bearbeitungsschritten ersichtlich, dass nicht nur ed selbst, sondern auch früher im Schedule entstandene Verzögerungen sich auf die zeitgerechte Durchführbarkeit auswirken. Dabei handelt es sich um die in Kapitel 3.2 eingeführten Störungskaskaden, die sich entlang einer Station (horizontal) oder eines Auftrags (vertikal) ausbreiten können. Man spricht dann auch von einer horizontalen bzw. vertikalen Verzögerung in Anlehnung an die Gantt-Notation, in der in den Spalten die Bearbeitungsschritte horizontal und in den Zeilen die Stationen vertikal abgetragen werden.

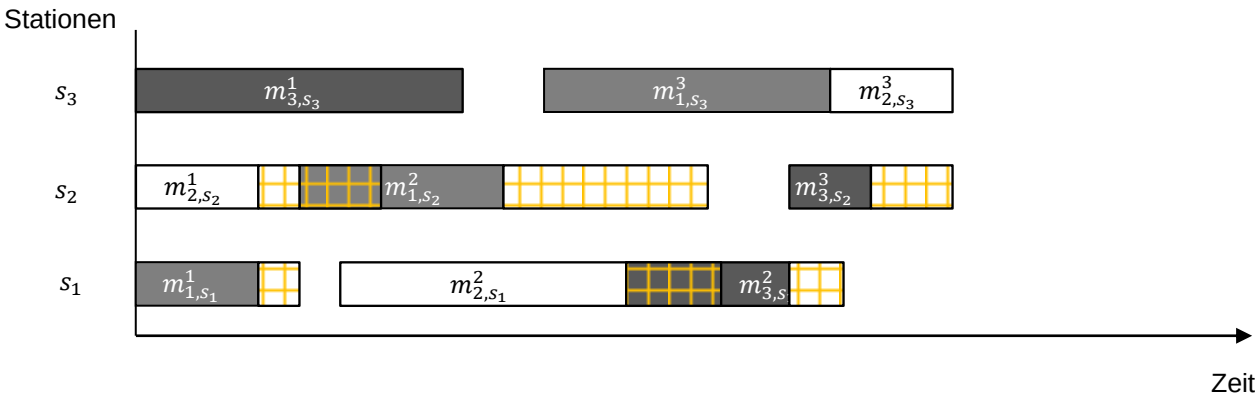


Abbildung 4-4: Nicht-robuster Schedule inkl. erwarteter Verzögerungen der Bearbeitungsschritte

Eine horizontale Störungskaskade entsteht, wenn die Verzögerung eines Bearbeitungsschrittes an einer Station dazu führt, dass sich mindestens ein an derselben Station durchzuführender nachfolgender Bearbeitungsschritt verzögert. Die horizontale Verzögerung eines Bearbeitungsschrittes hd durch eine horizontale Störungskaskade lässt sich rekursiv nach Formel 4-17 berechnen (Jamili 2016). Danach ergibt sich hd aus der gesamten erwarteten Verzögerung td des horizontal vorhergehenden Bearbeitungsschrittes an derselben Station abzüglich der ggf. bereits vorhandenen horizontalen Schlupfzeit vor dem Bearbeitungsschritt.

$$hd^{k_j}_{j,s} = td^{k'_j}_{j',s} - (x^{k_j}_{j,s} - x^{k'_j}_{j,s'} + p^{k'_j}_{j,s'} + r_{s,jt_{j'},jt_j}),$$
$$\forall j, j' \in J;\ s, s' \in S;\ k_j, k'_j \in K_j; jt_j \in JT \qquad \text{4-17}$$

Eine vertikale Störungskaskade entsteht, wenn die Verzögerung eines vorgelagerten Bearbeitungsschrittes an einer Station dazu führt, dass sich mindestens ein an einer

anderen Station durchzuführender nachfolgender Bearbeitungsschritt desselben Auftrags verzögert. Die vertikale Verzögerung vd eines Bearbeitungsschrittes durch eine vertikale Störungskaskade lässt sich rekursiv nach Formel 4-18 berechnen (Jamili 2016). Danach ergibt sich vd aus der gesamten erwarteten Verzögerung td des vertikal vorhergehenden Bearbeitungsschrittes desselben Auftrags abzüglich der ggf. bereits vorhandenen vertikalen Schlupfzeit vor dem Bearbeitungsschritt. Ebenfalls berücksichtigt werden muss dabei die Anforderungszeit rt_s, die Transportzeit $t_{s,s'}$ sowie die reihenfolgeabhängige Rüstzeit $r_{s,jt_{j'},jt_j}$.

$$vd_{j,s}^{k_j} = td_{j',s}^{k_{j'}} - \left(x_{j,s}^{k_j} - x_{j,'s}^{k'_j} + p_{j,'s}^{k'_j} + \max\left(rt_s + t_{s,s'}, r_{s,jt_{j'},jt_j}\right)\right), \quad \forall j,j' \in J;\ s,s' \in S;\ k_j,k'_j \in K_j; jt_j \in JT \tag{4-18}$$

Auf einen Bearbeitungsschritt können horizontale und vertikale Störungskaskaden gleichzeitig wirken, wobei deren Maximum für die erwartete Verzögerung ausschlaggebend ist. Damit lassen sich für alle Bearbeitungsschritte die gesamten erwartete Verzögerungen td nach Formel 4-19 ableiten.

$$td_{j,s}^{k_j} = ed_{j,s}^{k_j} + \max\left(hd_{j,s}^{k_j}, vd_{j,s}^{k_j}\right), \forall j \in J;\ s,s' \in S;\ k_j \in K_j \tag{4-19}$$

Die horizontalen und vertikalen Verzögerungen hd und vd beziehen die Störungskaskaden vorgelagerter Bearbeitungsschritte ein. Daraus folgt direkt, dass für den ersten eingeplanten Bearbeitungsschritt an jeder Station gilt: $td = ed$. Am Beispiel der horizontalen Überlappung von m_{1,s_2}^2 mit der erwarteten Verzögerung von m_{2,s_2}^1 in Abbildung 4-4 ist ersichtlich, dass eine horizontale Störungskaskade auf s_2 vorliegt. Vertikale Überlappungen sind nicht direkt ersichtlich, so z. B. die vertikale Überlappung von m_{3,s_2}^3 mit der erwarteten Verzögerung von m_{3,s_1}^2.

Grundidee des robusten Schedulings ist es nun, in den Schedule so Schlupfzeiten einzufügen, dass td auf einen Maximalwert begrenzt wird. Dazu führt Jamili (2016) den Robustheitsparameter λ ein. Die Schlupfzeiten müssen nun so eingefügt werden, dass gilt $\lambda \leq td_{j,s}^{k_j}$. Allerdings ist eine pauschale Festlegung von λ nicht hilfreich, wenn auftragsindividuelle Liefertermine d_j vorliegen und die Robustheit hinsichtlich der Liefertreue erreicht werden soll. Daher wird im Folgenden λ spezifisch je Bearbeitungsschritt ermittelt, sodass Ungleichung 4-20 gelten muss.

$$\lambda_{j,s}^{k_j} \leq td_{j,s}^{k_j}, \forall j \in J; s, s' \in S; k_j \in K_j \qquad 4\text{-}20$$

Für die Wahl der $\lambda_{j,s}^{k_j}$ wird ein benutzerdefinierbarer Parameter $R \in [0,1]$ zur Einstellung der gewünschten Robustheit eingeführt. R gibt an, um welchen Anteil der erwarteten gesamten Verzögerung je Bearbeitungsschritt td durch Schlupfzeiten sinken soll. Für $R = 0$ werden keine Schlupfzeiten eingefügt und der Schedule wird nicht verändert. Für $R = 1$ werden so vielen Schlupfzeiten eingeführt, dass td auf ed beschränkt wird. Sollten in der Realität also alle Störungen gemäß ihres Erwartungswertes eintreffen, würde sich für $R = 1$ jeder Bearbeitungsschritt zu seinem geplanten Beginn durchführen lassen. Werte von $R \geq 1$ haben keinen Effekt, da alle td bereits für $R = 1$ minimal sind.

Die Schlupfzeiten werden eingefügt, indem alle Bearbeitungsschritte so weit auf der Zeitachse nach rechts verschoben werden, dass sie von keinen Störungskaskaden einer erwarteten Verzögerung unterliegen. Das Ergebnis ist in Abbildung 4-5 dargestellt. Eine Verringerung der erwarteten Verzögerung auf 0 ist nicht möglich, da ed durch Schlupfzeiten nicht verringert werden kann, sondern bei der Bearbeitung unvermeidbar ist. Daher muss λ mindestens immer so groß wie ed sein.

$$\lambda_{j,s}^{k_j} = \max\left(td_{j,s}^{k_j} * \max\big((1-R), 0\big), ed_{j,s}^{k_j}\right), \forall j \in J; s \in S; k_j \in K_j \qquad 4\text{-}21$$

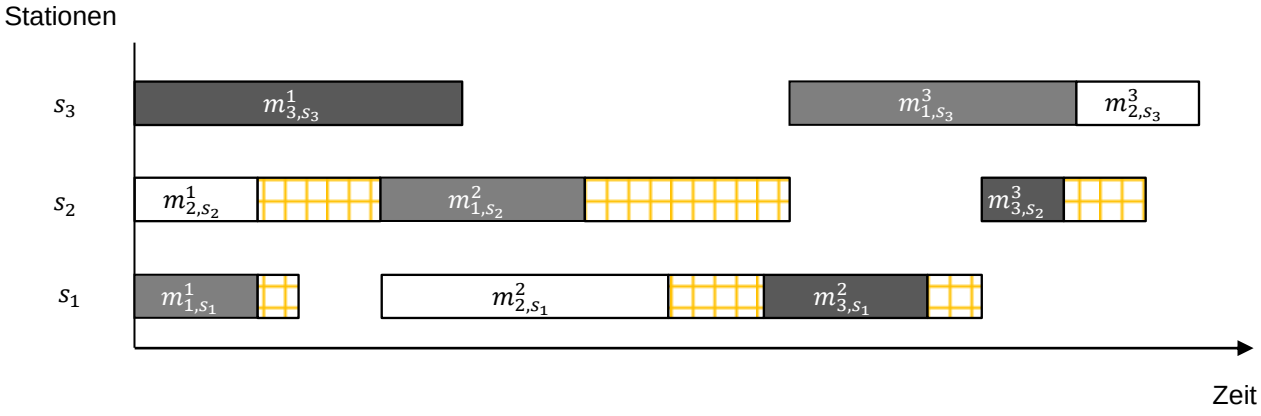


Abbildung 4-5: Robuster Schedule mit $R = 1$

4.3.3 Lösungsverfahren für das prädiktive robuste Scheduling

Der in Kapitel 4.3.1 erzeugte prädiktive Schedule $Sched_{Präd}$ soll nun im Rahmen eines Post Processing durch Einfügen geeigneter Schlupfzeiten an geeigneten Stellen im

Schedule auf die gewünschte Robustheit R gemäß des in Kapitel 4.3.2. beschriebenen Robustheitsmaßes eingestellt werden, sodass $Sched_{Rob}$ resultiert. Dazu wird der in Abbildung 4-6 dargestellte iterative Algorithmus durchlaufen, bis ausreichend Schlupfzeiten eingefügt und die Abbruchbedingung erfüllt ist. Im Wesentlichen besteht der Algorithmus aus zwei Teilen. Zunächst werden in den Schritten (1) - (3) die Berechnungen zum Einfügen der Schlupfzeiten in den Schedule durchgeführt. Anschließend werden in den Schritten (4) - (6) die notwendigen Operationen zur Wiederherstellung der Zulässigkeit des Schedules durchgeführt mit Bezug auf Überlappungen in horizontaler und vertikaler Richtung sowie hinsichtlich der Pufferkapazitäten an den Stationen.

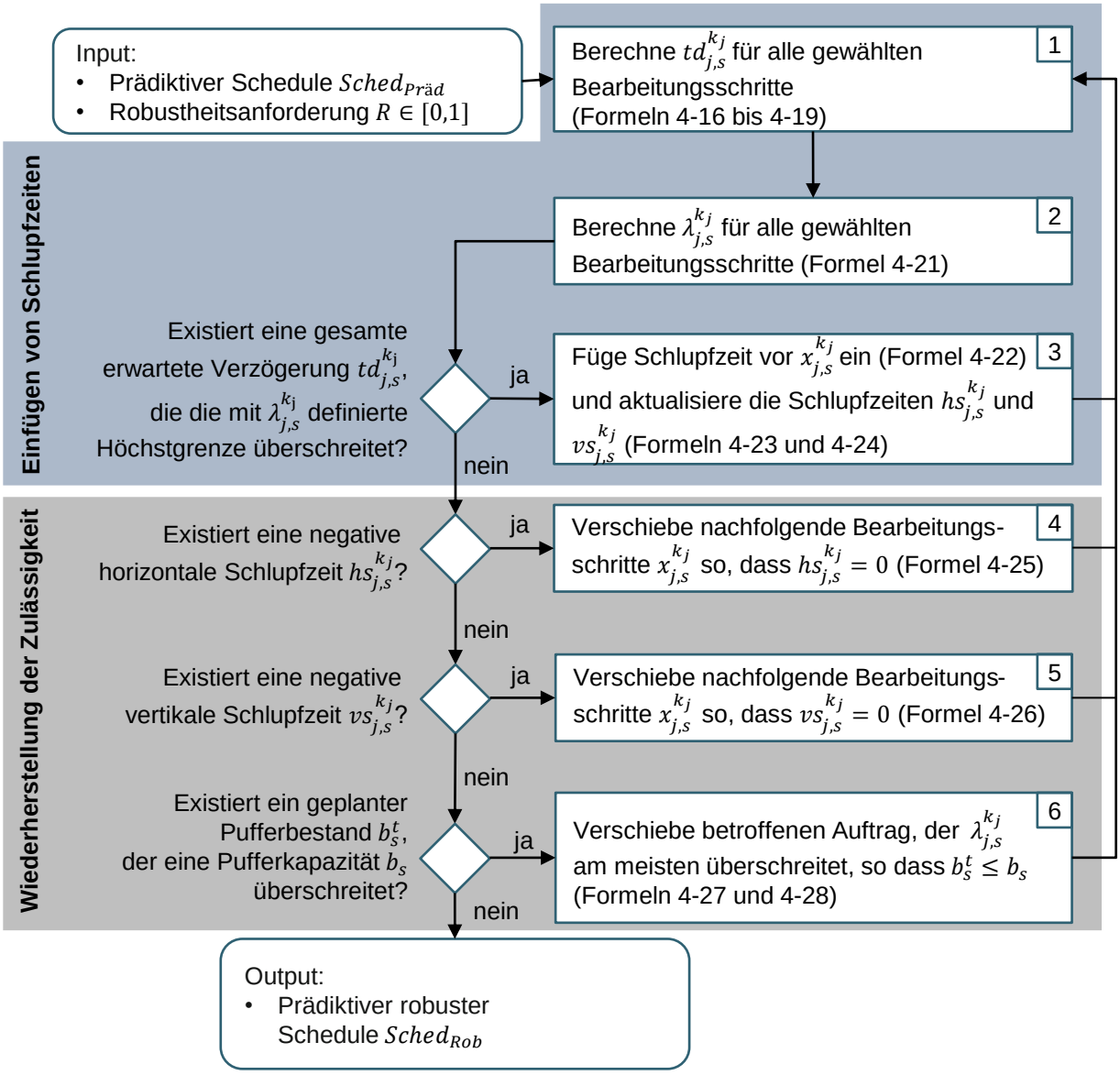


Abbildung 4-6: Iterativer Algorithmus zur Erzeugung eines zulässigen prädiktiven robusten Schedules

In Schritt (1) werden anhand der Formeln 4-16 bis 4-19 die gesamten erwartete Verzögerungen td für jeden Bearbeitungsschritt im prädiktiven Schedule berechnet. Anschließend werden in Schritt (2) die individuellen maximal zulässigen Verzögerungen im robusten Schedule λ nach Formel 4-21 bestimmt. Damit sind die Voraussetzungen zur Erzeugung eines robusten Schedules geschaffen.

In einer fortlaufenden Schleife werden nachfolgend alle Bearbeitungsschritte hinsichtlich von zwei Bedingungen überprüft: Zum einen wird geprüft, ob noch Bearbeitungsschritte existieren, deren gesamte erwartete Verzögerung td größer als die maximal zulässige Verzögerung λ ist, d. h. ob der Schedule seine gewünschte Robustheit erreicht hat. Zum anderen wird geprüft, ob durch die Verschiebungen der Bearbeitungsschritte ein unzulässiger Schedule entstanden ist, in dem sich Überlappungen zwischen Bearbeitungsschritten ergeben. Diese müssen innerhalb der Schleife behoben werden, um nach dem Algorithmus einen robusten und zulässigen Schedule zu erhalten. Das iterative Vorgehen ist notwendig, da das Einfügen von Schlupfzeiten an einer Stelle im Schedule Auswirkungen auf nachfolgende Störungskaskaden haben kann und damit später einzufügende Schlupfzeiten möglicherweise kleiner sein können.

In Schritt (3) wird für jeden Bearbeitungsschritt überprüft, ob die gesamte erwartete Verzögerung td die maximal zulässige Verzögerung λ übersteigt. Falls ja, wird mithilfe von Formel 4-22 der Startzeitpunkt des Bearbeitungsschritts angepasst. Der neue Startzeitpunkt x' ergibt sich, indem der alte Startzeitpunkt x um $td - \lambda$ in die Zukunft verschoben wird. Damit wird der Beitrag des Bearbeitungsschrittes zu nachfolgenden horizontalen und vertikalen Störungskaskaden auf den mithilfe von R festgelegten Anteil reduziert. Da eine Störungskaskade immer alle vorangegangenen Einflüsse berücksichtigt kann es sein, dass hierdurch keine nachfolgende Störungskaskade abgeschwächt wird, so lange noch weitere Verzögerungen existieren.

$$x'^{k_j}_{j,s} = x^{k_j}_{j,s} + td^{k_j}_{j,s} - \lambda^{k_j}_{j,s}, \forall j \in J; s, s' \in S; k_j \in K_j \qquad 4\text{-}22$$

Anschließend wird überprüft, ob $Sched_P$ durch das Einfügen von Schlupfzeit unzulässig geworden ist. Dazu werden zwei neue Größen hs und vs eingeführt, die die vorhandene horizontale bzw. vertikale Schlupfzeit der Bearbeitungsschritte darstellen. Die horizontale Schlupfzeit hs eines Bearbeitungsschrittes stellt die Zeit dar, die eine Station vor dessen Bearbeitungsbeginn unbeschäftigt ist. Sie kann nach Formel 4-23 berechnet

werden. Dabei sind $x_{j'',s}^{k_{j''}}$ und $p_{j'',s}^{k_{j''}}$ der Startzeitpunkt bzw. die Prozesszeit des an derselben Station s davor durchgeführten Bearbeitungsschrittes eines anderen Auftrags j'' und $r_{s,jt_{j''},jt_j}$ die reihenfolgeabhängige Rüstzeit an Station s von Auftrag j'' auf Auftrag j. Die vertikale Schlupfzeit vs eines Bearbeitungsschrittes stellt die Zeit dar, die ein Auftrag vor seiner Bearbeitung im Puffer einer Station verweilt. Diese kann nach Formel 4-24 berechnet werden. Dabei sind $x_{j,s''}^{k_j-1}$ und $p_{j,s''}^{k_j-1}$ der Startzeitpunkt bzw. die Prozesszeit des an einer anderen Station s'' davor durchgeführten Bearbeitungsschrittes desselben Auftrags j.

$$hs_{j,s}^{k_j} = x_{j,s}^{k_j} - x_{j'',s}^{k_{j''}} + p_{j'',s}^{k_{j''}} + r_{s,jt_{j''},jt_j} \quad \forall j, j'' \in J; s \in S; k_j \in K_j; k_{j''} \in K_{j''} \tag{4-23}$$

$$vs_{j,s}^{k_j} = x_{j,s}^{k_j} - x_{j,s''}^{k_j-1} + p_{j,s''}^{k_j-1} - t_{s,s''} - rt_s \quad \forall j \in J; s, s'' \in S; k_j \in K_j \tag{4-24}$$

Durch die eingefügten Schlupfzeiten verschieben sich die Start- und Endzeiten der Bearbeitungsschritte und es kann passieren, dass $Sched_{Präd}$ nicht mehr zulässig ist, also Restriktionen des Modells aus Kapitel 4.3.1.2 verletzt werden. Daher ist es erforderlich, dass der Schedule nach jeder eingefügten Schlupfzeit repariert wird. Eine einmalige Reparatur zum Schluss ist nicht möglich, da die Reparaturmaßnahmen die Störungskaskaden beeinflussen können und damit die Schlupfzeiten selbst nur auf Basis eines zulässigen Schedules korrekt berechnet werden können. Die Reparatur des Schedules wird in den Schritten (4) bis (6) des Algorithmus vorgenommen.

In Schritt (4) und (5) wird überprüft, ob durch die Verschiebungen horizontale oder vertikale Überlappungen von Aufträgen entstanden sind, also $hs < 0$ oder $vs < 0$ für mindestens einen Bearbeitungsschritt gilt. Ist dies der Fall, wird die Zulässigkeit wiederhergestellt, indem die nachfolgenden Bearbeitungsschritte um die negativen Schlupfzeiten hs und vs gemäß Formeln 4-25 und 4-26 auf der Zeitachse nach rechts verschoben werden. Da die Schlupfzeiten bereits Anforderungszeiten rt, Transportzeiten t und reihenfolgeabhängige Rüstzeiten r berücksichtigen, ist durch diese Verschiebungen in jedem Fall wieder Zulässigkeit bezüglich dieser Größen hergestellt.

$$x_{j,s}^{k_j''} = x_{j,s}^{k_j} - hs_{j,s}^{k_j} \quad \forall j, j'' \in J; s \in S; k_j \in K_j; k_{j''} \in K_{j''} \tag{4-25}$$

$$x_{j,s}^{k_j-1} = x_{j,s}^{k_j} - vs_{j,s}^{k_j} \quad \forall j \in J; s, s'' \in S; k_j, \in K_j \tag{4-26}$$

In Schritt (6) wird schließlich überprüft, ob durch die Verschiebungen der Bearbeitungsschritte unzulässige Überschreitungen der Pufferkapazitäten an den Stationen entstanden sind. Formel 4-27 berechnet dazu zu jedem Zeitpunkt t die Pufferbelegung b_s^t anhand der Anzahl der Aufträge, die bereits im Puffer angekommen sind ($x_{j,s}^{k_j} - tb_{j,s}^{k_j}$), aber deren Bearbeitungsbeginn noch nicht erreicht ist ($x_{j,s}^{k_j}$). Aus Gründen der Praktikabilität wird t in diskreten Zeitschritten von 1s berechnet.

$$b_s^t = \left| t \in \left[x_{j,s}^{k_j} - tb_{j,s}^{k_j}, x_{j,s}^{k_j} \right] \right| \tag{4-27}$$

Sollte dies der Fall sein, werden die Aufträge ermittelt, die beim Auftreten der Überschreitung im Puffer liegen würden. Anschließend wird derjenige Auftrag, bei dem die gesamte erwarteten Verzögerung td der verbleibenden Bearbeitungsschritte durchschnittlich am meisten über λ liegt so weit auf der Zeitachse nach rechts verschoben, dass die Überschreitung der Pufferkapazität aufgehoben wird (Formel 4-28). Dies liegt darin begründet, dass diese Bearbeitungsschritte im weiteren Verlauf des Algorithmus ohnehin weiter verschoben würden und verringert so die Wahrscheinlichkeit, dass Bearbeitungsschritte willkürlich verschoben werden. Es wird an dieser Stelle auch ersichtlich, dass eine hohe Pufferkapazität große Freiräume für das robuste Scheduling schafft.

$$tb_{j,s}^{k_j} = \underset{\forall j,s,k_j}{\operatorname{argmax}} \left(td_{j,s}^{k_j} - \lambda_{j,s}^{k_j} \right) \tag{4-28}$$

Die Iteration wird durchlaufen, bis überall ausreichend Schlupfzeiten eingefügt und keine Unzulässigkeit mehr besteht. Es entsteht damit der prädiktive robuste Schedule $Sched_{Rob}$, der die Auftragsreihenfolge an Stationen von $Sched_{Präd}$ beibehält und zudem über Schlupfzeiten verfügt, die die gesamte erwartete Verzögerung von Bearbeitungsschritten auf ein über den Robustheitsparameter R definierten Maximalwert begrenzt.

4.4 Ermittlung eines Rescheduling-Korridors

Im Folgenden wird das Vorgehen für die Ermittlung von Rescheduling-Korridoren der Struktur aus Abbildung 4-7 folgend beschrieben.

Das Zusammenwirken (im Folgenden: Kopplung) zwischen prädiktivem robusten Scheduling und reaktiven Scheduling erfolgt echtzeitnah während der Ausführung des Schedules. Dies wird in der vorliegenden Arbeit mit einer ereignisdiskreten Ablaufsimulation abgebildet. Das Vorgehen basiert auf der vom Autor angeleiteten Abschlussarbeit (A_Zöllner 2019).

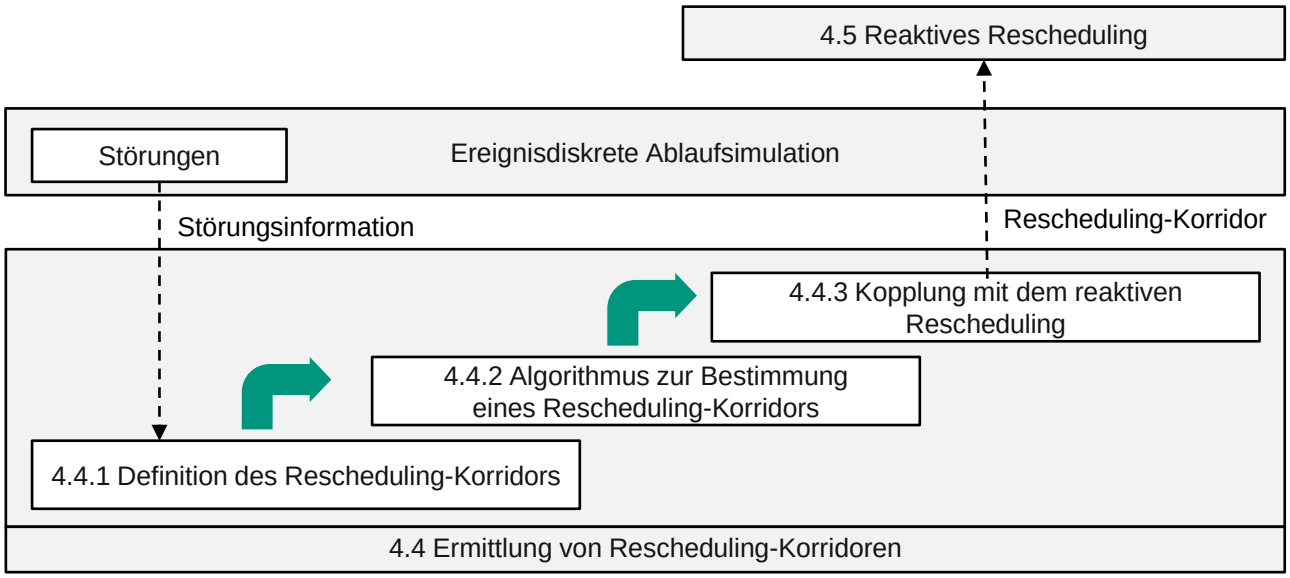


Abbildung 4-7: Struktur des Vorgehens zur Ermittlung von Rescheduling-Korridoren

Zum Zeitpunkt des Auftretens einer Störung an einer Station wird ein Störungsmanagement ausgelöst, das zunächst die nötige Informationsbasis für das reaktive Rescheduling erzeugt. Dazu wird in Kapitel 4.4.1 ein Steuerungsfall definiert, der den Rescheduling-Korridor beschreibt. Anschließend wird in Kapitel 4.4.2 ein Algorithmus zur Bestimmung und iterativen Erweiterung von Rescheduling-Korridoren beschrieben. Dieser nimmt kein Rescheduling vor, sondern ermittelt auf Basis der vorliegenden Störung und des prädiktiv robust erzeugten Schedules den Rescheduling-Korridor. Dessen iterative Erweiterung ist genau dann nötig, wenn für einen generierten Rescheduling-Korridor keine zulässige Lösung durch das reaktive Rescheduling gefunden werden kann. Die Kopplung zwischen der Ermittlung von Rescheduling-Korridoren mit dem reaktiven Rescheduling über die ereignisdiskrete Ablaufsimulation wird schließlich in Kapitel 4.4.3 dargestellt. Das Ergebnis wird an das reaktive Rescheduling in Kapitel 4.5 übergeben.

Ziel des reaktiven Rescheduling ist es, mit einem möglichst kleinen Rescheduling-Korridor zu einer zulässigen Lösung zu gelangen, da dann lediglich Ausschnitte des prädiktiv robust erzeugten Schedules umgeplant werden müssen. Dafür muss zuvor ein möglichst kleiner Rescheduling-Korridor ermittelt werden. Sobald eine zulässige Lösung gefunden wurde, muss das Ergebnis des Rescheduling in den prädiktiv robust erzeugten Schedule integriert werden. Es ist mit dem entwickelten Ansatz möglich, bereits bestehende Störungen bei Auftreten einer neuen Störung zu berücksichtigen.

In Abbildung 4-8 wird beispielhaft die Herausforderung bei der Entwicklung eines geeigneten Match-Up-Verfahrens für den Störungsfall aufgezeigt. Eine Störung an Station s_3 bestimmt den Start des Rescheduling-Korridors, allerdings bleibt offen, welchen Zeitraum er umfassen und welche Bearbeitungsschritte berücksichtigt werden sollen.

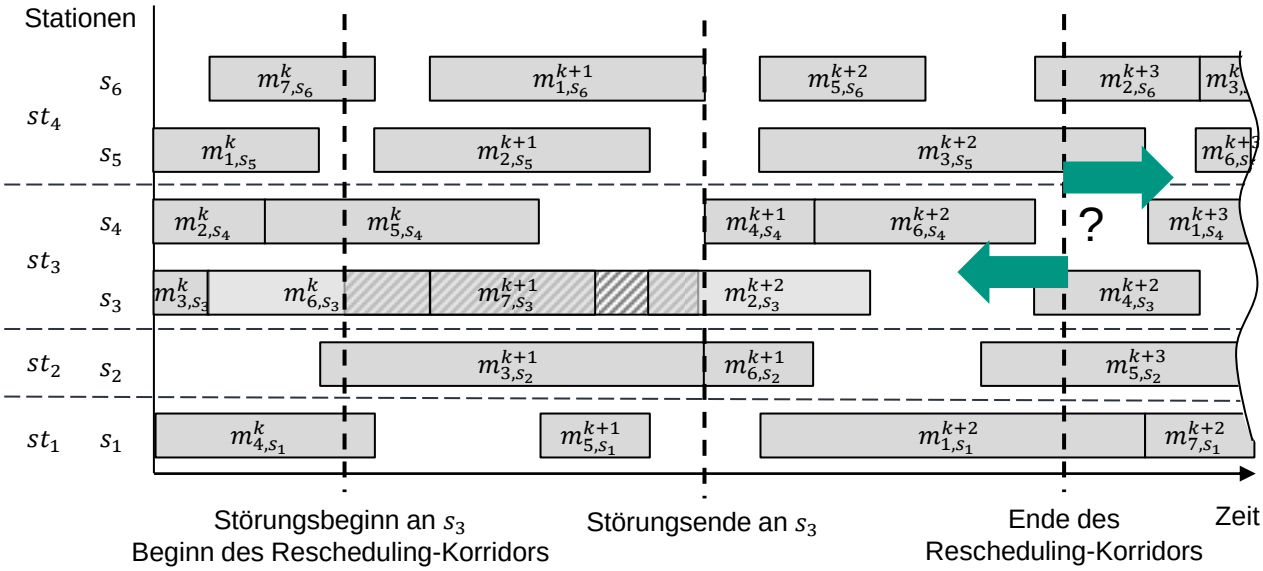


Abbildung 4-8: Herausforderung bei der Festlegung des Rescheduling-Korridors

4.4.1 Definition des Rescheduling-Korridors

Der Rescheduling-Korridor (abgekürzt mit RK) wird durch den Steuerungszeitpunkt, die Steuerungsdauer und den Steuerungsinhalt beschrieben und enthält damit die nötige Informationsbasis für das reaktive Rescheduling. Das Beispiel in Abbildung 4-8 zeigt einen Ausschnitt eines Schedules, in dem sieben Aufträge an sechs Stationen $s_1, \ldots, s_6$ eingeplant sind, die zu vier Stationstypen $st_1, .., st_4$ gehören. Zum Zeitpunkt $UG = t_0$ tritt eine Störung an Station s_3 auf, die die Ermittlung eines Rescheduling-Korridors erforderlich macht.

Der **Steuerungszeitpunkt** legt fest, wann das Rescheduling durchgeführt werden soll. In der vorliegenden Arbeit wird ein Match-Up-Ansatz verfolgt, der ereignisbasiert auf ressourcenbezogene Störungen reagiert. Dabei wird nur der durch den Rescheduling-Korridor definierte Ausschnitt des zugrundeliegenden Schedules umgeplant. Angelehnt an das in Abbildung 2-4 dargestellte Framework handelt es sich hierbei also um einen ereignisbasierten partiellen Rescheduling-Ansatz (A_Zöllner 2019).

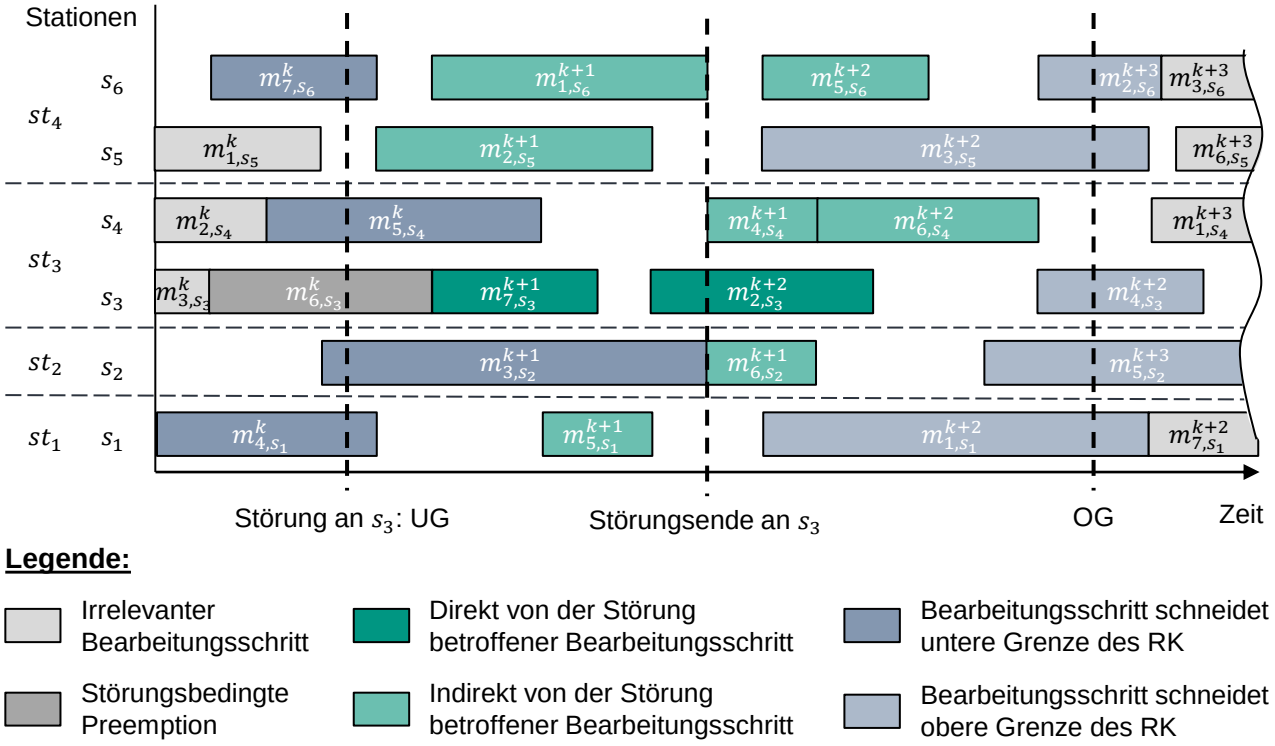


Abbildung 4-9: Relevante Typen von Bearbeitungsschritten für Rescheduling-Korridore

Die **Steuerungsdauer** legt die Zeitspanne des Rescheduling ab dem Steuerungszeitpunkt fest. Sie hängt von den Schlupfzeiten im zugrundeliegenden Schedule, dessen Abhängigkeiten in Form von Vorrangbeziehungen sowie dem Umfang der vorliegenden Störung ab. Eine kurze Störung kann zu einer langen Steuerungsdauer führen, wenn nur wenige Schlupfzeiten vorhanden sind. Umgekehrt können viele Schlupfzeiten auch größere Störungen in kurzer Zeit kompensieren. Analog zu den in Kapitel 3.3 vorgestellten Ansätzen für das Match-Up-Rescheduling wird das Ende der Steuerungsdauer als Match-Up-Zeitpunkt (obere Grenze OG) bezeichnet.

Der **Steuerungsinhalt** enthält die für das reaktive Rescheduling relevanten Bearbeitungsschritte derjenigen Aufträge, die ab dem Steuerungszeitpunkt bis zum Match-Up-

Zeitpunkt im zugrundeliegenden Schedule eingeplant waren. Aufträge auf Stationen, auf denen kein Auftrag von der Störung betroffen ist, können dabei ausgeschlossen werden. Die zur Bestimmung des notwendigen Steuerungsfalls unterschiedlichen Typen von Bearbeitungsschritten sind in Abbildung 4-9 beispielhaft dargestellt und werden im Folgenden beschrieben:

- **Irrelevante Bearbeitungsschritte** sind all diejenigen Bearbeitungsschritte, die komplett außerhalb des Korridors liegen, also entweder vor Beginn der Störung bei UG abgeschlossen sind oder nach Erreichen der oberen Grenze OG am Match-Up-Zeitpunkt anfangen. Sie sind nicht im Rescheduling-Korridor enthalten. Dies betrifft in Abbildung 4-9 z. B. die Bearbeitung von Auftrag 1 an s_5 und s_4.
- **Störungsbedingte Preemptions** betreffen diejenigen Bearbeitungsschritte, die zu Beginn einer Störung an einer Station bearbeitet werden. Bei deren Auftreten wird der betroffene Auftrag als Ausschuss aus dem System ausgeschleust sowie alle noch auszuführenden Bearbeitungsschritte aus dem Schedule entfernt. Dies betrifft in Abbildung 4-9 z. B. die Bearbeitung von Auftrag 6 an der gestörten Station s_3.
- **Direkt von einer Störung betroffene Bearbeitungsschritte** sind während der Dauer der Störung an der gestörten Station eingeplant und müssen daher zwingend umgeplant werden. Dies betrifft in Abbildung 4-9 die Bearbeitung der Aufträge 7 und 2 an der gestörten Station s_3.
- **Indirekt von einer Störung betroffene Bearbeitungsschritte** liegen vollständig innerhalb des Rescheduling-Korridors, gehören nicht zu einem Auftrag, der einer störungsbedingten Preemption unterliegt und sind auch nicht direkt durch eine Störung betroffen. Daher müssen diese Bearbeitungsschritte nicht zwingend umgeplant werden. Allerdings werden die Freiheitsgrade des Rescheduling entscheidend gesteigert, wenn diese Bearbeitungsschritte ebenfalls in den Rescheduling-Korridor aufgenommen werden, weshalb diese Strategie im Folgenden angewendet wird. Dies betrifft in Abbildung 4-9 z. B. die Bearbeitung von Auftrag 2 an s_5.
- **Bearbeitungsschritte schneiden untere bzw. obere Grenze des Rescheduling-Korridors**, wenn deren Bearbeitung zu Beginn einer Störung bei UG an einer ungestörten Station bereits mit deren Bearbeitung begonnen oder wenn bei Erreichen der oberen Grenze OG eine Bearbeitung noch nicht abgeschlossen ist.

Diese Bearbeitungsschritte dürfen nicht umgeplant werden, da keine geplanten Preemptions zulässig sind. Andernfalls könnte bei der Reintegration des umgeplanten Schedules in $Sched_{Rob}$ der Fall auftreten, dass zur selben Zeit für einen Auftrag zwei unterschiedliche Bearbeitungsschritte eingeplant sind. Die einzige Ausnahme bilden Bearbeitungsschritte, welche die obere Grenze des Rescheduling-Korridors schneiden und an einer gestörten Station eingeplant sind. Diese zählen ebenfalls zu den direkt von einer Störung betroffenen Bearbeitungsschritten, die zwingend umgeplant werden müssen.

Für das Rescheduling reicht es somit nicht aus, lediglich die jeweils direkt und indirekt von einer Störung betroffenen Bearbeitungsschritte zu ermitteln und störungsbedingte Preemptions zu entfernen. Zudem müssen Überlappungen zwischen dem Rescheduling-Ergebnis und dem prädiktiven robusten Schedule $Sched_{Rob}$ verhindert werden. Dazu ist zum einen die Information erforderlich, welche Stationen zu Beginn der Störung belegt sind. Zum anderen muss für den Match-Up-Zeitpunkt ermittelt werden, welche Aufträge zu diesem Zeitpunkt an welchen Stationen eingeplant sind.

4.4.2 Algorithmus zur Bestimmung eines Rescheduling-Korridors

Im Folgenden wird ein Algorithmus zur Bestimmung eines Rescheduling-Korridors beschrieben. Der Algorithmus nutzt das Prinzip des Match-Up-Schedulings. Dabei werden die im prädiktiven robusten Schedule vorhandenen Schlupfzeiten ausgenutzt, um den frühesten Zeitpunkt zu ermitteln, nach dem wieder mit dem prädiktiven robusten Schedule $Sched_{Rob}$ fortgefahren werden kann. Der Algorithmus baut im Wesentlichen auf der vom Autor angeleiteten studentischen Abschlussarbeit (A_Zöllner 2019) auf.

Bei dem Algorithmus handelt es sich um eine iterative Heuristik, die einen minimalen Match-Up-Zeitpunkt ermittelt. Aufgrund nicht berücksichtigter Vorrangbedingungen, Anforderungs- und Transport- sowie reihenfolgeabhängiger Rüstzeiten ist es wahrscheinlich, dass ein in der ersten Iteration erzeugter Rescheduling-Korridor keine zulässige Lösung besitzt und somit mehrere Iterationen notwendig sind.

Der Ablauf des Algorithmus ist in Abbildung 4-10 dargestellt. Im Folgenden wird dieser beschrieben und anhand des in Abbildung 4-11 dargestellten exemplarischen Beispiels veranschaulicht. In dem Beispiel existieren vier Stationstypen $ST = \{st_1, \dots, st_4\}$, die durch zehn Stationen $S = \{s_1, \dots, s_{10}\}$ teilweise redundant abgebildet werden. Zudem

sind in dem betrachteten Ausschnitt des Schedules 12 Aufträge auf den Stationen eingeplant, die sich entlang unterschiedlicher Transportwege durch die Produktion bewegen. Die Aufträge sind durch die Beschriftung der Bearbeitungsschritte gekennzeichnet. Beispielhaft wird von einer neu aufgetretenen Störung an s_5 ausgegangen. Zudem liegen zu dem Zeitpunkt noch Reststörungen an s_1 und s_6 vor.

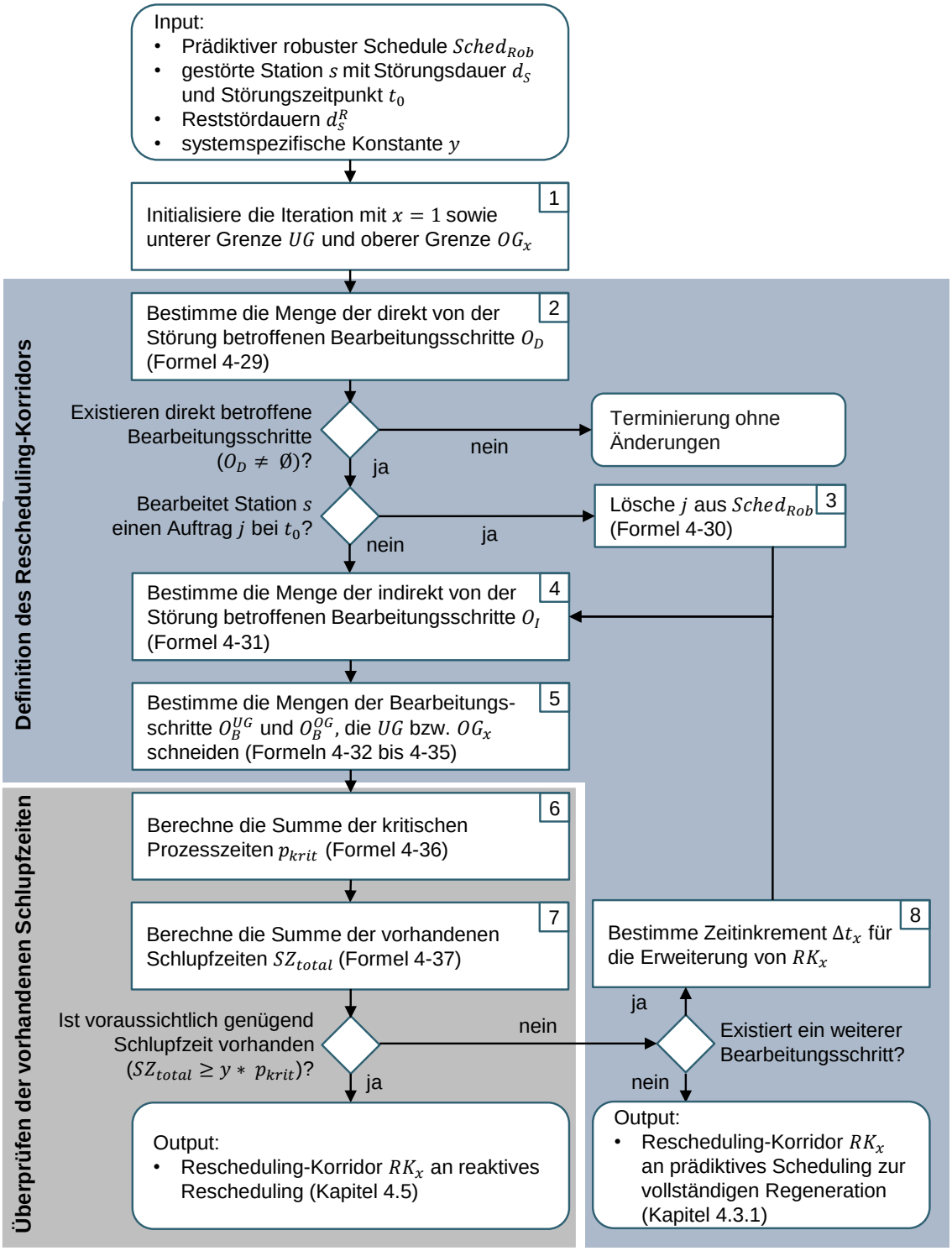


Abbildung 4-10: Pseudo-Code des Algorithmus zur Bestimmung eines Rescheduling-Korridors i. A. an (A_Zöllner 2019)

Der Algorithmus lässt sich in zwei Phasen unterteilen. Zunächst wird in jeder Iteration eine Definition des Rescheduling-Korridors durchgeführt (Schritte (1) - (5) und (8)), bevor anschließend eine Überprüfung der vorhandenen Schlupfzeiten stattfindet (Schritte (6) und (7)). Die Initialisierung in Schritt (1) startet, sobald eine Störung an einer Station auftritt. Input ist der prädiktive robuste Schedule $Sched_{Rob}$, die gestörte Station s und deren Störungsdauer d_s sowie vorhandene Reststörungsdauern d_s^R. Der Beginn einer Störung zum Störzeitpunkt t_0 an einer Station s stellt die untere Grenze UG des Rescheduling-Korridors dar. Im Beispiel wird eine Störung an s_5 betrachtet, die damit $UG = t_0$ determiniert. Die Störungsdauer d_s der betroffenen Station wird als bekannt angenommen. Auch die Reststördauern d_s^R ab dem Zeitpunkt t_0 aller bereits gestörten Stationen liegen vor. Die Störungsdauer d_s wird in der Initialisierung als Steuerungsdauer des Steuerungsfalls festgelegt. Der Rescheduling-Korridor muss mindestens so lang sein, wie die Störungsdauer, da diese Zeit mindestens benötigt wird, um wieder zu $Sched_{Rob}$ zurück zu kehren. In der Initialisierung ergibt sich damit $OG_1 = t_0 + d_s$. Die Menge $S' \subseteq S$ der zu s_5 parallelen Stationen besteht im Bespiel lediglich aus s_6. Dazu ist es zunächst unerheblich, dass dort noch eine Reststörung mit Dauer d_6^R vorliegt. Für die Bestimmung des Inhalts des Rescheduling-Korridors sind alle Bearbeitungsschritte relevant, die vor UG begonnen haben und bei UG noch nicht beendet sind oder deren Start vor OG_x eingeplant ist und die nach OG_x beendet werden.

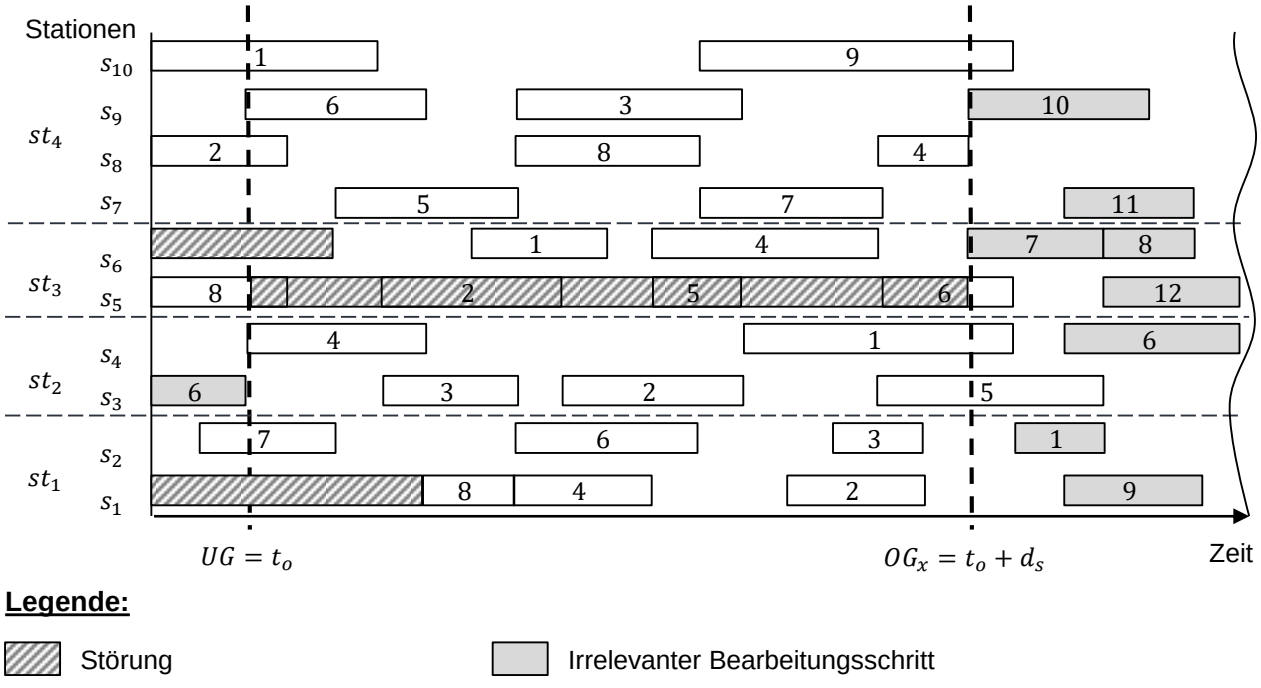


Abbildung 4-11: Schritt 1 - Initialisierung des Rescheduling-Korridors

In Schritt (2) werden die direkt betroffenen Bearbeitungsschritte O_D bestimmt. Dazu gehören alle Bearbeitungsschritte, die während der Störung auf s eingeplant sind und daher zwingend durch das Rescheduling umgeplant werden müssen. Die Menge O_D wird mit Formel 4-29 beschrieben. Es kann dabei auch vorkommen, dass Bearbeitungsschritte OG_x schneiden und daher trotzdem zwingend umgeplant werden müssen, auch wenn sie teilweise rechts von OG_x liegen.

$$O_D = \left\{ \forall o_j^{k_j} \in O : s_{m_{j,s}^{k_j}} = s \ \wedge \ t_0 \leq x_{j,s}^{k_j} < t_0 + d_S \right\} \quad \text{mit } s \in S;\ j \in J;\ k_j \in K_j \qquad 4\text{-}29$$

Gilt $O_D = \emptyset$, so ist kein Bearbeitungsschritt durch die Störung betroffen und der Algorithmus terminiert. Es kann dann mit $Sched_{Rob}$ direkt fortgefahren werden.

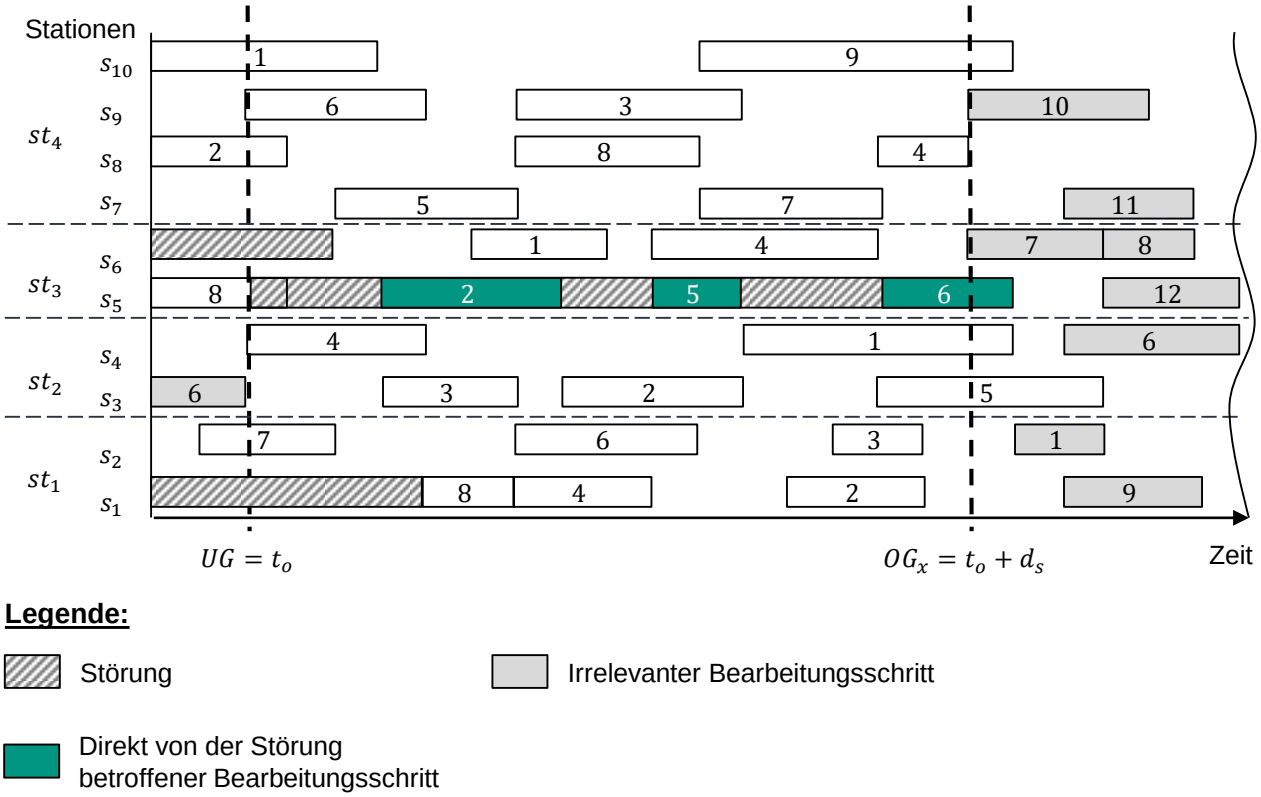


Abbildung 4-12: Schritt 2 - Ermittlung direkt von einer Störung betroffener Bearbeitungsschritte des initialen Rescheduling-Korridors

Im Beispiel in Abbildung 4-12 sind die Aufträge 2, 5 und 6 direkt von der Störung betroffen und müssen entsprechend umgeplant werden. Es gilt hier $O_D = \left\{ m_{2,s_5}^{k_j}, m_{5,s_5}^{k_j}, m_{6,s_5}^{k_j} \right\}$, weshalb es sich um eine signifikante Störung handelt.

In Schritt (3) werden störungsbedingte Preemptions behandelt. Dazu wird derjenige Auftrag j aus $Sched_{Rob}$ und RK_1 entfernt, der sich bei t_0 auf der gestörten Station s in Bearbeitung befindet, da diese störungsbedingte Preemption zu Ausschuss führt. Die Bedingung für einen betroffenen Auftrag ist in Formel 4-30 dargestellt.

$$\exists\, o_j^{k_j} \in O : s_{m_{j,s}^{k_j}} = s \;\wedge\; x_{j,s}^{k_j} < UG \;\wedge\; UG < x_{j,s}^{k_j} + p_{j,s}^{k_j} < OG \qquad 4\text{-}30$$

mit $s \in S;\ j \in J;\ k_j \in K_j$

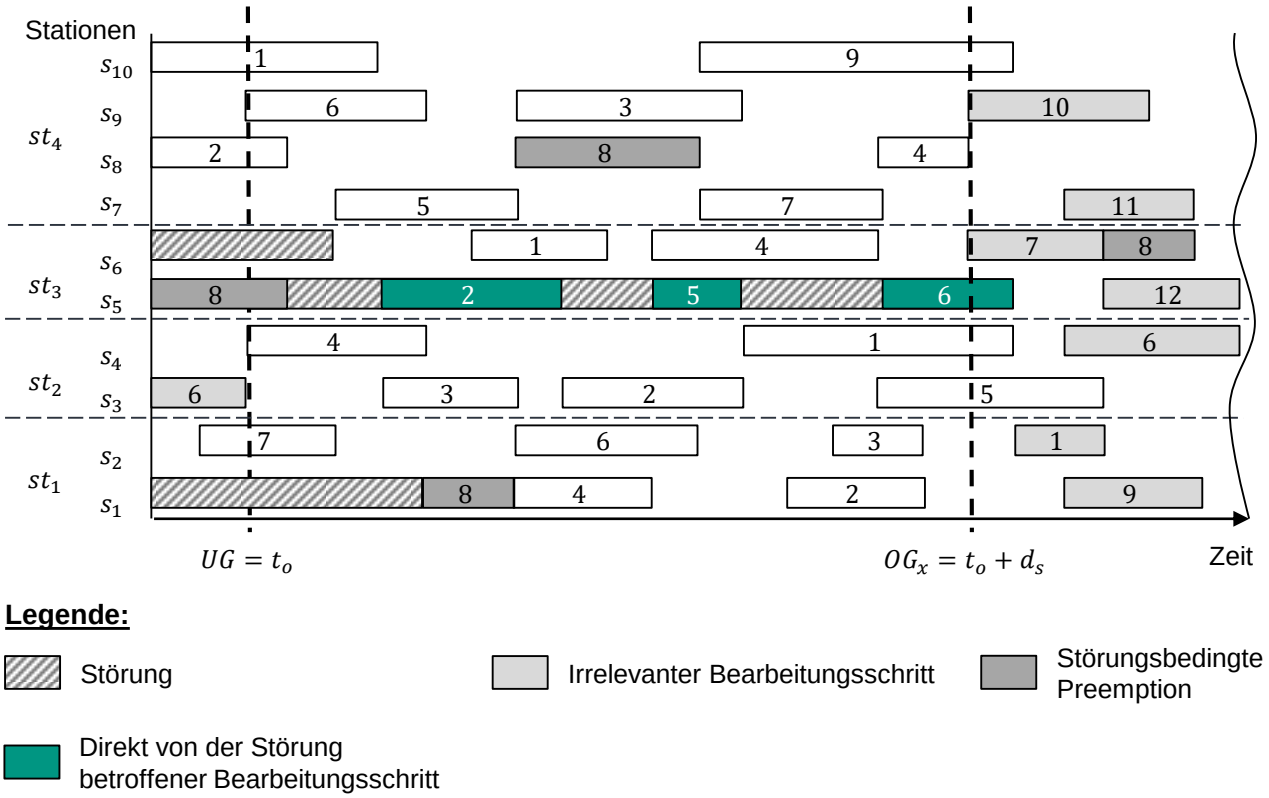


Abbildung 4-13: Schritt 3 - Störungsbedingtes Preemption-Handling des initialen Rescheduling-Korridors

Im Beispiel in Abbildung 4-13 sind die durch die störungsbedingte Preemption betroffen Bearbeitungsschritte von Auftrag 8 markiert, die anschließend aus $Sched_{Rob}$ und RK_1 gelöscht werden.

Anschließend wird in Schritt (4) die Menge der indirekt von der Störung an Station s betroffenen Bearbeitungsschritte O_I, die nicht während der Störungsdauer auf s eingeplant sind, bestimmt. Diese Bearbeitungsschritte müssen nicht zwingend aufgrund der Störung umgeplant werden, werden aber als zusätzliche Freiheitsgrade zur Kompensation der Störung verwendet. Die Menge O_I ist demnach nach Formel 4-31 definiert.

$$O_I = \left\{ \begin{array}{l} \forall o_j^{k_j} \in O : \left(s_{m_{j,s}^{k_j}} \neq s \ \wedge \ t_0 \leq x_{j,s}^{k_j} \wedge t_0 + p_{j,s}^{k_j} \leq OG_x \right) \\ \vee \left(s_{m_{j,s}^{k_j}} = s \ \wedge \ t_0 + d_s \leq x_{j,s}^{k_j} \wedge t_0 + p_{j,s}^{k_j} \leq OG_x \right) \end{array} \right\} \qquad 4\text{-}31$$

mit $s \in S;\ j \in J;\ k_j \in K_j$

Indirekt von einer Störung betroffene Bearbeitungsschritte in O_I dürfen frühestens zeitgleich mit Auftreten einer Störung in t_0 an einer nicht gestörten Station beginnen und müssen gleichzeitig spätestens mit Ende des Rescheduling-Korridors bei OG_x enden. Außerdem zählen auch diejenigen Bearbeitungsschritte zu O_I, die frühestens mit Ende der Störung auf der dann wieder zur Verfügung stehenden Station s beginnen und gleichzeitig spätestens bei OG_x enden. Im Beispiel in Abbildung 4-14 sind damit z. B. die Bearbeitungsschritte von Auftrag 3 betroffen.

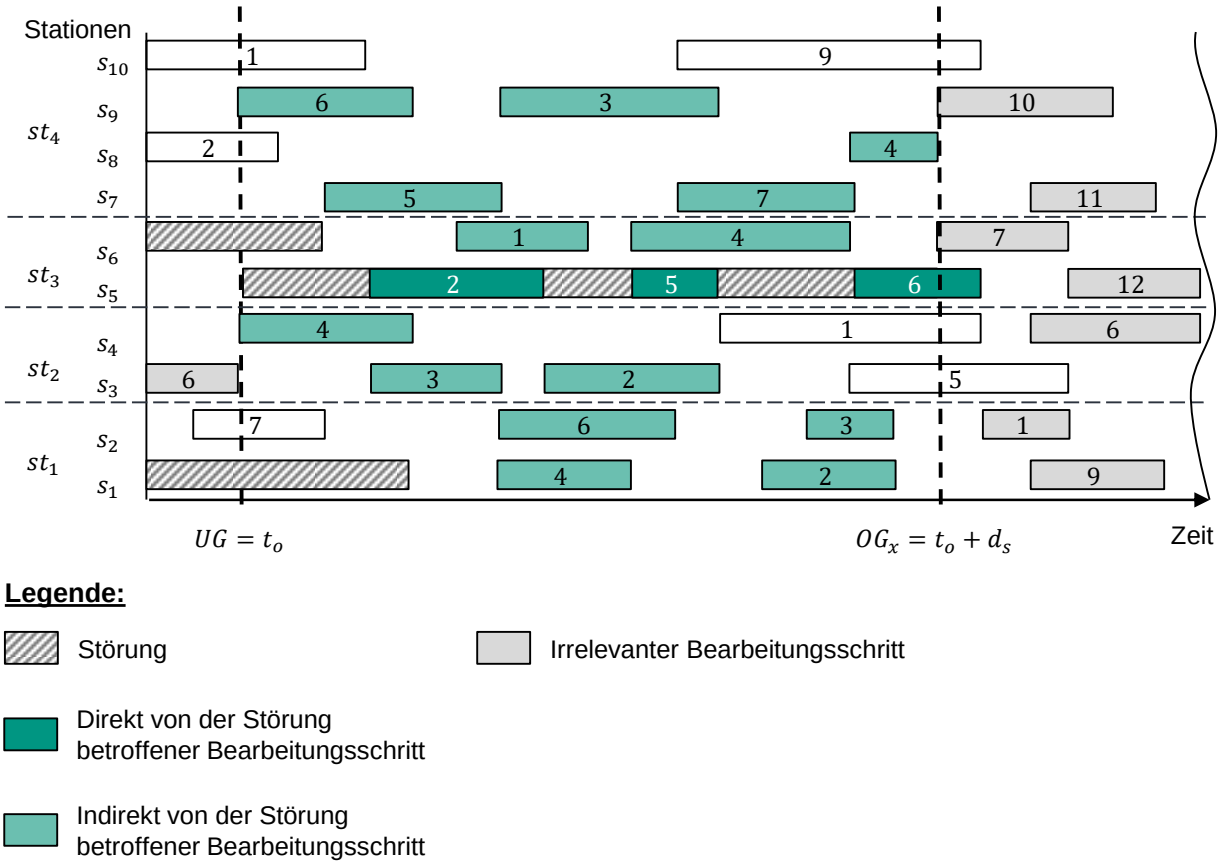


Abbildung 4-14: Schritt 4 - Ermittlung indirekt von einer Störung betroffener Bearbeitungsschritte des initialen Rescheduling-Korridors

Anschließend werden in Schritt (5) diejenigen Bearbeitungsschritte ermittelt, die die untere bzw. obere Grenze des Rescheduling-Korridors schneiden. Dabei bezeichne $O_i^{B,UG}$

die Menge der Bearbeitungsschritte, die die untere Grenze und O_B^{OG} die Menge der Bearbeitungsschritte, die die obere Grenze eines Rescheduling-Korridors RK_x schneiden. Auch auf s eingeplante Bearbeitungsschritte können zu O_B^{OG} gehören, wenn sie nach Ende der Störung beginnen und OG schneiden. Diese Bedingungen lassen sich mit Formeln 4-32 und 4-33 formal beschreiben. Im Beispiel in Abbildung 4-15 betrifft dies an UG und OG die dunkel- bzw. hellblau eingefärbten Bearbeitungsschritte.

$$O_B^{UG} = \left\{ \forall o_j^{k_j} \in O : s_{m_{j,s}^{k_j}} \neq s \ \wedge \ t_0 > x_{j,s}^{k_j} \wedge \ t_0 < x_{j,s}^{k_j} + p_{j,s}^{k_j} \right\} \quad 4\text{-}32$$

$$O_B^{OG} = \left\{ \begin{array}{c} \forall o_j^{k_j} \in O : \left(s_{m_{j,s}^{k_j}} \neq s \ \wedge \ t_0 \leq x_{j,s}^{k_j} < OG_x \wedge \ x_{j,s}^{k_j} + p_{j,s}^{k_j} > OG_x \right) \vee \\ \left(s_{m_{j,s}^{k_j}} = s \ \wedge \ t_0 + d_s \leq x_{j,s}^{k_j} \wedge \ x_{j,s}^{k_j} + p_{j,s}^{k_j} > OG_x \right) \end{array} \right\} \quad 4\text{-}33$$

mit $s \in S;\ j \in J;\ k_j \in K_j$

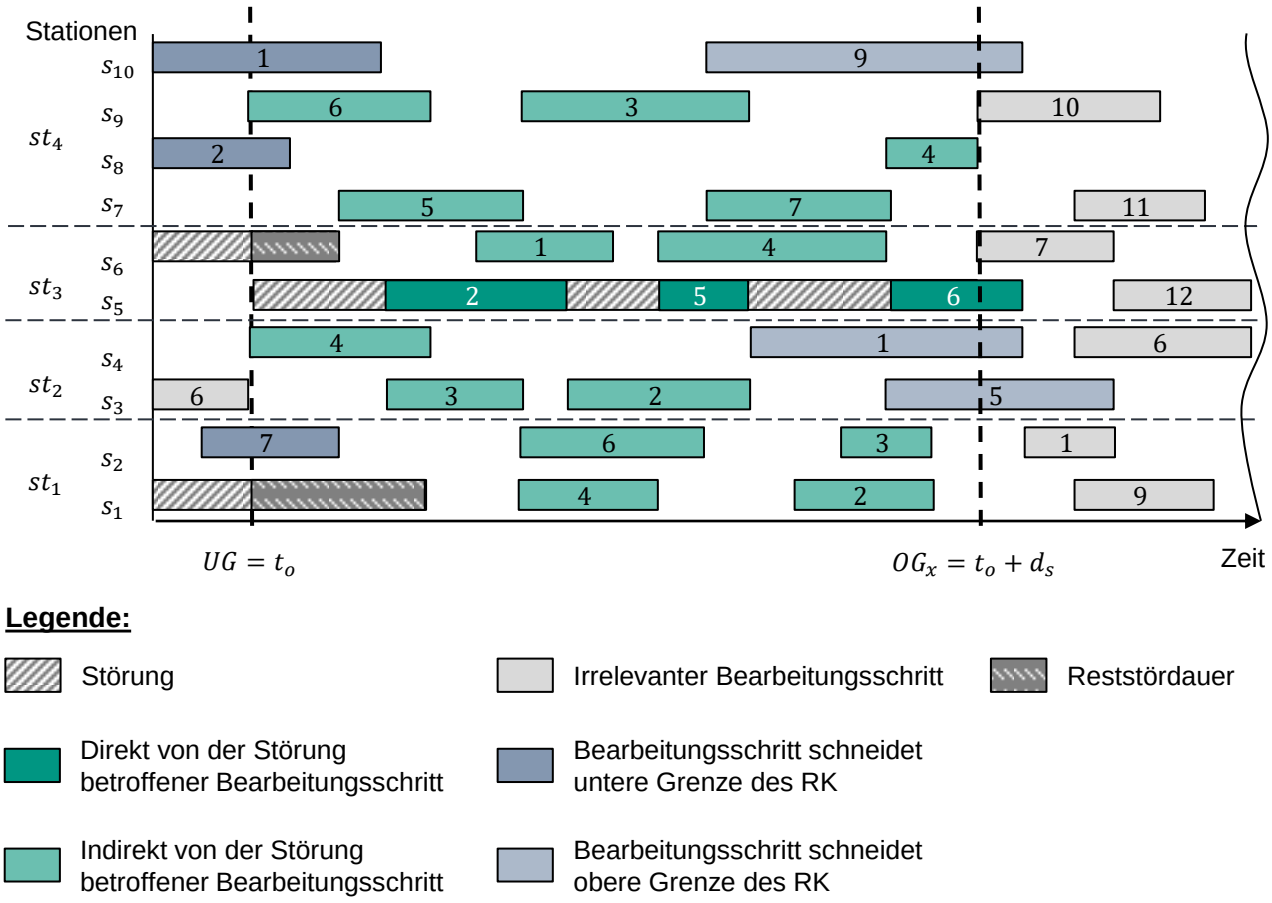


Abbildung 4-15: Schritt 5 – Ermittlung von Bearbeitungsschritten, die die untere bzw. obere Grenze des initialen Rescheduling-Korridors schneidenden

Einen Sonderfall stellen Bearbeitungsschritte dar, die UG und OG schneiden. Diese werden der Menge O_B^{UG} zugeordnet. Ist ein solcher Bearbeitungsschritt an einer Station $i \neq s$ eingeplant, so steht diese für das Rescheduling nicht zur Verfügung.

Elemente in O_B^{UG} und O_B^{OG} dürfen nicht durch das Rescheduling umgeplant, sondern müssen gemäß $Sched_{Rob}$ ausgeführt werden. Dadurch werden überlappende Bearbeitungsschritte vermieden, die die Reintegration eines umgeplanten Korridors in $Sched_{Rob}$ verhindern würden. Daher müssen dem Rescheduling die Informationen über die jeweiligen Belegungszeiträume der betroffenen Stationen $i \in S$ übergeben werden, sodass keine direkt oder indirekt betroffenen Bearbeitungsschritte währenddessen eingeplant werden. Die Belegungszeiträume werden durch den Belegungszeitpunkt sowie die Belegungsdauer charakterisiert. Die Belegungsdauern $d_i^{B,UG}$ bzw. $d_i^{B,OG}$ ergeben sich dabei, je nachdem, ob UG oder OG geschnitten werden, nach Formeln 4-34 und 4-35 aus der Differenz der geplanten Start- bzw. Endzeit eines betroffenen Bearbeitungsschrittes und dem Zeitpunkt der jeweiligen Horizontgrenze.

$$d_s^{B,UG} = x_{j,s}^{k_j} + p_{j,s}^{k_j} - t_0, \forall o_j^{k_j} \in O_B^{UG} \quad \forall s \in S;\ j \in J;\ x, k_j \in K_j \tag{4-34}$$

$$d_s^{B,OG} = OG_x - x_{j,s}^{k_j}, \forall o_j^{k_j} \in O_B^{OG} \quad \forall s \in S;\ j \in J;\ x, k_j \in K_j \tag{4-35}$$

Nachdem die unterschiedlichen Typen von Bearbeitungsschritten identifiziert sind, wird in Schritt (6) die kritische Prozesszeit p_{krit} ermittelt. Diese besteht aus der Summe aller Prozesszeiten der Bearbeitungsschritte in O_D, die zwingend umgeplant werden müssen (siehe Formel 4-36). Damit dies innerhalb von RK_x möglich ist, muss mindestens auf den parallelen Stationen ausreichende Schlupfzeit vorhanden sein. Ist dies nicht der Fall, muss zwangsläufig eine vollständigen Regeneration durchgeführt werden. Nur mithilfe ausreichender Schlupfzeit ist eine Rückkehr zu $Sched_{Rob}$ möglich.

$$p_{krit} = \sum_{\forall s \in S, j \in J:\ o_j^{k_j} \in O_D} p_{j,s}^{k_j} \tag{4-36}$$

Zur Bewertung des Rescheduling-Korridors müssen abschließend in Schritt (7) noch die vorhandenen Schlupfzeiten SZ_{total} berechnet werden. Diese bestehen aus der Summe der Zeiten, die auf den zur gestörten Station parallelen Stationen $s' \in S' \cup \{s\}$

im Intervall $[UG, OG_x]$ nicht belegt sind und lässt sich mit Formel 4-37 berechnen. Die berücksichtigten Schlupfzeiten für das Beispiel in Abbildung 4-16 sind als gepunktete Flächen an der zur gestörten Station s_5 parallelen Station s_6 gekennzeichnet.

$$SZ_{total} = \left|S'\right| * (OG_x - t_0) - \sum_{o_j^{k_j} \in O_I} p_{j,s'}^{k_j} - \sum_{o_j^{k_j} \in O_B^{UG}} d_{s'}^{B,UG} - \sum_{o_j^{k_j} \in O_B^{OG}} d_{s'}^{B,OG} - \sum_{i \in S'} d_{S,i}^{R} + (OG_x - OG_1) \quad \text{4-37}$$

mit $s \in S;\ j \in J;\ k_j \in K_j$

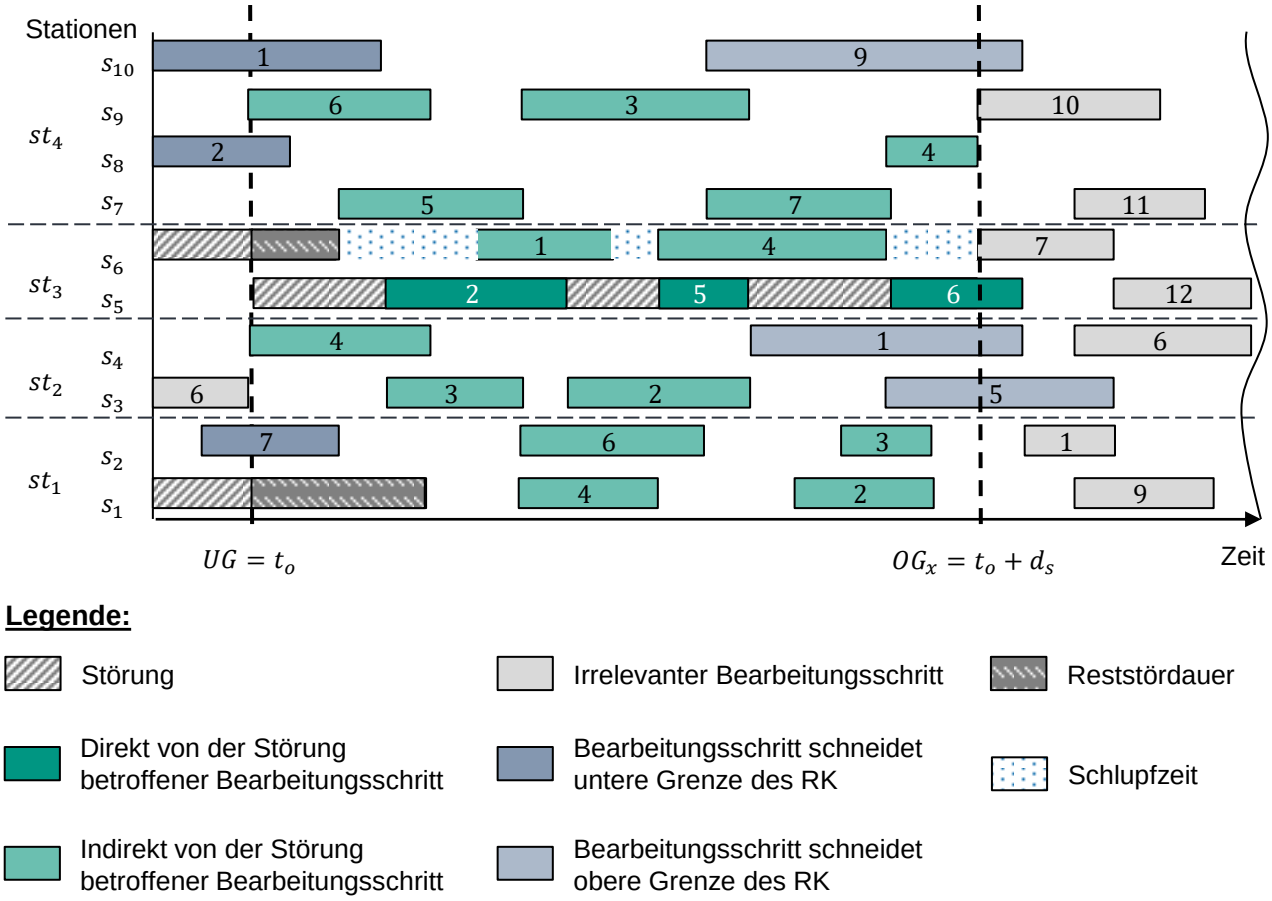


Abbildung 4-16: Schritt 7 - Berechnung der zur Verfügung stehenden Schlupfzeit innerhalb des Rescheduling-Korridors

Der Term $OG_x - UG$ beschreibt die Steuerungsdauer und damit die Breite eines in Iteration x betrachteten Rescheduling-Korridors. Diese stellt wiederum gleichzeitig die theoretisch maximal zur Verfügung stehende Schlupfzeit einer Station $i \in S'$ dar, wenn darauf innerhalb des Rescheduling-Korridors keine weiteren Bearbeitungsschritte in $Sched_{Rob}$ eingeplant wurden. Dieser maximale Wert wird entsprechend mit $|S'|$ multipliziert, um die maximal mögliche Schlupfzeit zu erhalten. Diese wird anschließend um

die Prozesszeiten $p_{j,s'}^{k_j}$ sowie die zuvor berechneten Stationsbelegungsdauern $d_{s'}^{B,UG}$ bzw. $d_{s'}^{B,OG}$ korrigiert. Der abschließende Term $OG_x - OG_1$ berücksichtigt hinzukommende Schlupfzeiten an der gestörten Station s bei einer notwendigen Verbreiterung des Rescheduling-Korridors.

Abschließend wird in Schritt (8) geprüft, ob der erzeugte Rescheduling-Korridor grundsätzlich zulässig ist, d. h. ob auf s bzw. auf den dazu parallelen Stationen aus S' mindestens so viel Schlupfzeit vorhanden ist, wie die Störungsdauer an s. Dies ist genau dann der Fall, wenn $SZ_{total} \geq y * p_{krit}$ erfüllt ist. Dabei ist y eine je Produktionssystem und Produktionsprogramm empirisch zu ermittelnde Konstante, die die zusätzlich anfallenden Anforderungs-, Transport- und reihenfolgeabhängigen Rüstzeiten berücksichtigt (siehe Formel 4-38). In Produktionssystemen, in denen die Prozesszeit etwa 25% der Durchlaufzeit von Aufträgen ausmacht, ist damit $y = 4$.

$$y = \frac{\sum_{\forall j \in J} \sum_{\forall s \in S} \left(p_{j,s}^{k_j} + t_{s,s'} + rt_s \right)}{\sum_{\forall j \in J} \sum_{\forall s \in S} p_{j,s}^{k_j}} \qquad \text{4-38}$$

Dann wird der Rescheduling-Korridor als zulässig bezeichnet und der Algorithmus abgebrochen. Der damit definierte Steuerungsfall beinhaltet die Informationen über die Steuerungsdauer vom Zeitpunkt des Störungsbeginns in t_0 bis zum Ende des Korridors in OG_x sowie den Steuerungsinhalt O_D und O_I. Als zusätzliche Randbedingung werden die Belegungszeiträume der Stationen über UG und OG hinweg angegeben um fehlerhafte Überlappungen von Bearbeitungsschritten zwischen $Sched_{Resch}$ und $Sched_{Rob}$ zu verhindern. Es muss allerdings mit einem zulässigen Rescheduling-Korridor nicht zwingend ein Rescheduling möglich sein, da bei der Berechnung von SZ_{total} Vorrangbeziehungen vernachlässigt werden.

Andernfalls muss der Rescheduling-Korridor schrittweise erweitert werden, bis $SZ_{total} \geq y * p_{krit}$ erfüllt ist. Dazu wird OG_x wie in Formel 4-39 dargestellt so erweitert, dass der Zeitpunkt des Bearbeitungsbeginns x eines nachfolgenden Bearbeitungsschrittes außerhalb der oberen Grenze OG_x an der gestörten Station s oder einer dazu parallelen Station aus S' die neue obere Grenze OG_{x+1} darstellt. Der Rescheduling-Korridor wird dadurch nur um ein Inkrement erweitert, das zu einer Vergrößerung des Lösungsraumes führt.

$$\Delta t_x = x_{j,s}^{k_j} - OG_x \text{ für } x_{j,s}^{k_j} = \min_{\forall j \in J, s' \in S' \cup s} \left\{ x_{j,s}^{k_j} - OG_{x-1} \right\} + OG_{x-1} \qquad 4\text{-}39$$

mit $s \in S' \cup s$

Sollten keine nachfolgenden Bearbeitungsschritte auf den jeweiligen Stationen existieren, wird der nächste Bearbeitungsschritt auf einer der anderen Stationen aus S gewählt. Ist auch dies nicht möglich, so ist $Sched_{Rob}$ beendet und es gibt keine Möglichkeit, einen Rescheduling-Korridor zu ermitteln. Stattdessen muss mit den verbleibenden Bearbeitungsschritten eine vollständige Neugenerierung durchgeführt werden.

Wie sich im Beispiel in Abbildung 4-17 erkennen lässt, steht im initialen Rescheduling-Korridor RK_1 nicht ausreichend Schlupfzeit an Station s_5 zur Verfügung, weshalb der Korridor in einer zweiten Iteration entsprechend erweitert werden muss. Da $m_{7,s_5}^{k_j}$ direkt bei OG_1 startet (und damit das Inkrement 0 wäre), wird dieser Bearbeitungsschritt für die Erweiterung vernachlässigt und stattdessen ist $m_{12,s_6}^{k_j}$ für die Erweiterung relevant.

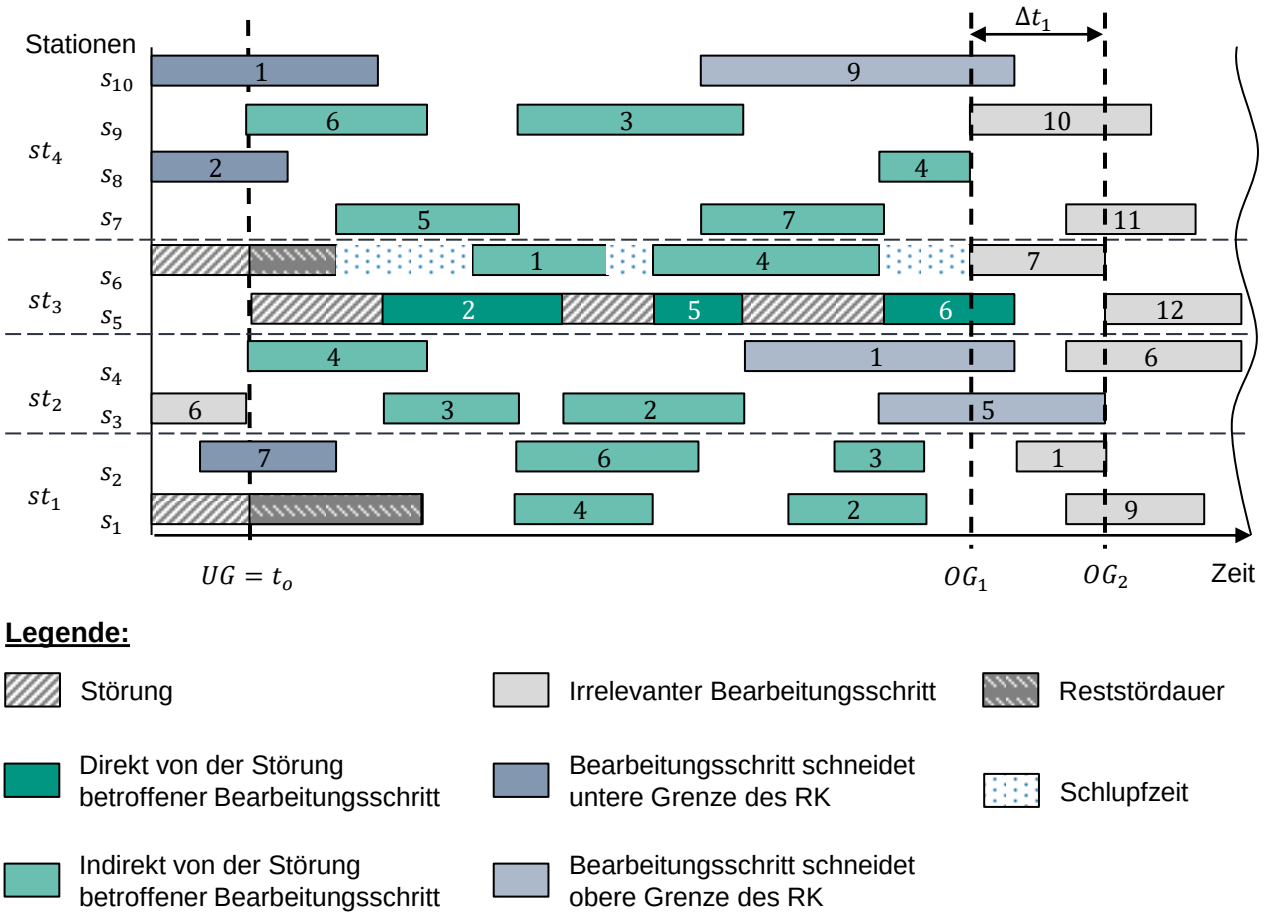


Abbildung 4-17: Schritt 8 – Erweiterung des initialen Rescheduling-Korridors

Durch die Erweiterung des Rescheduling-Korridors können sich die Typen von Bearbeitungsschritten verändern und neue Bearbeitungsschritte hinzukommen. Daher werden anschließend die Schritte (4) bis (11) des Algorithmus solange durchlaufen, bis ein zulässiger Rescheduling-Korridor ermittelt werden kann oder das Ende von $Sched_{Rob}$ erreicht wird. Von der Aktualisierung betroffen sind die Mengen O_I und O_B^{OG}, während sich O_D und O_B^{UG} aufgrund der konstanten unteren Grenze UG nicht verändern.

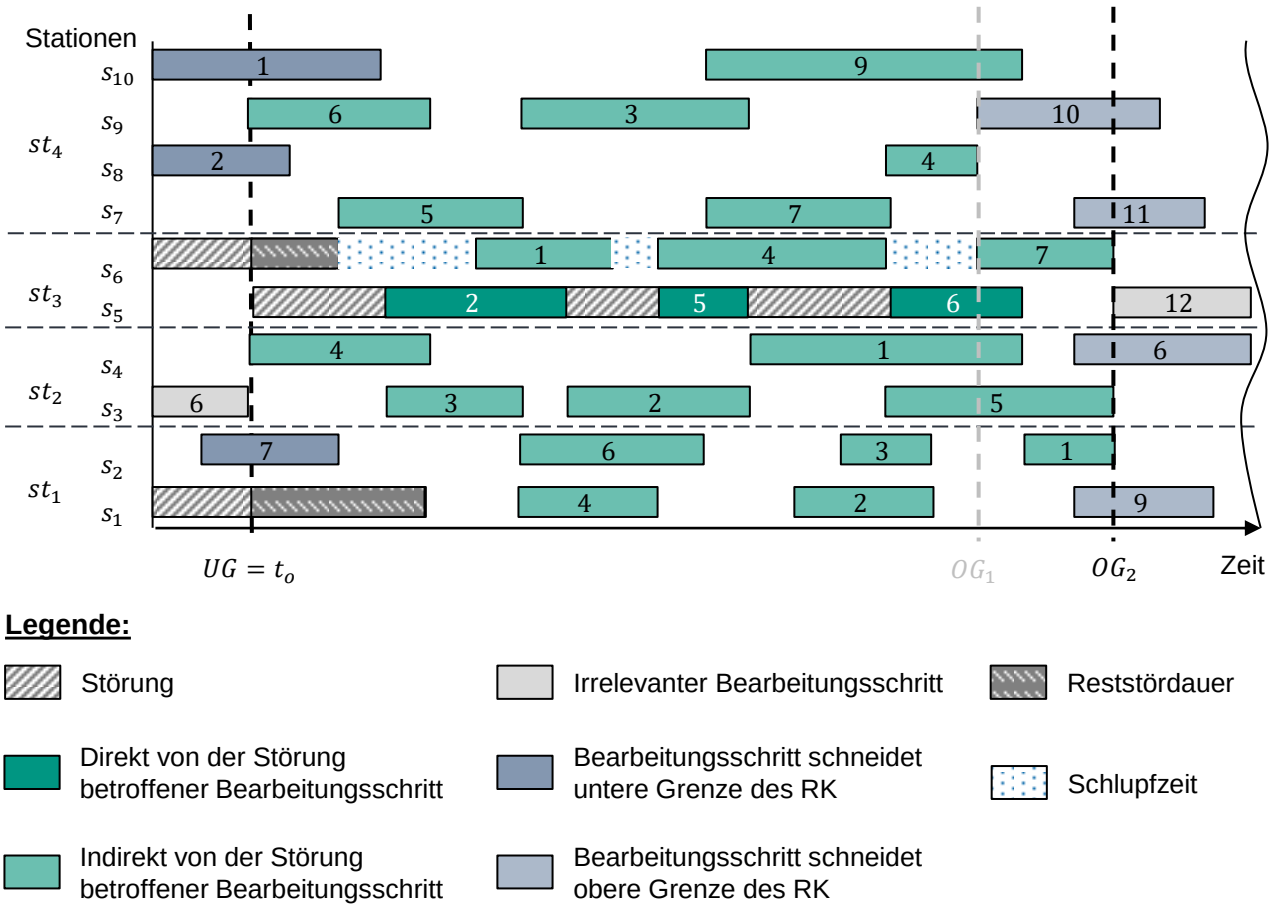


Abbildung 4-18: Rescheduling-Korridor als Ergebnis des Algorithmus

Im Beispiel in Abbildung 4-18 wird damit z. B. $m_{5,s_7}^{k_j}$ aus O_B^{OG} entfernt und neu zu O_I hinzugefügt. Außerdem kommt z. B. $m_{9,s_1}^{k_j}$ neu in O_B^{OG} aufgrund der Erweiterung des Rescheduling-Korridors hinzu. Sobald ein Rescheduling-Korridor gefunden wurde, wird als zusätzliche Information für das reaktive Rescheduling eine relative Distanzmatrix je Auftrag generiert. Dies ist erforderlich, da sich zum Zeitpunkt einer Störung Aufträge auf fahrerlosen Transportfahrzeugen zwischen Stationen befinden und für ein Rescheduling deren genaue Restentfernung zur geplanten Station sowie zu allen alternativ möglichen Stationen relevant ist. Wird nach dem reaktiven Rescheduling von RK_x

ein zulässiger Sub-Schedule gefunden, wird anschließend die Reintegration in $Sched_{Rob}$ durchlaufen. Dabei findet auch die Prüfung statt, ob ein Rescheduling innerhalb von RK_x möglich war. Sollte das nicht der Fall sein, wird RK_x analog zum Fall $SZ_{Total} < y * p_{krit}$ erweitert.

4.4.3 Kopplung mit dem reaktiven Rescheduling

Das iterative Zusammenspiel aus dem dargelegten Algorithmus zur Bestimmung eines Rescheduling-Korridors und dem reaktiven Rescheduling selbst ist in Abbildung 4-19 dargestellt. Bei Auftreten einer Störung wird zunächst durch den in Kapitel 4.4.2 beschriebenen Algorithmus ein Rescheduling-Korridor ermittelt. Dieser enthält einen Startzeitpunkt UG und einen Endzeitpunkt OG, Störungsinformation bezüglich der gestörten Station s und der Dauer der Störung d_s, die Restbelegungsdauern $d_s^{B,UG}$ und $d_s^{B,OG}$, die Menge der umzuplanenden Bearbeitungsschritte $O_D \cup O_I$ sowie den aktuellen Ort aller Aufträge und der jeweiligen relativen Distanzmatrizen. Mit dieser Information wird das reaktive Rescheduling durchgeführt, das in Kapitel 4.5 beschrieben wird.

Wie bereits erwähnt, wird der Rescheduling-Korridor so bestimmt, dass theoretisch ausreichend Schlupfzeit zur Kompensation von Störungen zum Match-Up-Zeitpunkt bereitsteht. Allerdings werden dabei keine Vorrangbeziehungen zwischen Bearbeitungsschritten berücksichtigt, sodass es sein kann, dass keine Lösung existiert. In dem Fall ist das Rescheduling nicht zulässig und der Rescheduling-Korridor muss erweitert werden. Andernfalls wird das Ergebnis des Rescheduling in den prädiktiven robusten Schedule $Sched_{Rob}$ integriert. Dazu werden für die umgeplanten Bearbeitungsschritte die neuen Start- und Endzeiten sowie die gewählte Bearbeitungsalternative und damit die gewählte Station übergeben.

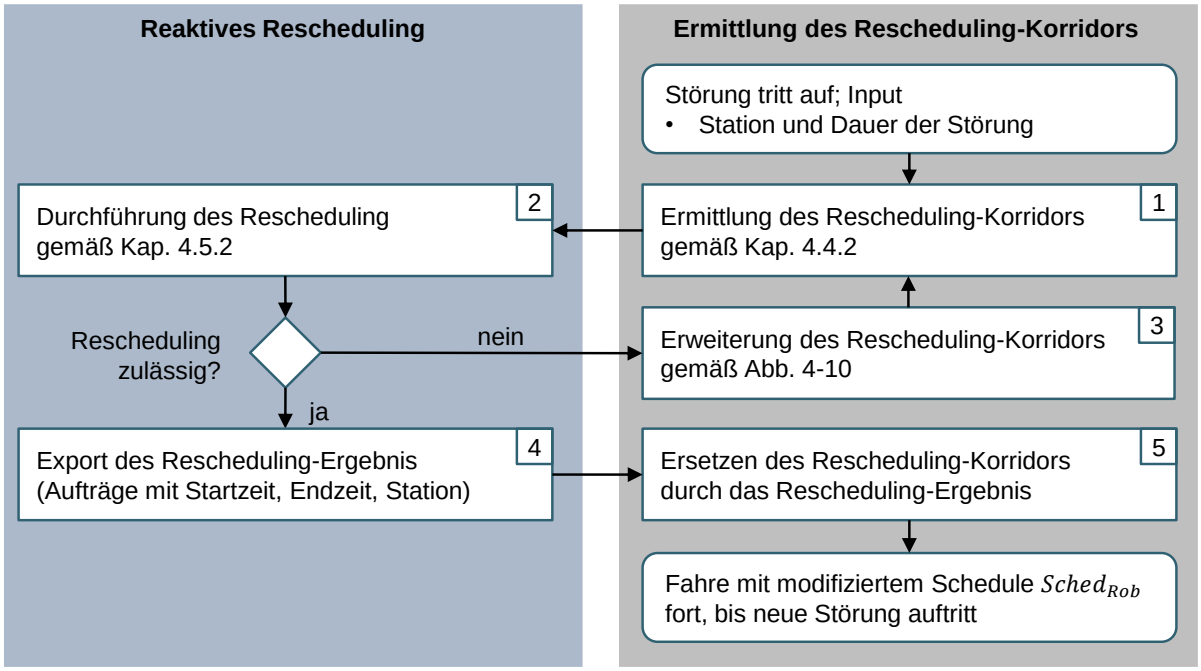


Abbildung 4-19: Kopplung mit dem reaktiven Rescheduling i. A. an (A_Hort 2019)

4.5 Reaktives Rescheduling

Im Folgenden wird das Vorgehen für das reaktive Rescheduling der Struktur aus Abbildung 4-20 folgend beschrieben.

Ziel ist es dabei, bei über das erwartete Maß hinausgehenden Störungen bei Bedarf in kurzer Zeit ein Rescheduling durchzuführen. Dieses soll den in Kapitel 4.4 ermittelten Rescheduling-Korridor aus $Sched_{Rob}$ so umplanen, dass anschließend mit diesem fortgefahren werden kann. Im Gegensatz zum prädiktiven robusten Scheduling ist es erforderlich, innerhalb von kurzer Zeit eine zulässige Lösung zu erhalten. Aufgrund der kurzen zur Verfügung stehenden Zeit für das Rescheduling kann die Forderung nach Optimalität, d. h. die Anwendung eines exakten Optimierungsverfahrens, nicht aufrechterhalten werden. Robustheit ist keine Zielgröße im reaktiven Rescheduling. Die in $Sched_{Rob}$ vorhandenen Schlupfzeiten sollen vielmehr genutzt werden, um die Störung zu kompensieren. Die Makespan soll minimiert werden, damit ein Match-Up mit $Sched_{Rob}$ zum Match-Up-Zeitpunkt, also der oberen Grenze OG des zugrundeliegenden Rescheduling-Korridors, möglich ist.

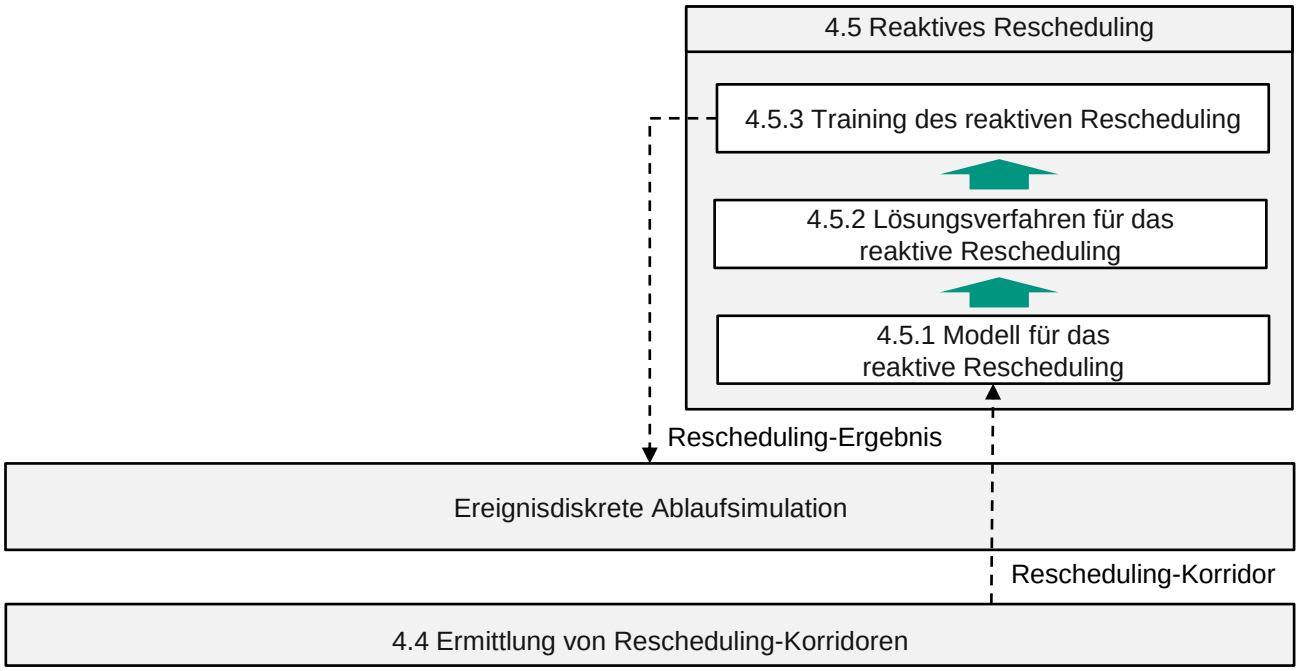


Abbildung 4-20: Struktur des Reaktiven Rechedulings in den Lösungsansätz

Das Problem des Rescheduling wird zunächst als dezentraler Markovscher Entscheidungsprozess (DEC-MDP) formuliert (Kapitel 4.5.1). Anschließend wird der von Gabel & Riedmiller (2012) stammende und für die vorliegende Arbeit weiterentwickelte Lösungsalgorithmus der verteilten Strategie-Iteration vorgestellt (Kapitel 4.5.2). Die Modifikationen basieren u. a. auf der vom Autor angeleiteten Abschlussarbeit (A_Hort 2019). Abschließend wird beschrieben, wie der Lernmechanismus des Lösungsverfahrens funktioniert (Kapitel 4.5.2.3).

4.5.1 Modell für das reaktive Rescheduling

Das reaktive Rescheduling wird als DEC-MDP modelliert. Dabei agieren alle Agenten autonom und können jeweils nur einen Teil des gesamten Systemzustands beobachten. Wie in Kapitel 2.4.1 eingeführt, wird ein DEC-MDP nach Bernstein & Givan et al. (2002) durch ein Tupel $< S, A, Z, P, G >$ beschrieben mit Agenten S, Aktionenraum A, Zustandsraum Z, Übergangsgesetz P und Belohnungsfunktion G. Das reaktive Rescheduling wird im Folgenden in Anlehnung an Echsler Minguillon & Lanza (2018) und Gabel & Riedmiller (2012) anhand der in Abbildung 4-21 dargestellten Bestandteile modelliert.

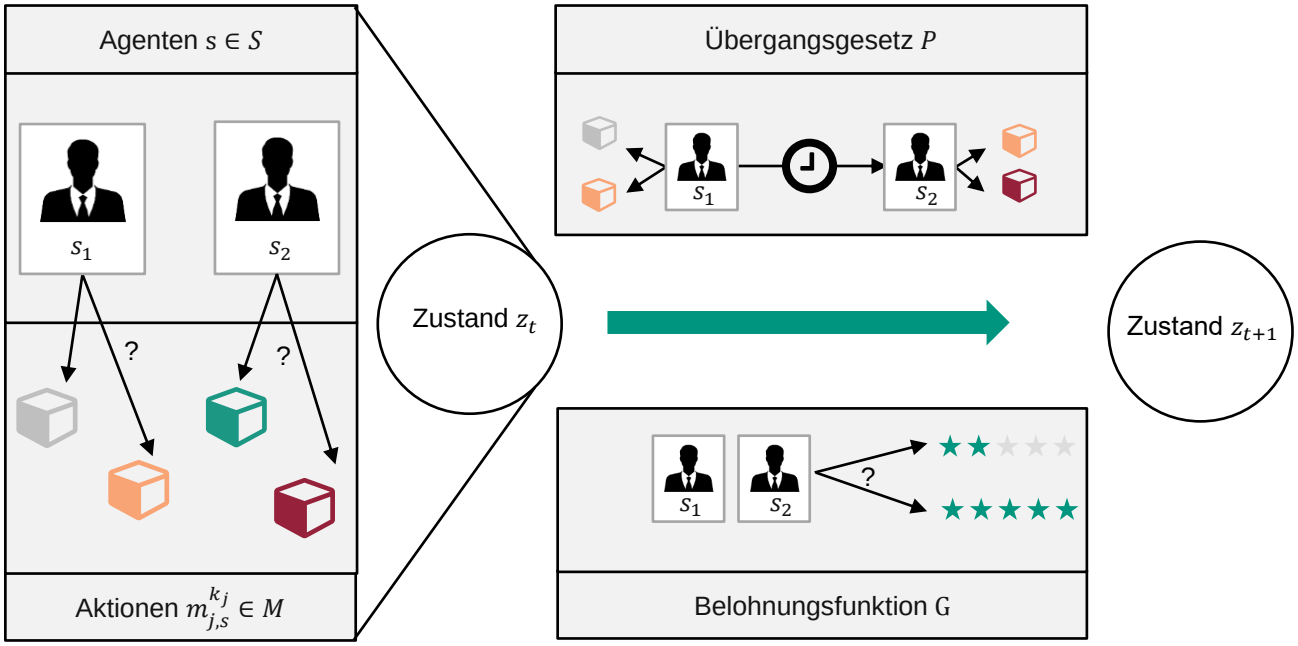


Abbildung 4-21: reaktives Rescheduling als DEC-MDP

Die **Agenten** S werden durch die Stationen in der Matrix-Produktion dargestellt. Dabei wird jede Station als unabhängiger Agent modelliert. Trotz mehrfach vorhandenen Stationen desselben Stationstyps wird die Unterscheidung getroffen, da Transportzeiten einen großen Anteil der gesamten Durchlaufzeit von Aufträgen ausmachen und aufgrund des Layouts hier große Unterschiede auftreten können. Entsprechend ist zu erwarten, dass sich die Strategie auch zwischen gleichartigen, an unterschiedlichen Stellen in der Produktion platzierten Stationen unterscheidet. In Abbildung 4-21 sind auf der linken Seite zwei Agenten s_1 und s_2 dargestellt.

Der **Aktionenraum** A enthält alle auswählbaren Aufträge aller Agenten und kann in disjunkte Aktionenräume A_s mit $s \epsilon S$ zerlegt werden, welche die auswählbaren Aufträge eines Agenten darstellen. Der Aktionenraum A_s wird durch eine Menge $A_s = \left\{m_{j,s}^{k_j} \dots, m_{|J|,s}^{|K_j|}\right\}$ beschrieben. Damit gibt es je möglicher Bearbeitungsalternative $m_{j,s}^{k_j}$ je Auftrag j eine Aktion. Es kann bei auswählbaren Aufträgen sowohl zwischen unterschiedlichen Auftragstypen als auch gleichen Auftragstypen mit unterschiedlichem Bearbeitungszustand unterschieden werden. Diese Unterscheidung ist in der Matrix-Produktion essentiell, da die Rezirkulation von Aufträgen grundsätzlich möglich ist. Die Aufträge können sich dabei sowohl im Puffer der Station, auf dem Weg zur Station oder in Bearbeitung an einer Vorgängerstation befinden. Es können somit bewusst Aufträge gewählt werden, die an Stationen zu Wartezeiten führen, wodurch aktive Schedules

erzeugt werden können. Die Wahl der nächsten Aktion erfolgt frühestens, wenn der Auftrag, auf den gewartet wird, an der betroffenen Stationen fertig bearbeitet wurde. In Abbildung 4-21 sind links die möglichen Aktionen der beiden Agenten s_1 und s_2 anhand der unterschiedlich eingefärbten Boxen dargestellt, die die im Produktionssystem befindlichen Aufträge repräsentieren.

Der **Zustandsraum** Z wird wie der Aktionenraum A abzüglich der Aktion $\propto_0$ definiert. Das heißt, dass ein Agent seine Aktion ausschließlich auf Basis der ihm zur Verfügung stehenden Aufträge ermittelt. Durch diese eingeschränkte Sicht der Agenten auf den Systemzustand wird dessen Entscheidung dezentralisiert. Der Zustandsraum kann wie der Aktionenraum in disjunkte Zustandsräume Z_s mit $s \epsilon S$ zerlegt werden. Der Zustandsraum entspricht im Beispiel in Abbildung 4-21 der Vereinigung aller möglichen Zustände $\bigcup_{s \in S} z_s$, also jeder möglichen Konstellation von Aufträgen an allen Stationen.

Entscheidet sich ein Agent für einen Auftrag j in einem Zustand, ist dieser im nächsten Zeitschritt dort nicht mehr verfügbar. Der neue Zustand z_s' ergibt sich dann nach Formel 4-40 aus dem bisherigen Zustand z_s. Für die der gewählten Bearbeitungsalternative $m_{j,s}^{k_j}$ nachgelagerten Stationen s' wird der Zustand um $m_{j,s'}^{k_j+1}$ nach Formel 4-41 erweitert. Je nach Stationstyp existieren mehrere Nachfolger. Der veränderliche Zustands- und damit Aktionsraum sorgt dafür, dass zu keinem Zeitpunkt eine unzulässige Aktion gewählt werden kann. Der Entscheidungsprozess ist damit auf die Auswahl einer besten Aktion aus einer Menge zulässiger Aktionen begrenzt.

$$z_s' = z_s \setminus \left\{ m_{j,s}^{k_j} \right\} \tag{4-40}$$

$$z_{s'}' = z_{s'} \cup \left\{ m_{j,s'}^{k_j+1} \right\} \tag{4-41}$$

Das **Übergangsgesetz** P beschreibt die Zustandsänderung von einem Zeitschritt zum nächsten bei Wahl einer Aktion. Da der Zustand der Agenten durch die an ihnen zur Verfügung stehenden Aufträge beschrieben wird, muss das Übergangsgesetz abbilden, wie sich diese Menge im Zeitverlauf ändert. Von einer expliziten Formulierung des Übergangsgesetzes wird hier abgesehen, da die Zustandsübergänge nicht zufallsverteilt sind, sondern von den Vorranggraphen der Aufträge abhängen. In Abbildung 4-21 ist das Übergangsgesetz exemplarisch so dargestellt, dass ein im aktuellen Zeitschritt von s_1 bearbeiteter Auftrag orange) im nächsten Zeitschritt als wählbare Aktion an einer Nachfolgestation s_2 zur Verfügung steht.

Die **Belohnungsfunktion** G muss sicherstellen, dass das globale Ziel des reaktiven Rescheduling erreicht wird. Im vorliegenden Fall soll der Makespan des Rescheduling minimiert werden, damit ein Match-Up mit $Sched_{Rob}$ zum Match-Up-Zeitpunkt, also der oberen Grenze OG des jeweiligen Rescheduling-Korridors, möglich ist. Wie (Gabel & Riedmiller 2012) beschreiben, wird daher in jedem Zeitschritt, zu dem noch offene Bearbeitungsschritte existieren (also mindestens ein Aktionsraum nicht-leer ist), eine Belohnung von -1 ausgezahlt. Die schrittweise Belohnung ist dabei für jede Aktion gleich. Gleichzeitig führt eine Folge von Aktionen dazu, dass die ausgezahlten Belohnungen einen numerischen Wert annehmen, der dem Makespan des Reschedulings entspricht.

Zudem wird ein Präferenz-Parameter $\theta_{jt_j,s}^{k_j} \in \mathbb{R}$ (im Folgenden verkürzt mit θ dargestellt) für jeden Besuch k_j eines Auftragstyps jt an Station s definiert, um die Strategie π_s zu beschreiben. Die θ werden im Lösungsverfahren verwendet, um aus dem Aktionsraum eines Agenten den nächsten Bearbeitungsschritt eines Auftrags anhand einer Präferenz zu wählen. In der Anlernphase des Algorithmus werden die θ ausgehend vom initialen Wert 0 schrittweise angepasst, bis sie zu einer die Makespan eines Rescheduling-Korridors minimierenden Prioritätsregel konvergieren. Diese kann dabei von Station zu Station unterschiedlich sein und stellt aufgrund der Dezentralität des Verfahrens lediglich ein lokales Optimum dar. Ein größerer Wert von $\theta_{jt_j,s}^{k_j}$ führt zu einer geringeren Wahrscheinlichkeit der Wahl einer Bearbeitungsalternative $m_{j,s}^{k_j}$.

θ wird in einer Matrix repräsentiert, was beispielhaft für neun Stationen und drei Auftragstypen in Abbildung 4-22 dargestellt ist. Die Parameter je Station stehen in den Zeilen, die möglichen Auftragstypen und deren Besuche an einer Station sind über die Spalten indiziert. Aufgrund der Darstellung wird die Anzahl der Spalten durch die maximale Anzahl unterschiedlicher Auftragstypen und Besuche über alle Stationen festgelegt. Daher sind im Beispiel 5 Spalten ausreichend, obwohl insgesamt 15 unterschiedliche Kombinationen aus Auftragstyp und Besuchen existieren. Für die Durchführung des Algorithmus wird jedem Agenten s der jeweils für ihn relevante Zeilenvektor $\overrightarrow{\theta_s}$ übergeben. In der ersten Zeile für Station 1 sind fünf Einträge vorhanden, d. h. es gibt insgesamt fünf unterschiedliche Arten von Aufträgen, die für die Auswahl an der Station relevant sind: Auftragstyp 1 kann seinen ersten und fünften Bearbeitungsschritt an Station 1 durchführen, Auftragstyp 2 kann dort seinen vierten und sechsten Bearbeitungsschritt durchführen; Auftragstyp 3 kann dort nur seinen dritten Bearbeitungsschritt

durchführen. Für jeden möglichen Auftragstyp und jeden Besuch an einer Station muss ein Parameter existieren, der in einem Zustand zur Auswahl herangezogen wird.

Auftragstypen jt und Besuche k_j

Stationen s

$$\begin{pmatrix} \theta_{1,1}^{1} & \theta_{1,1}^{5} & \theta_{2,1}^{4} & \theta_{2,1}^{6} & \theta_{3,1}^{3} \\ \theta_{1,2}^{1} & \theta_{1,2}^{5} & \theta_{2,2}^{4} & \theta_{2,2}^{6} & \theta_{3,2}^{3} \\ \theta_{1,3}^{1} & \theta_{1,3}^{5} & \theta_{2,3}^{4} & \theta_{2,3}^{6} & \theta_{3,3}^{3} \\ \theta_{1,4}^{2} & \theta_{1,4}^{4} & \theta_{2,4}^{2} & \theta_{3,4}^{2} & \\ \theta_{1,5}^{2} & \theta_{1,5}^{4} & \theta_{2,5}^{2} & \theta_{3,5}^{2} & \\ \theta_{1,6}^{3} & \theta_{2,6}^{5} & \theta_{3,6}^{1} & & \\ \theta_{1,7}^{3} & \theta_{2,7}^{5} & \theta_{3,7}^{1} & & \\ \theta_{1,8}^{3} & \theta_{2,8}^{5} & \theta_{3,8}^{1} & & \\ \theta_{2,9}^{1} & \theta_{2,9}^{3} & \theta_{3,9}^{4} & & \end{pmatrix}$$

Abbildung 4-22: Beispielhafte Matrix θ i. A. an (A_Hort 2019)

4.5.2 Lösungsverfahren für das reaktive Rescheduling

Zur Lösung des in Kapitel 4.5.1 modellierten DEC-MDP wird nachfolgend das gewählte Lösungsverfahren beschrieben. Dieses ist angelehnt an die verteilte Strategie-Iteration (DPS) nach (Gabel & Riedmiller 2012) dargelegt. Der Algorithmus ist in Abbildung 4-23 dargestellt.

Dabei wird anhand der Schritte (1) und (2) zunächst darauf eingegangen, wie die Agenten ihre Entscheidungen treffen. Anschließend wird beschrieben, wie durch Schritte (3) und (4) das Anlernen des Algorithmus für einen spezifischen Rescheduling-Korridor durchgeführt wird. Der Algorithmus wird von allen Agenten in jedem Zeitschritt nacheinander durchlaufen. Die folgenden Ausführungen beziehen sich aus Gründen der Übersichtlichkeit auf einen einzelnen Agenten.

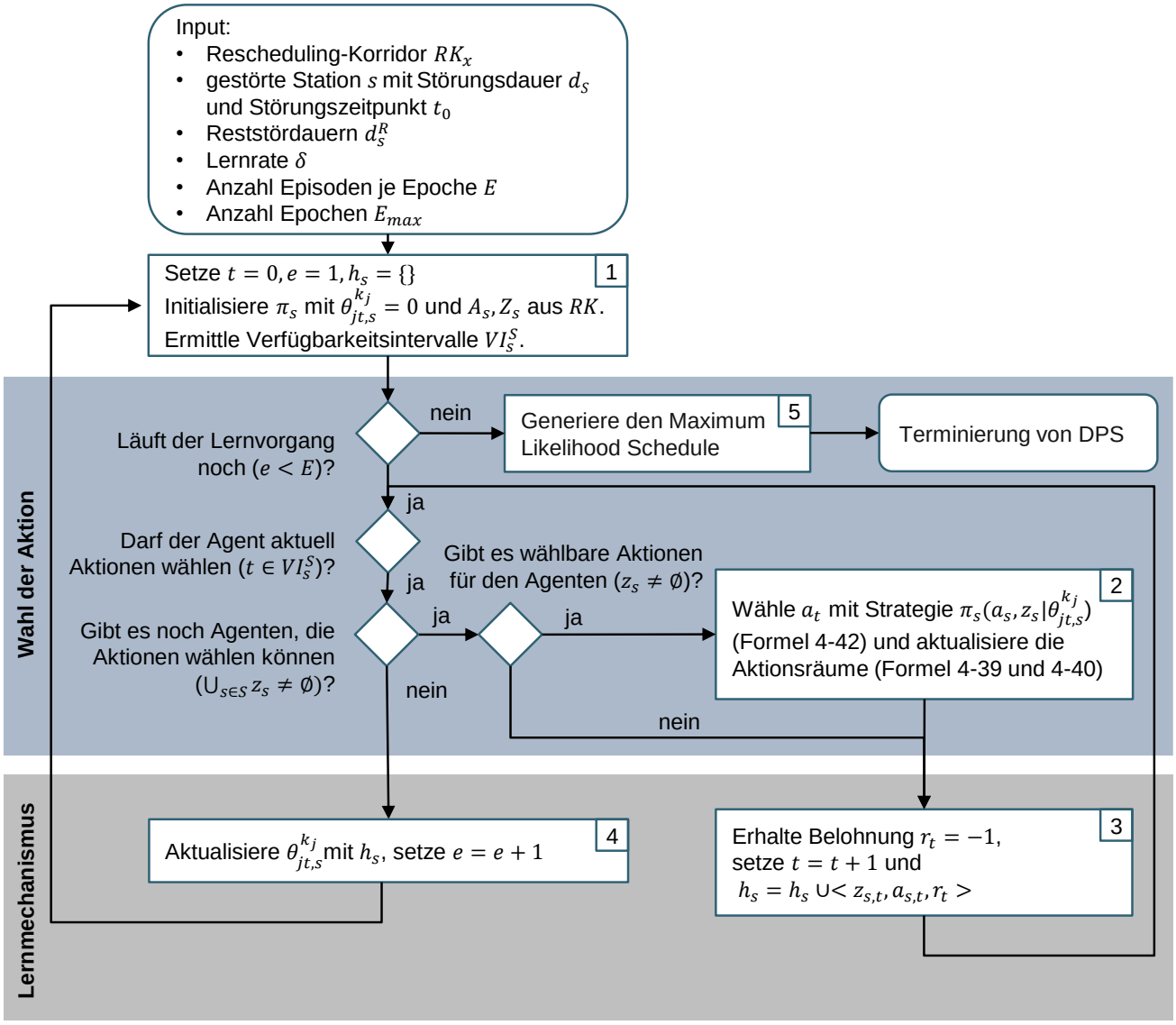


Abbildung 4-23: Ablaufdiagramm für DPS zum reaktiven Rescheduling aus Sicht eines Agenten i. A. an (Gabel & Riedmiller 2012)

Im Folgenden wird die Funktionsweise des Lösungsverfahrens dargelegt. Dazu wird zunächst auf die Aktionswahl auf der Ebene einzelner Agenten eingegangen (Kapitel 4.5.2.1), bevor die Interaktion der Agenten beschrieben wird (Kapitel 4.5.2.2).

4.5.2.1 Aktionswahl auf der Ebene einzelner Agenten

Zu Beginn des reaktiven Reschedulings werden die Elemente des DEC-MDP initialisiert (1). Für die Station s werden dem stationsspezifischen Aktionenraum A_s und dem Zustandsraum Z_s diejenigen Aufträge hinzugefügt, die zu Beginn des Rescheduling-Korridors RK dort zur Verfügung stehen, also sich entweder bereits im Puffer befinden oder auf dem Weg zu der Station sind. Damit spiegeln der Aktionenraum und der Zustands-

raum für alle Agenten zu Beginn des Rescheduling-Korridors den Zustand des Produktionssystems wieder. Als nächstes werden Verfügbarkeitsintervalle VI_s^S für die Stationen und VI_j^J für die Aufträge ermittelt. Diese geben an, in welchen Zeiträumen des Rescheduling-Korridors Aufträge bearbeitet werden können. Es lassen sich drei Fälle unterscheiden, die die Verfügbarkeitsintervalle von Stationen einschränken können:

- Bei der Station s handelt es sich um die gestörte Station mit Störungsdauer d_s. Damit ist das Verfügbarkeitsintervall auf $VI_s^S = [d_s, OG]$ eingeschränkt, da an der Station bis Ende der Störung keine Aufträge bearbeitet werden können.
- Die Station s unterliegt einer Reststörung mit Reststörungsdauer d_s^R aus einer vorangegangenen Störung. Das Verfügbarkeitsintervall wird dadurch auf $VI_s^S = [d_s^R, OG]$ eingeschränkt, d. h. die Station kann ebenfalls erst nach Ende ihrer Störung Aufträge bearbeiten.
- Die Station s bearbeitet aktuell noch einen Auftrag mit einer Restbearbeitungszeit $p_{j,s}^{k_j,R}$ (im Folgenden mit p^R annotiert). Dies schränkt das Verfügbarkeitsintervall auf $VI_s^S = [p^R, OG]$ ein, sodass frühestens nach Ablauf der Restbearbeitungszeit ein neuer Auftrag bearbeitet werden kann.

Zudem können sich in zwei Fällen Verfügbarkeitsintervalle VI_j^J für Aufträge ergeben:

- Der Auftrag j befindet sich aktuell noch in Bearbeitung an Station s. Da Bearbeitungsschritte mit Überschneidung von UG nicht Teil der Rescheduling-Korridore sind muss hier sichergestellt werden, dass nachfolgende Bearbeitungsschritte nicht während der Bearbeitung an einer Vorgängerstation eingeplant werden. Damit sind die Verfügbarkeitsintervalle von an Stationen in Bearbeitung befindlichen Aufträgen j auf $VI_j^J = [p_{j,s}^{k_j,R}, OG]$ beschränkt.
- Der Auftrag j wird aktuell zu einer Station transportiert mit einer Resttransportzeit von $t_{j,s'}^R$. Damit ist das Verfügbarkeitsintervall auf $VI_j^J = [t_{j,s'}^R, OG]$ beschränkt. Alternativ kann der Auftrag auch an einer parallelen Station allokiert werden. Die zu berücksichtigende Distanz hängt dann von der relativen Position des Transportmittels zu den parallelen Stationen ab. Es ergibt sich aus Sicht dieser Stationen daher $VI_j^J = [t_{j,s'}^R, OG]$ mit $s' \in S'$.

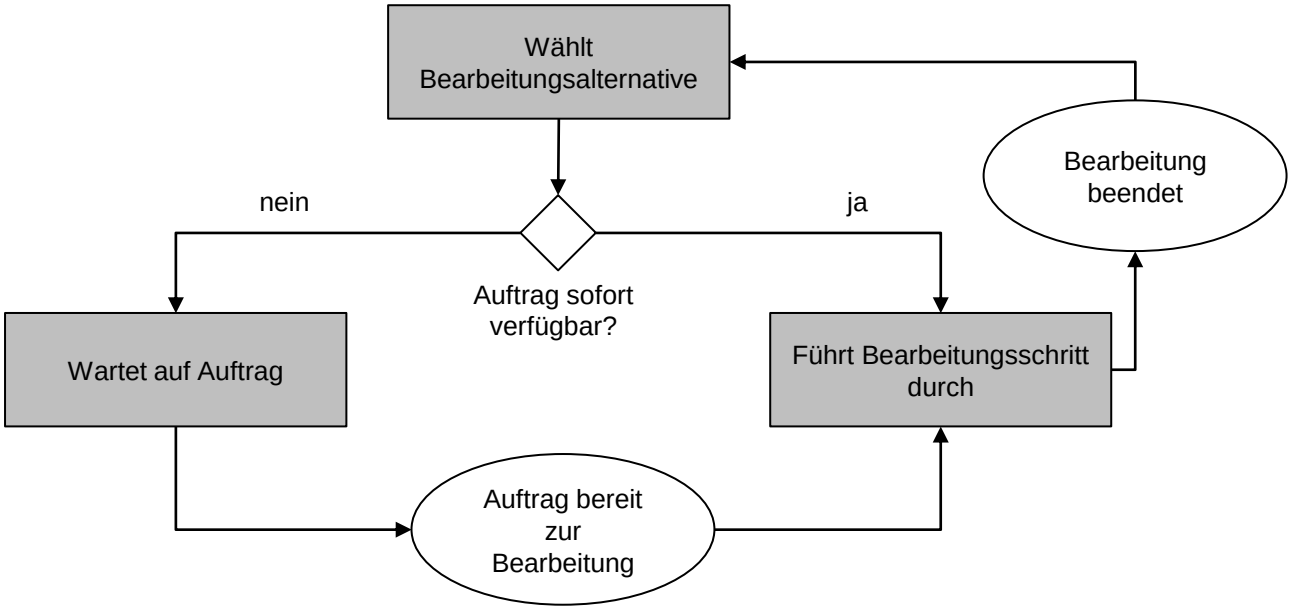


Abbildung 4-24: Bearbeitungszustände einer Station i. A. an (A_Hort 2019)

Nach dieser Initialisierung kann das eigentliche reaktive Rescheduling in Schritt (2) durchgeführt werden. Dabei befindet sich eine Station hinsichtlich ihrer Aktionswahl zu jedem Zeitpunkt in einem der drei in Abbildung 4-24 dargestellten Zuständen: Wenn die Station keinen Auftrag bearbeitet, wählt sie eine Bearbeitungsalternative aus ihrem Aktionsraum A_i gemäß ihrer Strategie π_s. Wenn kein Auftrag zur Verfügung steht, wartet die Station, bis erneut ein Auftrag verfügbar ist.

Je nachdem, ob der gewählte Auftrag sofort verfügbar ist, z. B. wenn dieser denselben Auftragstyp hat und sich bereits im Puffer der Station befindet, oder nicht, wartet die Station als nächstes bzw. führt den Bearbeitungsschritt aus. Aufgrund der berücksichtigten Anforderungs-, Transport- und reihenfolgeabhängigen Rüstzeiten steht ein Auftrag meist nicht sofort zur Bearbeitung zur Verfügung, sondern die Wartezeit muss eingehalten werden. Diese hängt davon ab, zu welchem Zeitpunkt ein Auftrag gewählt wird. Befindet sich dieser noch an einer Vorgängerstation in Bearbeitung, wird nach Formel 4-42 die verbleibende Restbearbeitungszeit $p_{j,s}^{k_{j,R}}$ des Auftrags miteinbezogen. Wartet der Auftrag an der Vorgängerstation auf ein Transportmittel, muss hingegen die Restanforderungszeit rt_s^R berücksichtigt werden. Ist der Auftrag bereits auf dem Weg zu der Station, wird die Resttransportzeit $t_{j,s}^R$, herangezogen. Wartet der Auftrag schließlich bereits im Puffer der Station, kann die Bearbeitung direkt nach der Rüstzeit $r_{s,j,j}$, erfolgen. Dadurch ist die Entscheidung für einen Auftrag und dessen Bearbeitung an

einer Station zeitlich entkoppelt. Die Auswahl eines Auftrags, der sich noch nicht an einer Station befindet, ermöglicht die Erstellung aktiver Schedules.

$$x_{j,s}^{k_j} = x_{j,s}^{k_j} + max\left(p_{j,s}^{k_{j,R}} + rt_s^R + t_{j,s\prime},, r_{s,j,j\prime}\right), \quad \forall j, j' \in J, s, s' \in S, k_j \in K_j \tag{4-42}$$

Wartet eine Station auf einen Auftrag, beginnt sie nach der jeweiligen Wartezeit mit der Durchführung des Bearbeitungsschrittes. Sobald die Bearbeitung beendet ist, wählt die Station die nächste Bearbeitungsalternative aus. Dieser Zyklus legt fest, zu welchen Zeitpunkten eine Station sich für eine Bearbeitungsalternative entscheiden darf.

Die Wahrscheinlichkeit für die Auswahl einer Bearbeitungsalternative aus dem Aktionsraum wird anhand der Strategie π_s je Agent mit Formel 4-43 berechnet. Wenn A_s leer ist, ist die Wahrscheinlichkeit für alle Bearbeitungsalternativen null. Andernfalls werden die Wahrscheinlichkeiten gemäß der Boltzmann-Exploration anhand der Softmax-Funktion bestimmt (Thrun 1992). Dies sorgt dafür, dass die Aktionen entsprechend ihrer Wahrscheinlichkeit, eine positive Belohnung zu erzielen, gewichtet werden. Dabei werden im Zähler alle Wahrscheinlichkeiten $e^{-\theta_{jt_{j,s}}^{k_j}}$ addiert, die zum selben Auftragstyp wie $m_{j,s}^{k_j}$ gehören. Die Normierung über den Nenner sorgt dafür, dass diese in Summe eins ergeben. Damit ist die Wahrscheinlichkeitsverteilung vollständig definiert und die Strategie π_s führt immer zur Auswahl einer Bearbeitungsalternative. Sollten mehrere Bearbeitungsalternativen von unterschiedlichen Aufträgen vorliegen, die zum selben Auftragstyp gehören und denselben Bearbeitungszustand haben, ist deren Auswahlwahrscheinlichkeit gleich groß und es wird zufällig gewählt. Dies hat keinen Einfluss auf die Minimierung der Makespan im Rescheduling. Während des Anlernens von DPS wird die zu wählende Aktion gemäß der berechneten Wahrscheinlichkeit ermittelt. Bei der nachfolgenden Anwendung wird in Schritt (5) ein Maximum Likelihood Schedule (MLS) erstellt, der jeweils die Aktion mit der höchsten Wahrscheinlichkeit wählt.

$$\pi_s(m_{j,s}^{k_j}, z_s|\theta) = \begin{cases} \dfrac{\sum_{\forall m_{i,s}^{k_i} \in A_{s:jt_i = jt_j}} e^{-\theta_{jt_{i,s}}^{k_i}}}{\sum_{\forall m_{j,s}^{k_j} \in A_s} e^{-\theta_{jt_{j,s}}^{k_j}}}, wenn\ A_s \neq \emptyset \\ 0, sonst \end{cases} \tag{4-43}$$

Beispielhaft sei angenommen, dass an einer Station drei Aufträge warten und ausgewählt werden können. Die Aufträge gehören zu zwei unterschiedlichen Auftragstypen A und B und die beiden Aufträge 1 und 2 desselben Auftragstyps A befinden sich im selben Bearbeitungszustand. Außerdem sei $\theta^1_{A,s} = 1$ und $\theta^1_{B,s} = 3$. Dann gilt: $\pi_s(m^1_{1,s}, z_s|\theta) = \frac{e^{-1}+e^{-1}}{e^{-1}+e^{-1}+e^{-3}} = 0{,}9366$ und $\pi_s(m^1_{3,s}, z_s|\theta) = \frac{e^{-3}}{e^{-1}+e^{-1}+e^{-3}} = 0{,}0634$. Das heißt mit einer Wahrscheinlichkeit von ca. 94 % wird Auftragstyp A gewählt und mit ca. 6 % Auftragstyp B. Da sich beide Aufträge mit Auftragstyp A im selben Bearbeitungszustand befinden, würde zufällig einer der beide zugrundeliegenden Aufträge gewählt.

4.5.2.2 Interaktion der Agenten

Im Folgenden wird auf die Interaktion der Agenten eingegangen. Dazu ist in Abbildung 4-25 für einen einzelnen Zeitpunkt ein Beispiel mit zwei Stationstypen und drei Stationen dargestellt. Es wird angenommen, dass Station 1 einen Auftrag bearbeitet und Station 2 und 3 mögliche Nachfolger sind.

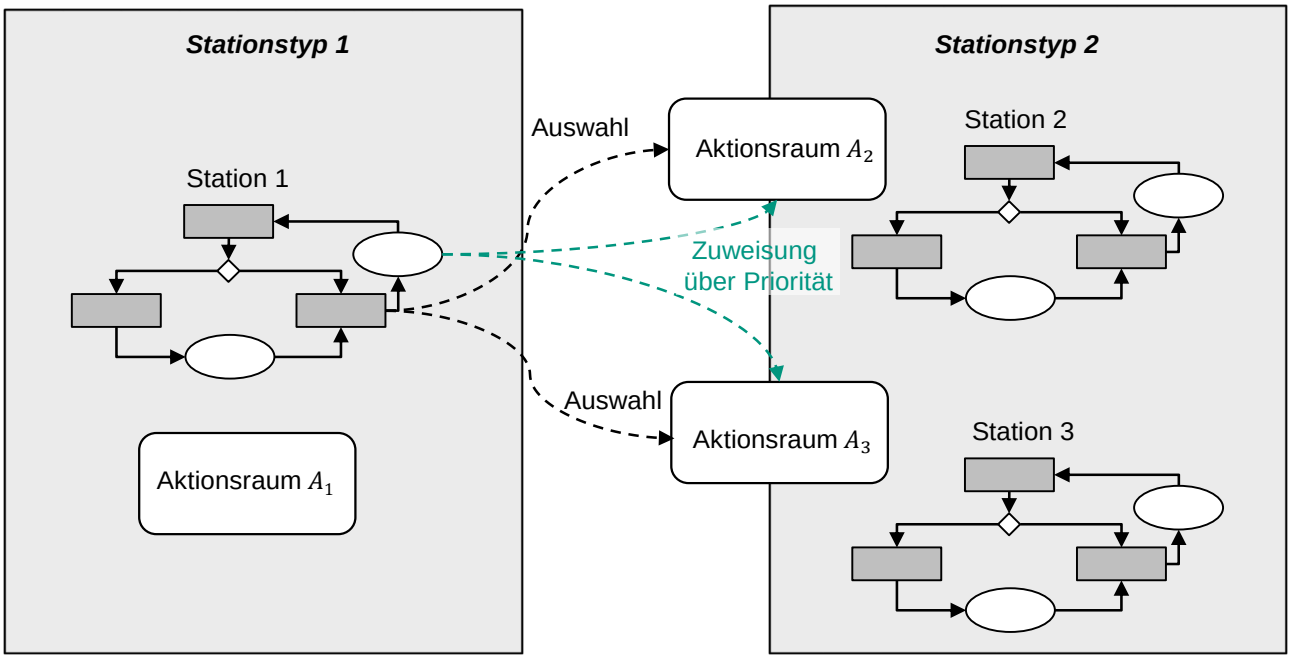


Abbildung 4-25: Interaktion zwischen Stationen i. A. an (A_Hort 2019)

Bei Bearbeitungsbeginn an Station 1 werden die Bearbeitungsalternativen den Aktionsräumen A_2 und A_3 hinzugefügt. Beendet Station 2 oder 3 ihre vorhergehende Bearbeitung, können sie auf Basis von Formel 4-43 den nachgelagerten Bearbeitungsschritt dieses Auftrags auswählen, müssen aber noch die durch Formel 4-42 ermittelte Zeit bis zum Bearbeitungsbeginn warten. Sollten Station 2 und 3 zum selben Zeitpunkt wählen,

kommt der sequenzielle Charakter von DPS zum Tragen. Da die Aktionen der Agenten den Aktionsraum anderer Agenten beeinflussen, ist eine simultane Entscheidung nicht möglich. In dem Fall würde der Agent mit dem kleineren Index zuerst den nächsten Auftrag wählen. Sollte es sich dabei um den an Station 1 in Bearbeitung befindlichen Auftrag handeln, so würde dieser aus A_3 entfernt, bevor Station 3 ihren nächsten Auftrag wählt. Wenn bis zum Ende der Bearbeitung an Station 1 keine nächste Station ermittelt wurde, wird aus der Menge möglicher Nachfolgerstationen diejenige ausgewählt, für die die Ankunftszeit als Summe der Anforderungs- und Transportzeit minimal ist und der Auftrag zu dieser transportiert. Das heißt nicht, dass der Auftrag als nächstes dort bearbeitet wird sondern sorgt lediglich dafür, dass der Transport zu der ausgewählten Station angestoßen wird und der Auftrag aus den Aktionsräumen der nicht gewählten Stationen entfernt wird.

Die beschriebene Interaktion zwischen den Agenten wiederholt sich über den gesamten Planungshorizont hinweg und ist in Abbildung 4-26 dargestellt. Die Interaktion aus Abbildung 4-25 erhält nun eine zusätzliche zeitliche Dimension, die zeigt, dass gewählte Aktionen in einem Zustand z_1 den Aktionsraum wiederum nachfolgender Stationen in z_{1+x} beeinflussen. Die Aktionsräume aller Agenten werden also durch Entscheidungen in jedem Zustand fortlaufend angepasst. Nach diesem Vorgehen werden in jedem Zeitschritt die Menge der Agenten im Zustand „wählt Auftrag aus“ ermittelt, die Aufträge nach Formel 4-43 ausgewählt und in die Aktionsräume aller möglichen Nachfolger transferiert. Für die Agenten in den Zuständen „wartet auf Auftrag“ und „bearbeitet Auftrag“ wird die verbleibende Warte- bzw. Prozesszeit um die Länge des Zeitschritts verringert. DPS wird beendet, wenn alle im Rescheduling-Korridor enthaltenen Bearbeitungsschritte durch Bearbeitungsalternativen an Stationen neu eingeplant wurden.

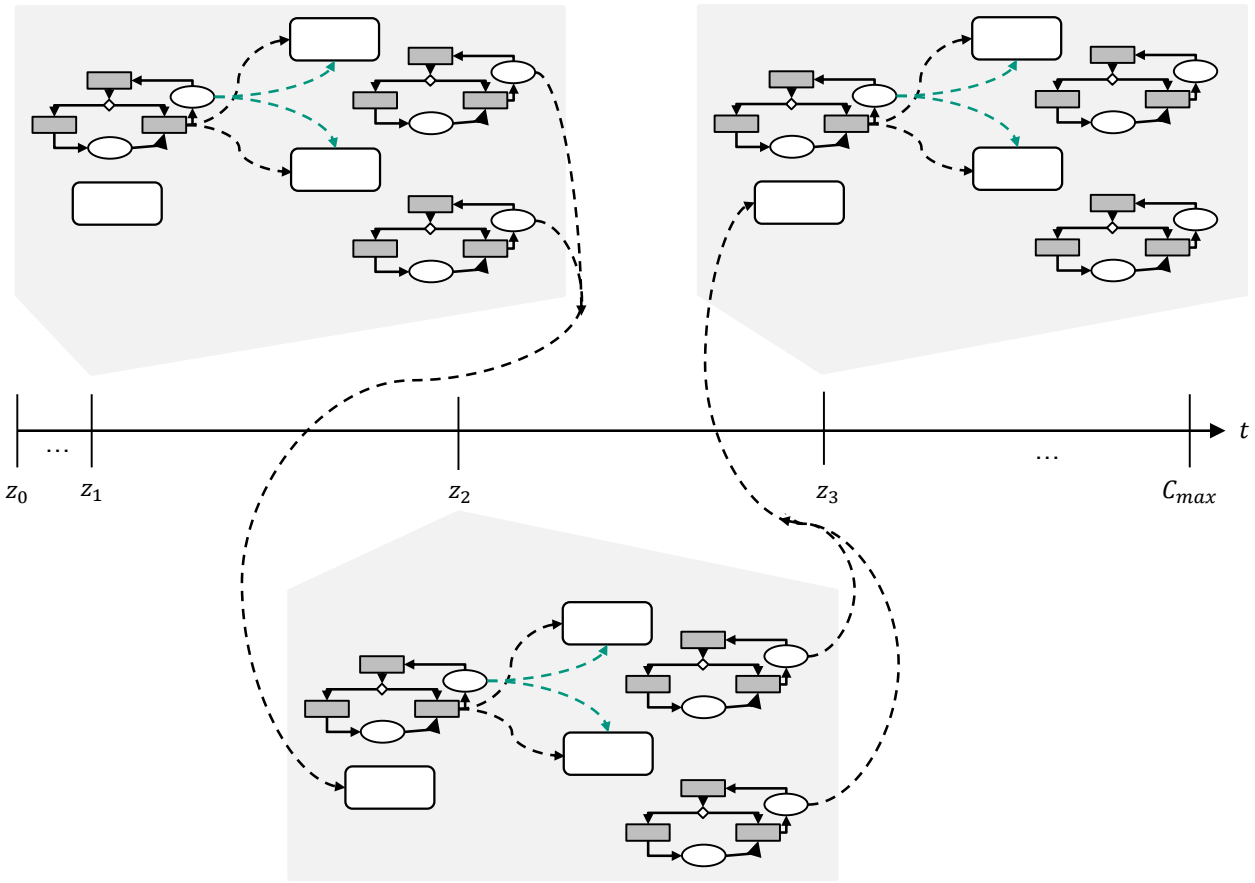


Abbildung 4-26: Interaktion zwischen Stationen über den Planungshorizont hinweg

4.5.2.3 Lernmechanismus des Lösungsverfahrens

Bisher wurde der logische Ablauf der DPS für das reaktive Rescheduling anhand der Schritte (1), (2) und (5) in Abbildung 4-23 beschrieben. Wesentlich ist dafür die Matrix θ zur Berechnung der nächsten Aktion. Im Folgenden werden daher die Schritte (3) und (4) in Anlehnung an Gabel & Riedmiller beschrieben, um das Anlernen der Matrix θ darzustellen.

Während des Anlernens wird in Schritt (3) in jedem Zeitschritt unabhängig davon, ob ein Agent eine Aktion gewählt hat, eine Belohnung $r_t = -1$ ausgezahlt, damit zum Ende des Planungshorizonts die gesamte Belohnung für alle Agenten der negativen Makespan entspricht. Anschließend wird die gewählte Aktion in die Historie h_s aufgenommen, sodass für das Lernen in Schritt (4) rekonstruiert werden kann, welcher Agent in welchem Zustand welche Aktion gewählt hat (Oliehoek & Amato 2016).

In Schritt (4) schließlich werden die $\theta^{k_j}_{jt_j,s}$ aktualisiert und der Episodenzähler erhöht. Nachfolgend wird beschrieben, wie die Aktualisierung der $\theta^{k_j}_{jt_j,s}$ berechnet wird. Optimierungsziel von DPS ist die Maximierung der erwarteten Performance der Strategie

π_s nach Formel 2-11, sodass der Makespan minimiert wird. Für das reaktive Rescheduling zur Minimierung des Makespan kann dies mit Formel 4-44 dargestellt werden. Die Leistungsfähigkeit $L(\pi_s)$ einer Strategie π_s entspricht dem Erwartungswert der Makespan $-C_{max}(\pi_s)$.

$$L(\pi_s) = E[-C_{max}(\pi_s)] \quad \text{4-44}$$

Zur Bewertung der Veränderung der Leistungsfähigkeit wird deren Gradient nach θ gebildet. Dies erfolgt nach (Gabel & Riedmiller 2012) durch Formel 4-45. Dabei steht e für eine Episode, d. h. einen kompletten Durchlauf des Algorithmus über den Planungshorizont hinweg, bis alle Aufträge vollständig eingeplant sind. Die im Laufe einer Episode vorgefundenen Zustände und die daraufhin gewählten Aufträge werden in der Historie h_s (Oliehoek & Amato 2016) gespeichert. Dies entspricht Schritt (3) in Abbildung 4-23. Der Erwartungswert $E[-C_{max}(\pi_s)]$ ergibt sich aus der Makespan $-C_{max}(e)$ einer spezifischen Episode gewichtet mit der Wahrscheinlichkeit $\pi_s(a_{s,t}, z_{s,t}|\theta)$ aus der Strategie π_s, mit der eine Episode unter Berücksichtigung der θ so auftritt. Zusammengefasst kann dies als $E[-C_{max}(e) * \ln \pi_s(a_{s,t}, z_{s,t}|\theta)]$ geschrieben werden.

$$\frac{dL(\pi_s)}{d\pi_s} = E[-C_{max}(e) * \ln \pi_s(a_{s,t}, z_{s,t}|\theta)] \quad \text{4-45}$$

Der Term $\ln \pi_s(a_{s,t}, z_{s,t}|\theta)$ wird nach Ende einer Episode e (Gabel & Riedmiller 2012) mithilfe der Historie h_s nach Formel 4-46 ausgewertet (Schritt (4) in Abbildung 4-23). Dazu werden die natürlichen Logarithmen des Gradienten der Strategie über alle Schritte aufsummiert (Peshkin & Kim et al. 2000). Dabei entspricht T nicht dem zeitlichen Horizont C_{max}, sondern der je Agent individuellen Anzahl an Zeitpunkten, in denen er eine Bearbeitungsalternative gewählt hat. Für jede Entscheidungssituation wird die Historie h_s bzgl. des vorliegenden Zustandes $z_{s,t}$ und der gewählten Aktion $a_{s,t}$ ausgewertet. Für jede Entscheidungssituation, in der eine Aktion vorliegt und gewählt wurde, wird der entsprechende Gradient um die Gegenwahrscheinlichkeit der Wahl dieser Aktion erhöht, andernfalls wird er um die Wahrscheinlichkeit der Wahl der Aktion verringert. Das heißt, der Wert der Ableitung für eine Aktion an einer Station in einem Zustand wird jedes Mal um die Auswahlwahrscheinlichkeit verringert, wenn die Aktion in dem Zustand nicht gewählt wurde. Wenn die Aktion in dem Zustand jedoch gewählt wurde,

wird die Ableitung um die Gegenwahrscheinlichkeit erhöht. So werden aus der Historie die in Zuständen gewählten Aktionen mit der erzielten Makespan in Formel 4-45 verknüpft.

$$\ln \pi_s(a_{s,t}, z_{s,t}|\theta) = \begin{cases} 1 - \pi_s(a_{s,t}, z_{s,t}|\theta), wenn\ a\ =\ a_{s,t} \\ -\pi_s(a_{s,t}, z_{s,t}|\theta), sonst \end{cases} \quad 4\text{-}46$$

Der Gradient kann mit einer unendlichen Anzahl von Episoden nicht ermittelt werden, daher wird für die Umsetzung mit einer fixen Anzahl von Epochen E eine Schätzungsfunktion verwendet (Gabel & Riedmiller 2012). Zur Verringerung der Varianz wird der Makespan um die durchschnittliche Leistungsfähigkeit $\bar{L}(\pi_s)$ der Strategie π_s korrigiert (Greensmith & Bartlett et al. 2004). Damit ergibt sich die Schätzung des Gradienten nach Formel 4-47.

$$\frac{dL(\pi_s)}{d\pi_s} = \frac{1}{E}\sum_{k=1}^{E}[(-C_{max}(e_k) - \bar{L}(\pi_s)) * \sum_{t=0}^{T_e} ln\, \pi_s(a_{s,t}, z_{s,t}|\theta) \quad 4\text{-}47$$

Nachdem eine Epoche E durchlaufen wurde, wird θ schließlich nach Formel 2-13 aktualisiert. Für einen beispielhaften Rescheduling-Korridor RK ist in Abbildung 4-27 die Entwicklung des Makespans auf der Ordinate in Abhängigkeit von der Anzahl ausgeführter Update-Schritte auf der Abszisse abgetragen. Für das Beispiel wurde eine Epochenlänge von $e = 100$ Episoden, eine maximale Anzahl von Epochen $E_{max} = 100$ und eine Lernrate $\delta = 0{,}01$ gewählt. Die Optimierung der Lernparameter ist anwendungsfallspezifisch und wird hier daher nicht weiter betrachtet.

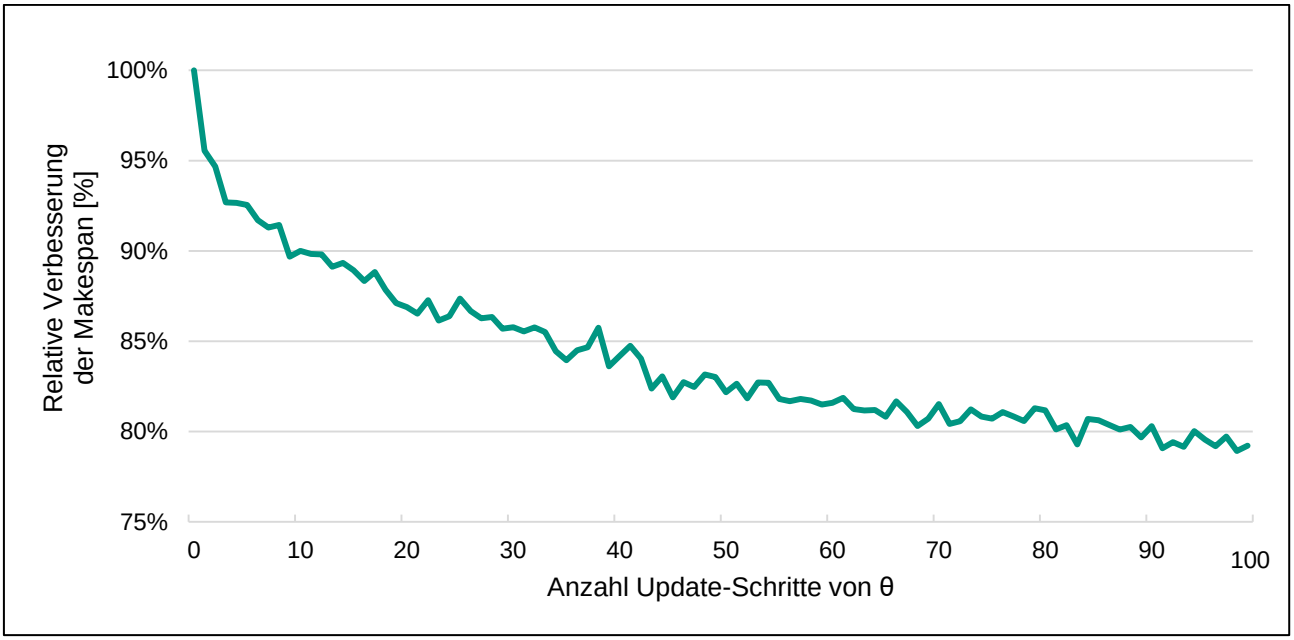


Abbildung 4-27: Beispielhafte Entwicklung des Makespans C_{max} in Abhängigkeit von der Anzahl durchgeführter Update-Schritte von θ

Für einen einzelnen Agenten ist in Abbildung 4-28 die Entwicklung der Parameter θ über die Anzahl der Update-Schritte dargestellt. Die insgesamt neun farbigen Linien stellen dar, dass in dem zugrundeliegenden Rescheduling-Korridor insgesamt neun unterschiedliche Kombinationen von Auftragstyp jt und Bearbeitungszuständen k_j an der Station vorliegen. Zu Beginn sind alle Parameter mit 0 initialisiert und es lässt sich erkennen, wie sich im Zeitverlauf eine davon abweichende positive oder negative Ausprägung ergibt. Für die Erstellung von Maximum-Likelihood-Schedules in der Anwendung sind die konkreten Werte von θ nicht ausschlaggebend, sondern deren Rangfolge untereinander. Unter den maximal neun verfügbaren unterschiedlichen Aktionen einer Station wird im Maximum-Likelihood-Schedule immer der geringste Wert gewählt. Das heißt z. B. für die $\theta = -1{,}6$ (dunkelrote Kurve), dass diese Aktion immer gewählt wird, wenn eine entsprechende Bearbeitungsalternative vorliegt. Die Aktion für $\theta = 1{,}4$ (hellgrüne Linie) hingegen wird an dieser Station nur dann gewählt, wenn kein anderer Auftrag verfügbar ist. In der Anwendung stellen die θ also eine stationsspezifische Prioritätsregel dar, die angibt, wie Bearbeitungsalternativen an den Stationen priorisiert werden sollen, um einen minimalen Makespan zu erhalten. Die Lösungsgüte des beschriebenen Lösungsverfahrens hängt maßgeblich von einem geeigneten Anlernen ab. Dieses muss es erlauben, für spezifische Rescheduling-Korridore eine Strategie zu erlernen, die möglichst geringfügig vom Optimum abweicht.

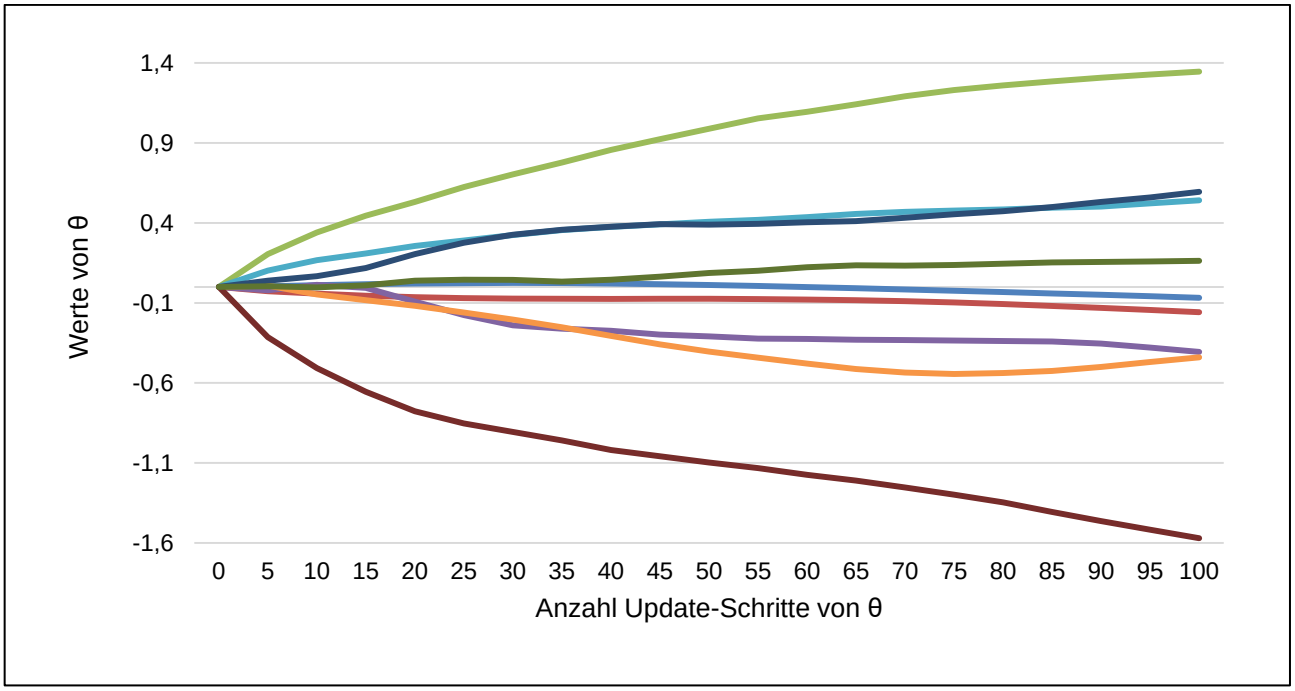


Abbildung 4-28: Beispielhafte Entwicklung von θ in Abhängigkeit von der Anzahl durchgeführter Update-Schritte von θ für einen Agenten

Für das Anlernen spezifischer Rescheduling-Korridore müssen der Lernparameter δ, die Anzahl zu durchlaufender Episoden E zwischen zwei Update-Schritten von θ sowie die maximale Anzahl zu durchlaufender Episoden E_{max} ermittelt werden. Zudem ist die Festlegung eines Abbruchkriteriums ω erforderlich. Damit kann das Anlernen vor Erreichen von E_{max} abgebrochen werden, falls keine Verbesserung der Lösung mehr erreicht werden kann. Dies kann dann der Fall sein, wenn die Lösung für einen Rescheduling-Korridor deutlich vor Erreichen von E_{max} konvergiert bzw. ein oszillierendes Verhalten aufweist. Unter Konvergenz wird hier das Auftreten eines stationären Verhaltens bezeichnet, bei dem sich die θ in einem Update-Schritt um weniger als ω % verändern. In diesem Fall kann durch ein geeignetes Abbruchkriterium Rechenzeit eingespart werden, ohne die Lösungsgüte zu verringern. Im Rahmen der vorliegenden Arbeit werden diese Parameter empirisch ermittelt, da dazu keine allgemeingültigen Vorgehensweisen existieren. Das Ergebnis des reaktiven Reschedulings ist gültig, sobald dessen Makespan unterhalb der Länge des Rescheduling-Korridors RK liegt. Es kann dabei vorkommen, dass eine Überlappung von Bearbeitungsschritten an der oberen Grenze auftritt. In dem Fall werden die betroffenen Bearbeitungsschritte im zugrundeliegenden Schedule auf der Zeitachse durch Right Shifting verschoben.

5 Erprobung und prototypische Softwarerealisierung

Die entwickelte Methode zum prädiktiv-reaktiven Scheduling wurde im Rahmen des durch das Bundesministerium für Wirtschaft und Energie (BMWi) geförderten Verbund-Forschungsprojektes „SmartBodySynergy - Smarte Rohbauzellen für einen synergetischen Hochlauf elektrifizierter Fahrzeuge“ (Fkz. 01MX15007) entwickelt und bei einem Partnerunternehmen aus der Automobilindustrie in einer Matrix-Produktion für einen Teilbereich des Karosseriebaus erprobt. Für die Erprobung wurde eine Ablaufsimulation der konzipierten Matrix-Produktion in der Software Plant Simulation® von Siemens aufgebaut, die als Testumgebung verwendet wird.

Nachfolgend wird in Kapitel 5.1 die grundlegende Systemstruktur beschrieben und dabei auf den Aufbau des Produktionssystems und der Produktionsprogramme eingegangen sowie deren Zusammenwirken im Rahmen der entwickelten Ablaufsimulation dargestellt. Das prädiktive robuste Scheduling (Kapitel 4.3) wird in Kapitel 5.2 erprobt. Dabei werden unterschiedlich robuste Schedules erzeugt und ohne Berücksichtigung des reaktiven Reschedulings in der Ablaufsimulation durchgeführt. Über eine Sensitivitätsstudie wird der Einfluss der $MTTR_s$ auf die Effektivität des robusten Schedulings aufgezeigt. Das reaktive Rescheduling (Kapitel 4.5) wird in Kapitel 5.3 erprobt. Hierzu werden die Ergebnisse nach Anlernen simulativ erzeugter Störungen vorgestellt und einer mathematischen Optimierung sowie Prioritätsregeln gegenübergestellt. In Kapitel 5.4 schließlich wird untersucht, wie sich die Robustheit im prädiktiven Scheduling auf das reaktive Rescheduling und damit auf die Verschiebung von Bearbeitungsschritten sowie die Verspätung von Aufträgen auswirkt. Es wird zudem exemplarisch ein optimaler Robustheitswert für den vorliegenden Anwendungsfall simulativ ermittelt, bevor abschließend die wesentlichen Funktionsbausteine des entwickelten Softwareprototypen in Kapitel 5.5 beschrieben werden.

5.1 Systemstruktur

5.1.1 Aufbau des Produktionssystems

Die Einordnung des Karosseriebaus in der Fahrzeugproduktion ist in Abbildung 5-1 übersichtsartig dargestellt: Auf Basis einer Kundenbestellung wird die Auftragseinplanung vorgenommen, die den Produktionsprozess auslöst. Zu Beginn steht das Presswerk, in dem die benötigten Karosserieteile gefertigt werden. Im Karosseriebau werden

anschließend die Karosserieteile zur Gesamtkarosserie gefügt. Diese wird anschließend lackiert, bevor schließlich in der Montage der Antriebsstrang und das Interieur montiert werden. Die einzelnen Produktionsschritte sind durch Puffer entkoppelt, um eine jeweils optimierte Reihenfolgebildung zu ermöglichen. Über die Distribution wird abschließend die Übergabe an den Kunden realisiert und der Auftragsabwicklungsprozess ist abgeschlossen. (Meissner 2009)

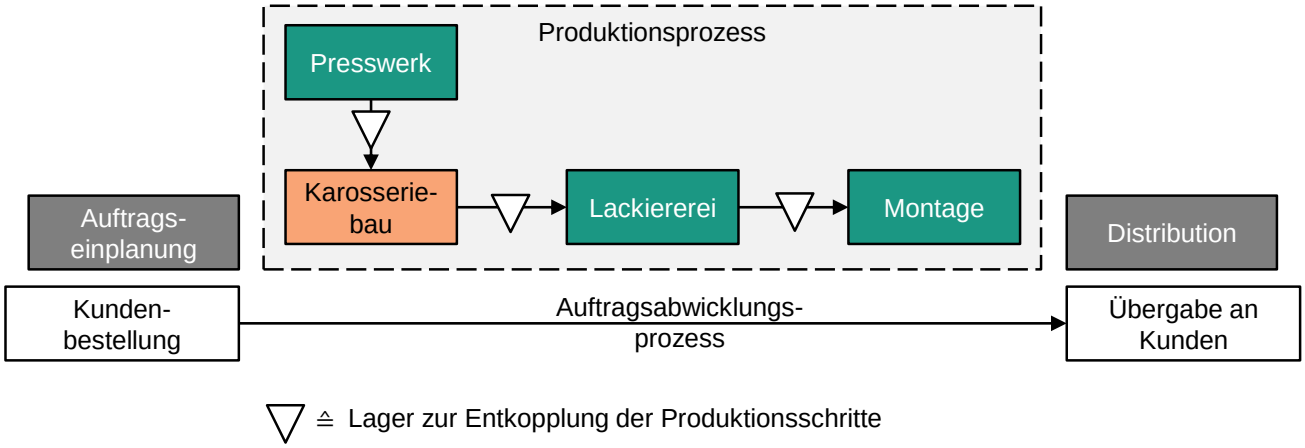


Abbildung 5-1: Positionierung des Karosseriebaus in der Fahrzeugproduktion (Meissner 2009)

Betrachtungsgegenstand der vorliegenden Arbeit ist die Fertigung von Heckwagen (HeWa) und Vorderwagen (VoWa) innerhalb der Bodengruppenfertigung eines automobilen Karosseriebaus. Für die heute noch variantenspezifischen und starr verketteten Fließfertigungssysteme wurden im Forschungsprojekt SmartBodySynergy die Matrix-Produktion als Alternative in unterschiedlichen Ausbaustufen untersucht, um zukünftigen Anforderungen bezüglich steigender Variantenzahlen und schwankendem Variantenmix bei gleichzeitig heterogeneren Produkten zu begegnen. Wesentliche Herausforderungen bestanden dabei sowohl in der technischen Ausgestaltung der Stationen aufgrund der hohen Flexibilitätsanforderungen sowie in der organisatorischen Realisierung der Produktionssteuerung der flexiblen und vom Takt entkoppelten Produktion mit Losgröße 1.

Die Systemgrenze der Heckwagen- und Vorderwagenfertigung ist in Abbildung 5-2 dargestellt. Die benötigten Bleche werden JIS in der Produktion zur Verfügung gestellt und die produzierten Vorder- und Heckwagen anschließend per JIS-Anlieferung an die Hauptlinie übergeben, um sie gemeinsam mit dem Unterboden zur Bodengruppe zu fügen.

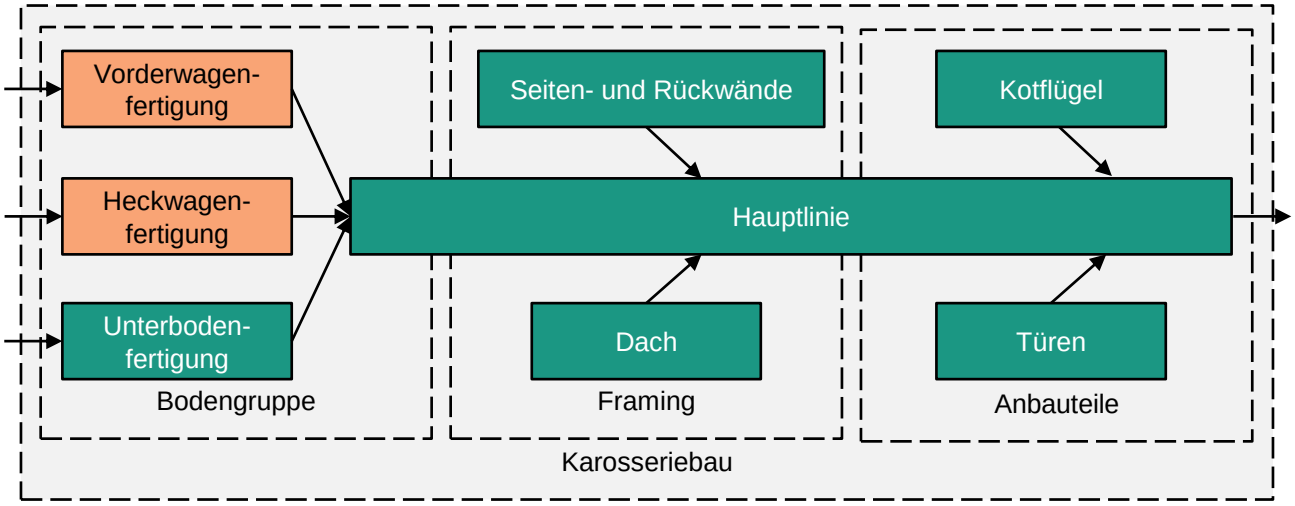


Abbildung 5-2: Positionierung der Vorder- und Heckwagenfertigung im Karosseriebau i.A. an (Wemhöner et al. 2006)

Die konzipierte Matrix-Produktion ist in Abbildung 5-3 dargestellt. Beim Fügen der Vorder- und Heckwagen kann grundsätzlich zwischen Geometrie- und Ausfügestationen unterschieden werden. An den Geometriestationen werden diejenigen Fügepunkte gesetzt, die notwendig sind, um eine ausreichende Steifigkeit des Zusammenbaus für den Transport zu erreichen. Dabei müssen die Bleche exakt zueinander positioniert werden, da sich ihre Lage anschließend nicht mehr verändern lässt. Geometriestationen sind zudem ausschließlich entweder für Varianten von Heck- oder Vorderwagen verwendbar. In den technologiespezifischen Ausfügestationen werden anschließend weitere Fügeverbindungen hergestellt, sodass die Verbindung der Bleche die nötige Steifigkeit für den Betrieb aufweist. Die Ausfügestationen sind produktunabhängig und beinhalten die in Tabelle 5-1 dargestellten Fügetechnologien mit den hinterlegten Verfügbarkeiten.

Tabelle 5-1: Betrachtete Fügetechnologien

Abkürzung	Bezeichnung	Verfügbarkeit
WPS	Widerstandspunktschweißen	98,31 %
HSN Typ 1/ Typ 2	Halbhohlstanznieten (Werkzeug groß – Typ 1/ Werkzeug klein – Typ 2)	96,70 %
FLS	Fließlochschrauben	97,33 %
Impact	Hochgeschwindigkeits-Bolzensetzen	95,72 %
Geo (VoWa/HeWa)	Geometriestation (Vorderwagen/Heckwagen)	100,00 %

Während die Geometriestationen jeweils nur ein einziges Mal vorhanden sind, können Ausfügestationen redundant sein. Es existieren in dem betrachteten System vier WPS-

Stationen, jeweils zwei Stationen mit HSN Typ 1, HSN Typ 2 und FLS sowie eine Impact-Station. Für den Bereich der Heck- bzw. Vorderwagenfertigung gibt es zudem fünf bzw. sechs unterschiedliche Geometriestationen (Geo1 HeWa bis Geo5 HeWa bzw. Geo1 VoWa bis Geo6 VoWa). Jede Station verfügt über einen physischen Puffer von 5 Karossen, über den die Produktion von Störungen an Stationen zur Stabilisierung der Ausbringungsmenge entkoppelt werden kann.

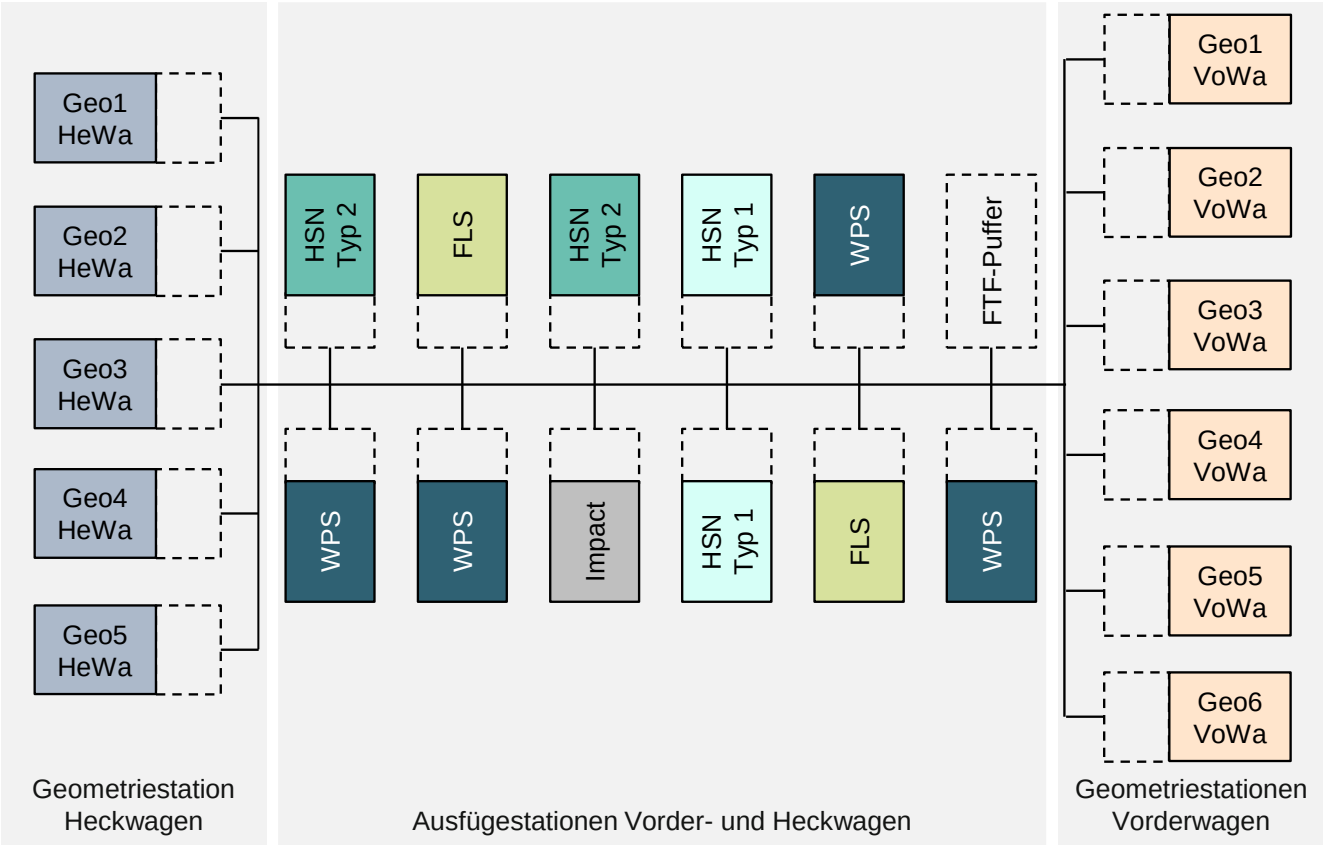


Abbildung 5-3: Schematische Darstellung der Matrix-Produktion im Projekt SmartBodySynergy i.A. an (A_Hort 2019)

Der Transport zwischen den Stationen wird mithilfe von FTF realisiert, die jeweils eine Karosserie laden können. Die Dimensionierung der Anzahl benötigter FTF wurde simulativ durchgeführt und anhand repräsentativer Produktionsprogramme auf 30 FTF festgelegt, sodass ein Durchsatz von 80 Vorder- und Heckwagen pro Stunde erreicht werden kann. Dies entspricht einer Einsteuerung eines Vorder- bzw. Heckwagens alle $\frac{3600s}{40} = 90\ s$. Die Verteilung der Stationen im Ausfügebereich wurde mithilfe eines quadratischen Zuordnungsproblems ermittelt, sodass die Gesamtfahrstrecke für einen bestimmten Variantenmix minimal und relativ unempfindlich gegenüber dessen Schwankungen ist. Die Stationen sind über Fahrwege so verbunden, dass Transporte

zwischen allen Stationen realisiert werden können. Zudem existiert ein FTF-Puffer, in dem nicht benötigte FTF zwischenzeitlich geparkt werden können.

Die FTF-Steuerung ist nicht auftragsgebunden, d. h. ein FTF begleitet einen Auftrag in der Regel nicht durch die gesamte Produktion, sondern führt auftragsunabhängig Transportaufträge durch. Diese umfassen einen Auftrag, eine Abholstation sowie eine Zielstation. Nachdem ein FTF einen Auftrag zu einer Station transportiert hat, wird ihm aus einer zentralen Liste der örtlich am nächsten gelegene Transportauftrag zur Abarbeitung übermittelt. Warten keine Transportaufträge, fährt das FTF in den FTF-Puffer. Im Gegenzug sucht ein Transportauftrag erst das nächste verfügbare FTF und wird nur dann auf der zentralen Warteliste eingetragen, wenn kein FTF verfügbar ist. Zur Entkopplung der Transporte von der Bearbeitung werden die physischen Puffer an den Stationen verwendet. Um die Wartezeiten auf FTF an den Stationen zu minimieren, werden FTF bereits vor Bearbeitungsende angefordert, sodass diese möglichst direkt abgeholt werden können. Für das vorliegende Produktionssystem hat sich durch Simulationsexperimente empirisch eine Anforderung maximal 45 s vor Bearbeitungsende als zielführend heraus gestellt. Die danach dennoch verbleibenden Wartezeiten entsprechen den stationsspezifischen Anforderungszeiten rt_s, die in Kapitel 4.3.1 eingeführt wurden. Das simulationsbasierte Vorgehen zu deren Ermittlung ist in Anhang A1 dargestellt.

Die Systemgrenzen der Matrix-Produktion sind durch die Materialbereitstellung an den Geometriestationen für Vorder- bzw. Heckwagen sowie dem Übergabepunkt zur Hauptlinie definiert. Eine Materialbereitstellung an den Ausfügestationen wird nicht benötigt, da dort lediglich die Steifigkeit eines bereits gefügten Zusammenbaus über das Setzen weiterer Fügepunkte erhöht wird. Am Übergabepunkt zur Hauptlinie befindet sich ein kapazitiv unbegrenzter Sortierpuffer, um die unabhängig voneinander durch das Produktionssystem laufenden Vorder- und Heckwagen synchronisiert an die Hauptlinie übergeben zu können. In den durchgeführten Experimenten sammeln sich jedoch im Sortierpuffer nie mehr als 8 Vorder- bzw. Heckwagen.

5.1.2 Aufbau der Produktionsprogramme

In der beschriebenen Matrix-Produktion können Vorder- und Heckwagen für eine Fahrzeugbaureihe gefertigt werden. Die Variantenbildung erfolgt anhand der folgenden frei kombinierbaren Kriterien:

- Antriebsstrang: konventionell (V) / elektrifiziert (E)
- Lenkung: Linkslenker (LL) / Rechtslenker (RL)
- Laststufe: Motorisierung Standard (LS1) / Sport (LS2)

Der Antriebsstrang wirkt sich auf die Karosserie aus, da je nach gewähltem Konzept z. B. kein Mitteltunnel im Unterboden erforderlich ist, stattdessen aber Halterungen und zusätzliche Versteifungen zur Aufnahme einer Batterie benötigt werden. Die Lenkung wirkt sich vor allem auf die Trennung zwischen Fahrgastzelle und Motorraum im Vorderwagen aus. Die Laststufe schließlich ermöglicht unterschiedliche Motorisierungen des Fahrzeugs, die wiederum angepasste Steifigkeiten von Vorder- und Heckwagen erfordern.

Es ergeben sich insgesamt $2^3 = 8$ unterschiedliche Heck- bzw. Vorderwagen und damit $2 * 2^3 = 16$ unterschiedliche Produkte zur Fertigung. Jedes Produkt folgt einem definierten Vorranggraphen, der beschreibt, welche Bearbeitungsschritte in welcher Reihenfolge durchzuführen sind. Ein exemplarischer Vorranggraph für einen Vorder- und einen dazugehörigen Heckwagen ist in Anhang A2 dargestellt. Es ist ersichtlich, dass sich die in Kapitel 5.1.1 beschriebene abwechselnde Reihenfolge aus Geometrie- und Ausfügeschritten ergibt. Zudem sind die Ausfügeschritte in wahlfreier Reihenfolge ausführbar, da sich die Zugänglichkeit zum Zusammenbau während des Ausfügens nicht ändert und keine sonstigen Effekte berücksichtigt werden müssen.

Da die Geometriestationen jeweils nur einmal existieren und zwangsläufig für jeden Vorder- bzw. Heckwagen verwendet werden, wird deren Bearbeitungszeit von 90s angenommen, um den Zu- und Abfluss der Produkte in die Matrix-Produktion zu modellieren. Betrachtet wird der Zeitraum einer Schicht von 8 Stunden, wobei aufgrund der vollständigen Automatisierung der Stationen keine Pausen- oder sonstige Übergangszeiten anzunehmen sind. Es ergibt sich daraus, dass je Schicht Vorder- und Heckwagen für $\frac{3600s/h}{90s} * 8\,h = 320$ Fahrzeuge produziert werden müssen. Dazu werden im Produktionsprogramm 640 Aufträge für entsprechende Vorder- und Heckwagen angelegt, wobei zu einem Fahrzeug gehörende Vorder- und Heckwagen denselben Fälligkeitstermin besitzen. Die Fälligkeitstermine für je ein Fahrzeug sind in Abständen von 90s gestaffelt und so terminiert, dass ohne Störungen gerade noch eine Termintreue von 100% realisiert werden kann. Dadurch wird der Effekt von Störungen auf die Verspätung von Aufträgen besonders deutlich. Die Zugänge in das Produktionssystem

werden über das prädiktive robuste Scheduling vorgegeben. Die Produktionsprogramme enthalten somit eine Menge zu fertigender Vorder- und Heckwagen, den dazugehörigen Fälligkeitsterminen, Vorranggraphen und den Informationen über parallele Stationen sowie deren Bearbeitungszeiten.

5.2 Anwendung des prädiktiven robusten Scheduling

Aufgabe des prädiktiven robusten Schedulings ist es nun, die in Kapitel 5.1.2 beschriebenen Produktionsprogramme in die Matrix-Produktion zur Fertigung von Vorder- und Heckwagen aus Kapitel 5.1.1 einzuplanen. Für die Untersuchungen wurde ein Anteil von Fahrzeugen mit elektrifiziertem Antriebsstrang von 10 % angenommen, wobei 80 % der batterieelektrischen Fahrzeuge und 60 % der konventionellen Verbrenner als Linkslenker ausgeführt sind. Außerdem sind 70 % der konventionellen Verbrenner in Laststufe 1 ausgeführt, während dies für 30 % der Fahrzeuge mit elektrifiziertem Antriebsstrang gilt. Damit ergeben sich für den Zeitraum einer Schicht die in Tabelle 5-2 dargestellten Stückzahlen.

Tabelle 5-2: Datengrundlage für das prädiktive robuste Scheduling

Antriebsstrang	Lenkung	Laststufe	Stückzahl
V	LL	LS1	162
V	LL	LS2	69
V	RL	LS1	41
V	RL	LS2	17
E	LL	LS1	7
E	LL	LS2	12
E	RL	LS1	4
E	RL	LS2	8

Die Anwendung der prädiktiven Scheduling ohne Berücksichtigung der Robustheit wird in Kapitel 5.2.1 durchgeführt. Anschließend wird unter Berücksichtigung der Robustheit in Kapitel 5.2.2 eine Simulationsstudie durchgeführt, um den Einfluss der $MTTR_s$ auf die erzielbaren Ergebnisse zu analysieren. Die Grenzen der Anwendung werden abschließend in Kapitel 5.2.3 diskutiert.

5.2.1 Anwendung des prädiktiven Scheduling

Durch Lösung des in Kapitel 4.3.1 beschriebenen Optimierungsmodells wurden die Produktionsprogramme in prädiktive Schedules überführt. Mit einer begrenzten Rechenzeit als Abbruchkriterium werden Schedules erzeugt, deren minimaler gewichteter

Schlupf ca. 240 s beträgt. Nach einer Rechenzeit von ca. 75 min können erstmals Ergebnisse mit positiven Zielfunktionswert erzielt werden (siehe Anhang A3). Diese langen Rechenzeiten sind zu erwarten, da es sich bei den Produktionsprogrammen um sehr große und NP-schwere Scheduling-Probleme handelt. Ein Abbruchkriterium ω muss in diesem Kontext rechenzeitbasiert sein, da die relative Abweichung zur optimalen relaxierten Lösung zu groß ist, um ein ω-Kriterium sinnvoll einsetzen zu können. Eine optimale Lösung kommt aus diesem Grund ebenfalls nicht infrage. Die minimale gewichtete Schlupfzeit von 240 s gehört zu einem Auftrag, dessen Gewichtung 0,952 beträgt. Es handelt sich dabei um einen Vorderwagen einer Verbrennerfahrzeugs mit Linkslenker ausgeführt für Laststufe 2 (VoWa_V_LL_LS2). Dieser durchläuft die in Tabelle 5-3 dargestellten Bearbeitungsschritte, aus denen sich die Gewichtung anhand der Verfügbarkeit der Stationen sowie der parallelen Stationen desselben Stationstyps ergibt.

Tabelle 5-3: Exemplarischer Produktionsdurchlauf VoWa_V_LL_LS2

Besuch k_j	Bearbeitungsschritt $o_{j,s}^{k_j}$	Verfügbarkeit $1 - \sum_{\forall m_{j,s}^{k_j}} (1 - v_s)$
1	VoWa_Geo1	$100\ \%$
2	VoWa_Geo2	$100\ \%$
3	WPS	$1 - (1 - 0{,}9831)^4 * 100\ \% = 99{,}99\ \%$
4	HSNTyp2	$1 - (1 - 0{,}967)^2 * 100\ \% = 99{,}89\ \%$
5	VoWa_Geo3	$100\ \%$
6	WPS	$1 - (1 - 0{,}9831)^4 * 100\ \% = 99{,}99\ \%$
7	HSNTyp1	$1 - (1 - 0{,}967)^2 * 100\ \% = 99{,}89\ \%$
8	VoWa_Geo4	$100\ \%$
9	Impact	$1 - (1 - 0{,}9572)^1 * 100\ \% = 95{,}72\ \%$
10	HSNTyp1	$1 - (1 - 0{,}967)^2 * 100\ \% = 99{,}89\ \%$
11	VoWa_Geo5	$100\ \%$
12	WPS	$1 - (1 - 0{,}9831)^4 * 100\ \% = 99{,}99\ \%$
13	FLS	$1 - (1 - 0{,}9733)^2 * 100\ \% = 99{,}93\ \%$
14	VoWa_Geo6	$100\ \%$
15	HSNTyp2	$1 - (1 - 0{,}967)^2 * 100\ \% = 99{,}89\ \%$
	Resultierende Gewichtung w_j	$\prod_{\forall o_{j,s}^{k_j}} \left(1 - \sum_{\forall m_{j,s}^{k_j}} (1 - v_s) \right) = 0{,}9523$

Die Gewichtung wird maßgeblich durch die Verwendung der Station Impact bestimmt, da diese nur einmal vorhanden ist und zudem eine vergleichsweise geringe Verfügbar-

keit besitzt. Aufträge von Varianten, die diese Station nicht verwenden, werden entsprechend höher gewichtet, sodass die tatsächlichen Schlupfzeiten dort geringer ausfallen. Dahinter steht die Überlegung, dass ein Auftrag vor allem dann von Schlupfzeiten im Schedule profitiert, wenn sich voraussichtlich entlang seiner Bearbeitung verfügbarkeitsbedingte Zeitverluste ergeben.

Für den Beispielauftrag bedeutet die Gewichtung von ca. 0,952 eine ungewichtete Schlupfzeit von 253 s, die gleichzeitig im betrachteten Schedule die minimale Schlupfzeit darstellt. Das heißt, dass im Schedule alle Aufträge so eingeplant sind, dass diese mindestens 253 s vor ihrem Fälligkeitstermin fertiggestellt sind.

Die Erzeugung von Schlupfzeiten ist nur dann möglich, wenn die Kapazitätsauslastung im betrachteten Zeitraum unter 100 % liegt. Im untersuchten Produktionsprogramm liegt die errechnete Kapazitätsauslastung der Stationstypen zwischen 67,14 % und 89,14 % (siehe Anhang A4). Erst dadurch wird es möglich, bei Störungen den Schedule wieder zu reparieren, damit ein Match-Up-Zeitpunkt erreicht werden kann. Bei vollständiger Auslastung der Stationen führt jede Störung unweigerlich zu einer Verzögerung aller nachgelagerten Bearbeitungsschritte und damit zu einer Verlängerung der Makespan. Der erwartete gesamte Kapazitätsbedarf ist gegenüber einer nicht robusten Planung dabei nicht erhöht, da sich dort Störungen an Stationen implizit ebenfalls auf den Kapazitätsbedarf auswirken.

Die sich ergebende Verteilung der ungewichteten Schlupfzeiten in Abhängigkeit von der Rechenzeit ist in Abbildung 5-4 dargestellt. Sie werden direkt aus dem prädiktiv erzeugten Schedule abgeleitet und stellen die Differenz zwischen Fälligkeitstermin und geplanter Fertigstellung dar. Während nach 2 h Rechenzeit der minimale ungewichtete Schlupf noch lediglich 84s beträgt, erhöht sich dieser im Zeitverlauf nach weiteren 2 h auf 210 s und schließlich auf 253 s nach 6 h Rechenzeit. Die im Zeitverlauf erreichte Lösungsgüte ist in Anhang A3 dargestellt. Zwischen Minute 260 und 320 konnte keine Verbesserung der Lösung erzielt werden, daher existieren dort keine Datenpunkte.

Die Mehrzahl der Schlupfzeiten im finalen Ergebnis liegt zwischen 253 s und 600 s, es existieren allerdings auch einige Ausreißer mit einer Schlupfzeit von mehr als 1000 s. Da die Gesamtsumme der Schlupfzeiten durch das Kapazitätsangebot der Stationen begrenzt ist, wird die Verteilung umso linksschiefer, je länger die verfügbare Rechenzeit ist. Wenn das Minimum der Schlupfzeiten größer wird, bedingt dies eine Verringerung der Schlupfzeiten von Aufträgen mit höherer Schlupfzeit.

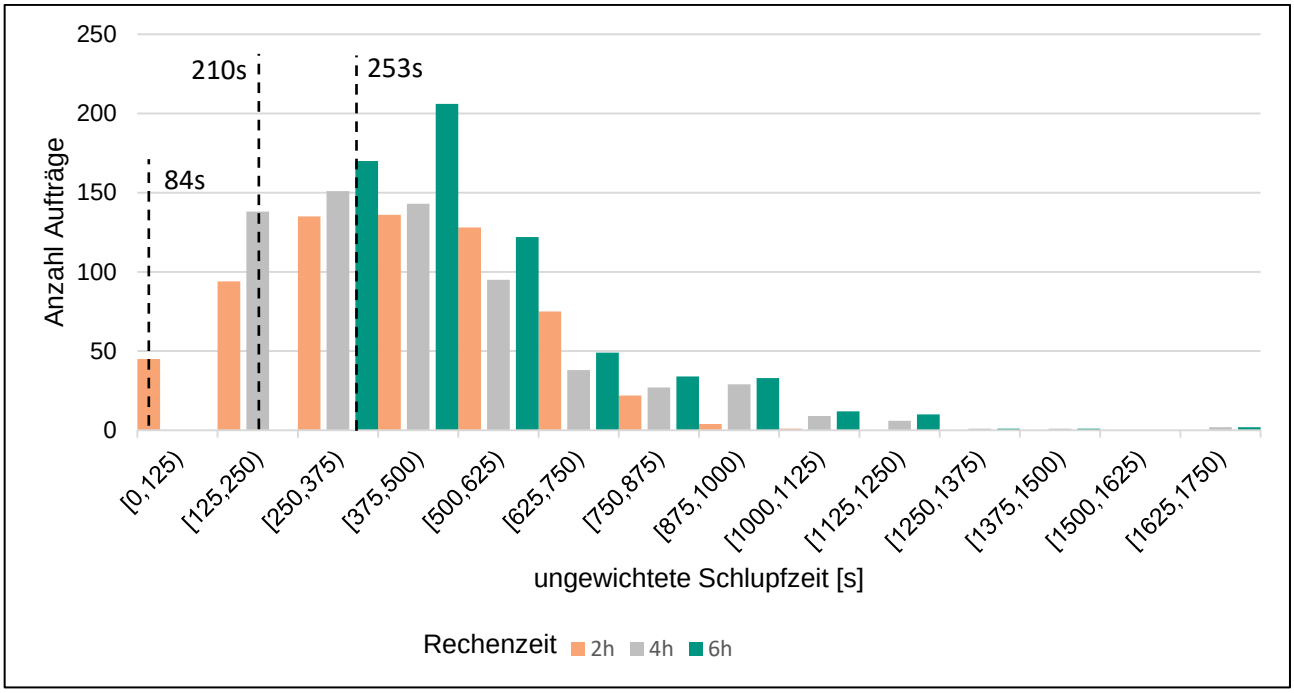


Abbildung 5-4: Verteilung der Schlupfzeiten über alle Aufträge

Die relative Belegung der Puffer im Schedule ist in Abbildung 5-5 dargestellt. Auf der Abszisse sind die Stationen der Matrix-Produktion dargestellt. Die Ordinate gibt an, welchen relativen Pufferfüllstand jede Station im Laufe einer Schicht voraussichtlich aufweist. So ist z. B. der Puffer der Station Geo 2 HeWa voraussichtlich 30 % der Zeit komplett leer und enthält weitere 46 % der Zeit einen Auftrag. Die Stationen Geo 1 HeWa und Geo 2 VoWa sind nicht abgebildet, da sich an ihnen per Definition keine Bestände aufbauen können. Dies ist dadurch begründet, dass sie eine Verfügbarkeit von 100 % aufweisen und neue Aufträge auch erst dann freigegeben werden, wenn die Stationen die Bearbeitung der vorangegangenen Aufträge abgeschlossen haben.

Der Puffer ist im Planungshorizont nur an den Stationen FLS 2 und Geo 3 HeWa zwischenzeitlich vollständig gefüllt. In dieser Situation ist ein Rescheduling aufwändiger, da weniger Freiheitsgrade bei der Umplanung von Aufträgen bestehen. Die Station Geo 3 HeWa weist mit 1,41 Heckwagen die höchste mittlere Pufferbelegung auf. Die Pufferbelegung im prädiktiven Scheduling hat eine hohe Wichtigkeit, da das Hinzufügen von Schlupfzeiten im nachfolgenden Schritt zu längeren Liegezeiten in den Puffern und damit deren höherer Auslastung einhergeht. Ähnlich wie bei der Auslastung der Stationen sind Überkapazitäten in den Puffern vorteilhaft. Andernfalls kann aufgrund gegenseitiger Blockaden im Störungsfall nur ein langsamer Abfluss von Vorder- und Heckwagen aus dem Produktionssystem erfolgen.

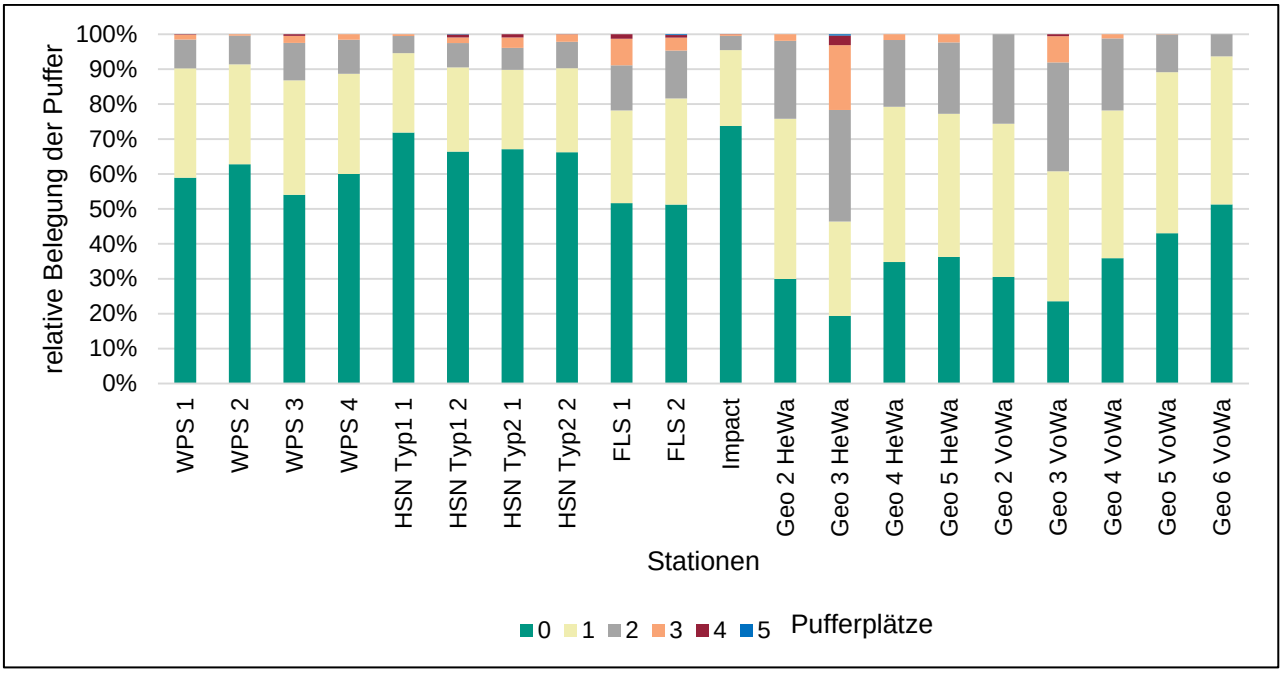


Abbildung 5-5: Relative Pufferbelegung an allen Stationen

5.2.2 Simulationsstudie für das prädiktive robuste Scheduling

Die erwartete Verzögerung aller Aufträge im prädiktiven Scheduling kann anhand der Formeln 4-16 bis 4-19 ermittelt werden und ist für den beispielhaften Schedule in Abbildung 5-6 dargestellt. Es fällt dabei auf, dass die gesamte erwartete Verzögerung tendenziell zunimmt, je später im Schedule die Aufträge fertig gestellt werden. Das hängt damit zusammen, dass zu einem frühen Zeitpunkt der Abarbeitung in viel geringerem Ausmaß vorgelagerte Störungskaskaden erwartet werden. Die Ausreißer oberhalb und unterhalb der Diagonalen sind Aufträge, die bereits im prädiktiven Schedule unter- bzw. überdurchschnittlich hohe Schlupfzeiten zwischen Bearbeitungsschritten besitzen. Aufgrund der insgesamt hohen angenommenen Verfügbarkeiten haben Störungskaskaden einen vergleichsweise geringen Effekt auf die Verzögerung. So wird die Durchlaufzeit eines spät startenden Auftrags von ca. 40 min durch erwartete Störungskaskaden nur um maximal ca. 6 min verlängert, während die mittlere Verlängerung lediglich ca. 3 min beträgt.

Zusätzlich sind die gesamten erwarteten Verzögerungen je Auftrag für die Robustheitswerte $R = 0{,}5$ und $R = 0{,}9$ dargestellt. Es wird deutlich, dass das Robustheitsmaß die relative Skalierung der erwarteten Verzögerung ermöglicht. Für $R \geq 1$ sind alle erwar-

teten Verzögerungen null. Mithilfe des in Kapitel 4.3.3 beschriebenen Vorgehens werden die Schlupfzeiten entsprechend so eingefügt, dass die erwartete Verzögerung begrenzt wird. Zudem werden Schlupfzeiten vor allem später im Planungshorizont eingefügt, da hier höhere Auswirkungen durch vorgelagerte Störungskaskaden erwartet werden.

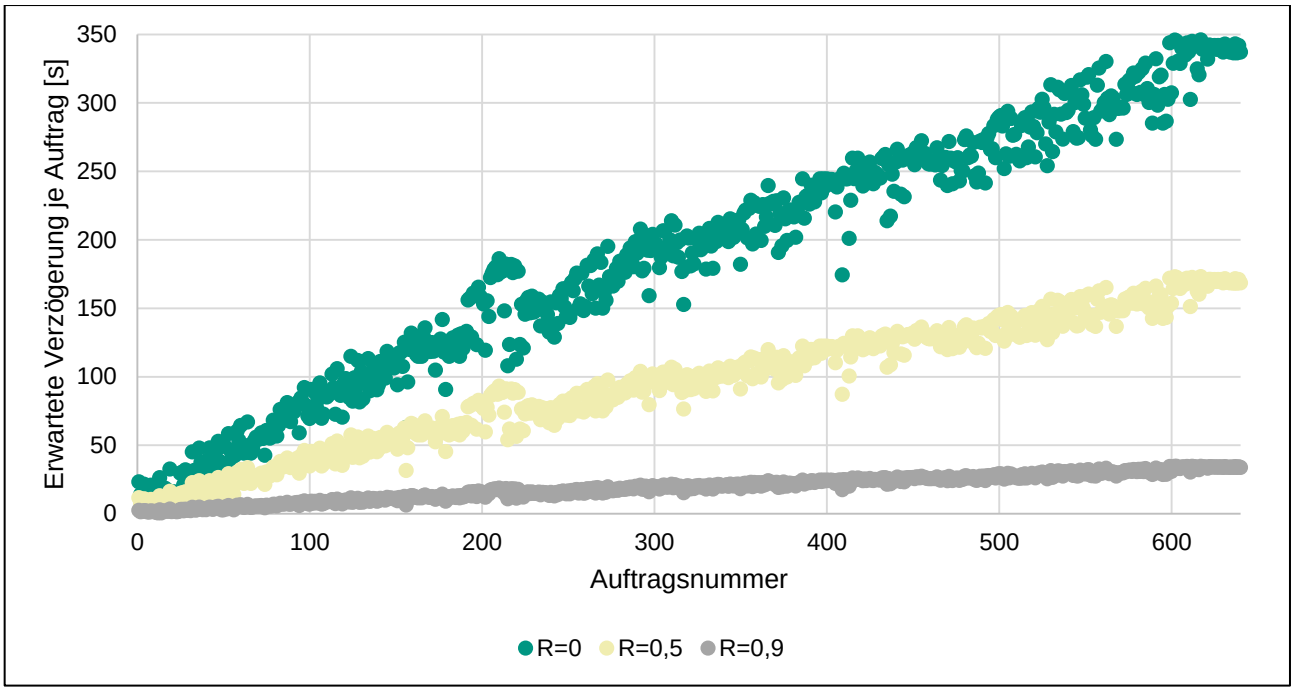


Abbildung 5-6: Erwartete Verzögerung aller Aufträge im prädiktiven Schedule

Aufgrund der vergleichsweise hohen Verfügbarkeit der Stationen sind die Verschiebungen einzelner Bearbeitungsschritte zu Beginn des Planungshorizontes sehr gering. Zu Beginn entsprechen sie der erwarteten Verzögerung eines Bearbeitungsschrittes aufgrund der realen Verfügbarkeiten und bewegen sich daher im Bereich <1s. Durch Störungskaskaden nehmen die Verschiebungen der Bearbeitungsschritte im Zeitverlauf zu, sodass einzelne Schritte für $R = 1$ um mehr als 5 Minuten nach hinten verschoben werden.

Zur Erprobung wurden für $R \in [0,1]$ in Stufen von 0,1 prädiktive robuste Schedules erzeugt und in der Ablaufsimulation durchgeführt. Das reaktive Rescheduling wurde dabei deaktiviert, um die Funktionsweise des prädiktiven robusten Schedulings isoliert untersuchen zu können. Eine ganzheitliche Betrachtung wird in Kapitel 5.4 durchgeführt. Da das Robustheitsmaß die erwarteten Verzögerungen begrenzt, wird dessen Wirksamkeit für verschiedene Konfigurationen im Folgenden untersucht. Hierfür wird

ein vollfaktorieller Versuchsplan verwendet, in dem die Robustheit R sowie die $MTTR_s$ der Stationen variiert werden. Die $MTTR_s$ wird dabei jeweils für alle Stationen um denselben Betrag variiert.

Die Verfügbarkeit bleibt für alle Stationen auf den in Tabelle 5-1 dargestellten Werten fixiert. Die Störungen werden anhand einer Erlang-Verteilung an allen Ausfügestationen erzeugt. Die Erlang-Verteilung kann zur Approximation des Ausfallverhaltens technischer Systeme verwendet werden (Eberlin & Hock 2014). Für R werden 11 Faktorstufen zwischen 0 und 1 mit Abstand von jeweils 0,1 untersucht. Die $MTTR_s$ wird ausgehend von 30 s in 10 Schritten mit Schrittweite 30 s auf bis zu 300 s erhöht. Ziel ist es, den Einfluss des Robustheitsmaßes auf die Robustheit für die Matrix-Produktion mit unterschiedlichem Störungsverhalten zu untersuchen. Damit kann untersucht werden, wie das prädiktive robuste Scheduling sich bei vielen kurzen (geringe $MTTR_s$) gegenüber wenigen langen Störungen (hohe $MTTR_s$) verhält.

Als Zielgrößen wird die mittlere absolute Verschiebung aller Bearbeitungsschritte zwischen Soll- und Ist-Schedule sowie die Termintreue als mittlere Verspätung aller Aufträge gemessen. Die mittlere absolute Verschiebung aller Bearbeitungsschritte wird in Sekunden erfasst und gibt die Differenz zwischen tatsächlicher und geplanter Startzeit an. Die mittlere Verspätung von Aufträgen wird ebenfalls in Sekunden gemessen und gibt die Differenz zwischen tatsächlicher Fertigstellung und Fälligkeitstermin jedes Auftrags an. Da kein Rescheduling durchgeführt wird, werden beim Auftreten von Störungen die betroffenen Bearbeitungsschritte durch Right Shifting verschoben. Der in Tabelle 5-4 dargestellte Versuchsplan umfasst damit 11*10=110 Simulationsexperimente.

Tabelle 5-4: Versuchsplan zur Untersuchung des Einflusses von R und $MTTR_s$

Experiment	Faktoren		Zielgrößen	
	R	$MTTR_s$	Mittlere absolute Verschiebung von Bearbeitungs-schritten	Mittlere Verspätung aller Aufträge
1	0	30s		
2	…	60s		
…	…	…		
11	0	300s		
12	0,1	30s		
…	…	…		
110	1	300s		

Die Ergebnisse der Simulationsstudie hinsichtlich der mittleren Verschiebung von Bearbeitungsschritten ist in Abbildung 5-7 dargestellt. Die zugrundeliegenden Werte sind in Anhang A5 aufgelistet. Auf den Ordinaten sind die beiden variierten Größen R und $MTTR_s$ abgebildet, die Abszisse zeigt den Mittelwert der resultierenden mittleren Verschiebung von Bearbeitungsschritten. Eine alternative Darstellung im dreidimensionalen Raum ist in Anhang A6 enthalten. Es sei an dieser Stelle zudem erwähnt, dass die Streuung der mittleren Verschiebung zunimmt, je höher die $MTTR_s$ ist. Während für $MTTR_s = 30\ s$ die mittlere absolute Verschiebung zwischen 74 s und 451 s schwankt, sind für $MTTR_s = 300\ s$ Werte zwischen 172 s und 1403 s zu beobachten. Die dazugehörigen Verteilungen sind in Anhang A7 dargestellt.

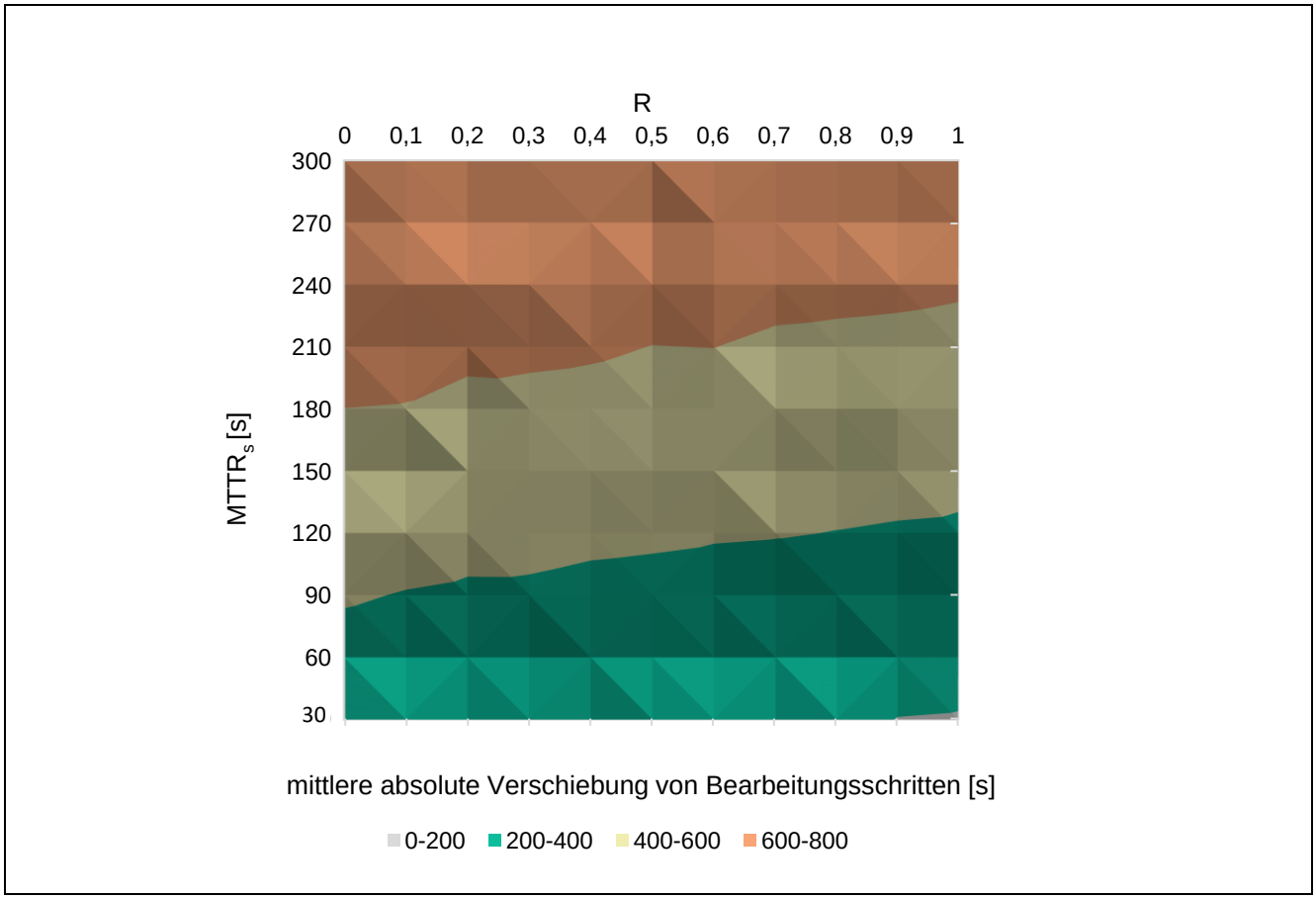


Abbildung 5-7: Ergebnisse der Simulationsstudie bezüglich der mittleren absoluten Verschiebung von Bearbeitungsschritten

Aus Abbildung 5-7 wird zum einen ersichtlich, dass der überwiegende Einfluss auf die mittlere absolute Verschiebung von Bearbeitungsschritten von einer Variation der $MTTR_s$ herrührt. Bei $R = 0$ steigt dieser Wert bei einer Erhöhung der $MTTR_s$ von 30 s

auf 300 s von ca. 307 s auf 795 s. Das heißt, dass in einem nicht robust geplanten Schedule im Mittel jeder Bearbeitungsschritt 795 s später als geplant ausgeführt wird, wenn die $MTTR_s$ aller Stationen bei 300 s liegt. Es kann daraus geschlossen werden, dass das Ersatzmaß zur Bewertung der Robustheit basierend auf dem Erwartungswert der Verspätungen deutlich besser für viele kurze Störungen funktioniert. Bei wenigen langen Störungen hingegen entstehen längere Auftragswarteschlangen in den Puffern, die erst im Laufe der Zeit abgearbeitet werden können und die während der Wartezeit große Verschiebungen gegenüber dem ursprünglichen Schedule erfahren.

Zum anderen wird ersichtlich, dass eine Steigerung von des Robustheitsmaßes R zu einer Reduzierung der mittleren Verschiebungen führt. Dieser Effekt ist deutlich in den Experimenten mit $MTTR_s = 30\ s$ zu beobachten, da die Vielzahl kurzer Störungen besonders effizient durch die statistisch im Schedule verteilten Schlupfzeiten kompensiert werden kann. Für $R = 0$ beträgt die durchschnittliche Verschiebung von Bearbeitungsschritten unabhängig von der $MTTR_s$ ca. 558 s, während diese für $R = 1$ um 103s auf 455 s reduziert werden kann. Dies entspricht einer relativen Verringerung um 18,5 %. Wenn die $MTTR_s$ schrittweise erhöht wird, nimmt der relative Effekt ab. Für $MTTR_s = 300\ s$ liegt bei $R = 0$ eine durchschnittliche Verschiebung von 795 s vor, während diese für $R = 1$ um 107 s auf 688 s reduziert werden konnte. Dies entspricht einer relativen Verringerung um 13,5 %. Eine alternative Darstellung mit den durchschnittlichen Verschiebungen von Bearbeitungsschritten ist in Anhang A6 dargestellt.

Die alleinige Betrachtung der mittleren Verschiebung von Bearbeitungsschritten legt nahe, in jedem Fall $R = 1$ zu wählen, unabhängig von der tatsächlichen $MTTR_s$. Nicht berücksichtigt ist dabei allerdings die Auswirkung von Schlupfzeiten auf die Termintreue. Die Ergebnisse der Simulationsstudie hinsichtlich der mittleren Verspätung von Aufträgen ist mit analogem Aufbau in Abbildung 5-8 dargestellt.

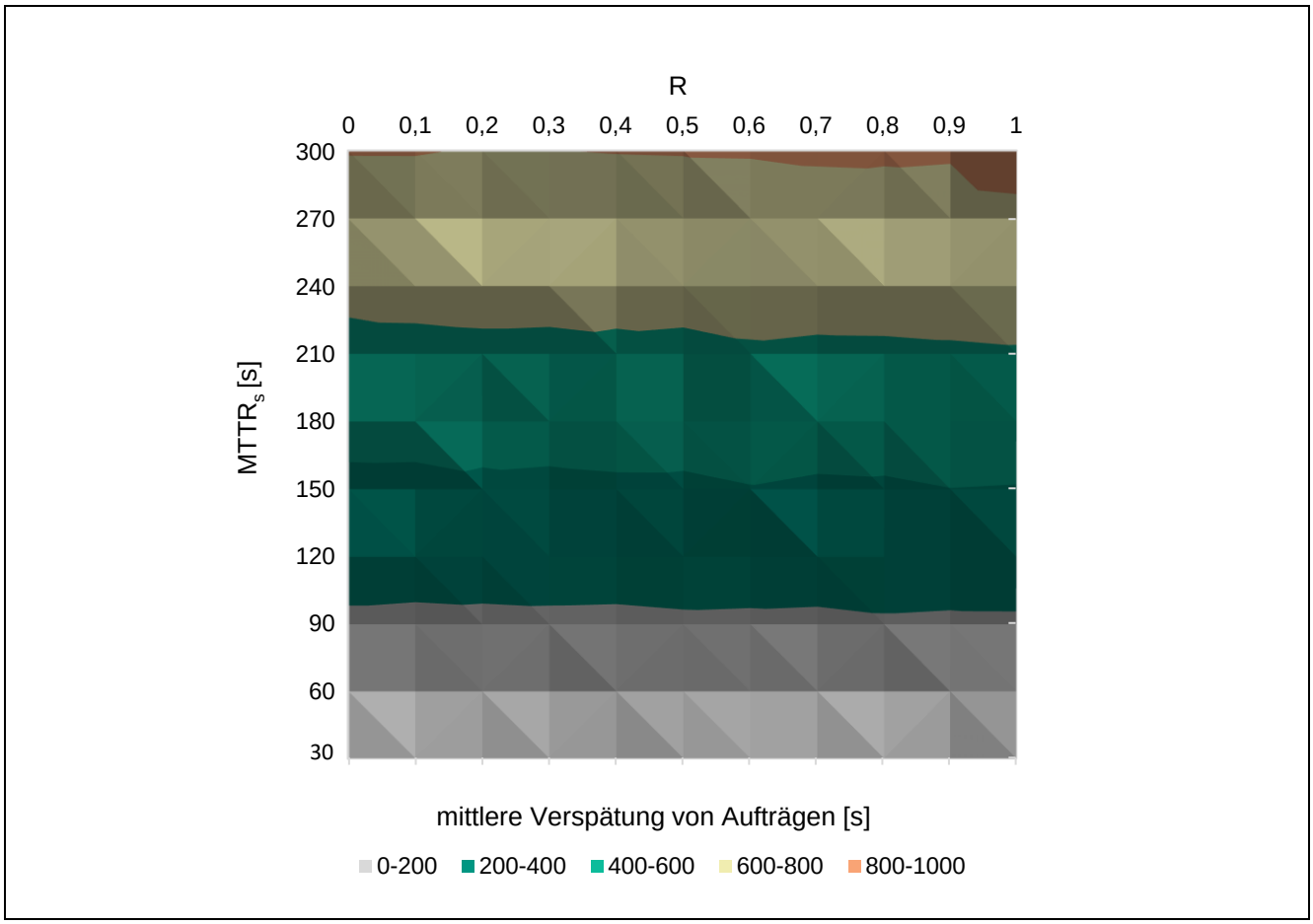


Abbildung 5-8: Ergebnisse der Simulationsstudie bezüglich der mittleren Verspätung von Aufträgen

Wie bei der Betrachtung der mittleren Verschiebung von Bearbeitungsschritten wird in Abbildung 5-8 deutlich, dass die $MTTR_s$ auch hier den wesentlichen Einfluss darstellt. Die dazugehörigen Rohdaten sind in Anhang A8 dargestellt. Bei $R = 0$ steigt die mittlere Verspätung von Aufträgen bei einer Erhöhung der $MTTR_s$ von 30s auf 300s von ca. 60s auf 807s. Dies ist eine direkte Folge der durch lange Störungen bedingten großen Verschiebungen von Bearbeitungsschritten.

Zudem zeigt sich allerdings auch, dass eine Steigerung von R zu einer Erhöhung der mittleren Verspätungen führt. Allerdings ist dieser Einfluss deutlich schwächer gegenüber der Verschiebung von Bearbeitungsschritten. Der relative Effekt ist wieder für $MTTR_s = 30\ s$ am deutlichsten zu beobachten. Für $R = 0$ wird simulativ eine mittlere Verspätung von 60 s ermittelt, die bis $R = 1$ auf 71 s ansteigt. Dies entspricht einer relativen Steigerung von 18,3 %. Für größere Werte von $MTTR_s$ nimmt der relative Effekt ab. Bei $MTTR_s = 300\ s$ steigt die mittlere Verspätung je Auftrag von 807 s bei $R =$

0 bis auf 892 s bei $R = 1$ an. Dies entspricht einem relativen Anstieg von 10,5 %. Eine alternative Darstellung mit den durchschnittlichen Verspätungen ist in Anhang A9 dargestellt. Die Häufigkeitsverteilung ist in Anhang A10 für die beiden Extremfälle $MTTR_s = 30\ s$ und $MTTR_s = 300\ s$ dargestellt.

5.2.3 Grenzen der Anwendung

Es wird aus der untersuchten Matrix-Produktion deutlich, dass im prädiktiven robusten Scheduling ein Trade-Off zwischen einer Verringerung der mittleren Verschiebung von Bearbeitungsschritten und der mittleren Verspätung von Aufträgen besteht. Große Schlupfzeiten verringern die Wahrscheinlichkeit, dass Störungen an Stationen sich auf den Schedule auswirken. Sie können aber bei Nichteintreten oder Abweichungen von der Gleichverteilung von Störungen ebenfalls dazu führen, dass die Verspätung von Aufträgen ansteigt.

Es wurde auf Basis einer Simulationsstudie gezeigt, dass sich mit dem in Kapitel 4.3 entwickelten Vorgehen prädiktive robuste Schedules erzeugen lassen, die eine Verringerung der durchschnittlichen Verschiebung je Bearbeitungsschritt ermöglichen. Während der absolute Effekt nahezu unabhängig von der $MTTR_s$ ist, nimmt der relative Effekt mit steigender $MTTR_s$ deutlich ab. Ohne weitere Informationen zum Ausfallverhalten der Stationen ist es also allein durch ein prädiktives robustes Scheduling nicht möglich, eine Verschiebung von Bearbeitungsschritten zu verhindern. Gleichzeitig wurde deutlich, dass die durchschnittliche Verspätung je Auftrag ansteigt, je höher die Robustheit im prädiktiven robusten Scheduling gewählt wird. Wenn für die Anwendung allein die mittlere Verspätung relevantes Kriterium ist, ist der isolierte Einsatz des beschriebenen prädiktiven robusten Schedulings nicht zu empfehlen. Dieser Trade-Off wird im Zusammenspiel mit dem reaktiven Rescheduling erneut in Kapitel 5.4 aufgegriffen.

5.3 Anwendung des reaktiven Rescheduling

Nachdem in Kapitel 5.2 das prädiktive robuste Scheduling erprobt wurde, wird nun daran anschließend das reaktive Rescheduling isoliert erprobt. Dazu werden die Schedules aus Kapitel 5.2 mit $R = 0$ herangezogen. Die Funktionsweise wird also zunächst ohne Berücksichtigung von Schlupfzeiten gezeigt. Eine kombinierte Betrachtung mit $R = 0,..,1$ findet im nachfolgenden Kapitel 5.4 statt.

Der prädiktive robuste Schedule wird nun in der in Kapitel 5.1.1 erwähnten Ablaufsimulation durchgeführt. Für die Anwendung werden zunächst die zugrundeliegenden Verteilungen der Störungen beschrieben. Anschließend wird die Lösungsgüte des reaktiven Rescheduling exemplarisch vergleichend gegenüber anderen Verfahren beschrieben. Aus dem Anwendungsfall wird eine mittlere Störungsdauer von 150 s angegeben, die zwei Peaks (bei 60s und bei 300s) hat. D. h. die mittlere Störungsdauer ergibt sich durch eine gleichgewichtete Kombination kurzer und längerer Störungen.

Die zur Abbildung dieses Sachverhaltes angenommenen Erlang-Verteilungen mit Erwartungswert der $MTTR_s$ von 60 s bzw. 300 s sind in Abbildung 5-9 dargestellt. Die Überlagerung der Erlang-Verteilungen ist aus der Anwendung motiviert, um sowohl Häufungen kürzerer als auch längerer Störungen adäquat abbilden zu können. Die resultierende Häufigkeitsverteilung nach 1000 Störungen in der Simulation ist ebenfalls abgebildet. Es wird ersichtlich, dass die entstehende Häufigkeitsverteilung Charakteristika beider zugrundeliegender Erlang-Verteilungen aufweist.

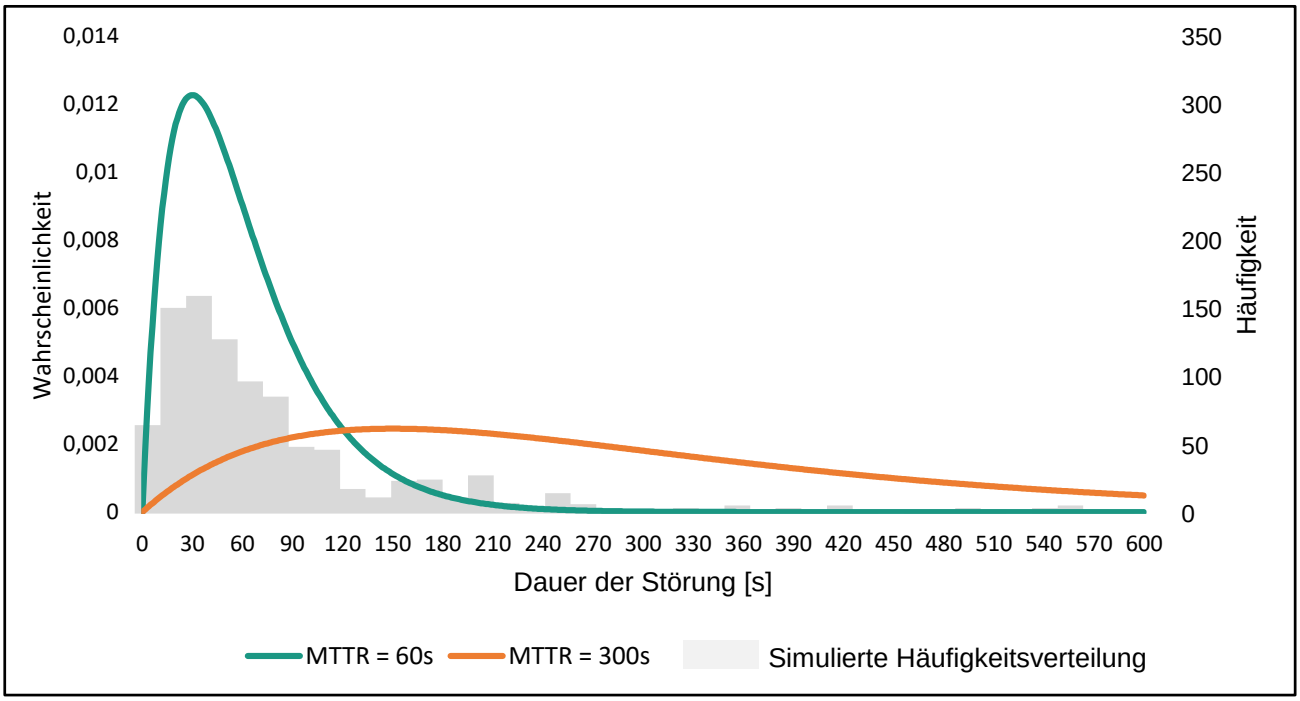


Abbildung 5-9: Erlang-Verteilungen der Störungen je Station mit Mittelwerten 30s und 300s und die simulierte Häufigkeitsverteilung

In der Ablaufsimulation wird bei jeder Störung entschieden, ob ein Rescheduling-Korridor erzeugt werden muss oder nicht. Bei sehr kurzen Störungen oder Störungen während geplanter Leerzeiten verschiebt sich unter Umständen nur ein einziger Bearbeitungsschritt. Es ist dann keine Umplanung erforderlich. Andernfalls wird mithilfe des in

Kapitel 4.4 beschriebenen Verfahrens ein Rescheduling-Korridor erzeugt, der anschließend durch das reaktive Rescheduling umgeplant wird.

Der Lernerfolg der verteilten Strategie-Iteration wird anhand exemplarischer Rescheduling-Korridore unterschiedlicher Länge in Kapitel 5.3.1 ausgewählten Prioritätsregeln vergleichend gegenübergestellt. Die Bewertung der Lösungsgüte in Kapitel 5.3.2 wird für einen umfangreichen Datensatz aus der Ablaufsimulation durchgeführt. Die Grenzen der Anwendung schließlich werden in Kapitel 5.3.3 diskutiert.

5.3.1 Bewertung des Lernerfolgs

Für zwei exemplarische Rescheduling-Korridore ist in Abbildung 5-10 und Abbildung 5-11 die Lösungsgüte der verteilten Strategie-Iteration (DPS) illustriert. Es werden insgesamt $E_{max} = 100$ Updates der θ mit der Lernrate $\delta = 0{,}01$ durchgeführt wird. Die Herleitung der Parameter ist in Anhang A11 beschrieben. Die Anzahl der Update-Schritte der θ ist auf der Abszisse gegenüber dem Makespan auf der Ordinate abgetragen. Die erzielten Ergebnisse mit DPS werden mit den Prioritätsregeln First-In-First-Out (FIFO) und kürzeste Bearbeitungszeit (SPT) sowie einer zufälligen Auswahl von Aufträgen (ZUFALL) basierend auf $\theta = 0$ in Anlehnung an Gabel und Riedmiller (2012) verglichen.

Die Prioritätsregeln sind dabei in dasselbe sequenzielle Framework eingebunden wie DPS. Wenn in einem Zustand nur eine Aktion verfügbar ist, wird diese automatisch gewählt, da keine Priorisierung vorgenommen werden kann. Für Rescheduling-Korridore mit sehr wenigen Aufträgen ist daher zu erwarten, dass sich kaum Unterschiede zwischen DPS und Prioritätsregeln abzeichnen.

Da die Prioritätsregeln durch das Anlernen nicht verbessert werden können, sind deren Verläufe horizontal. Zudem wird ein Vergleich mit einer constraint-basierten Optimierung (CP-OPT) gezogen, in der die Rescheduling-Korridore jeweils 15 min optimiert werden. Das dazu verwendete Modell ist im Wesentlichen angelehnt an das in Kapitel 4.3.1 gezeigte Modell aufgebaut und näher in Anhang A12 beschrieben.

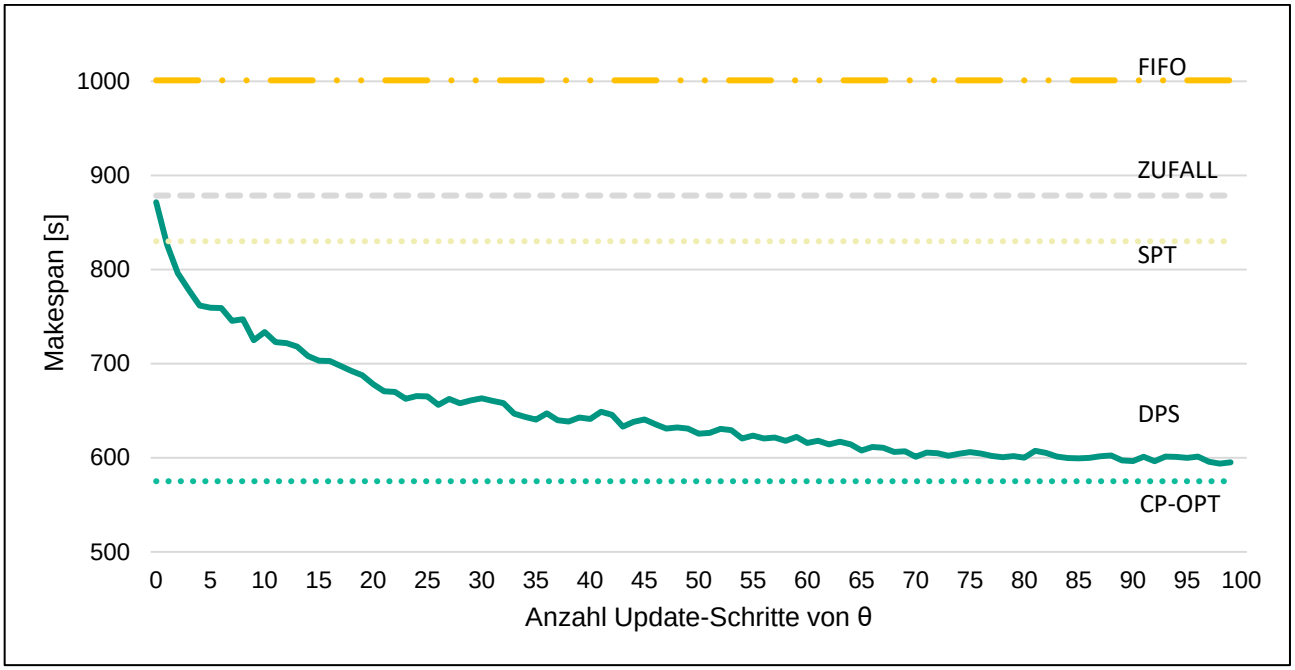


Abbildung 5-10: Lernverhalten für einen Rescheduling-Korridor mit ca. 600 s Länge

Für den in Abbildung 5-10 dargestellten Rescheduling-Korridor fällt auf, dass DPS bereits nach wenigen Update-Schritten von θ die Leistungsfähigkeit der Prioritätsregel SPT erreicht. Es fällt außerdem auf, dass die Prioritätsregel FIFO in dem gewählten Beispiel schlechtere Ergebnisse als eine zufällige Auftragsauswahl erreicht. Dies kann als Hinweis darauf gedeutet werden, dass die Leistungsfähigkeit heuristscher Prioritätsregeln stark situativ bestimmt und nicht generalisierbar ist. Im Laufe der Anlernphase verbessert sich bei DPS der Makespan von 871 s auf 595 s über 100 Update-Schritte. Dies entspricht einer relativen Verbesserung von 31,7 %. Die Rechenzeit für das Anlernen beträgt dabei ca. 5min. Mit CP-OPT wird in 15 min Rechenzeit ein Makespan von 575 s für den betrachteten Rescheduling-Korridor erreicht. Wie erwartet erreicht DPS diesen Wert nicht, da nicht sichergestellt werden kann, dass ein globales Optimum gefunden werden kann. Allerdings beträgt die Optimalitätslücke in diesem Fall lediglich 2,8 %.

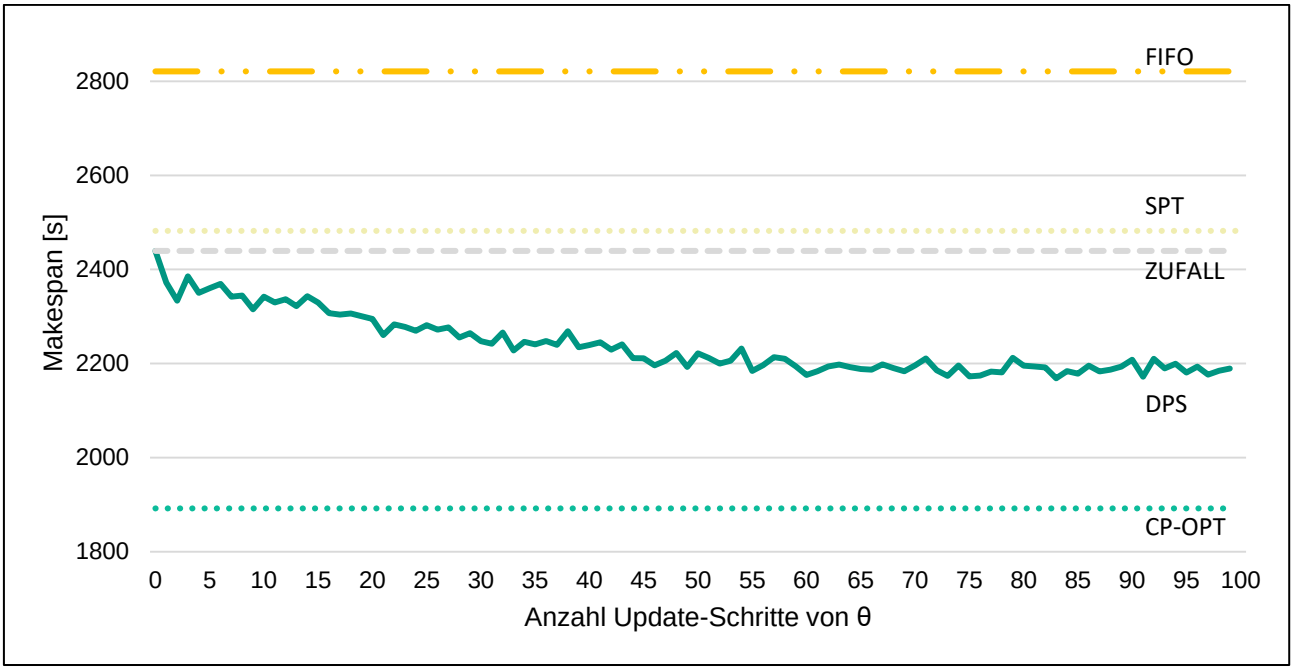


Abbildung 5-11: Lernverhalten für einen Rescheduling-Korridor mit ca. 2000 s Länge

Für den in Abbildung 5-11 dargestellten Rescheduling-Korridor liefern die Prioritätsregeln durchgehend schlechtere Ergebnisse als DPS und sind sogar der zufälligen Auftragsauswahl unterlegen. Im Laufe des Anlernens verbessert sich bei DPS der Makespan von 2439 s auf 2189 s über 100 Update-Schritte. Dies entspricht einer relativen Verbesserung von 10,3 %. Die Dauer für das Anlernen beträgt dabei ca. 12 min. Mit CP-OPT wird in 15min Rechenzeit einen Makespan von 1892s für den betrachteten Rescheduling-Korridor erreicht. Die Optimalitätslücke steigt in diesem Fall auf 15,6 % an. Die große Optimalitätslücke ist dadurch zu erklären, dass die Aussagekraft von θ für sehr lange Rescheduling-Korridore abnimmt, da lediglich die verfügbaren Aufträge an einer Station ausschlaggebendes Entscheidungskriterium sind. Die beispielhaften Betrachtungen des Lernverhaltens für die beiden Rescheduling-Korridore können als erstes Indiz gewertet werden, dass kürzere Rescheduling-Korridore eine geringere Optimalitätslücke ermöglichen, unabhängig davon, ob das Rescheduling mit DPS oder Prioritätsregeln erfolgt. Dieser Zusammenhang wird nochmals in Kapitel 5.4 aufgegriffen.

5.3.2 Bewertung der Lösungsgüte

Zur Erprobung der Anwendung wird exemplarisch für Störungen an einer Station HSN Typ2 wie folgt vorgegangen: Zunächst werden für 100 relevante Störungen die dazugehörigen 100 Rescheduling-Korridore erzeugt. Für die Ermittlung ausreichend langer

Rescheduling-Korridore ist das Ergebnis von DPS ausschlaggebend, andere Verfahren können daher längere oder kürzere Gesamtdurchlaufzeiten für die jeweiligen Rescheduling-Korridore ergeben. Für jeden Korridor werden die Parameter θ separat erlernt. Zusätzlich wird wie zuvor der Abstand zu den Prioritätsregeln FIFO und SPT, zur zufallsbasierten Auswahl (ZUFALL) und zur constraint-basierten Optimierung (OPT-CP) untersucht. Die Rohdaten der Optimierung sind in Anhang A13 dargestellt.

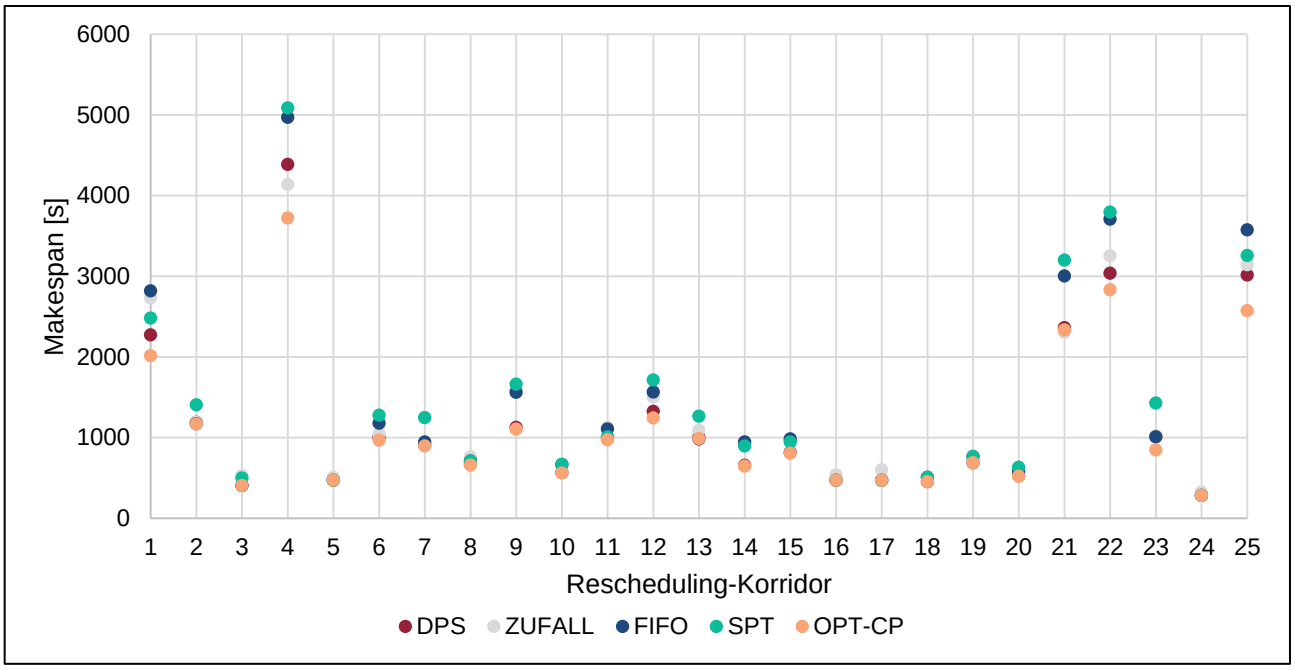


Abbildung 5-12: Absolute Lösungsgüte exemplarischer Rescheduling-Korridore

Ein Ausschnitt der Ergebnisse der generierten Rescheduling-Korridore (1-25) ist zusammenfassend in Abbildung 5-12 (absolut) und Abbildung 5-13 (relativ) dargestellt. Auf der Abszisse sind die Rescheduling-Korridore dargestellt, die Ordinate zeigt die mit den unterschiedlichen Verfahren erlangten Makespans bzw. die prozentuale Abweichung zum Optimum. In manchen Fällen erreichen für einen Rescheduling-Korridor mehrere Verfahren dasselbe Ergebnis, sodass die Datenpunkte in der Darstellung überlappt dargestellt sind. Die Optimalitätslücke von DPS hängt zum großen Teil von der Länge der Rescheduling-Korridore ab und schwankt zwischen 0 % und 30 %. Die durchschnittliche Optimalitätslücke beträgt für DPS 5,1 %, während für ZUFALL, FIFO und SPT 14,4 %, 18,2 % und 23,3 % erreicht wurden. Die Prioritätsregel ZUFALL erzielt verhältnismäßig gute Resultate für kurze Rescheduling-Korridore, allerdings ist die Lösungsgüte für längere Rescheduling-Korridore oft unterlegen. Es ist zu vermuten, dass

die guten Ergebnisse aufgrund der Kürze erzeugter Rescheduling-Korridore erzielt werden. Zudem beträgt die Standardabweichung der Optimalitätslücke für DPS 6,3 %, während für ZUFALL, FIFO und SPT 12,2 %, 16,7 % und 19,7 % erreicht wurden. Unter der Annahme, dass die 100 zufällig generierten Störungen bereits eine hinreichend genaue Abbildung des tatsächlichen Störungsverhaltens darstellen, kann ein Rescheduling mit OPT-CP im Mittel über 1116 s bewältigt werden, während für DPS 1208 s, für ZUFALL 1288 s, für FIFO 1400 s und für SPT 1466 s benötigt werden.

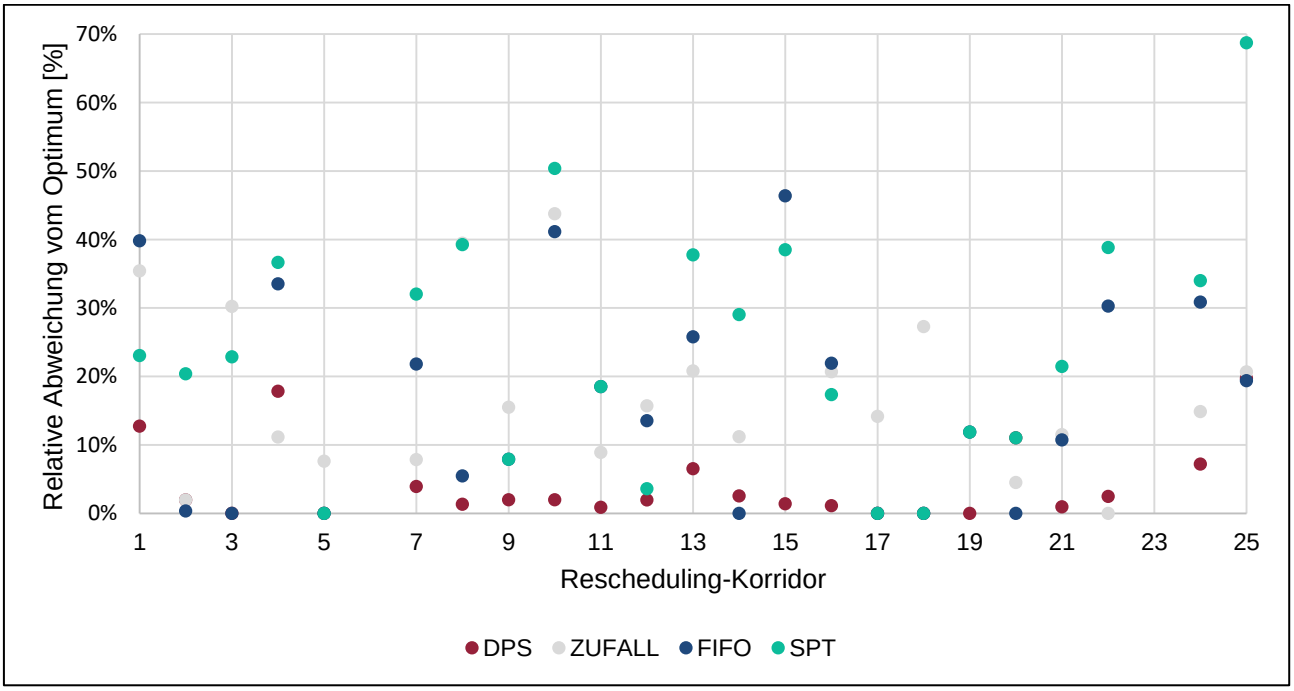


Abbildung 5-13: Relative Lösungsgüte exemplarischer Rescheduling-Korridore

Damit zeigt DPS sowohl bezüglich der Höhe als auch der Performance eine überlegene Leistung gegenüber den ausgewählten Prioritätsregeln. Die Ergebnisse für alle Rescheduling-Korridore sind in Anhang A14 und Anhang A15 dargestellt. Es lässt sich damit zusammenfassend sagen, dass mit DPS sehr gute Ergebnisse erzielt werden können, die klassische Prioritätsregeln in den meisten Fällen nicht erreichen. Eine Einschränkung besteht bei längeren Rescheduling-Korridoren, da die Optimalitätslücke hier deutlich größer wird und in einigen Fällen sogar schlechtere Ergebnisse liefert als eine zufällige Auswahl von Aufträgen (z. B. Rescheduling-Korridore 4, 20, 25 in Abbildung 5-13). Für eine hohe Lösungsgüte ist es daher erstrebenswert, möglichst kurze Rescheduling-Korridore zu verwenden.

5.3.3 Grenzen der Anwendung

Anhand exemplarischer Rescheduling-Korridore wurde das prinzipielle Verhalten von DPS im Vergleich zu ausgewählten Prioritätsregeln und einer Optimierung aufgezeigt. Während für Rescheduling-Korridore mit ca. 5-10 min Länge noch geringe Abweichungen von dem besten mit der Optimierung erreichbaren Makespan erreicht werden konnten, steigt die Optimalitätslücke bei längeren Rescheduling-Korridoren an. Dieser Zusammenhang ist in Abbildung 5-14 dargestellt.

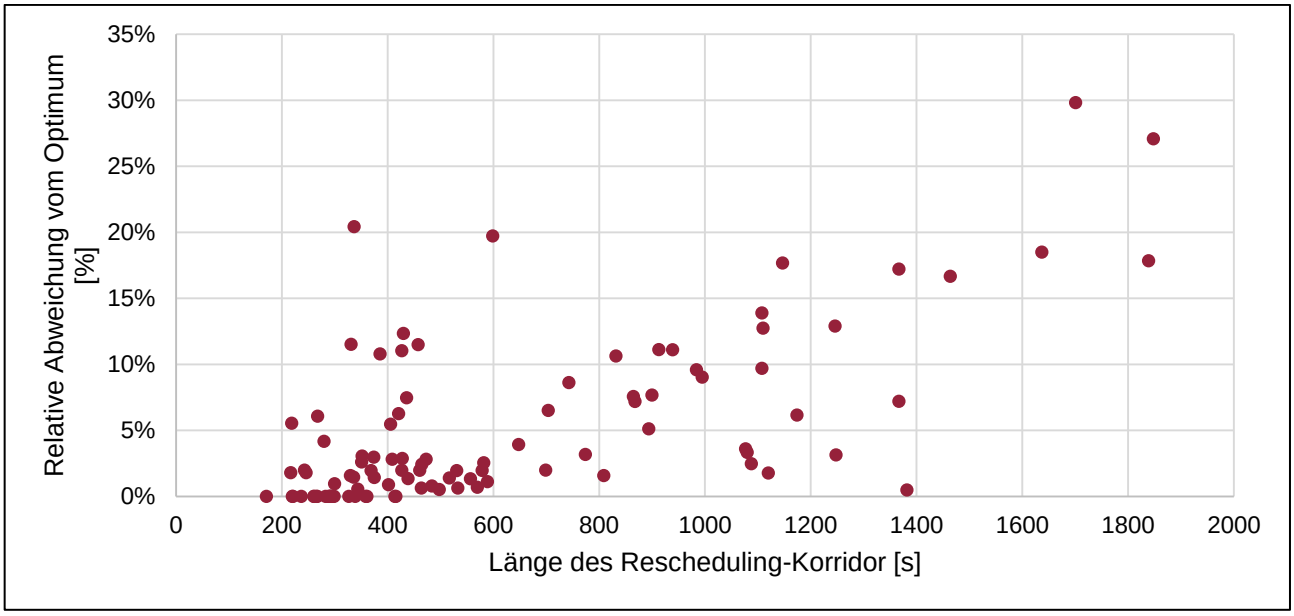


Abbildung 5-14: Relative Abweichung vom Optimum in Abhängigkeit der Länge der Rescheduling-Korridore für DPS

Dies ist bedingt durch die unterschiedliche Modellierung als simultanem (OPT-CP) gegenüber einem sequenziellen (DPS) Entscheidungsproblem. Zudem kann aus einer großen Optimalitätslücke nicht auf eine unzureichende Funktionsweise von DPS geschlossen werden, da dort ein inhärent dezentraler Entscheidungsprozess abgebildet wird. Es lassen sich jedoch Hinweise auf die praktische Anwendbarkeit gewinnen, die damit für längere Rescheduling-Korridore als nur eingeschränkt gegeben eingestuft werden muss.

5.4 Optimale Robustheit im prädiktiven robusten Scheduling

In Kapitel 5.2 wurde gezeigt, dass ein Trade-Off zwischen durchschnittlicher Verschiebung von Bearbeitungsschritten und durchschnittlicher Verspätung je Auftrag besteht. Es wurde dabei deutlich, dass ein rein statistisches Einbringen von Schlupfzeiten in

einen Schedule einen negativen Effekt auf die Liefertreue hat und damit nur dann empfehlenswert ist, wenn die Kosten durch Verschiebungen von Bearbeitungsschritten, z. B. aufgrund begonnener Rüstvorgänge oder aufwändiger Materialbereitstellung, die Kosten durch Verspätungen von Aufträgen überwiegen. Zudem legt die Erprobung des reaktiven Reschedulings in Kapitel 5.3 nahe, dass die mittlere Optimalitätslücke umso geringer ist, je kürzer die zugrundeliegenden Rescheduling-Korridore sind.

In Kapitel 5.4.1 wird im Folgenden untersucht, ob ein optimaler Wert für die Robustheit im prädiktiven robusten Scheduling ermittelt werden kann, der die Liefertreue zulasten der Verschiebung von Bearbeitungsschritten durch das reaktive Rescheduling maximiert. Die Grenzen der Anwendung werden in Kapitel 5.4.2 aufgezeigt.

5.4.1 Simulationsstudie für das prädiktiv-reaktive Scheduling

Es wird davon ausgegangen, dass eine Verschiebung von Bearbeitungsschritten ohne Verzögerungen der Bearbeitung ganzer Aufträge unproblematisch ist, da der Auftrag und das benötigte Material stets auf den FTF mitgeführt werden.

Hierzu wird wiederum eine Simulationsstudie durchgeführt, bei der dieselben prädiktiven robusten Schedules wie in Kapitel 5.2 sowie die in Kapitel 5.3 zum Einsatz kommenden Verteilungen von Störungen mit $MTTR_s = 60\ s$ und $MTTR_s = 300\ s$ verwendet werden. Zur Untersuchung des Mehrwertes des reaktiven Reschedulings wird zudem jede Konfiguration auch ohne Rescheduling untersucht. Ohne Rescheduling wird wie in Kapitel 5.2 ein Right Shifting bei Störungen durchgeführt, mit Rescheduling werden wie in Kapitel 5.3 Rescheduling-Korridore generiert, die mit DPS umgeplant werden.

Tabelle 5-5: Aufbau des Versuchsplans zur Untersuchung des Einflusses von R und des Rescheduling mit DPS

Experiment	Faktoren		Zielgrößen	
	R	Rescheduling	Mittlere absolute Verschiebung von Bearbeitungsschritten	Mittlere Verspätung aller Aufträge
1	0	ja		
2	0,1	ja		
…	…	…		
11	1	ja		
12	0	nein		
…	…	…		
22	1	nein		

Als Antwort wird wiederum die mittlere absolute Verschiebung von Bearbeitungsschritten sowie die mittlere Verspätung von Aufträgen erfasst. Zur Sicherstellung der Vergleichbarkeit werden die Experimente mit jeweils denselben Seed-Werten in der Ablaufsimulation erzeugt. Das heißt, die Störungen treten unabhängig davon, welcher Schedule durchlaufen wird, an denselben Stationen auf und besitzen dieselbe Dauer. Lediglich die umzuplanenden Bearbeitungsschritte und der Zustand des Produktionssystems zu Beginn der Störung unterscheiden sich dann zwischen den einzelnen Experimenten. Die Betrachtung erfolgt im Gegensatz zu Kapitel 5.3 nun für alle Stationen, sodass in einem Simulationslauf sämtliche auftretende Störungen an Stationen im Ausfügebereich während einer Schicht berücksichtigt werden.

Der Aufbau des Versuchsplans ist in Tabelle 5-5 dargestellt. Aufgrund des hohen Aufwandes für das Anlernen je Störung werden hier nunmehr 10 Simulationsläufe je Experiment durchgeführt. Die Ergebnisse der Simulationsstudie sind in Abbildung 5-15 und Abbildung 5-16 dargestellt.

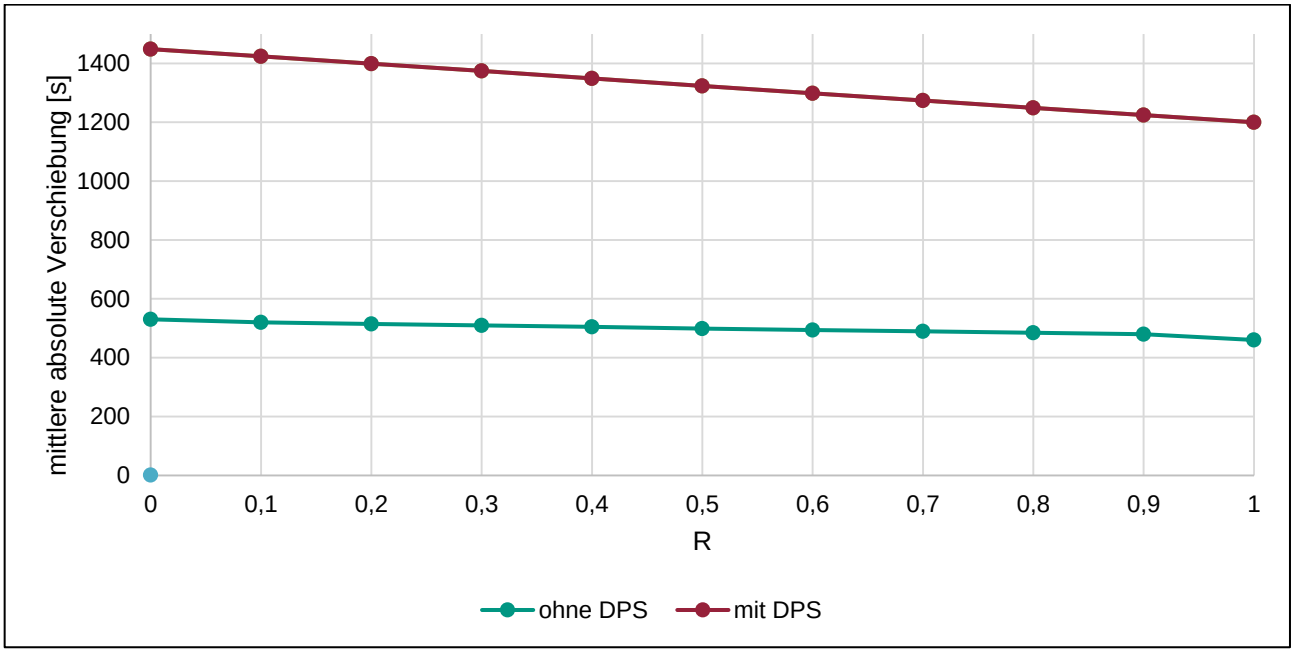


Abbildung 5-15: Ergebnisse der Simulationsstudie bezüglich der mittleren Verschiebung von Bearbeitungsschritten

Abbildung 5-15 stellt die mittlere absolute Verschiebung von Bearbeitungsschritten in Abhängigkeit von der gewählten Robustheit R des ermittelten Schedules dar. Die Datenbasis der Darstellung ist in Anhang A16 dargestellt. Ohne DPS gilt für $R = 0$ ein Mittelwert von 530 s, der bis $R = 1$ auf 460 s absinkt. Mit DPS ist eine starke Steigerung

der mittleren Verschiebung um ca. den Faktor 3 zu beobachten, sodass für $R = 0$ die mittlere absolute Verschiebung 2448 s beträgt und bis $R = 1$ auf 2200 s absinkt. Dieser hohe Unterschied ist darauf zurückzuführen, dass ohne DPS lediglich ein Right Shifting zum Einsatz kommt, während mit DPS jeweils alle Bearbeitungsschritte innerhalb eines Rescheduling-Korridors komplett frei umgeplant werden können. In Produktionssystemen, in denen durch diese großen Verschiebungen keine zusätzlichen logistischen Aufwände entstehen, können diese vernachlässigt werden, so lange die Fälligkeitstermine der Aufträge eingehalten werden.

Zudem ist ersichtlich, dass die grundlegende Tendenz einer Verringerung der mittleren Verschiebung bei höherer Robustheit auch in der Kopplung mit DPS erhalten bleibt. Dies ist darauf zurückzuführen, dass eine hohe Anzahl kürzerer Störungen bei höherer Robustheit zu kürzeren Rescheduling-Korridoren führen, für die ein Rescheduling mit geringerer Optimalitätslücke (siehe 5.3.2) möglich ist.

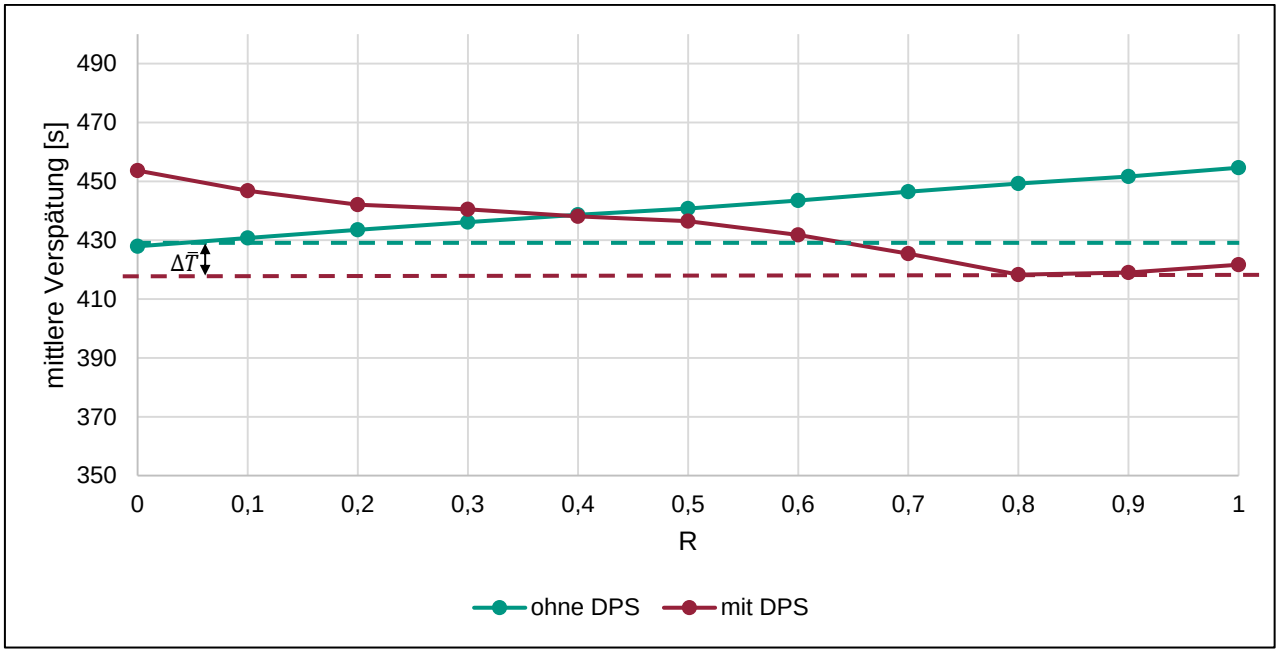


Abbildung 5-16: Ergebnisse der Simulationsstudie bezüglich der mittleren Verspätung von Aufträgen

Abbildung 5-16 stellt analog zu Abbildung 5-15 die Mittelwerte der mittleren Verspätung von Aufträgen in Abhängigkeit von der gewählten Robustheit des zugrundeliegenden Schedules dar. Die gestrichelten Linien repräsentieren das jeweilige Minimum der mittleren Verspätung mit bzw. ohne DPS. Die Datenbasis der Darstellung ist in Anhang

A16 dargestellt. Für das prädiktive robuste Scheduling ohne DPS zeigt sich wie in Kapitel 5.2, dass eine Erhöhung der Robustheit R zu einer Erhöhung der mittleren Verspätung führt. Dabei steigt die mittlere Verspätung von 428 s bei $R = 0$ auf 455 s bei $R = 1$ an.

Bei der Betrachtung mit DPS wird deutlich, dass im Bereich bis $R = 0{,}4$ größere Verspätungen als ohne DPS entstehen. Dies ist dadurch zu begründen, dass in diesem Bereich verhältnismäßig lange Rescheduling-Korridore erzeugt werden, deren Optimalitätslücke daher relativ groß ist. Die Länge der Rescheduling-Korridore rührt dabei nicht primär von der Länge der auftretenden Störungen her, sondern ist durch die Abwesenheit ausreichender Schlupfzeiten zu begründen. Allerdings ist in dem Bereich ein gegenläufiger Effekt der Robustheit auf die mittlere Verspätung im Vergleich zum prädiktiven robusten Scheduling zu beobachten. Während diese ohne DPS nahezu linear zunimmt, wird sie mit DPS mit zunehmender Robustheit geringer. Dies ist dadurch zu erklären, dass die Schlupfzeiten des prädiktiven robusten Scheduling durch das reaktive Rescheduling so ausgenutzt werden können, dass im verbleibenden Schedule Verspätungen verringert werden.

Das Minimum der mittleren Verspätungen mit DPS wird bei $R = 0{,}8$ erreicht. Dabei wird die mittlere Verspätung ohne DPS um 7,3 % von 449 s auf 418 s verringert. Gegenüber dem besten Wert ohne DPS wird eine Verbesserung von 2,3 % von 428 s auf 418 s erreicht. Bei einer Erhöhung der Robustheit auf $R > 0{,}8$ ist wieder ein Ansteigen der mittleren Verspätung zu beobachten, die nahezu komplett parallel zum Scheduling ohne DPS verläuft. Es ist zu vermuten, dass die zusätzlichen Schlupfzeiten nur bis ca. $R = 0{,}8$ einen Effekt auf die Qualität des Rescheduling besitzen. Aus den Auswertungen in Kapitel 5.3 folgt, dass die Lösungsgüte direkt in Zusammenhang mit der Länge von Rescheduling-Korridoren steht. Daher sind in Abbildung 5-17 die in den Simulationsexperimenten aufgetretenen Längen der Rescheduling-Korridore dargestellt. Für jeden Wert von R ergibt sich mithilfe der jeweils 10 durchgeführten Simulationsläufe ein Datensatz von ca. 1200 Störungen. Der Boxplot je Simulationsexperiment stellt die aufgetretenen Minima und Maxima sowie das erste, zweite und dritte Quartil der Verteilung dar. Während die minimale Länge der Rescheduling-Korridore für alle untersuchten R keinen klaren Trend ausweist, ist zu beobachten, dass sich die ersten drei Quartile stetig nach unten verschieben. Während ein mittlerer Rescheduling-Korridor für $R = 0$ noch 1274 s umfasst, sinkt dieser Wert bis $R = 1$ auf 971 s. Dies entspricht einer Verringerung von 23,8 %. Gemeinsam mit der bis zu einem gewissen Grad steigenden

Lösungsgüte ergibt sich eine stagnierende Qualität des reaktiven Rescheduling bei Werten von $R > 0{,}8$.

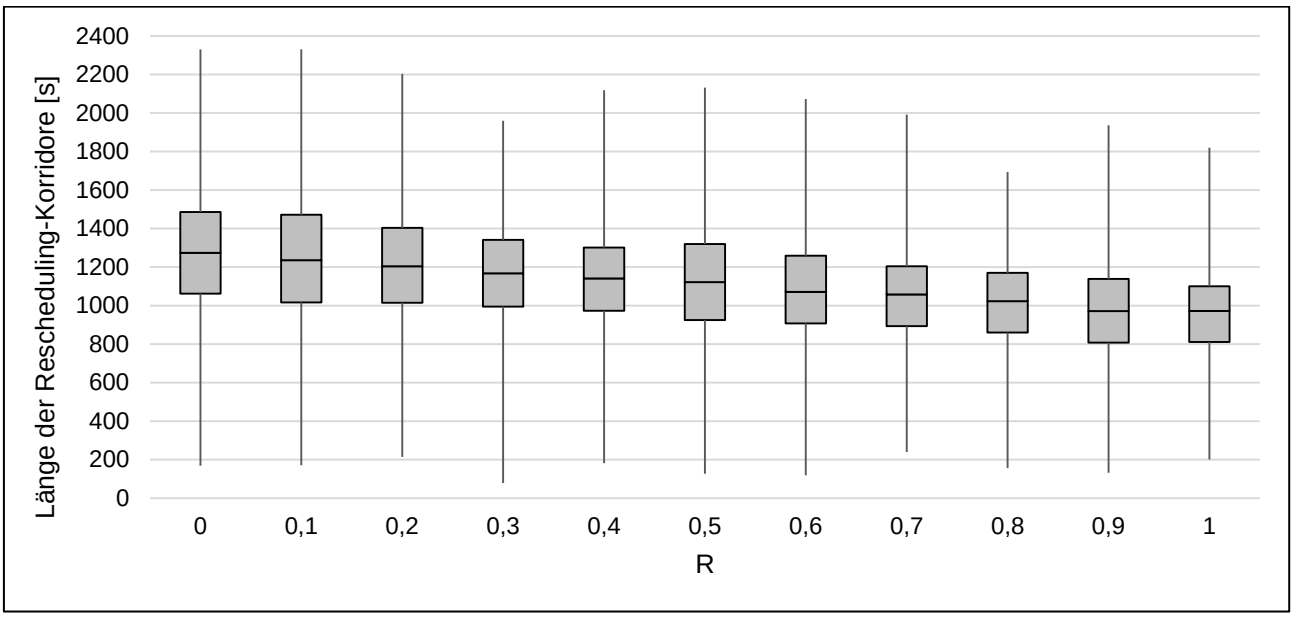


Abbildung 5-17: Boxplot der Länge der Rescheduling-Korridore in Abhängigkeit von der Robustheit R

Bei noch höheren Werten von R tritt der bereits in Kapitel 5.2 beschriebene Effekt auf, dass die Gleichverteilung der Schlupfzeiten gegenüber Erlang-verteilten Störungen zu einer Verringerung der Termintreue führt, da neben nicht kompensierten Störungszeiten nun auch noch nicht benötigte Schlupfzeiten die Bearbeitungsdauer erhöhen. Demgegenüber steht ein reaktives Rescheduling, dessen Lösungsgüte bezüglich der Makespan durch die noch höhere Robustheit nicht weiter ansteigt, sondern stagniert. Für den untersuchten Anwendungsfall kann damit festgestellt werden, dass eine Robustheit von $R = 0{,}8$ den optimalen Kompromiss zwischen zusätzlichen Schlupfzeiten und einer Befähigung des reaktiven Reschedulings darstellt. Der direkte Vergleich der jeweiligen Simulationsläufe der Experimente zeigt zudem, dass die mittlere Verspätung je Auftrag konsistent geringer ausfällt und zwischen 1,8 % und 3,7 % niedriger liegt.

5.4.2 Grenzen der Anwendung

Im beschriebenen Anwendungsfall wurden zwei wesentliche Beobachtungen gemacht. Zum einen ist die mittlere Verspätung je Auftrag bis $R = 0{,}8$ umso geringer, je höher die gewählte Robustheit ist. Damit liegt ein gegenläufiger Trend im Vergleich zur Anwendung des prädiktiven robusten Schedulings ohne das reaktive Rescheduling vor. Zum

anderen hat sich herausgestellt, dass mit einem Robustheitswert von $R = 0{,}8$ mit DPS die mittlere Verspätung gegenüber $R = 0$ ohne DPS verringert werden kann. Damit wird deutlich, dass die nachteilige Auswirkung des prädiktiven robusten Schedulings auf die Termintreue in einem bestimmten Robustheitsbereich durch DPS kompensiert werden kann. Aufgrund der geringen Anzahl betrachteter Simulationsläufe und der verhältnismäßig geringen Abweichung kann nicht mit Sicherheit davon ausgegangen werden, dass der ermittelte Robustheitswert optimal ist. Eine wesentliche Herausforderung für die Anwendung besteht darin, möglichst repräsentative Verläufe der auftretenden Störungen zu ermitteln, um eine verlässliche Aussage zu erhalten.

5.5 Prototypische Softwarerealisierung

Die Methode zum prädiktiv-reaktiven Scheduling wurde in einem Softwaredemonstrator prototypisch umgesetzt, dessen wesentliche Bausteine in Abbildung 5-18 dargestellt sind und nachfolgend erläutert werden.

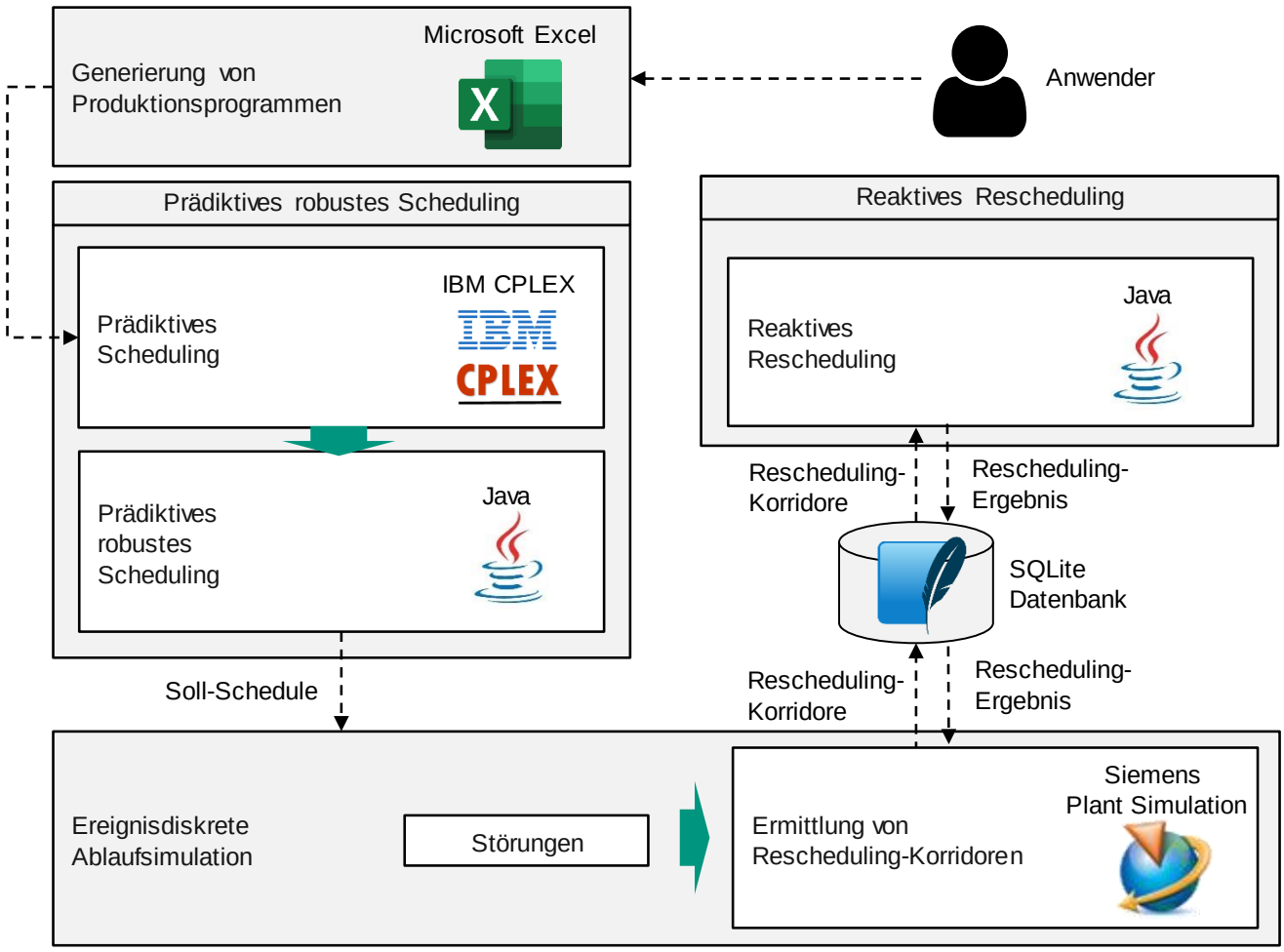


Abbildung 5-18: Aufbau der prototypischen Softwarerealisierung

Zu Beginn wird ein Produktionsprogramm benötigt, das wie in Kapitel 5.1.2 aufgebaut ist. Zur Erzeugung dieser Produktionsprogramme wurde ein Excel-basiertes Planungstool entwickelt, in dem die Anzahl der Fahrzeuge sowie der gewünschte Variantenmix durch den Benutzer definiert werden kann. Es lassen sich ebenfalls neue Stationen und Produkte frei definieren, sodass die Übertragbarkeit gegeben ist. Daraus wird ein Produktionsprogramm erstellt, in dem die Fahrzeuge in eine zufällige Reihenfolge gebracht werden und mit entsprechenden Fälligkeitsterminen versehen werden. Die Vorder- und Heckwagen für das erste Fahrzeug sollen nach einer Stunde fertiggestellt sein. Die Fälligkeitstermine für nachfolgende Vorder- und Heckwagen sind in Abständen von je 90 s gestaffelt, um einen gleichmäßigen Zu- und Abfluss von Aufträgen in das Produktionssystem zu gewährleisten.

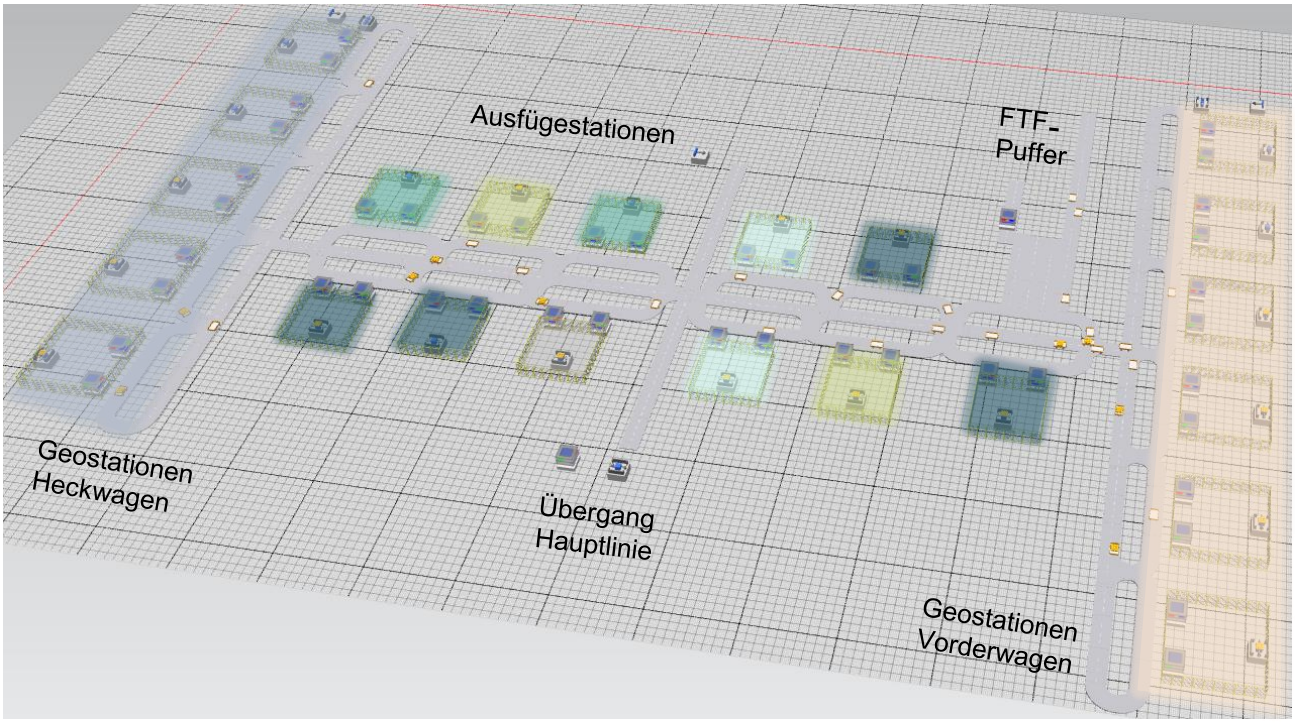


Abbildung 5-19: Simulationsmodell der Matrix-Produktion von Vorder- und Heckwagen in Siemens Plant Simulation®

Zunächst wird das prädiktive robuste Scheduling aus Kapitel 4.3 durchgeführt. Das prädiktive Scheduling aus Kapitel 4.3.1 wird mithilfe der in der Software ILOG CPLEX® 12.9 von IBM implementierten Constraintprogrammierung gelöst. Dabei wird das Produktionsprogramm in einen prädiktiven Schedule überführt, in dem der mit der Verfügbarkeit der benötigten Stationen gewichtete Schlupf je Auftrag maximiert wird. Für das prädiktive robuste Scheduling aus Kapitel 4.3.3 wurde das Post Processing in Java 8

implementiert. Dieses lässt sich über den zu verwendenden prädiktiven Schedule sowie die gewünschte Robustheit parametrieren und generiert schließlich den Soll-Schedule als Input für die ereignisdiskrete Ablaufsimulation. Diese wurde in der Simulationssoftware Plant Simulation® 14.2 von Siemens umgesetzt und stellt ein digitales Abbild der in Kapitel 5.1.1 beschriebenen Matrix-Produktion dar. Das Simulationsmodell mit seinen Elementen ist in Abbildung 5-19 dargestellt. Die Einfärbung der Stationen ist dabei analog zu Abbildung 5-3. Der Soll-Schedule wird in der Simulation verfolgt, so lange keine Störungen an Stationen auftreten. Bei Auftreten einer Störung wird ein Rescheduling-Korridor wie in Kapitel 4.4 beschrieben ermittelt. Der Algorithmus aus Kapitel 4.4.2 ist in die Simulation eingebettet und wurde in der Skriptsprache SimTalk® implementiert. Während der Ermittlung des Rescheduling-Korridors und dem Rescheduling pausiert die Simulation, bis ein Rescheduling-Ergebnis vorliegt.

Zur in Kapitel 4.4.3 beschriebenen Kopplung mit dem reaktiven Rescheduling wird eine SQLite Datenbank verwendet, mit der die relationale Logik der Daten abgebildet werden kann. Die Datenbank speichert die ermittelten Rescheduling-Korridore sowie die angelernten Parameter θ und die in der Anwendung erzeugten Rescheduling-Ergebnisse zur Reintegration in den prädiktiven robusten Schedule $Sched_{Rob}$. Ferner wurde in der Simulation und im reaktiven Rescheduling ein sogenannter Datenbank-Listener implementiert, sodass die jeweiligen Module selbständig aktiv werden, sobald diese benötigt werden.

Das reaktive Rescheduling aus Kapitel 4.5 ist ebenfalls in Java 8 implementiert worden. Sobald ein neuer Rescheduling-Korridor in der SQLite Datenbank vorliegt, werden die hinterlegten θ zur Durchführung des reaktiven Reschedulings verwendet. Anschließend wird dessen Makespan mit der Länge des Rescheduling-Korridors verglichen. Liegt diese unterhalb des Rescheduling-Korridors, wird das Rescheduling-Ergebnis an die Simulation weitergereicht und integriert, andernfalls wird der Rescheduling-Korridor in der Simulation erweitert und wiederum über die Datenbank ein reaktives Rescheduling angestoßen. Die Rechenzeiten für die iterative Erweiterung und Lösung sind aufgrund des vergleichsweise einfachen Aufbaus der Algorithmen vernachlässigbar. Die Simulation bricht ab, wenn alle Bearbeitungsschritte des Soll-Schedules durchgeführt wurden und keine Störungen mehr auftreten.

Für einen potenziellen Anwender besteht der wesentliche Aufwand in der Anpassung der prototypischen Softwarerealisierung in dem Aufbau einer adäquaten Ablaufsimula-

tion, die für die Ermittlung eines optimalen Robustheitswertes unerlässlich ist. Die Module zum prädiktiven robusten Scheduling und dem reaktiven Rescheduling sind so aufgebaut, dass nur geringfügige Änderungen der Parameter (Anzahl und Verfügbarkeit von Stationen, Anforderungs-, Transport-, Rüst- und Prozesszeiten, sowie Vorrangbeziehungen der zu produzierenden Auftragstypen) vorgenommen werden müssen.

6 Bewertung und Ausblick

Die in dieser Arbeit entwickelte Methode zum prädiktiv-reaktiven Scheduling und die aus der Erprobung gewonnenen Erkenntnisse werden in diesem Kapitel den Anforderungen an die Methode aus Kapitel 4.1 und der Zielsetzung aus Kapitel 1.2 gegenübergestellt und bewertet. Abschließend wird ein Ausblick auf mögliche weitergehende Forschungsthemen gegeben.

6.1 Bewertung

Die entwickelte Methode ist für vielfältige Produktionssysteme einsetzbar und damit breit anwendbar. Durch die Modellierung als flexibler Job Shop lassen sich unterschiedlichste Systemkonfigurationen ableiten, mit denen sich neben der Matrix-Produktion alle in Kapitel 2.1.2 eingeführten Fertigungskonzepte abbilden lassen. Sowohl im prädiktiven robusten Scheduling als auch im reaktiven Rescheduling werden Transportzeiten zwischen Stationen, Anforderungszeiten für Transportmittel, reihenfolgeabhängige Rüstzeiten sowie Puffer berücksichtigt. Eine explizite Betrachtung von Losgrößen findet nicht statt. Lose könnten allerdings als einzelner Auftrag mit längeren Bearbeitungszeiten modelliert werden. Es ist dann lediglich ein losweiser Transport zwischen den Stationen realisierbar, der zwangsläufig mit einem einzelnen FTF durchführbar sein muss. Eine Aufteilung von Losen ist hiermit allerdings nicht möglich. Die in Kapitel 4.2 formulierten Anforderungen an den Lösungsansatz wurden vollständig berücksichtigt und konnten bis auf eine später nochmal aufgegriffene Einschränkung hinsichtlich der Echtzeitfähigkeit des reaktiven Reschedulings vollumfänglich erfüllt werden. Im Folgenden wird abschließend auf die wesentlichen entwickelten Teilmodelle eingegangen, deren Gesamtzusammenhang übersichtsartig nochmal in Abbildung 6-1 dargestellt ist.

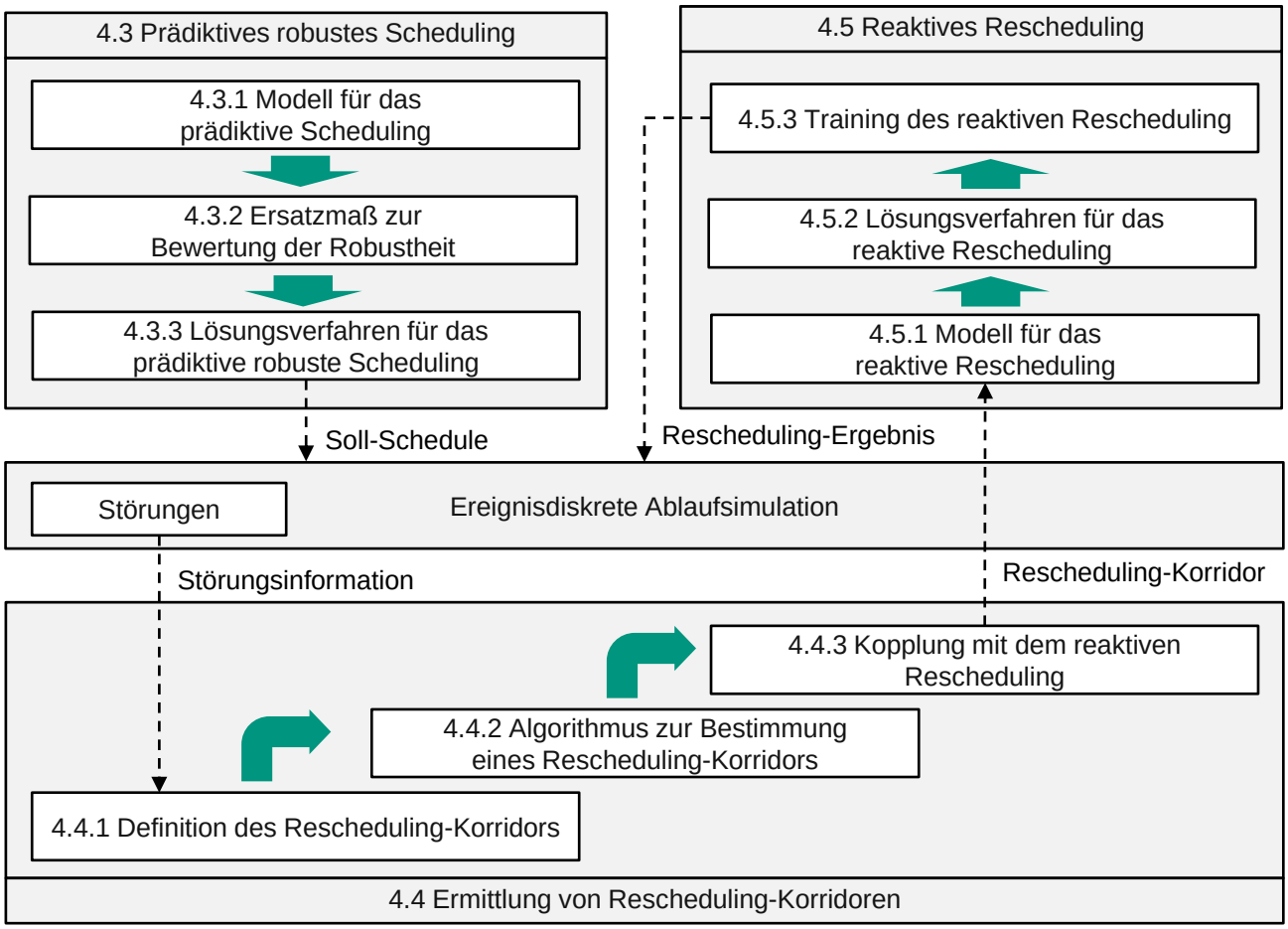


Abbildung 6-1: Überblick über den entwickelten Lösungsansatz

Das prädiktive robuste Scheduling (Kapitel 4.3) wurde als constraint-basiertes Problem formuliert, sodass eine nahezu optimale Lösung nach wenigen Stunden erreicht werden kann. Die Lösungsgüte im Vergleich zum Optimum ließ sich für die Produktionsprogramme aus der Erprobung aufgrund deren Größe und Komplexität nicht nachweisen. Das Robustheitsmaß verwendet die Verfügbarkeit von Stationen zur Bestimmung von Schlupfzeiten. Eine Integration in das constraint-basierte Problem war nicht möglich, da die große Anzahl zusätzlicher Restriktionen zu einer zu starken Erhöhung der Rechenzeit geführt hätte. Aus diesem Grund wurde dem prädiktiven Scheduling nachgelagert ein Algorithmus definiert, der Schlupfzeiten nachträglich in einen bestehenden prädiktiven Schedule integriert. Das Ausmaß der Schlupfzeiten kann dabei als kontrollierbarer Parameter durch den Anwender festgelegt werden. Es wurde damit beschrieben, wie eine prädiktive Planung Robustheit gegenüber dynamischen Störungen erlangen kann. In der Erprobung hat sich gezeigt, dass das entwickelte Vorgehen besonders effektiv für geringe Werte der $MTTR_s$ ist, aufgrund der statistischen Herangehensweise

aber umso schlechter geeignet ist, je seltener und länger Störungen an Stationen auftreten. Bei sporadischen und langen Störungen ist grundsätzlich fraglich, inwiefern ein Match-Up-Ansatz basierend auf statistischen Störungsverteilungen sinnvoll ist.

Zur Ermittlung eines Rescheduling-Korridors bei längeren Störungen wurde ein Algorithmus (Kapitel 4.4) entwickelt. Dieser ermittelt bei Auftreten einer Störung ausgehend von der Störungsdauer sowie dem prädiktiven robusten Schedule einen Ausschnitt des Schedules (Rescheduling-Korridor genannt), der durch das reaktive Rescheduling umgeplant werden soll. Der Algorithmus basiert auf der Idee, die Verzögerungen durch die Störungen durch die Ausnutzung von Schlupfzeiten zu kompensieren und entsprechend einen Rescheduling-Korridor zu erzeugen, der über ausreichend Schlupfzeit verfügt. Dabei stellt der Algorithmus keine Optimierung, sondern eine Heuristik dar, da bei der Ermittlung der Schlupfzeiten logische Restriktionen wie Vorrangbeziehungen oder zusammenhängende Schlupfzeiten nicht berücksichtigt werden. Dies wurde so gewählt, da sonst bei der Bestimmung des Rescheduling-Korridors bereits wesentliche Entscheidungen des reaktiven Reschedulings vorweggenommen würden und somit dessen Lösungsraum eingeschränkt wäre. Stattdessen wurde der Ansatz so aufgebaut, dass ein Rescheduling-Korridor mit zu wenig Schlupfzeit schrittweise erweitert werden kann. So können die Freiheitsgrade für das reaktive Rescheduling erhalten werden.

Im reaktiven Rescheduling (Kapitel 4.5) wurde mit der Modellierung als dezentraler Markovscher Entscheidungsprozess eine Möglichkeit geschaffen, das Rescheduling in sehr kurzer Zeit mit einer Lösungsgüte oberhalb ausgewählter statischer Prioritätsregeln durchzuführen. Aufgrund der hohen Unterschiedlichkeit der Rescheduling-Korridore hinsichtlich der umzuplanenden Aufträge ist eine Verallgemeinerung des Lernvorgangs nicht möglich. Stattdessen wird beim Auftreten einer längeren Störung jeder Rescheduling-Korridor zunächst angelernt und anschließend ein reaktives Rescheduling mit den spezifisch angelernten Parametern durchgeführt. Damit wurde ein Match-Up-Ansatz realisiert, bei dem nur ein möglichst kleiner Teil der ursprünglichen Schedules umgeplant wird. Es wurde hiermit eine Möglichkeit geschaffen, auf längere Störungen zu reagieren und das nötige Rescheduling in kurzer Zeit mit hoher Lösungsgüte durchzuführen.

In der Erprobung wurde die entwickelte Methode systematisch analysiert. Es wurde in der Anwendung des prädiktiven robusten Schedulings (Kapitel 5.2) gezeigt, dass ein Trade-Off zwischen der mittleren absoluten Verschiebung von Bearbeitungsschritten

und der mittleren Verspätung von Aufträgen besteht: Je höher die Robustheitsanforderung an den prädiktiven robusten Schedule, desto mehr Schlupfzeiten sind dort vorhanden. Das führt dazu, dass einerseits weniger Verschiebungen von Bearbeitungsschritten auftreten, andererseits steigen die Verspätungen von Aufträgen ohne ein geeignetes Rescheduling stark an. Es wurde ersichtlich, dass zur Erreichung einer hohen Termintreue entsprechend ein Verfahren für das reaktive Rescheduling erforderlich ist.

In der Anwendung des reaktiven Reschedulings (Kapitel 5.3) hat sich gezeigt, dass die entwickelte Methode vor allem für kurze Rescheduling-Korridore eine sehr kleine Optimalitätslücke aufweist und Rescheduling-Ergebnisse klar oberhalb der Leistungsfähigkeit einfacher Prioritätsregeln erzielt werden können. Einschränkend muss gesagt werden, dass die Rechenzeiten für das Anlernen im Vergleich zu Prioritätsregeln dazu führen können, dass die Lösungsgüte des reaktiven Reschedulings insgesamt weniger stark vorteilhaft ist. Es hat sich zudem gezeigt, dass die Optimalitätslücke ansteigt, je länger die umzuplanenden Rescheduling-Korridore sind. Dies hat nochmals das Bestreben untermauert, möglichst kurze Rescheduling-Korridore zu erhalten, was über ein größeres Ausmaß von Schlupfzeiten und damit eine höhere Robustheitsanforderung realisiert werden kann.

Bei der Ermittlung der optimalen Robustheit im prädiktiven robusten Scheduling (Kapitel 5.4) wurde gezeigt, dass eine höhere Robustheit im prädiktiven robusten Schedule zu kürzeren Rescheduling-Korridoren führt und damit das Ausmaß der Umplanungen insgesamt verringert werden kann. Es hat sich zudem gezeigt, dass für den untersuchten Anwendungsfall ein Trade-Off zwischen Termintreue und Robustheit besteht: Eine zu hohe Robustheit im prädiktiven robusten Scheduling bedeutet große Schlupfzeiten, die in Verbindung mit dem reaktiven Rescheduling nicht benötigt werden. Eine zu geringe Robustheit hingegen führt zu längeren Rescheduling-Korridoren und damit ebenfalls geringerer Termintreue. Exemplarisch wurde für den untersuchten Anwendungsfall gezeigt, dass ein Optimum der Robustheit im prädiktiven robusten Scheduling gefunden werden kann. Dieses hängt davon ab, welches Verfahren für das reaktive Rescheduling verwendet wird und wie die Matrix-Produktion und das Produktionsprogramm aufgebaut sind. Zudem besteht eine Abhängigkeit von den untersuchten Störungsszenarien, sodass auf deren möglichst gute Realitätsnähe geachtet werden muss. Die optimale Robustheit im prädiktiven robusten Scheduling lässt sich nicht allgemein herleiten, sondern muss für jeden Anwendungsfall neu ermittelt werden.

Hinsichtlich der eingangs aufgestellten These, dass eine reaktive Umplanung durch eine robuste prädiktive Planung befähigt wird kann abschließend gesagt werden, dass die erlangten Erkenntnisse diese stützen.

6.2 Ausblick

Aus der vorliegenden Arbeit ergeben sich folgende konkrete Anknüpfungspunkte zur Weiterentwicklung der Methodik:

Im prädiktiven Schedule wird auftragsbezogen jeweils möglichst viel Schlupf zum Fälligkeitstermin vorgehalten, der in einem nachgelagerten Schritt im Schedule dann verteilt wird. Allerdings ist nicht garantiert, dass der gesamte Schlupf auch tatsächlich verteilt wird, da dieser Wert sich an den erwarteten Verschiebungen durch Störungen orientiert. In einem Einsatzszenario ohne Störungen würden damit alle Aufträge gleichmäßig früh vor ihrem Fälligkeitstermin fertiggestellt und es würde ein großes Zwischenlager am Ausgang des Systems notwendig, während Pufferkapazitäten im Produktionssystem ungenutzt blieben. Für die robuste prädiktive Planung könnte daher untersucht werden, inwiefern eine Rückkopplung der Ergebnisse nach der Verteilung der Schlupfzeiten an die Optimierung zur Verbesserung beitragen kann, indem Schlupfzeiten noch gezielter dort eingebracht werden, wo sie benötigt werden.

Bei der Ermittlung von Rescheduling-Korridoren wurden keine Informationen zu den Vorrangbeziehungen zwischen Bearbeitungsschritten berücksichtigt, um den Rechenaufwand gering zu halten und keine Rescheduling-Entscheidungen implizit vorab zu treffen. In der Anwendung kommt es daher sehr oft zu Iterationsschleifen, in denen der Rescheduling-Korridor erweitert werden muss, bevor mit dem reaktiven Rescheduling eine zulässige Lösung gefunden werden kann. Hier könnte untersucht werden, welches Ausmaß an zusätzlicher Information bei der Ermittlung der Rescheduling-Korridore verwendet werden kann, um die Anzahl nötiger Iterationen und damit den Rechenaufwand zu verringern.

Zudem könnte eine sinnvolle Erweiterung des Ansatzes darin bestehen, bei Auftreten einer Störung zu prüfen, ob ein reaktives Rescheduling mit Match-Up-Zeitpunkt überhaupt noch sinnvoll durchführbar ist, oder ob eine vollständige Neugenerierung mit Berücksichtigung aller Fälligkeitstermine zielführender zur Aufrechterhaltung einer hohen Liefertreue ist. Bei sehr kurzen Rescheduling-Korridoren könnte zudem geprüft werden,

ob ein Verfahren wie DPS überhaupt nötig ist, oder ob mithilfe einer Prioritätsregel vergleichbar gute Ergebnisse bei deutlich geringerer Komplexität und ohne ein Anlernen erzielt werden können.

Beim reaktiven Rescheduling könnte weiterhin erforscht werden, ob noch weitere Parameter als θ für die Rescheduling-Entscheidungen berücksichtigt werden sollen. Zur Verbesserung der Liefertreue könnte beispielsweise ein zusätzlicher Parameter eingeführt werden, der angibt, ob ein wartender Auftrag sich kurz vor seinem Fälligkeitstermin befindet oder nicht. So könnte eine Dringlichkeitsunterscheidung zwischen Aufträgen getroffen werden, die noch feingranularere Scheduling-Entscheidungen zulässt. Demgegenüber steht allerdings ein deutlich erhöhter Rechenaufwand, da sich die Anzahl der zu lernenden Parameter selbst bei einer Unterscheidung in „dringend“ und „nicht dringend“ bereits verdoppeln. Bei Erweiterungen dieser Art muss also stets der Trade-Off zwischen tatsächlicher Verbesserung der Lösungsgüte und Erhöhung des Anlernaufwandes berücksichtigt werden.

Letztlich ist es nach derzeitigem Stand der Technik nicht ohne weiteres möglich, die Ergebnisse des Lernprozesses auf bisher ungesehene Rescheduling-Korridore zu übertragen, weshalb ein separates Anlernen bei jeder Störung derzeit unumgänglich ist. Hierzu könnten in Zukunft Ansätze untersucht werden, wie eine Verallgemeinerung realisiert werden kann, um den Anlernaufwand zu verringern und noch kürzere Reaktionszeiten beim Rescheduling zu ermöglichen.

7 Zusammenfassung

Bedingt durch eine steigende Individualisierung von Produkten bieten produzierende Unternehmen immer mehr Varianten bei gleichzeitig sinkenden Stückzahlen je Variante an. Zudem wird die Kundennachfrage zunehmend volatiler und immer schwerer prognostizierbar. Die wesentliche Herausforderung besteht darin, einen schwankenden Variantenmix bei schwankenden Gesamtstückzahlen wirtschaftlich herzustellen.

Die Matrix-Produktion als dynamisch an interne und externe Einflussfaktoren anpassbares Produktionssystem erlaubt eine vom Takt entkoppelte Produktion in Losgröße 1 und ist daher aktueller Forschungsgegenstand. Neben der Gestaltung dieser Systeme kommt der PPS eine zunehmend wichtigere Rolle zu, da die Materialflüsse in solchen Produktionssystemen eine hohe Komplexität aufweisen, die mit bisher eingesetzten einfachen Verfahren nicht beherrscht werden kann. Der Stand der Forschung wird durch eine Vielzahl von prädiktiv-reaktiven Methoden zum Scheduling auch in komplexen Produktionssystemen geprägt. Es existiert jedoch kein Ansatz, der gezielt die Robustheit in der prädiktiven Planung berücksichtigt, um das reaktive Rescheduling zu befähigen, trotz unvorhergesehener Störungen die gewünschte Liefertreue zu erhalten.

Daher wurde in der vorliegenden Arbeit eine Methode zum prädiktiv-reaktiven Scheduling in der Matrix-Produktion entwickelt, welche die Ermittlung eines optimalen Grades der Robustheit im prädiktiven robusten Scheduling erlaubt und damit einen optimalen Mix aus Prävention und Reaktion in der Produktionssteuerung ermöglicht. Die Methode besteht aus drei wesentlichen Bausteinen. Im prädiktiven robusten Scheduling wird zunächst auf Basis des Produktionsprogramms ein Schedule erzeugt, in dem anschließend ein gewünschtes Ausmaß an Schlupfzeiten zwischen Bearbeitungsschritten eingefügt werden. Die robusten Schedules werden simulativ ausgeführt. Bei Eintreten längerer Störungen wird ein Rescheduling-Korridor ermittelt der in Abhängigkeit der Störung und des zugrundeliegenden Schedules angibt, welche Bearbeitungsschritte welcher Aufträge umgeplant werden müssen. Die Rescheduling-Korridore werden anschließend im reaktiven Rescheduling umgeplant und in den ursprünglichen Schedule übertragen. Das reaktive Rescheduling setzt dabei ein auf einem dezentralen Markov-Prozess basierendes RL ein, um stationsabhängig optimale Auswahlstrategien von Aufträgen zu erlernen. Die Methode wurde im Rahmen des BMWi-geförderten Verbundprojektes „SmartBodySynergy“ bei einem Anwendungspartner aus der Automobilindustrie im Karosseriebau erprobt.

Die entwickelte Methode leistet einen wesentlichen Beitrag zum Einsatz des Konzeptes der Robustheit in der Produktionssteuerung. Die vorliegende Arbeit stellt nach Wissen des Autors den ersten Ansatz der, der die Robustheit gezielt in ein prädiktiv-reaktives Scheduling integriert.

8 Literaturverzeichnis

Verweise gemäß dem Schema (A_Name Jahr) beziehen sich auf studentische Arbeiten, die vom Verfasser der Dissertation angeleitet wurden.

(A_Hort 2019)

Hort, S. (2019), *Optimierung der Materialflusssteuerung in der Matrix-Produktion durch maschinelles Lernen*. Masterarbeit, Karlsruher Institut für Technologie, Karlsruhe, wbk Institut für Produktionstechnik.

(A_Neupert 2018)

Neupert, N. (2018), *Robuste Produktionsplanung in der Matrix-Produktion*. Masterarbeit, Karlsruher Institut für Technologie, Karlsruhe, wbk Institut für Produktionstechnik.

(A_Zhang 2018)

Zhang, G. (2018), *Robuste Produktionsplanung in der Matrix-Produktion*. Masterarbeit, Karlsruher Institut für Technologie, Karlsruhe, wbk Institut für Produktionstechnik.

(A_Zöllner 2019)

Zöllner, J. (2019), *Echtzeitfähige Kopplung von zentralem und dezentralem Scheduling in der Matrixproduktion*. Masterarbeit, Karlsruher Institut für Technologie, Karlsruhe, wbk Institut für Produktionstechnik.

(Abele & Reinhart 2011)

Abele, E. & Reinhart, G. (2011), *Zukunft der Produktion*. *Herausforderungen, Forschungsfelder, Chancen,* Hanser, München. ISBN: 978-3-446-42805-4.

(Abumaizar & Svestka 1997)

Abumaizar, R. J. & Svestka, J. A. (1997), „Rescheduling job shops under random disruptions“, *International Journal of Production Research*, 35(7), S. 2065–2082.

(Adams & Balas et al. 1988)

Adams, J.; Balas, E. & Zawack, D. (1988), „The Shifting Bottleneck Procedure for Job Shop Scheduling“, *Management Science*, 34(3), S. 391–401.

(AhmadBeygi & Cohn et al. 2008)

AhmadBeygi, S.; Cohn, A. & Lapp, M. (2008), „Decreasing Airline Delay Propagation by Re-Allocating Scheduled Slack“, *IIE Transactions*, 42(7), S. 478–489.

(Akturk & Gorgulu 1999)

Akturk, M.S. & Gorgulu, E. (1999), „Match-up scheduling under a machine breakdown“, *European Journal of Operational Research*, 112(1), S. 81–97.

(Al-Fawzan & Haouari 2005)

Al-Fawzan, M. A. & Haouari, M. (2005), „A bi-objective model for robust resource-constrained project scheduling“, *International Journal of Production Economics*, 96(2), S. 175–187.

(Al-Hinai & ElMekkawy 2011)

Al-Hinai, N. & ElMekkawy, T. Y. (2011), „Robust and stable flexible job shop scheduling with random machine breakdowns using a hybrid genetic algorithm“, *International Journal of Production Economics*, 132(2), S. 279–291.

(Allahverdi & Ng et al. 2008)

Allahverdi, A.; Ng, C.T.; Cheng, T.C.E. & Kovalyov, M. Y. (2008), „A survey of scheduling problems with setup times or costs“, *European Journal of Operational Research*, 187(3), S. 985–1032.

(Aloulou & Portmann 2003)

Aloulou, M. A. & Portmann, M.-C. (2003), „An Efficient Proactive-Reactive Scheduling Approach to Hedge Against Shop Floor Disturbances. Multidisciplinary Scheduling: Theory and Applications“. *Multidisciplinary International Scheduling Conference,* 13. - 16. August 2003, Nottingham, UK, S. 223–245.

(Audi 2016)

Audi (2016), *Die modulare Montage*. https://www.audi-mediacenter.com/de/audi-techday-smart-factory-7076/die-modulare-montage-7078 [14.08.2019].

(Aytug & Lawley et al. 2005)

Aytug, H.; Lawley, M. A.; McKay, K.; Mohan, S. & Uzsoy, R. (2005), „Executing production schedules in the face of uncertainties. A review and some future directions“, *European Journal of Operational Research*, 161(1), S. 86–110.

(Baird 1999)

Baird, L. C. (1999), *Reinforcement Learning through Gradient Descent*. Dissertation, Carnegie Mellon University, Pittsburgh.

(Bauer & Arnold et al. 2010)

Bauer, W.; Arnold, H.; Kuhnert, F. & Kurtz, R. (2010), *Elektromobilität - Herausforderung für Industrie und öffentliche Hand*. https://wiki.iao.fraunhofer.de/images/studien/elektromobilitaet-herausforderungen-fuer-industrie-und-oeffentliche-hand.pdf [10.12.2019].

(Bauernhansl & Hompel et al. 2014)

Bauernhansl, T.; Hompel, M. ten & Vogel-Heuser, B. (Hrsg.) (2014), *Industrie 4.0 in Produktion, Automatisierung und Logistik,* Springer Fachmedien, Wiesbaden. ISBN: 978-3-658-04681-1.

(Bean & Birge et al. 1991)

Bean, J. C.; Birge, J. R.; Mittenthal, J. & Noon, C. E. (1991), „Matchup Scheduling with Multiple Resources, Release Dates and Disruptions“, *Operations Research*, 39(3), S. 470–483.

(Bellman 1954)

Bellman, R. (1954), „The theory of dynamic programming“, *Bulletin of the American Mathematical Society*, 60(6), S. 503–515.

(Ben-Tal & Nemirovski 2002)

Ben-Tal, A. & Nemirovski, A. (2002), „Robust optimization. Methodology and applications“, *Mathematical Programming*, 92(3), S. 453–480.

(Bernstein & Givan et al. 2002)

Bernstein, D. S.; Givan, R.; Immerman, N. & Zilberstein, S. (2002), „The Complexity of Decentralized Control of Markov Decision Processes“, *Mathematics of Operations Research*, 27(4), S. 819–840.

(Billaut & Moukrim et al. 2008)

Billaut, J.-C.; Moukrim, A. & Sanlaville, E. (Hrsg.) (2008), *Flexibility and Robustness in Scheduling,* ISTE Ltd and John Wiley & Sons Inc, London. ISBN: 978-0-470-61143-2.

(Blazewicz & Ecker et al. 2007)

Blazewicz, J.; Ecker, K. H.; Pesch, E.; Schmidt, G. & Weglarz, J. (2007), *Handbook on Scheduling,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-540-28046-0.

(BMBF 2015)

BMBF (2015), *Zukunftsbild „Industrie 4.0“*. https://www.plattform-i40.de/PI40/Redaktion/DE/Downloads/Publikation/zukunftsbild-industrie-4-0.html [07.12.2019].

(Bochmann 2018)

Bochmann, L. S. (2018), *Entwicklung und Bewertung eines flexiblen und dezentral gesteuerten Fertigungssystems für variantenreiche Produkte*. Dissertation, ETH Zürich, Innovation Center Virtual Reality, Zürich.

(Bouazza & Sallez et al. 2017)

Bouazza, W.; Sallez, Y. & Beldjilali, B. (2017), „A distributed approach solving partially flexible job-shop scheduling problem with a Q-learning effect“. *20th IFAC World Congress,* S. 15890–15895.

(Box & Andersen 1955)

Box, G. E. P. & Andersen, S. L. (1955), „Permutation Theory in the Derivation of Robust Criteria and the Study of Departures from Assumption“, *Journal of the Royal Statistical Society: Series B (Methodological) banner*, 17(1), S. 1–34.

(Brandimarte 1993)

Brandimarte, P. (1993), „Routing and scheduling in a flexible job shop by tabu search“, *Annals of Operations Research*, 41(3), S. 157–183.

(Brandmüller & Önnerfors et al. 2018)

Brandmüller, T.; Önnerfors, Å. & Reinecke, P. (2018), *Eurostat regional yearbook*. *2018 edition,* Publications Office of the European Union, Luxembourg. ISBN: 978-9-279-87878-7.

(Brucker & Schlie 1990)

Brucker, P. & Schlie, R. (1990), „Job-shop scheduling with multi-purpose machines“, *Computing*, 45(4), S. 369–375.

(Bürgin 2018)

Bürgin, J. (2018), *Robuste Auftragsplanung in Produktionsnetzwerken*. *Mittelfristige Planung der variantenreichen Serienproduktion unter Unsicherheit der Kundenauftragskonfigurationen*. Dissertation, Karlsruher Institut für Technologie, wbk Institut für Produktionstechnik.

(Burke & De Causmaecker et al. 2010)

Burke, E. K.; De Causmaecker, P.; Mulder, J.; Paelnick, M. & Vanden Berghe, G.

(2010), „A multi-objective approach for robust airline scheduling“, *Computers & Operations Research*, 37(5), S. 822–832.

(Busoniu & Babuska et al. 2008)
Busoniu, L.; Babuska, R. & Schutter, B. de (2008), „A Comprehensive Survey of Multiagent Reinforcement Learning“, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(2), S. 156–172.

(Çaliş & Bulkan 2015)
Çaliş, B. & Bulkan, S. (2015), „A research survey. Review of AI solution strategies of job shop scheduling problem“, *Journal of Intelligent Manufacturing*, 26(5), S. 961–973.

(Caprara & Galli et al. 2010)
Caprara, A.; Galli, L.; Kroon, L.; Maróti, G. & Toth, P. (2010), „Robust Train Routing and Online Re-scheduling“. *ATMOS 2010 - 10th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems,* 9. September 2010, Liverpool, England,S. 24–33.

(Chaudhry & Khan 2016)
Chaudhry, I. A. & Khan, A. A. (2016), „A research survey. Review of flexible job shop scheduling techniques“, *International Transactions in Operational Research*, 23(3), S. 551–591.

(Church & Uzsoy 1992)
Church, L. K. & Uzsoy, R. (1992), „Analysis of periodic and event-driven rescheduling policies in dynamic shops“, *International Journal of Computer Integrated Manufacturing*, 5(3), S. 153–163.

(Claus & Herrmann et al. 2015)
Claus, T.; Herrmann, F. & Manitz, M. (2015), *Produktionsplanung und –steuerung,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-662-43541-0.

(Corsten & Gössinger 2012)
Corsten, H. & Gössinger, R. (2012), *Produktionswirtschaft*. *Einführung in das industrielle Produktionsmanagement,* Oldenbourg, München. ISBN: 978-3-486-70569-0.

(Cowling & Johansson 2002)

Cowling, P. & Johansson, M. (2002), „Using real time information for effective dynamic scheduling“, *European Journal of Operational Research*, 139(2), S. 230–244.

(Csaji & Monostori 2006)

Csaji, B. & Monostori, L. (2006), „Adaptive Algorithms in Distributed Resource Allocation“. *Proceedings of the 6th International Workshop on Emergent Synthesis (IWES 2006),* 18. - 19. August 2006, Tokio, Japan, S. 69–75.

(Daimler 2017)

Daimler (2017), *Der Rohbau der Zukunft ist flexibel*. https://media.daimler.com/marsMediaSite/de/instance/ko/Mercedes-Benz-Cars-Der-Rohbau-der-Zukunft-ist-flexibel.xhtml?oid=30023981 [14.08.2019].

(Davenport & Gefflot et al. 2001)

Davenport, A.; Gefflot, C. & Beck, C. (2001), „Slack-Based Techniques for Robust Schedules“. *Proceedings of the sixth european conference on planning,* 12. - 14. September 2001, Toledo, Spanien, S. 43–49.

(Dempster & Lenstra et al. 1982)

Dempster, M. A. H.; Lenstra, J. K. & Rinnooy Kan, A. H. G. (1982), *Deterministic and Stochastic Scheduling,* Springer Netherlands, Dordrecht. ISBN: 978-9-400-97803-4.

(Dichtl & Issing 1987)

Dichtl, E. & Issing, O. (Hrsg.) (1987), *Vahlens großes Wirtschaftslexikon,* Beck, München. ISBN: 978-3-800-61698-X.

(Duenas & Petrovic 2008)

Duenas, A. & Petrovic, D. (2008), „An approach to predictive-reactive scheduling of parallel machines subject to disruptions“, *Annals of Operations Research*, 159(1), S. 65–82.

(Eberlin & Hock 2014)

Eberlin, S. & Hock, B. (2014), *Zuverlässigkeit und Verfügbarkeit technischer Systeme,* Springer Fachmedien Wiesbaden, Wiesbaden. ISBN: 978-3-658-03572-3.

(Engell & Märkert et al. 2001)

Engell, S.; Märkert, A.; Sand, G.; Schultz, R. & Schulz, C. (2001), „Online Scheduling of Multiproduct Batch Plants under Uncertainty“ in *Online optimization of large scale systems,* Hrsg. M. Grötschel, S. O. Krumke & J. Rambau, Springer, Berlin, S. 649–676.

(Ertel 2016)

Ertel, W. (2016), *Grundkurs Künstliche Intelligenz,* Springer Fachmedien Wiesbaden, Wiesbaden. ISBN: 978-3-658-13548-5.

(Eversheim 1989)

Eversheim, W. (1989), *Organisation in der Produktionstechnik Band 4,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-642-64800-7.

(Eversheim 1996)

Eversheim, W. (1996), *Organisation in der Produktionstechnik*. *Band 1: Grundlagen,* Springer, Berlin, Heidelberg. ISBN: 978-3-642-87738-4.

(Eversheim 2002)

Eversheim, W. (2002), *Organisation in der Produktionstechnik 3,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-642-62640-1.

(Eversheim & Schuh 1999)

Eversheim, W. & Schuh, G. (1999), *Produktion und Management 3,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-540-65453-7.

(Gabel 2009)

Gabel, T. (2009), *Learning in Cooperative Multi-Agent Systems*. *Distributed Reinforcement Learning Algorithms and their Application to Scheduling Problems*. Dissertation, Universität Osnabrück, Osnabrück.

(Gabel & Riedmiller 2012)

Gabel, T. & Riedmiller, M. (2012), „Distributed policy search reinforcement learning for job-shop scheduling tasks“, *International Journal of Production Research*, 50(1), S. 41–61.

(Ganschar & Gerlach et al. 2013)

Ganschar, O.; Gerlach, S.; Hämmerle, M.; Krause, T. & Schlund, S. (2013), *Produktionsarbeit der Zukunft - Industrie 4.0,* Fraunhofer Verlag, Stuttgart. ISBN: 978-3-8396-0570-7.

(Gao & Gen et al. 2007)

Gao, J.; Gen, M.; Sun, L. & Zhao, X. (2007), „A hybrid of genetic algorithm and bottleneck shifting for multiobjective flexible job shop scheduling problems“, *Computers & Industrial Engineering*, 53(1), S. 149–162.

(Garey & Johnson et al. 1976)

Garey, M. R.; Johnson, D. S. & Sethi, R. (1976), „The Complexity of Flowshop and Jobshop Scheduling“, *Mathematics of Operations Research*, 1(2), S. 117–129.

(Gören & Sabuncuoglu 2008)

Gören, S. & Sabuncuoglu, I. (2008), „Robustness and stability measures for scheduling. Single-machine environment“, *IIE Transactions*, 40(1), S. 66–83.

(Graham & Lawler et al. 1979)

Graham, R. L.; Lawler, E. L.; Lenstra, J. K. & Kan, A.H.G.R. (1979), „Optimization and Approximation in Deterministic Sequencing and Scheduling. A Survey“, *Annals of Discrete Mathematics,* Vol. 5, S. 287–326.

(Greensmith & Bartlett et al. 2004)

Greensmith, E.; Bartlett, P. L. & Baxter, J. (2004), „Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning“, *Journal of Machine Learning Research*, 5(11), S. 1471–1530.

(Greschke 2016)

Greschke, P. I. (2016), *Matrix-Produktion als Konzept einer taktunabhängigen Fließfertigung*. Dissertation, Technische Universität Braunschweig, Braunschweig.

(Grinninger 2012)

Grinninger, J. (2012), *Schlanke Produktionssteuerung zur Stabilisierung von Auftragsfolgen in der Automobilproduktion*. Dissertation, Technische Universität München, München, fml – Lehrstuhl für Fördertechnik Materialfluss Logistik.

(Günther & Tempelmeier 2005)

Günther, H.-O. & Tempelmeier, H. (2005), *Produktion und Logistik,* Springer, Berlin. ISBN: 978-3-540-23246-9.

(Gutenberg 1955)

Gutenberg, E. (1955), *Grundlagen der Betriebswirtschaftslehre,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-642-53039-5.

(Hackstein 1989)

Hackstein, R. (1989), *Produktionsplanung und -steuerung (PPS)*. *Ein Handbuch für die Betriebspraxis,* VDI-Verl., Düsseldorf. ISBN: 978-3-184-00924-3.

(Haupt 1989)

Haupt, R. (1989), „A survey of priority rule-based scheduling“, *OR Spektrum*, 11(1), S. 3–16.

(Hazır & Haouari et al. 2010)

Hazır, Ö.; Haouari, M. & Erel, E. (2010), „Robust scheduling and robustness measures for the discrete time/cost trade-off problem“, *European Journal of Operational Research*, 207(2), S. 633–643.

(Herrmann & Lee et al. 1993)

Herrmann, J. W.; Lee, C.-Y. & Snowdon, J. L. (1993), „A Classification of Static Scheduling Problems“ in *Complexity in numerical optimization,* Hrsg. P. M. Pardalos, World Scientific Pub. Co, Singapore, River Edge, N.J, S. 203–253.

(Herroelen & Leus 2004)

Herroelen, W. & Leus, R. (2004), „Robust and reactive project scheduling. A review and classification of procedures“, *International Journal of Production Research*, 42(8), S. 1599–1620.

(IBM 2017)

IBM (2017), *IBM ILOG CPLEX Optimization Studio OPL Language Reference Manual*. https://www.ibm.com/support/knowledgecenter/SSSA5P_12.7.0/ilog.odms.studio.help/pdf/opl_languser.pdf [07.12.2019].

(Jamili 2016)

Jamili, A. (2016), „Robust job shop scheduling problem. Mathematical models, exact and heuristic algorithms“, *Expert Systems with Applications*, 27(55), S. 341–350.

(Jensen 2003)

Jensen, M. T. (2003), „Generating robust and flexible job shop schedules using genetic algorithms“, *IEEE Transactions on Evolutionary Computation*, 7(3), S. 275–288.

(Jorge Leon & David Wu et al. 1994)

Jorge Leon, V.; David Wu, S. & Storer, R. H. (1994), „Robustness measures and robust scheduling for job shops“, *IIE Transactions*, 26(5), S. 32–43.

(Kadipasaoglu & Sridharan 1995)

Kadipasaoglu, S. N. & Sridharan, V. (1995), „Alternative approaches for reducing schedule instability in multistage manufacturing under demand uncertainty“, *Journal of Operations Management*, 13(3), S. 193–211.

(Kaelbling & Littman et al. 1996)

Kaelbling, L. P.; Littman, M. L. & Moore, A. W. (1996), „Reinforcement Learning:. A Survey“, *Journal of Artificial Intelligence Research*, 5(4), S. 237–285.

(Kaluza & Blecker et al. 2005)

Kaluza, B.; Blecker, T. & Behrens, S. (Hrsg.) (2005), *Erfolgsfaktor Flexibilität*. *Strategien und Konzepte für wandlungsfähige Unternehmen,* Schmidt, Berlin. ISBN: 978-3-503-08367-1.

(Karle 2018)

Karle, A. (2018), *Elektromobilität*. *Grundlagen und Praxis,* Fachbuchverlag Leipzig im Carl Hanser Verlag, München. ISBN: 978-3-446-45657-0.

(Kern & Rusitschka et al. 2015)

Kern, W.; Rusitschka, F.; Kopytynski, W.; Keckl, S. & Bauernhansl, T. (2015), „Alternatives to Assembly Line Production in the Automotive Industry“. *23rd International Conference on Production Research (ICPR)*, 01. - 04. August 2015, Manila, Philippinen, 9 S.

(Koren 2010)

Koren, Y. (2010), *The global manufacturing revolution*. *Product-process-business integration and reconfigurable systems,* Wiley, Hoboken, N.J. ISBN: 978-0-470-58377-7.

(Kouvelis & Yu 1997)

Kouvelis, P. & Yu, G. (1997), *Robust Discrete Optimization and Its Applications,* Springer, Boston, MA. ISBN: 978-1-441-94764-2.

(Kraftfahrtbundesamt 2019)

Kraftfahrtbundesamt (2019), *Veränderungen der Neuzulassungen von Personenkraftwagen von Januar bis Juli 2019 nach Marken und Modellreihen in Prozent*.

https://www.kba.de/DE/Statistik/Fahrzeuge/Neuzulassungen/MonatlicheNeuzulassungen/2019/201907_GImonatlich/201907_n_top50.html?nn=2162804 [14.08.2019].

(KUKA 2016)
KUKA (2016), *Matrix-Produktion*. *Ein Beispiel für Industrie 4.0*. https://www.kuka.com/de-de/branchen/loesungsdatenbank/2016/10/solution-systems-matrix-produktion [14.08.2019].

(Küpper & Lüder et al. 2013)
Küpper, W.; Lüder, K. & Streitferdt, L. (2013), *Netzplantechnik,* Physica-Verlag, Heidelberg, ISBN: 978-3-662-12568-7.

(Kurbel 2003)
Kurbel, K. (2003), *Produktionsplanung und -steuerung*. *Methodische Grundlagen von PPS-Systemen und Erweiterungen,* Oldenbourg, München. ISBN: 978-3-486-27299-4.

(Lan & Clarke et al. 2006)
Lan, S.; Clarke, J.-P. & Barnhart, C. (2006), „Planning for Robust Airline Operations. Optimizing Aircraft Routings and Flight Departure Times to Minimize Passenger Disruptions“, *Transportation Science*, 40(1), S. 15–28.

(Lanza & Nyhuis et al. 2017)
Lanza, G.; Nyhuis, P.; Fisel, J.; Jacob, A.; Nielsen, L.; Schmidt, M. & Stricker, N. (2017), *Wandlungsfähige, menschzentrierte Strukturen in Fabriken und Netzwerken der Industrie 4.0 (acatech Studie),* Herbert Utz Verlag, München.

(Larsen & Pranzo 2018)
Larsen, R. & Pranzo, M. (2018), „A framework for dynamic rescheduling problems“, *International Journal of Production Research*, 57(1), S. 16–33.

(Li & Wang 2009)
Li, Q. & Wang, B. (2009), „A stable scheduling for single machine under uncertainty“. *ICAL 2009 – IEEE International Conference on Automation,* 05. - 07. August 2009, Shenyang, China*,* S. 526–531.

(Liu & Kozan 2017)
Liu, S. Q. & Kozan, E. (2017), „A hybrid shifting bottleneck procedure algorithm for

the parallel-machine job-shop scheduling problem“, *Journal of the Operational Research Society*, 63(2), S. 168–182.

(Lödding 2016)

Lödding, H. (2016), *Verfahren der Fertigungssteuerung*. *Grundlagen, Beschreibung, Konfiguration,* Springer Vieweg, Berlin and Heidelberg. ISBN: 978-3-662-48458-6.

(Mahajan 2007)

Mahajan, K. R. (2007), *A combined simulation and optimization based method for predictive - reactive scheduling of flexible production systems subject to execution exceptions*. Dissertation, Universität Paderborn, Paderborn.

(Manne 1960)

Manne, A. S. (1960), „On the Job-Shop Scheduling Problem“, *Operations Research*, 8(2), S. 219–223.

(Martinez 2012)

Martinez, Y. (2012), *A Generic Multi-Agent Reinforcement Learning Approach for Scheduling Problems*. Dissertation, Vrije Universiteit Brussel, Brüssel.

(Mehta & Uzsoy 1999)

Mehta, S. V. & Uzsoy, R. M. (1999), „Predictable scheduling of a single machine subject to breakdowns“, *International Journal of Computer Integrated Manufacturing*, 12(1), S. 15–38.

(Meissner 2009)

Meissner, S. (2009), *Logistische Stabilität in der automobilen Variantenfließfertigung*. Dissertation, Technische Universität München, Garching.

(Melzer-Ridinger 2018)

Melzer-Ridinger, R. (2018), *PPS: Systemgestützte Produktionsplanung: Konzeption und Anwendung,* De Gruyter, Oldenbourg. ISBN: 978-3-486-22998-1.

(Mnih & Kavukcuoglu et al. 2015)

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S. & Hassabis, D. (2015), „Human-level control through deep reinforcement learning“, *Nature*, 518(7540), S. 529–533.

(Mönch & Drießel 2005)

Mönch, L. & Drießel, R. (2005), „A distributed shifting bottleneck heuristic for complex job shops“, *Computers & Industrial Engineering*, 49(3), S. 363–380.

(Moratori & Petrovic et al. 2011)

Moratori, P.; Petrovic, S. & Vázquez-Rodríguez, J. A. (2011), „Match-up approaches to a dynamic rescheduling problem“, *International Journal of Production Research*, 50(1), S. 261–276.

(Mulvey & Vanderbei et al. 1995)

Mulvey, J. M.; Vanderbei, R. J. & Zenios, S. A. (1995), „Robust Optimization of Large-Scale Systems“, *Operations Research*, 43(2), S. 264–281.

(Neuhaus 2008)

Neuhaus, U. (2008), *Reaktive Planung in der chemischen Industrie*. *Verfahren zur operativen Plananpassung für Mehrzweckanlagen*. Dissertation, Technische Universität Berlin, Berlin.

(Nickel & Stein et al. 2014)

Nickel, S.; Stein, O. & Waldmann, K.-H. (2014), *Operations Research,* Springer Gabler, Berlin, Heidelberg. ISBN: 978-3-642-54367-8.

(Niehues 2016)

Niehues, M. R. (2016), *Adaptive Produktionssteuerung für Werkstattfertigungssysteme durch fertigungsbegleitende Reihenfolgebildung*. Dissertation, Techn. Univ und Herbert Utz Verlag GmbH, München.

(Nyhuis & Reinhart et al. 2008)

Nyhuis, P.; Reinhart, G. & Abele, E. (2008), *Wandlungsfähige Produktionssysteme*. *Heute die Industrie von morgen gestalten,* TEWISS, Garbsen, ISBN: 978-3-939-02696-9.

(Oliehoek & Amato 2016)

Oliehoek, F. A. & Amato, C. (2016), *A Concise Introduction to Decentralized POMDPs,* Springer International Publishing, Cham. ISBN: 978-3-319-28927-4.

(Ouelhadj & Petrovic 2009)

Ouelhadj, D. & Petrovic, S. (2009), „A survey of dynamic scheduling in manufacturing systems“, *Journal of Scheduling*, 12(4), S. 417–431.

(Panwalkar & Iskander 1977)

Panwalkar, S. S. & Iskander, W. (1977), „A Survey of Scheduling Rules“, *Operations Research*, 25(1), S. 45–61.

(Patig & Thorhauer 2002)

Patig, S. & Thorhauer, S. (2002), „Ein Planungsansatz zum Umgang mit Störungen bei der Produktion. Die flexible Produktionsfeinplanung mithilfe von Planungsschritten“, *WIRTSCHAFTSINFORMATIK*, 44(4), S. 355–366.

(Peshkin & Kim et al. 2000)

Peshkin, L.; Kim, K.-E.; Meuleau, N. & Kaelbling, L. P. (2000), „Learning to Cooperate via Policy Search“. *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence,* 30. Juni - 03. Juli 2000, Stanford, USA, Hrsg. C. Boutilier & M. Goldszmidt, Morgan Kaufmann Publishers Inc., San Francisco, CA, S. 489–496. ISBN: 978-1-55860-709-9.

(Petrovic & Fayad et al. 2008)

Petrovic, S.; Fayad, C.; Petrovic, D.; Burke, E. & Kendall, G. (2008), „Fuzzy job shop scheduling with lot-sizing“, *Annals of Operations Research*, 159(1), S. 275–292.

(Pinedo 2016)

Pinedo, M. L. (2016), *Scheduling*. *Theory, Algorithms, and Systems,* Springer International Publishing and Imprint: Springer, Cham. ISBN: 978-3-319-26578-0.

(Policella & Cesta et al. 2007)

Policella, N.; Cesta, A.; Oddi, A. & Smith, S. F. (2007), „From precedence constraint posting to partial order schedules“, *AI Communications*, 20(3), S. 163–180.

(Puterman 1994)

Puterman, M. L. (1994), *Markov decision processes*. *Discrete stochastic dynamic programming,* Wiley, New York. ISBN: 978-0-471-61977-2.

(Qu & Wang et al. 2016)

Qu, S.; Wang, J.; Govil, S. & Leckie, J. O. (2016), „Optimized Adaptive Scheduling of a Manufacturing Process System with Multi-skill Workforce and Multiple Machine Types. An Ontology-based, Multi-agent Reinforcement Learning Approach“. *Factories of the Future in the digital environment - Proceedings of the 49th CIRP*

Conference on Manufacturing Systems, 25. - 27. Mai 2016, Stuttgart, S. 55–60. ISBN: 978-1-510-83493-4.

(Rajendran & Holthaus 1999)
Rajendran, C. & Holthaus, O. (1999), „A comparative study of dispatching rules in dynamic flowshops and jobshops“, *European Journal of Operational Research*, 116(1), S. 156–170.

(Rossi & van Beek et al. 2006)
Rossi, F.; van Beek, P. & Walsh, T. (Hrsg.) (2006), *Handbook of constraint programming,* Elsevier, Amsterdam. ISBN: 978-0-444-52726-4.

(Russell & Norvig et al. 2016)
Russell, S. J.; Norvig, P. & Davis, E. (2016), *Artificial intelligence*. *A modern approach,* Pearson and MyiLibrary, Boston and La Vergne TN. ISBN: 978-1-292-15397-1.

(Sabuncuoglu & Karabuk 1999)
Sabuncuoglu, I. & Karabuk, S. (1999), „Rescheduling frequency in an FMS with uncertain processing times and unreliable machines“, *Journal of Manufacturing Systems*, 18(4), S. 268–283.

(Sabuncuoglu & Bayız 2000)
Sabuncuoglu, I. & Bayız, M. (2000), „Analysis of reactive scheduling problems in a job shop environment“, *European Journal of Operational Research*, 126(3), S. 567–586.

(Sabuncuoglu & Gören 2009)
Sabuncuoglu, I. & Gören, S. (2009), „Hedging production schedules against uncertainty in manufacturing environment with a review of robustness and stability research“, *International Journal of Computer Integrated Manufacturing*, 22(2), S. 138–157.

(Salido & Barber et al. 2008)
Salido, M. A.; Barber, F. & Ingolotti, L. (2008), „Robustness in railway transportation scheduling“. *IEEE 7th World Congress on Intelligent Control and Automation,* S. 2880–2885.

(Sarin & Varadarajan et al. 2010)
Sarin, S. C.; Varadarajan, A. & Wang, L. (2010), „A survey of dispatching rules for

operational control in wafer fabrication“, *Production Planning & Control*, 22(1), S. 4–24.

(Scholl 2000)

Scholl, A. (2000), *Robuste Planung und Optimierung*. *Grundlagen - Konzepte und Methoden - experimentelle Untersuchungen*. Habilitationsschrift, Technische Universität Darmstadt, Heidelberg.

(Schomburg 1980)

Schomburg, E. (1980), *Entwicklung eines betriebstypologischen Instrumentariums zur systematischen Ermittlung der Anforderungen an EDV-gestützte Produktionsplanungs-und-steuerungssysteme im Maschinenbau*. Dissertation, RWTH Aachen, Aachen.

(Schönemann & Herrmann et al. 2015)

Schönemann, M.; Herrmann, C.; Greschke, P. & Thiede, S. (2015), „Simulation of matrix-structured manufacturing systems“, *Journal of Manufacturing Systems*, 37(57), S. 104–112.

(Schuh 2007)

Schuh, G. (Hrsg.) (2007), *Produktionsplanung und -steuerung*. *Grundlagen, Gestaltung Und Konzepte,* Springer, Dordrecht. ISBN: 978-3-540-40306-7.

(Schulte 1995)

Schulte, J. (1995), *Werkstattsteuerung mit genetischen Algorithmen und simulativer Bewertung*, Dissertation, Universität Stuttgart, Stuttgart.

(Shen & Han et al. 2017)

Shen, X.-N.; Han, Y. & Fu, J.-Z. (2017), „Robustness measures and robust scheduling for multi-objective stochastic flexible job shop scheduling problems“, *Soft Computing*, 21(21), S. 6531–6554.

(Statista 2019)

Statista (2019), *Anzahl der Neuzulassungen von Personenkraftwagen mit Dieselmotor in Deutschland von Juli 2017 bis Juli 2019*. https://de.statista.com/statistik/daten/studie/468652/umfrage/monatliche-pkw-neuzulassungen-in-deutschland-dieselmotor/ [19.08.2019].

(Statistisches Bundesamt 2018)

Statistisches Bundesamt (2018), *Statistisches Jahrbuch Deutschland 2018,* Statistisches Bundesamt, Wiesbaden. ISBN: 978-3-8246-1074-7.

(STAUFEN 2018)

STAUFEN (2018), *Deutscher Industrie 4.0 Index 2018*. https://www.staufen.ag/fileadmin/HQ/02-Company/05-Media/2-Studies/STAUFEN.-Studie-Industrie-4.0-Index-2018-Web-DE-de.pdf [14.08.2019].

(Stricker 2016)

Stricker, N. (2016), *Robustheit verketteter Produktionssysteme*. *Robustheitsevaluation und Selektion des Kennzahlensystems der Robustheit*. Dissertation, Karlsruher Institut für Technologie, Karlsruhe.

(Stricker & Lanza 2014)

Stricker, N. & Lanza, G. (2014), „The Concept of Robustness in Production Systems and its Correlation to Disturbances“. *2nd CIRP Robust Manufacturing Conference (RoMac 2014),* 07. - 09. Juli 2014, Bremen*,* Hrsg. K. Windt, S. 87–92. ISBN: 978-1-634-39362-1.

(Suresh & Chaudhuri 1993)

Suresh, V. & Chaudhuri, D. (1993), „Dynamic scheduling—a survey of research“, *International Journal of Production Economics*, 32(1), S. 53–63.

(Sutton & McAllester et al. 2000)

Sutton, R. S.; McAllester, D.; Singh, S. & Mansour, Y. (2000), „Policy Gradient Methods for Reinforcement Learning with Function Approximation“, *Advances in Neural Information Processing Systems*(12), S. 1057–1063.

(Sutton & Barto 2018)

Sutton, R. S. & Barto, A. (2018), *Reinforcement learning*. *An introduction,* The MIT Press, Cambridge, Massaschusetts and London. ISBN: 978-0-262-19398-6.

(Szepesvári 2010)

Szepesvári, C. (2010), *Algorithms for Reinforcement Learning,* Morgan & Claypool Publishers, San Rafael, California. ISBN: 978-1-608-45492-1.

(Thrun 1992)

Thrun, S. B. (1992), *Efficient Exploration In Reinforcement Learning*., Carnegie Mellon University Pittsburgh, PA, USA, Tech. Rep. CMU-CS-92-102.

(Tolio & Urgo et al. 2011)

Tolio, T.; Urgo, M. & Váncza, J. (2011), „Robust production control against propagation of disruptions“, *CIRP Annals*, 60(1), S. 489–492.

(Urgo & Váncza 2014)

Urgo, M. & Váncza, J. (2014), „A Robust Scheduling Approach for a Single Machine to Optimize a Risk Measure“. *2nd CIRP Robust Manufacturing Conference (RoMac 2014),* 07. - 09. Juli 2014, Bremen*,* Hrsg. K. Windt, S. 148–153. ISBN: 978-1-634-39362-1.

(van Brackel 2008)

van Brackel, T. (2008), *Adaptive Steuerung flexibler Werkstattfertigungssysteme*. *Nutzung moderner Informations- und Kommunikationstechnologien zur effizienten Produktionssteuerung unter Echtzeitbedingungen*. Disseration, Universität Paderborn, Paderborn.

(Van de Vonder, S. & Demeulemeester, E. et al. 2008)

Van de Vonder, S.; Demeulemeester, E. & Herroelen, W. (2008), „Proactive heuristic procedures for robust project scheduling. An experimental analysis“, *European Journal of Operational Research*, 189(3), S. 723–733.

(van de Vonder 2006)

van de Vonder, S. (2006), *Proactive-reactive procedures for robust project scheduling*. Dissertation, Katholieke Universiteit Leuven, Leuven.

(van Hoeve & Laborie 2018)

van Hoeve, W.-J. & Laborie, P. (2018), „An Update on the Comparison of MIP, CP and Hybrid Approaches for Mixed Resource Allocation and Scheduling. Integration of Constraint Programming, Artificial Intelligence, and Operations Research“. *International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR 2018),* 26. - 29. Juni 2018, Delft, Niederlande, Hrsg. W.-J. van Hoeve, S. 403–411.

(Váncza & Monostori et al. 2011)

Váncza, J.; Monostori, L.; Lutters, D.; Kumara, S. R.; Tseng, M.; Valckenaers, P. & van Brussel, H. (2011), „Cooperative and responsive manufacturing enterprises“, *CIRP Annals*, 60(2), S. 797–820.

(Vieira & Herrmann et al. 2003)

Vieira; Herrmann & Lin (2003), „Rescheduling Manufacturing Systems“, *Journal of Scheduling*, 6(1), S. 39–62.

(Volk 2017)

Volk, R. (2017), *Proactive-reactive, robust scheduling and capacity planning of deconstruction projects under uncertainty*. Disseration, Karlsruher Institut für Technologie, Karlsruhe.

(von Heynitz & Bremicker et al. 2016)

von Heynitz, H.; Bremicker, M.; Amadori, D. M. & Reschke, K. (2016), *Fabrik der Zukunft*. *Industrie 4.0 - die Herausforderungen von morgen*. https://assets.kpmg/content/dam/kpmg/pdf/2016/02/broschuere-industrie-4-0.pdf [07.12.2019].

(Wallentowitz & Freialdenhoven et al. 2009)

Wallentowitz, H.; Freialdenhoven, A. & Olschewski, I. (2009), *Strategien in der Automobilindustrie*. *Technologietrends und Marktentwicklungen,* Vieweg+Teubner Verlag, Wiesbaden. ISBN: 978-3-834-80725-0.

(Wannenwetsch 2014)

Wannenwetsch, H. (2014), *Integrierte Materialwirtschaft, Logistik und Beschaffung,* Springer Berlin Heidelberg, Berlin, Heidelberg. ISBN: 978-3-642-45022-8.

(Waschneck & Reichstaller et al. 2018)

Waschneck, B.; Reichstaller, A.; Belzner, L.; Altenmüller, T.; Bauernhansl, T.; Knapp, A. & Kyekb, A. (2018), „Optimization of global production scheduling with deep reinforcement learning“. *51st CIRP Conference on Manufacturing Systems,* 16. - 18. Mai 2018, Stockholm, Schweden, Hrsg. L. Wang, S. 1264–1269.

(Watkins & Dayan 1992)

Watkins, C. J. C. H. & Dayan, P. (1992), „Q-learning“, *Machine Learning*(8), S. 279–292.

(Weber & Kabst et al. 2018)

Weber, W.; Kabst, R. & Baum, M. (2018), *Einführung in die Betriebswirtschaftslehre,* Springer Fachmedien Wiesbaden, Wiesbaden. ISBN: 978-3-658-18251-9.

(Wemhöner et al. 2006)
Wemhöner, N. & Schuh, G. (2006), *Flexibilitätsoptimierung zur Auslastungssteigerung im Automobilrohbau*. Dissertation, RWTH Aachen, Aachen.

(Wiendahl 1997)
Wiendahl, H.-P. (1997), *Fertigungsregelung*. *Logistische Beherrschung von Fertigungsabläufen auf Basis des Trichtermodells,* Hanser, München. ISBN: 978-3-446-19084-9.

(Wiendahl 2009)
Wiendahl, H.-P. (2009), *Betriebsorganisation für Ingenieure,* Carl Hanser Verlag GmbH & Co. KG, München. ISBN: 978-3-446-41878-3.

(Winter 2014)
Winter, E. (Hrsg.) (2014), *Gabler Wirtschaftslexikon,* Springer Gabler, Wiesbaden. ISBN: 978-3-834-93464-2.

(Wissenschaftlicher Dienst des Deutschen Bundestags 2016)
Wissenschaftlicher Dienst des Deutschen Bundestags (2016), *Zur Diskussion um die Verkürzung von Produktlebenszyklen*. https://www.bundestag.de/resource/blob/438002/42b9bf2ae2369fd4b8dd119d968a1380/wd-5-053-16-pdf-data.pdf [14.08.2019].

(Wu & Byeon et al. 1999)
Wu, S. D.; Byeon, E.-S. & Storer, R. H. (1999), „A Graph-Theoretic Decomposition of the Job Shop Scheduling Problem to Achieve Scheduling Robustness“, *Operations Research*, 47(1), S. 113–124.

(Xiong & Xing et al. 2013)
Xiong, J.; Xing, L.-n. & Chen, Y.-w. (2013), „Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns“, *International Journal of Production Economics*, 141(1), S. 112–126.

(Yen & Birge 2006)
Yen, J. W. & Birge, J. R. (2006), „A Stochastic Programming Approach to the Airline Crew Scheduling Problem“, *Transportation Science*, 40(1), S. 3–14.

(Yuan 2006)
Yuan, J. (2006), *Stochastic modelling of train delays and delay propagation in stations*. Dissertation, TU Delft, Delft.

Publikationsliste

(Greinacher & Echsler Minguillon et al. 2016)

Greinacher, S.; Echsler Minguillon, F.; Häfner, B.; Stricker, N. & Lanza, G. (2016), „Skalierbare Automatisierung und Industrie 4.0“, wt Werkstattstechnik online, 106(9), S. 659-665.

(Greinacher & Echsler Minguillon et al. 2017)

Greinacher, S.; Echsler Minguillon, F. & Lanza, G. (2017), „Montagesysteme: Skalierbare Automatisierung in der Lernfabrik Globale Produktion“ in *Handbuch Industrie 4.0: Geschäftsmodelle, Prozesse, Technik,* Hrsg. G. Reinhart, Carl Hanser Verlag, München, S. 605–620.

(Bürgin & Echsler Minguillon et al. 2017)

Bürgin, J.; Echsler Minguillon, F.; Wehrle, F.; Häfner, B. & Lanza, G. (2017), „Demonstration of a Concept for Scalable Automation of Assembly Systems in a Learning Factory“. 7th Conference on Learning Factories (CLF 2017), 04. - 05. April 2017, Darmstadt, Hrsg. Procedia Manufacturing, S. 111-118.

(Stricker & Echsler Minguillon et al. 2017)

Stricker, N.; Echsler Minguillon, F. & Lanza, G. (2017), „Selecting key performance indicators for production with a linear programming approach“, International Journal of Production Research, 55(19), S. 5537-5549.

(Echsler Minguillon & Lanza 2017)

Echsler Minguillon, F. & Lanza, G. (2017), „Maschinelles Lernen in der PPS“, wt Werkstattstechnik online, 107(9), S. 630-634.

(Echsler Minguillon & Lanza 2018)

Echsler Minguillon, F. & Lanza, G. (2018), „Coupling of centralized and decentralized scheduling for robust production in agile production systems“. 12th CIRP Conference on Intelligent Computation in Manufacturing Engineering (ICME 2018), 18. - 20. Juli 2018, Golf von Neapel, Italien, Hrsg. Roberto Teti, S. 385-390.

(Echsler Minguillon & Schömer et al. 2019)

Echsler Minguillon, F.; Schömer, J.; Stricker, N.; Lanza, G.; Duffie, N. (2019), „Planning for changeability and flexibility using a frequency perspective”. CIRP Annals - Manufacturing Technology, Volume 68, Issue 1, S. 427-430

(Echsler Minguillon & Stricker 2020)

Echsler Minguillon, F.; Stricker, N. (2020), „Robust predictive-reactive scheduling and its effect on machine disturbance mitigation”. CIRP Annals – Manufacturing Technology [im Druck]

Abbildungsverzeichnis

Tabellenverzeichnis

Anhang

A1 Ermittlung stationsspezifischer Anforderungszeiten

Die stationsspezifischen Anforderungszeiten wurden simulationsbasiert ermittelt. Dazu wurden für einen vorgegebenen Variantenmix zufällige Aufträge im Abstand von 90s erzeugt und die Wartezeiten von Aufträgen im Puffer bis zum Weitertransport je Auftrag ermittelt. Jedes der zehn durchgeführten Simulationsexperimente wurde mit einem randomisierten Seed-Wert gestartet und brach nach 60.000 absolvierten Transportaufträgen ab. Es ergibt sich damit je Station eine kumulierte Häufigkeitsverteilung wie in Abbildung A-1 exemplarisch für die Station FLS 1 dargestellt.

Um den Einfluss der dynamischen FTF-Steuerung auf die Untersuchung der Methode zum prädiktiv-reaktiven Scheduling zu minimieren, wurde jeweils das 95 %-Quantil der Häufigkeitsverteilungen zur Ermittlung der rt_s herangezogen. Damit wird sichergestellt, dass trotz der dynamischen FTF-Steuerung 95 % der Anforderungszeiten kleiner oder gleich den rt_s sind und deren Störeinfluss minimal wird. Im Beispiel sind sowohl der Median $\tilde{x}$ als auch der Mittelwert $\bar{x}$ ungeeignet, da vergleichsweise viele Anforderungszeiten darüber liegen. Daher wird auf das 95 %-Quantil $x_{0,95} = 23{,}74\ s$ zurückgegriffen.

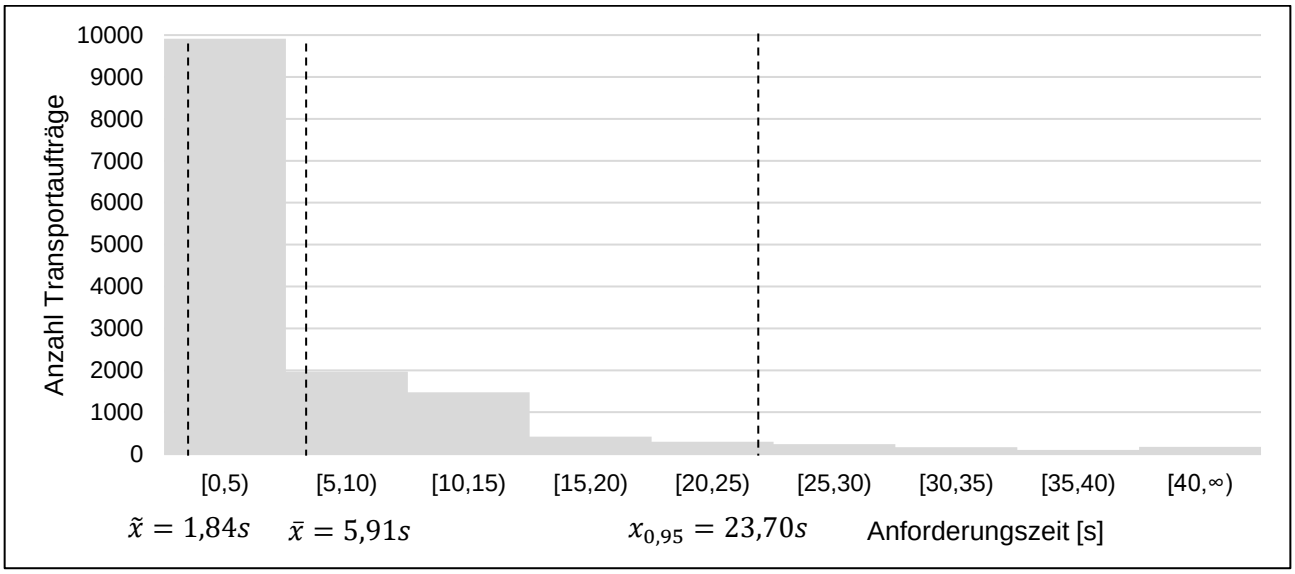


Abbildung A-1: Häufigkeitsverteilung der Anforderungszeiten an der Station FLS 1

Es ergeben sich mit dem Vorgehen die in Tabelle A-1 dargestellten Anforderungszeiten rt_s. Auffällig ist dabei die Bandbreite von ca. 19 bis 93 Sekunden. Dies ist auf die räumliche Anordnung der Stationen im Layout zurückzuführen: An Ausfügestationen fahren aufgrund ihrer zentralen Lage sehr viele FTF vorbei, wodurch die Wahrscheinlichkeit höher ist, dass sich zu einem beliebigen Zeitpunkt ein benötigtes leeres FTF in unmittelbarer Nähe befindet. Zu den Geostationen am Rande des Layouts hingegen fahren FTF in der Regel nur dann, wenn sie die spezifische Station anfahren.

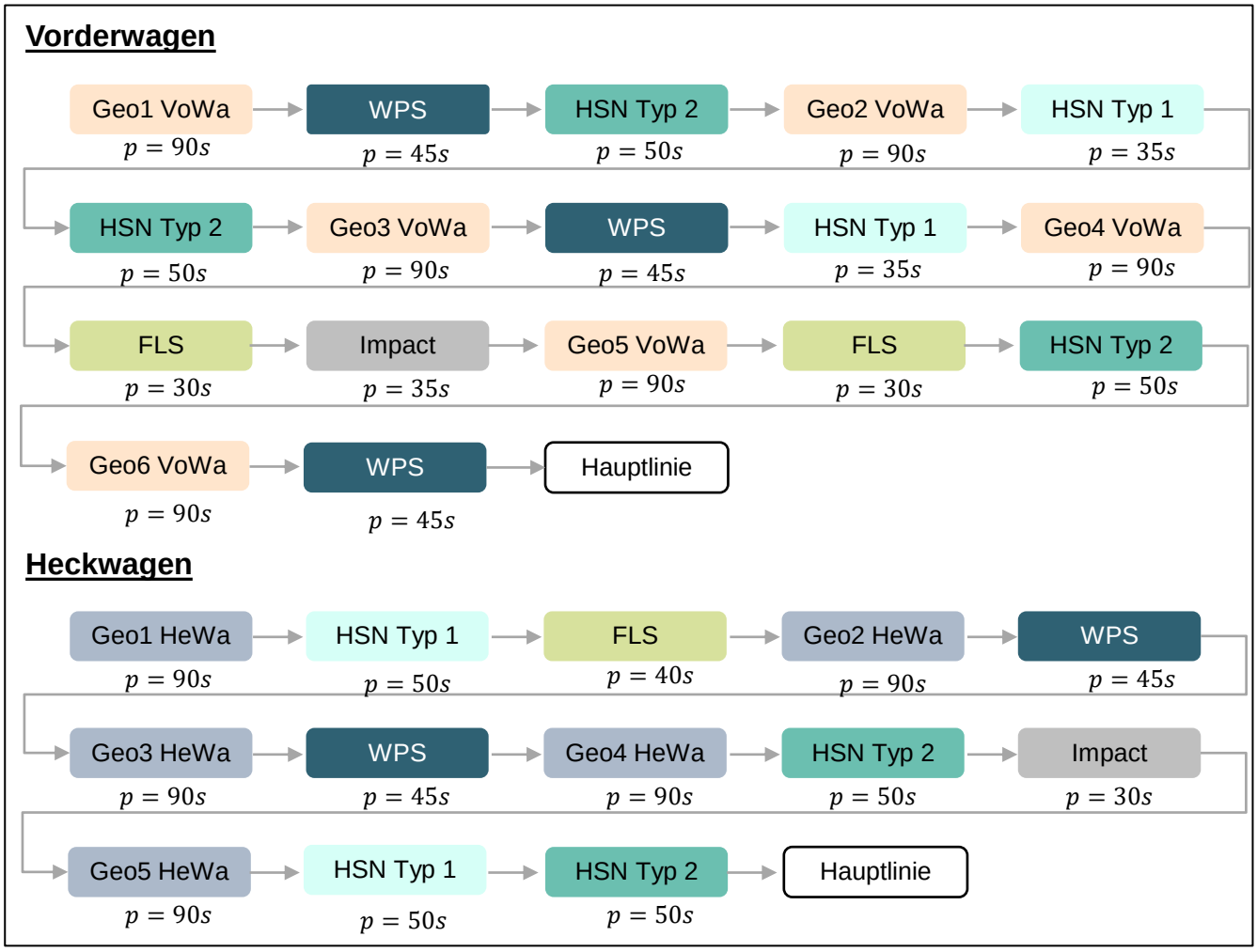
Tabelle A-1: Anforderungszeiten rt_s aller Stationen

Station	Anforderungszeit [s]
Geo1 HeWa	93,15
Geo2 HeWa	51,29
Geo3 HeWa	40,41
Geo4 HeWa	29,86
Geo5 HeWa	40,96
Geo1 VoWa	62,43
Geo2 VoWa	35,58
Geo3 VoWa	31,15
Geo4 VoWa	36,93
Geo5 VoWa	46,02
Geo6 VoWa	61,76
HSN Typ1 1	22,06
HSN Typ1 2	22,27
HSN Typ2 1	23,22
HSN Typ2 2	18,77
FLS 1	23,70
FLS 2	24,23
Impact	27,74
WPS 1	22,71
WPS 2	22,21
WPS 3	21,99
WPS 4	19,45

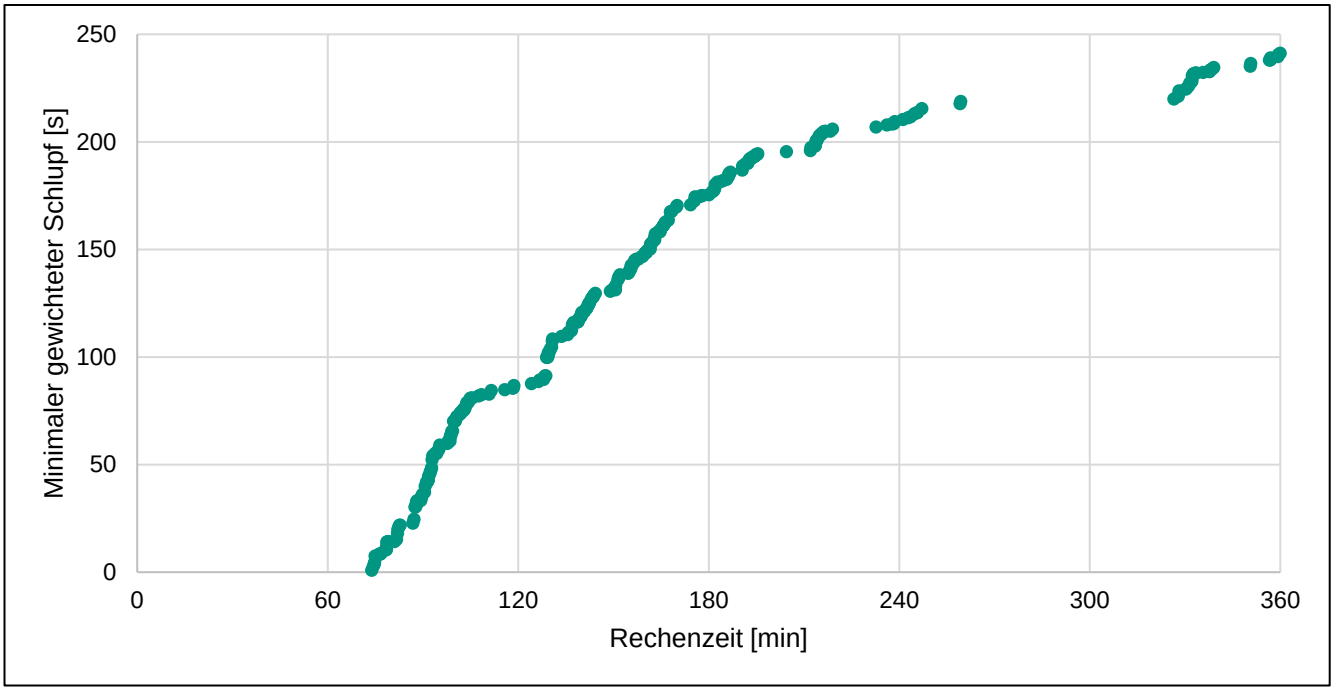
Die Simulation eines nicht robusten prädiktiven Schedules mit $R = 0$ unter Verwendung der in Tabelle A-1 dargestellten Werte hat eine mittlere absolute Verschiebung von Bearbeitungsschritten unterhalb 1s ergeben, sodass das Vorgehen als hinreichend genau für die vorliegende Arbeit betrachtet werden kann. Durch die Berücksichtigung der ermittelten Anforderungszeiten können dynamische Verzögerungen

aufgrund der Steuerung der stochastischen Anforderung von FTF kompensiert werden, sodass keine zusätzliche Quelle von Verzögerungen neben Stationsausfällen besteht.

A2 Exemplarische Fügefolgen für eine Vorder- (VoWa) und eine Heckwagenvariante (HeWa)



A3 Zielfunktionswert im prädiktiven Scheduling über die Zeit



A4 Kapazitätsangebot und -bedarf im Produktionsprogramm

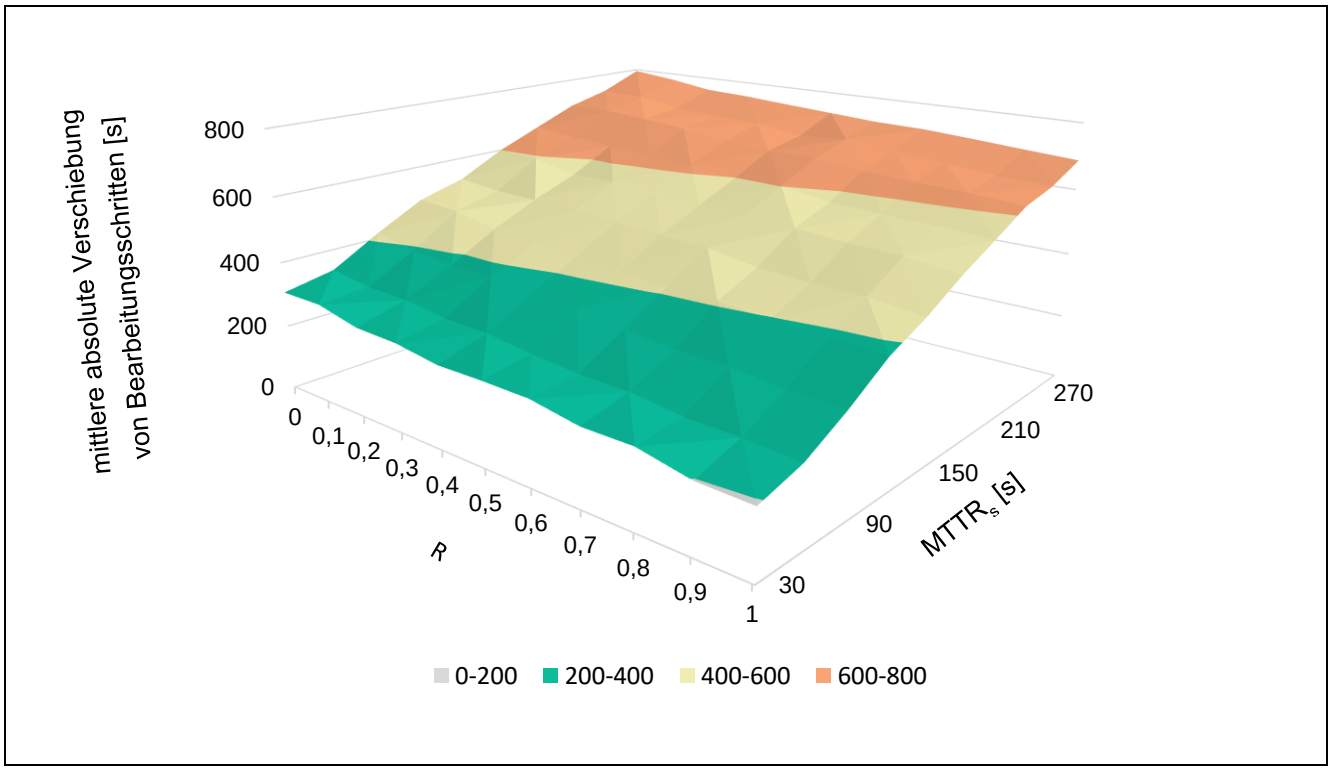
Variante				WPS	HSN Typ1	HSN Typ2	FLS	Impact	Geo
Antriebsstrang	**Lenkung**	**Laststufe**	**Kapazitätsangebot [s]**	127.056	62.488	62.488	62.895	30.927	32.310
V	LL	LS1	**Kapazitätsbedarf [s]**	48.600	22.680	18.630	19.440	10.530	14.580
V	LL	LS2		24.495	8.970	10.350	11.730	4.485	6.210
V	RL	LS1		13.530	9.020	5.330	6.150	2.665	3.690
V	RL	LS2		6.205	2.380	3.315	2.380	1.105	1.530
E	LL	LS1		1.470	1.435	1.505	805	420	630
E	LL	LS2		2.760	1.380	2.040	1.560	780	1.080
E	RL	LS1		1.620	920	600	320	260	360
E	RL	LS2		1.800	1.360	1.600	880	520	720
			Auslastung	79,08 %	77,05 %	69,41 %	68,79 %	67,14 %	89,14 %

A5 Ergebnisse der Simulationsstudie zur mittleren Verschiebung [s]

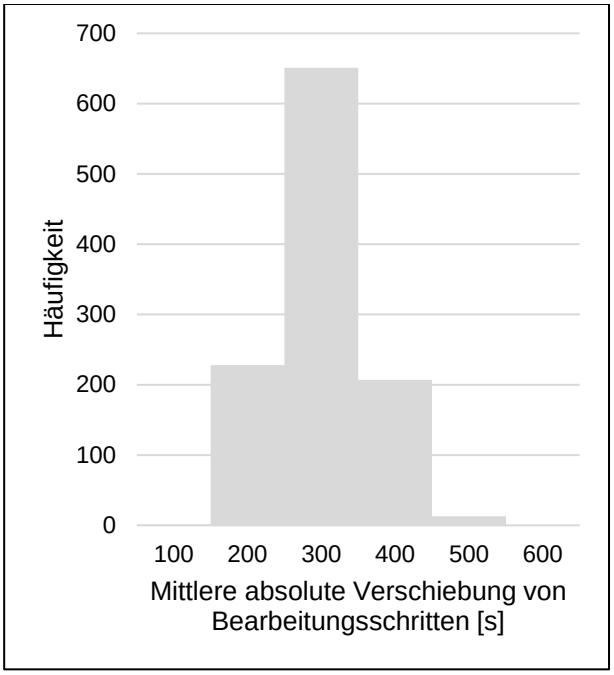
$MTTR_S$ \ R	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1	μ
30	306,7	306,4	275,8	271,8	251,9	253,6	251,7	227,7	228,7	198,2	195,0	251,6
60	335,5	317,2	313,4	296,2	293,0	275,4	262,6	260,7	243,3	240,6	231,6	279,1
90	417,4	392,4	377,7	378,2	362,4	355,9	336,2	323,2	320,4	305,1	301,0	351,8
120	490,9	481,7	452,9	443,9	430,0	422,0	413,2	407,6	397,2	387,1	381,8	428,0
150	530,1	516,3	514,0	509,4	500,2	490,4	485,1	461,4	453,6	451,7	435,4	486,1

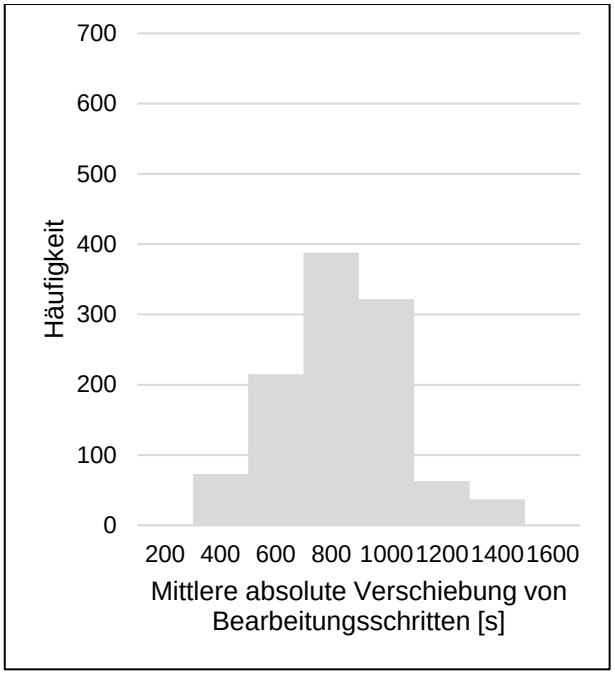
180	599,4	595,2	564,5	568,7	557,2	544,6	542,5	536,5	524,1	520,3	502,0	550,5
210	653,6	641,6	632,6	622,8	616,8	598,4	601,3	578,6	572,8	566,4	558,0	603,9
240	707,7	709,0	699,0	686,1	665,7	657,2	656,0	641,5	633,0	627,8	616,3	663,6
270	741,3	736,1	715,7	707,4	700,1	680,3	691,1	676,2	662,7	648,2	642,4	691,1
300	794,9	779,5	758,1	751,8	741,4	731,9	722,3	718,8	709,0	697,6	688,3	735,8
Mittelwert	557,8	547,5	530,4	523,6	511,9	501,0	496,2	483,2	474,5	464,3	455,2	

A6 Einfluss von $MTTR_s$ und R auf die mittlere absolute Verschiebung von Bearbeitungsschritten



A7 Häufigkeitsverteilung der mittleren absoluten Verschiebung für $MTTR_s = 30s$ (links) und $MTTR_s = 300s$ (rechts)

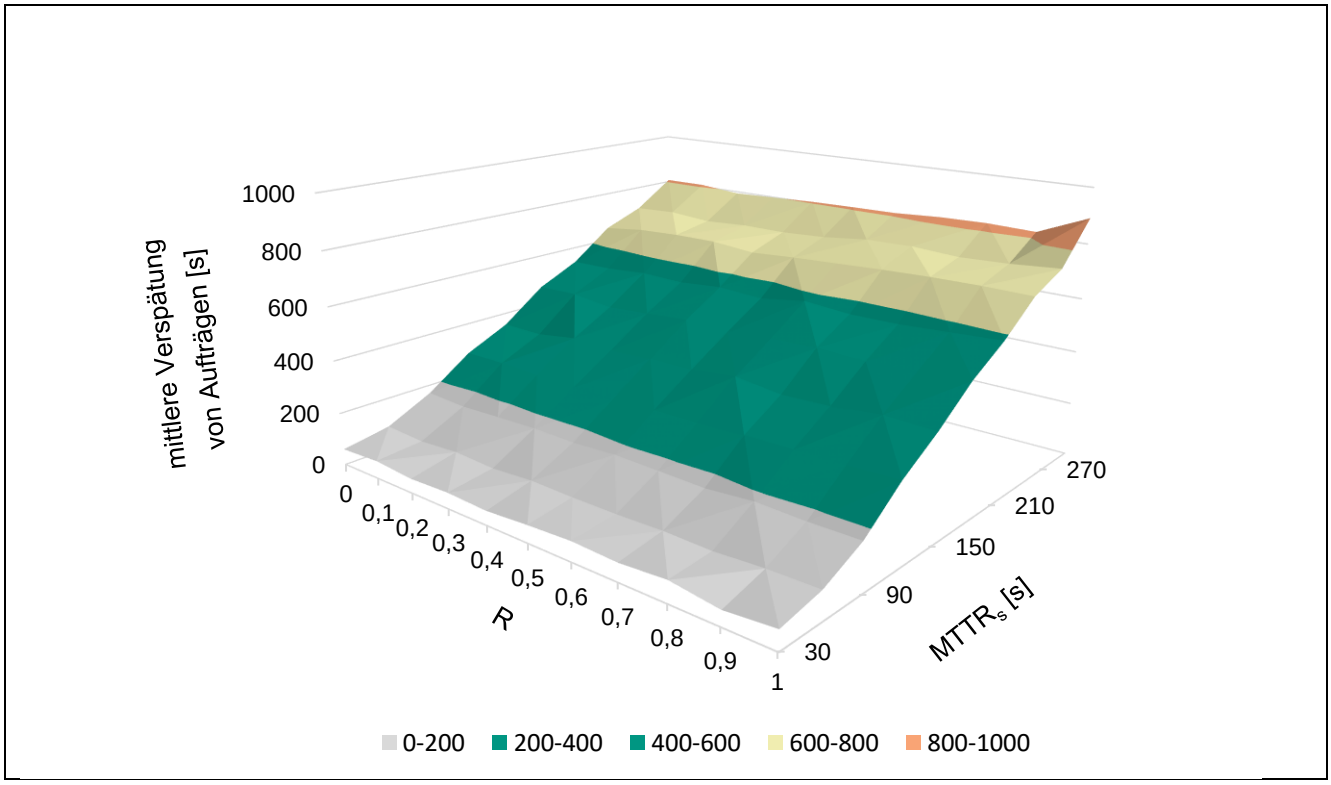




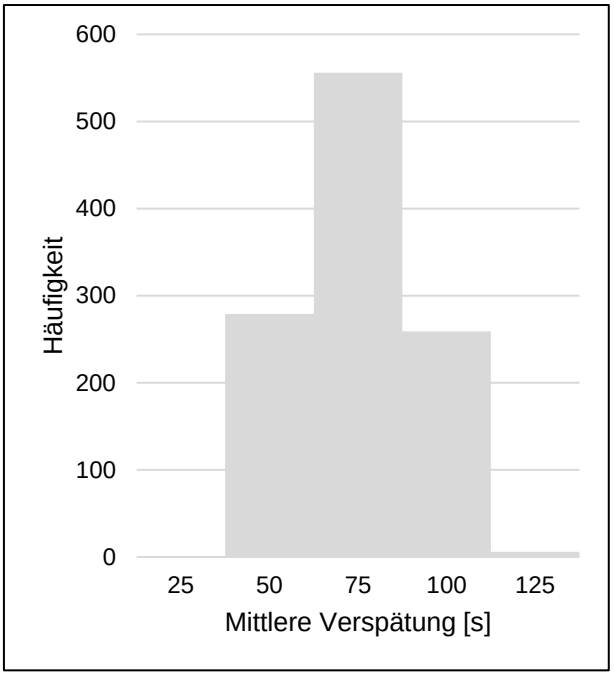
A8 Ergebnisse der Simulationsstudie zur mittleren Verspätung

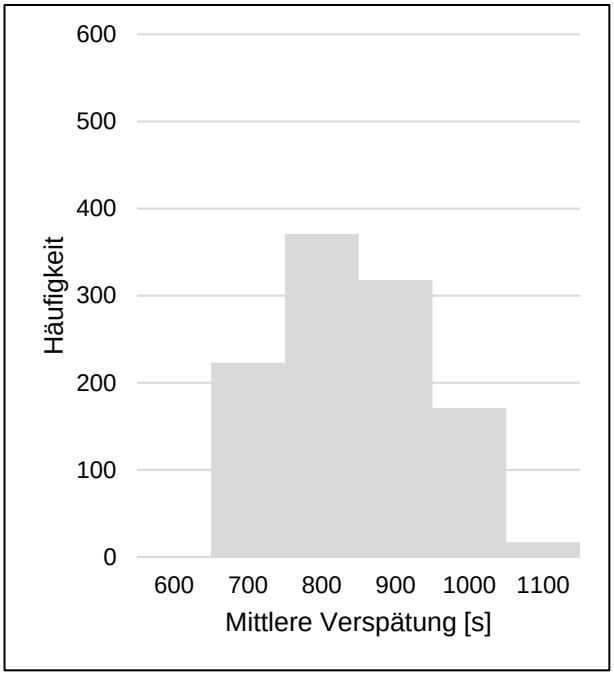
MTTR$_s$ \ R	0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	1	μ
30	59,8	67,2	54,6	60,5	53,6	65,1	73,5	66,7	78,7	54,9	70,5	64,1
60	89,4	80,8	91,0	83,0	93,3	90,0	90,4	95,2	91,5	101,7	103,6	91,8
90	168,6	160,0	166,7	172,0	169,3	177,5	173,6	168,8	182,3	176,2	176,6	172,0
120	280,9	281,5	274,6	272,1	271,2	279,5	282,5	287,7	289,9	289,7	297,7	282,5
150	353,8	351,2	374,2	366,7	374,6	375,3	394,0	374,0	379,8	398,1	392,3	375,8
180	469,0	473,0	454,7	465,1	476,8	467,0	483,3	489,8	481,4	487,9	503,0	477,4
210	539,6	542,1	554,1	549,0	563,2	553,5	578,5	565,1	566,5	576,0	585,9	561,2
240	651,7	670,6	677,1	677,5	662,3	673,0	680,3	688,1	693,6	694,9	691,5	678,2
270	708,8	715,9	703,0	711,8	710,3	718,6	737,2	739,9	728,8	735,0	748,3	723,4
300	806,9	807,2	792,8	798,4	804,5	807,1	808,5	818,2	821,3	815,6	892,0	815,7
Mittelwert	412,9	414,9	414,3	415,6	417,9	420,7	430,2	429,3	431,4	433,0	446,2	

A9 Einfluss von $MTTR_s$ und R auf die mittlere Verspätung



A10 Häufigkeitsverteilung der mittleren Verspätung von Aufträgen für $MTTR_s = 30s$ (links) und $MTTR_s = 300s$ (rechts)





A11 Empirische Ermittlung der Lernparameter

Die maximale Anzahl zu durchlaufender Epochen E_{max} hat maßgeblichen Einfluss auf die erreichbare Lösungsgüte, da sie die Anzahl der Lernschritte festlegt. Allerdings ist ein höherer Wert von E_{max} auch mit erhöhtem Rechenaufwand verbunden. Daher wird zur Festlegung von E_{max} das Update-Intervall bzw. die Länge einer Epoche E heran gezogen. Mit dem Update-Intervall E wird die Varianz der Monte-Carlo-Schätzung determiniert. Kleine Update-Intervalle führen zu einer hohen Varianz und damit auch zu einer hohen Anzahl an nötigen Lernschritten bis zur Erreichung eines lokalen Optimums. Bei großen Update-Intervallen ist hingegen eine hohe Anzahl an Episoden E_{max} erforderlich, um eine ausreichende Anzahl an Lernschritten zu erlauben. In (Gabel & Riedmiller 2012) wurden für Update-Intervalle zwischen 10 und 100 ähnliche Ergebnisse erzielt. Aufgrund der geänderten Problemstruktur mit tendenziell größeren Rescheduling-Problemen wurden im Rahmen der Arbeit noch die Update-Intervalle 100, 200 und 1000 untersucht (mit $\beta = 0{,}01, E_{max} = 100 * E$). Für typische Rescheduling-Korridore konnten für $E > 100$ keine geringeren Gesamtdurchlaufzeiten erzielt werden, allerdings waren die Rechenzeiten entsprechend deutlich höher. Es wird daher $E = 100$ verwendet. Zudem hat sich gezeigt, dass bei dieser Konfiguration jenseits von 100 Lernschritten kaum noch eine Verbesserung der Gesamtdurchlaufzeit erzielbar war, sodass $E_{max} = 100 * E = 10000$ verwendet wird. (A_Hort 2019).

Der Lernparameter δ schließlich determiniert den Trade-Off zwischen Exploration und Exploitation, wobei Werte nahe 0 die Exploration befördern. In der Literatur werden oft Werte $\delta < 0{,}01$ verwendet, um eine zu schnelle Konvergenz zu verhindern. Diese würde dazu führen, dass nach wenigen Lernschritten bereits nur noch dieselben Aktionen gewählt werden und eine weitere Verbesserung nicht möglich ist. Hierfür wurde keine gesonderte Untersuchung durchgeführt, sondern durchgängig $\delta = 0{,}01$ verwendet. Für eine nähere empirische Untersuchung optimaler Lernparameter sei auf A_Hort (2019) verwiesen.

A12 Benchmark-Optimierungsproblem

Für die Reoptimierung von Rescheduling-Korridoren zu Vergleichszwecken kann auf dem in Kapitel 4.3.1 formulierten Optimierungsmodell aufgebaut werden.

$$\min C_{max} \qquad \text{A12-1}$$

Die neue Zielfunktion A12-1 minimiert den Makespan, da dies auch das Optimierungsziel des reaktiven Reschedulings darstellt. Die Nebenbedingungen 4-4 - 4-15 gelten weiterhin. Es werden nachfolgend die neuen Nebenbedingungen erläutert, die für die Berücksichtigung der Randbedingungen beim Rescheduling benötigt werden.

$$startOf(IP_o) = 0, \forall o \in O: m \in M_j^k | \left(x_{j,s}^{k_j} < UG \wedge x_{j,s}^{k_j} + p_{j,s}^{k_j,R} > UG \right) \quad \text{A12-2}$$

Nebenbedingung A12-2 stellt sicher, dass alle Aufträge, die sich bei Beginn der Störung an einer nicht gestörten Station befinden, am Anfang des Rescheduling-Korridors mit ihrer Restbearbeitungszeit eingeplant werden. So wird sichergestellt, dass keine belegte Station durch das Rescheduling beansprucht werden.

$\forall\, j \in J;\ k_1, k_2 \in K_j; o_1, o_2 \in O; m_1, m_2 \in M_j^k: k_2 = k_1 + 1$:

$$startOf(IP_o) * presenceOf(m) >= \left(rt_s^R + t_{j,s'}^R \right) * presenceOf(m) \quad \text{A12-3}$$

$$startAtEnd(IT_o, IP_o) \quad \text{A12-4}$$

$$stendOf(IT_o) * presenceOf(m) = \left(endOf(IP_o) + rt_s^R + t_{j,s'}^R \right) * presenceOf(m) \quad \text{A12-5}$$

Die Nebenbedingungen A12-3 bis A12-5 behandeln den Sonderfall beim Rescheduling, dass ein umzuplanender Auftrag zu Beginn der Planungsperiode noch an einer anderen Station bearbeitet wird und entsprechend frühestens nach Ende der Bearbeitung dort anderweitig eingeplant werden darf. Nebenbedingung A12-3 stellt sicher, dass in dem Fall wie bei anderen Aufträgen auch der nachfolgende Bearbeitungsschritt frühestens beginnen darf, wenn der vorhergehende Bearbeitungsschritt und der Transport zuzüglich Wartezeit an die nachfolgende Station abgeschlossen ist. Der Beginn des Transportintervalls erfolgt nach Nebenbedingung A12-4, sobald die Restbearbeitungszeit verstrichen ist und endet nach Nebenbedingung A12-5, wenn der Transport zur nachfolgenden Station beendet ist.

$$t_{j,s'}^R = s' \rightarrow startOf(IP_o) * presenceOf(m) \geq rt_{j,s'}^R, \forall o \in O; m \in M_j^k \quad \text{A12-6}$$

Ein weiterer Sonderfall beim Rescheduling sind Aufträge, die sich zu Beginn der Planungsperiode bereits auf dem Weg zu einer Station befinden. Zur Berechnung der Transportzeit im Fall einer Umplanung kann nicht auf die Distanzmatrix zurückgegriffen

werden, sondern es muss der tatsächliche Ort und die resultierende relative Transportzeit zu den alternativen Stationen berücksichtigt werden. Nebenbedingung A12-6 sorgt dafür, dass bei der Umplanung eines solches Auftrags mindestens die relative Transportzeit verstreicht.

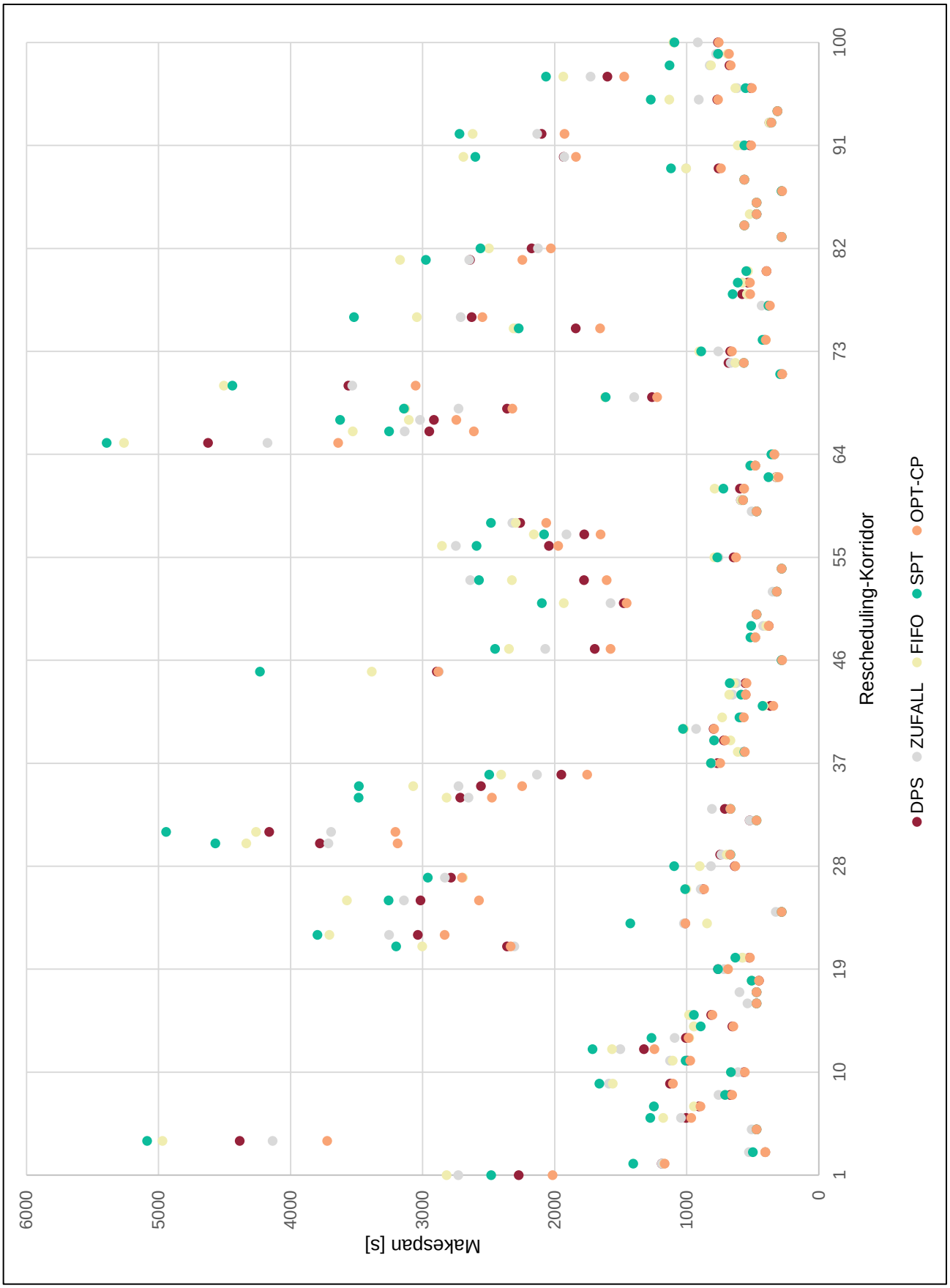
A13 Ergebnisse des Reschedulings für 100 Rescheduling-Korridore

Daten aus der Ermittlung der Rescheduling-Korridore				Gesamtdurchlaufzeiten des Rescheduling [s]				
RK-ID	**Länge RK [s]**	**Iterationen**	**Bearbeitungs-schritte**	**DPS**	**ZU-FALL**	**FIFO**	**SPT**	**OPT-CP**
1	2.497	27	243	2.274	2.731	2.820	2.482	2.017
2	1.191	11	80	1.191	1.191	1.172	1.406	1.168
3	423	1	7	407	530	407	500	407
4	4.478	53	494	4.387	4.138	4.971	5.087	3.723
5	473	1	12	473	509	473	473	473
6	1.024	6	51	1.006	1.044	1.179	1.278	968
7	1.042	5	41	909	1.251	946	1.249	897
8	693	3	14	671	760	710	710	658
9	1.427	12	96	1.128	1.590	1.561	1.663	1.106
10	585	4	31	567	612	666	666	562
11	1.038	5	41	994	1.128	1.107	1.010	975
12	1.442	12	94	1.326	1.504	1.566	1.715	1.245
13	1.011	5	41	1.007	1.092	982	1.267	988
14	864	5	22	656	947	947	896	647
15	869	5	30	816	974	984	947	807
16	473	1	13	473	540	473	473	473
17	473	3	18	473	602	473	473	473
18	483	4	23	455	509	509	509	455
19	745	6	38	765	720	689	765	689
20	579	3	18	527	582	578	634	522
21	2.438	32	293	2.363	2.306	3.004	3.201	2.335
22	3.124	34	373	3.038	3.255	3.708	3.797	2.834
23	1.016	8	50	1.014	1.022	1.011	1.429	847
24	283	1	4	283	326	283	283	283
25	3.140	32	313	3.017	3.143	3.575	3.259	2.574
26	884	6	35	876	896	1.004	1.014	870
27	2.801	28	275	2.786	2.832	2.696	2.961	2.703
28	774	5	35	637	818	903	1.096	633
29	726	4	24	747	738	698	670	671
30	4.055	44	422	3.780	3.715	4.337	4.571	3.190
31	4.194	46	446	4.163	3.695	4.263	4.944	3.207
32	501	2	11	524	524	473	473	473
33	702	4	23	712	811	670	670	670

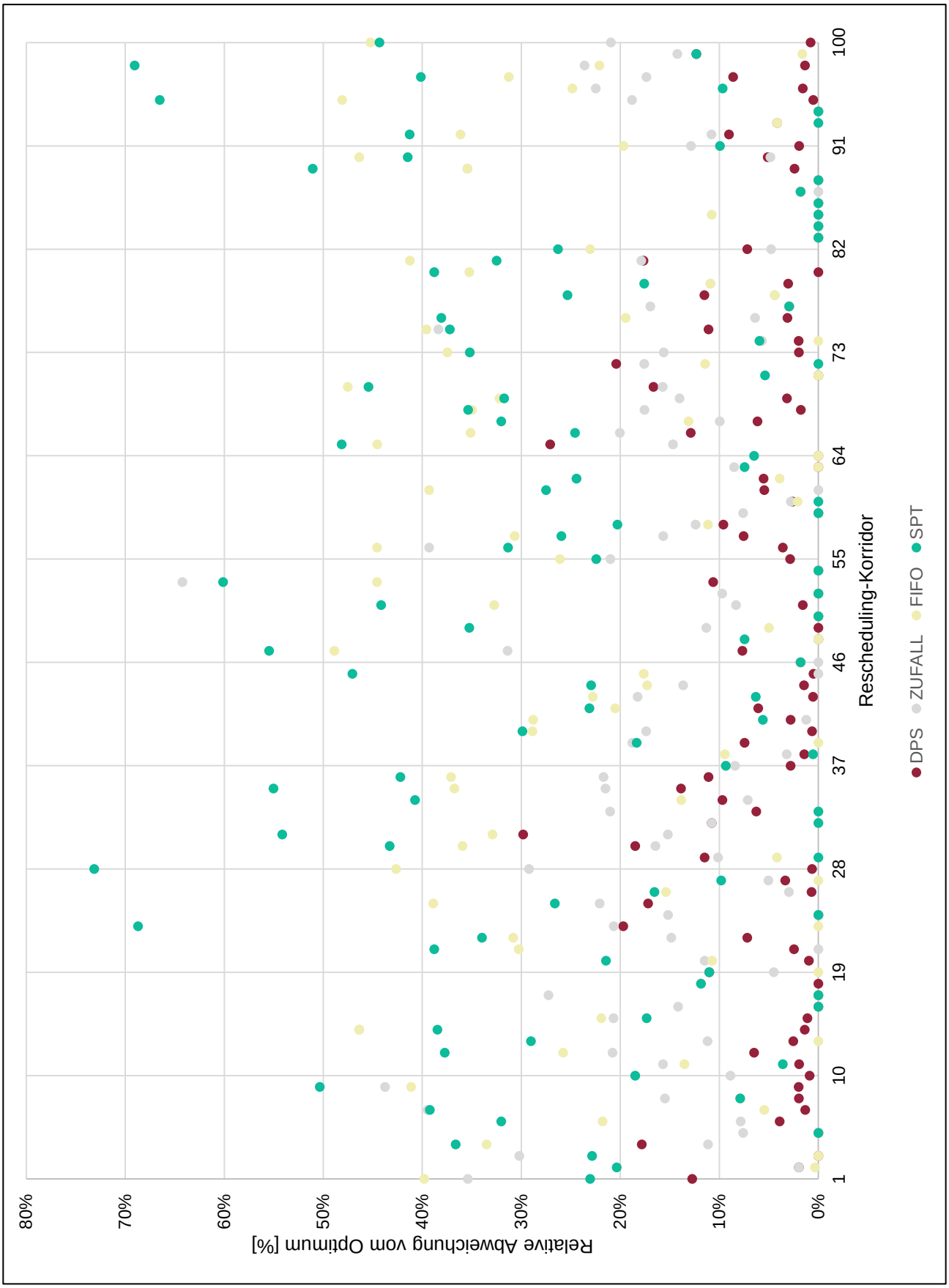
34	2.767	30	298	2.716	2.653	2.819	3.485	2.476
35	2.833	30	294	2.559	2.730	3.073	3.484	2.247
36	2.153	20	201	1.950	2.136	2.406	2.496	1.755
37	797	5	27	769	811	818	818	748
38	573	3	11	569	579	614	564	561
39	747	4	22	720	796	670	793	710
40	878	5	30	798	931	1.022	1.030	793
41	585	3	14	585	576	733	601	569
42	412	1	11	367	426	417	426	346
43	565	2	15	556	654	679	588	553
44	616	4	28	557	624	644	675	549
45	3.034	37	355	2.893	2.879	3.387	4.234	2.882
46	283	1	7	283	278	283	283	278
47	1.929	24	223	1.698	2.072	2.348	2.452	1.577
48	482	4	19	482	482	482	518	482
49	399	2	8	380	423	399	514	380
50	473	2	11	473	473	473	473	473
51	1.686	17	142	1.479	1.577	1.933	2.099	1.456
52	319	2	2	319	350	319	319	319
53	2.032	14	174	1.779	2.641	2.325	2.575	1.608
54	283	1	7	283	283	283	283	283
55	746	4	27	646	760	792	769	628
56	2.631	24	232	2.045	2.750	2.854	2.593	1.974
57	1.895	19	169	1.778	1.912	2.160	2.082	1.653
58	2.300	28	252	2.263	2.321	2.295	2.484	2.065
59	473	1	9	473	509	473	473	473
60	590	2	15	590	591	587	575	575
61	645	5	22	598	567	790	723	567
62	334	3	14	324	382	319	382	307
63	499	4	16	482	523	482	518	482
64	338	2	3	338	338	338	360	338
65	4.784	57	520	4.627	4.176	5.263	5.394	3.641
66	3.185	33	323	2.950	3.137	3.531	3.255	2.613
67	3.051	27	276	2.915	3.020	3.106	3.626	2.746
68	2.722	24	279	2.362	2.729	3.133	3.142	2.321
69	1.328	17	147	1.265	1.398	1.621	1.615	1.226
70	3.769	38	381	3.563	3.534	4.506	4.442	3.054
71	278	3	9	278	278	278	293	278
72	655	2	14	684	668	633	568	568
73	789	8	47	672	762	906	891	659
74	416	1	15	411	426	403	427	403
75	2.086	21	192	1.841	2.293	2.313	2.274	1.657
76	2.850	31	300	2.629	2.712	3.045	3.520	2.549
77	382	2	11	382	434	382	382	371
78	576	3	14	581	544	544	653	521

79	540	3	11	539	580	580	615	523
80	443	1	10	397	551	537	551	397
81	2.739	28	260	2.643	2.648	3.173	2.976	2.246
82	2.182	20	209	2.176	2.127	2.498	2.564	2.030
83	283	2	3	283	283	283	283	283
84	564	2	11	564	564	564	564	564
85	473	3	18	473	473	524	473	473
86	473	1	12	473	473	473	473	473
87	283	1	15	283	278	283	283	278
88	564	2	12	564	564	564	564	564
89	836	3	31	760	1.005	1.005	1.121	742
90	1.982	19	194	1.933	1.928	2.692	2.602	1.839
91	536	3	11	523	579	614	564	513
92	2.186	24	227	2.100	2.134	2.622	2.721	1.926
93	375	1	3	375	375	375	360	360
94	314	1	5	314	314	314	314	314
95	897	4	47	769	909	1.133	1.274	765
96	561	3	22	515	621	633	556	507
97	1.649	15	153	1.601	1.730	1.935	2.066	1.474
98	745	7	36	678	827	817	1.131	669
99	765	5	25	765	778	692	765	681
100	987	5	43	764	917	1.101	1.094	758

A14 Absolute Lösungsgüte der Rescheduling-Korridore



A15 Relative Lösungsgüte der Rescheduling-Korridore



A16 Ergebnisse der Simulationsstudie für das prädiktiv-reaktive Scheduling

	mittlere absolute Verschiebung [s]		mittlere Verspätung [s]	
R	**Ohne DPS**	**Mit DPS**	**Ohne DPS**	**Mit DPS**
0	530	1.449	428	454
0,1	520	1.424	431	447
0,2	515	1.399	434	442
0,3	510	1.374	436	440
0,4	505	1.349	439	438
0,5	499	1.324	441	436
0,6	494	1.298	444	432
0,7	489	1.274	447	425
0,8	485	1.249	449	418
0,9	480	1.224	452	419
1	460	1.200	455	422