

Hochschule Anhalt

Promotionszentrum Sozial-, Gesundheits- und Wirtschaftswissenschaften

Feldsynchrone Ablaufplanung dynamischer Fertigungsprozesse mit Techniken des maschinellen Lernens

KUMULATIVE DISSERTATION

zur Erlangung des akademischen Grades Dr. rer. pol.

von Felix Johannes Grumbach, M.Sc.

Betreuer	Prof. Dr.-Ing. Sebastian Trojahn, Hochschule Anhalt
1. Gutachter	Prof. Dr.-Ing. Fabian Behrendt, Hochschule Magdeburg-Stendal
2. Gutachter	Prof. Dr.-Ing. Hartmut Zadek, Otto-von-Guericke-Universität Magdeburg
Tag der Einreichung	14.09.2023
Tag der Disputation	23.02.2024
Erscheinungsort/-jahr	Bernburg, 2024

Kontakt

Felix Johannes Grumbach
Hochschule Bielefeld
Interaktion 1, D-33619 Bielefeld
E-Mail: felix.grumbach@hsbi.de

Bibliografische Informationen

ISBN (Print): 978-3-96057-173-5
ISBN (Online): 978-3-96057-174-2

Gemäß APA-Richtlinien (7. Auflage) kann diese Arbeit wie folgt zitiert werden:

Grumbach, F. (2024). *Feldsynchrone Ablaufplanung dynamischer Fertigungsprozesse mit Techniken des maschinellen Lernens* [Dissertation, Hochschule Anhalt]. ISBN 978-3-96057-174-2.

Lizenzbestimmungen und Urheberrecht

Dieses Werk ist unter der Creative Commons Attribution 4.0 International License (CC BY 4.0) lizenziert. Um eine Kopie dieser Lizenz einzusehen, besuchen Sie www.creativecommons.org/licenses/by/4.0/ oder senden Sie einen Brief an Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Diese Lizenz erlaubt es Ihnen, das Werk zu teilen – das heißt, es zu kopieren und weiterzuverteilen – sowie das Werk zu bearbeiten, zu verändern und darauf aufzubauen, und das für jegliche Zwecke, einschließlich kommerzieller Nutzung. Voraussetzung dafür ist, dass Sie eine angemessene Anerkennung geben, einen Link zur Lizenz bereitstellen und angeben, ob Änderungen vorgenommen wurden. Diese Bedingungen können auf jede angemessene Art und Weise erfüllt werden, allerdings nicht in einer Weise, die den Eindruck erweckt, der Lizenzgeber unterstütze Sie oder Ihre Nutzung des Werks.

© 2024 Felix Johannes Grumbach

»Die Kunst, Pläne zu machen, besteht darin, den Schwierigkeiten ihrer Ausführung zuvorzukommen.«

Luc de Clapiers

Für Johannes Grumbach (1938-2015)

Zusammenfassung

Aktuelle Lösungen zur ganzheitlichen und echtzeitnahen Planung dynamischer Fertigungsprozesse stoßen an ihre Grenzen. Dies gilt vor allem für komplexe, soziotechnische Produktionsumgebungen mit flexiblen Materialflüssen sowie unbestimmten Ereignissen und Schwankungen. Verfahren der Optimierung unter Unsicherheit sind sehr rechenintensiv und entscheidende Wechselwirkungen der realen Welt werden nur unzureichend berücksichtigt. Dieser Mangel an Feldsynchronität mindert die Qualität der Produktionspläne, führt zu manuellen Aufwänden (*Fire Fighting*) und hat einen negativen Effekt auf die logistische Gesamtleistung.

Die vorliegende Arbeit basiert auf vier Journalartikeln, welche neuartige Methoden und Modelle zur Verbesserung der feldsynchronen Ablaufplanung demonstrieren. Durch die Vereinigung von Instrumenten des Operations Research und des Machine Learnings werden generische und prädiktive Algorithmen erarbeitet, um die Effizienz und Effektivität der Planungsprozeduren zu verbessern. Die gewonnenen Erkenntnisse legen nahe, dass Regressionsmodelle rechenaufwändige stochastische Simulationen zur Erhebung von Robustheitsmetriken ersetzen können. Zudem können mithilfe von Reinforcement Learning sowohl unsicherheitsrobuste als auch realistische Ablaufpläne für die humanzentrierte Fertigung in kurzer Zeit erzeugt werden. Dazu werden diskrete Simulationsmodelle benutzt, welche auf Basis einer allgemeingültigen Steuerungslogik datengetrieben initialisiert werden. Die Algorithmen können in eine virtuelle Fabrik integriert werden, welche als digitale Repräsentanz der realen Welt Grundlage smarterer und feldsynchroner Dispositionssysteme ist. In diesem Zusammenhang kann ein prototypisches System für die Planung dynamischer Fertigungsprozesse vorgestellt werden, welches von Forschungspartnern aus der Industrie erprobt und in Kooperation weiterentwickelt wird.

Über die Publikationen hinaus lassen sich weitere Forschungsbedarfe ableiten. Um die Übertragbarkeit der Verfahren sicherzustellen, müssen sie zunächst im Kontext weiterer und umfassenderer Umgebungen evaluiert werden. Aus wissenschaftlicher und praktischer Sicht ist es eine entscheidende Herausforderung, holistische und vorausschauende Ablaufplanungssysteme zu entwickeln, welche ein umfassendes Instrumentarium an datengetriebenen Analyse- und Entscheidungsprozessen orchestrieren. In dieser Hinsicht müssen die vorgestellten Methoden und Modelle weiterentwickelt und in einem generischen Gesamtkonzept vereint werden. In der Arbeit werden vier Fokusfelder abgeleitet, die in künftigen Arbeiten auf interdisziplinäre Weise weiter beleuchtet werden müssen: (1) Generische Simulationsmodelle, (2) Humanzentrierte Optimierung, (3) Feldsynchroner Ablaufplanung sowie (4) Systementwicklung und Rollout.

Abstract

Current solutions for holistic and real-time planning of dynamic manufacturing processes are reaching their limits. This is particularly applicable to complex socio-technical production environments with flexible material flows as well as undetermined events and fluctuations. Methods of optimization under uncertainty are very computationally intensive and crucial interactions with the real world are insufficiently considered. This lack of field synchronicity reduces the quality of production schedules, leads to manual efforts (firefighting), and has a negative impact on the logistical performance.

The present work is based on four journal articles that demonstrate novel methods and models for improving field-synchronous scheduling. Through the combination of instruments from operations research and machine learning, generic and predictive algorithms are developed to improve the efficiency and effectiveness of planning procedures. The findings suggest that regression models can replace computation-heavy stochastic simulations in obtaining robustness metrics. Additionally, using reinforcement learning, uncertainty-robust and realistic production schedules for human-centered manufacturing can be generated in a short time. For this purpose, discrete simulation models are used, which are data-driven initialized based on a general control logic. The algorithms can be integrated into a virtual factory, which serves as a digital representation of the real world and is the basis for smart and field-synchronous scheduling systems. In this context, a prototype distributed system for the planning of dynamic manufacturing processes can be presented, which is being tested by industry research partners and further developed in collaboration.

Beyond the publications, further research needs can be derived. In order to ensure the transferability of the methods, they need to be evaluated in the context of additional and more comprehensive environments. From a scientific and practical perspective, it is a crucial challenge to develop holistic and proactive scheduling systems that orchestrate a comprehensive set of data-driven analysis and decision-making processes. In this regard, the presented methods and models need to be further developed and integrated into a generic overall concept. The work identifies four focus areas that future research should address in an interdisciplinary manner: (1) Generic simulation models, (2) Human-centered optimization, (3) Field-synchronous scheduling, and (4) System development and rollout.

Danksagung

Es war wohl ein Quäntchen Glück dabei, dass ich direkt die passende Forschungsstelle und Promotionsmöglichkeit gefunden habe. Insofern hatte ich Glück, dass die Ausrichtung der Forschungsprojekte perfekt auf meinen Werdegang passte, ich mich direkt mit der Thematik wohlfühlte und stets die richtigen Personen zur richtigen Zeit kennenlernen durfte. In Reihenfolge der räumlichen Distanz möchte ich folgendermaßen meinen Dank an jene Organisationen und Personen aussprechen, die mein Promotionsvorhaben erst möglich gemacht haben.

Zuerst gilt mein Dank dem Promotionszentrum SGW und besonders meinem Doktorvater Sebastian Trojahn. Lieben Dank für die stets angenehme Betreuung, die richtungsweisende Unterstützung sowie für das wertvolle Feedback zu meinen Veröffentlichungen und zu meiner Vorgehensweise. Besten Dank auch an meine Gutachter Fabian Behrendt und Hartmut Zadek für die umfassende Begutachtung meiner Publikationen und der Mantelschrift.

Zudem bedanke ich mich bei meiner Arbeitgeberin, der Hochschule Bielefeld, dem Land Nordrhein-Westfalen sowie bei den Projektpartnern Bio-Circle Surface Technology GmbH, Fraunhofer IOSB-INA, Isringhausen GmbH & Co. KG, Miele & Cie. KG, MIT Moderne Industrietechnik GmbH & Co. KG und PerFact Innovation GmbH & Co. KG. Ohne die Forschungsstelle, Drittmittel, Ressourcen und spannenden Anwendungsfälle wären meine Studien nicht möglich gewesen.

Besten Dank auch an das Center for Applied Data Science und an meine Arbeitsgruppe. Auch wenn es im operativen Forschungs- und Lehrbetrieb manchmal drunter und drüber geht, ist auf euch immer Verlass. Vielen Dank auch an meinen Chef Pascal Reusch für die kontinuierliche Förderung meines Promotionsvorhabens. Ich bin außerordentlich dankbar dafür, dass sich die Projektarbeit, Lehre und Forschung so wunderbar synchronisieren ließen. Ein großer Dank geht an meine unmittelbaren Kollegen und Mitautoren Nour Badr, Anna Müller und Lukas Vollenkemper für die tolle und erfolgreiche Zusammenarbeit. Des Weiteren möchte ich mich bei Daniel Fischer, Stefan Görlitz, Hannah Hüsener, Anna Müller, Pascal Reusch und Sebastian Trojahn für die sorgfältige Prüfung und das wertvolle Feedback zu meinem Manteltext bedanken.

Herzlichen Dank an meine Familie und Freunde für die mentale Unterstützung und Motivation auf der einen oder anderen Durststrecke. Besonderer Dank gebührt meiner Partnerin Raquel Koltzsch, die mich während des gesamten Prozesses unermüdlich unterstützt und motiviert hat. Durch euch habe ich die nötige Energie geschöpft, um am Ball bleiben zu können.

Assoziierte Publikationen

Die Dissertation basiert auf folgenden vier Artikeln, die in *Scopus*-indizierten Fachjournalen veröffentlicht wurden (siehe Anhang *Journal-Rankings*):

1. Felix Grumbach, Nour E. A. Badr, Pascal Reusch und Sebastian Trojahn (2023):
A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling. *IEEE Access* 11. DOI: 10.1109/ACCESS.2023.3292548
2. Lukas Vollenkemper, Felix Grumbach, Martin Kohlhase und Pascal Reusch (2023):
Humanzentrierte Ablaufplanung von Montagelinien. *wt Werkstattstechnik online* 113.04. S. 158-163. DOI: 10.37544/1436-4980-2023-04-58
3. Felix Grumbach, Anna Müller, Pascal Reusch und Sebastian Trojahn (2023):
Robustness Prediction in Dynamic Production Processes - A new Surrogate Measure based on Regression Machine Learning. *Processes* 11.4. DOI: 10.3390/pr11041267
4. Felix Grumbach, Anna Müller, Pascal Reusch und Sebastian Trojahn (2022):
Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning. *Journal of Intelligent Manufacturing*. DOI: 10.1007/s10845-022-02069-x

Abkürzungen

APS	Advanced Planning and Scheduling
C_{max}	Zykluszeit
DRC-FJSSP	Dual Resource Constrained Flexible Job Shop Scheduling Problem
GAN	Generative Adversarial Networks
JSSP	Job Shop Scheduling Problem
KNN	Künstliches neuronales Netz
LLM	Large Language Model
MCE	Monte-Carlo-Experiment
MDP	Markov Decision Process
ML	Machine Learning
NP	Nichtdeterministisch polynomielle Zeit
PPS	Produktionsplanung und -steuerung
RL	Reinforcement Learning
SM	Surrogatmaß

Inhaltsverzeichnis

1	Einleitung	1
1.1	Ausgangslage und Forschungsbedarf	1
1.2	Begriffsbestimmungen und Konzeptualisierung	4
1.2.1	Ablaufplanung dynamischer Fertigungsprozesse	5
1.2.2	Anforderungen an eine feldsynchrone Ablaufplanung	8
1.2.3	Effiziente Planung mit Optimierungsmethoden	10
1.2.4	Techniken des ML	16
1.3	Zielsetzung und Forschungsdesign	23
2	Publikationen	27
2.1	Publikation 1: A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling	27
2.2	Publikation 2: Humanzentrierte Ablaufplanung von Montagelinien	44
2.3	Publikation 3: Robustness Prediction in Dynamic Production Processes	51
2.4	Publikation 4: Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning	66
3	Synthese der Ergebnisse	87
3.1	Hauptergebnisse der durchgeführten Studien	87
3.2	Perspektiven künftiger Forschung, Entwicklung und Anwendung	90
3.2.1	Generische Simulationsmodelle	91
3.2.2	Humanzentrierte Optimierung	95
3.2.3	Feldsynchrone Ablaufplanung	98
3.2.4	Systementwicklung und Rollout	105
3.3	Entwickeltes Scheduling-Framework	106
3.3.1	Architektur und Anwendungskontext	106
3.3.2	Workflow der Datenverarbeitung	107
3.4	Schlussbemerkungen	109
	Literaturverzeichnis	113

Anhang	123
Liste aller Veröffentlichungen	123
Forschungsdaten	124
Eingesetzte Technologien und Hilfsmittel	124
Scheduling-Framework	125
Journal-Rankings	129
Eidesstattliche Erklärung	130

Tabellenverzeichnis

1.1 Kernaufgaben des Aachener PPS-Modells	7
1.2 Teilschritte zur Planung von Fertigungsprozessen	7
1.3 Qualitätsmerkmale für Verfahren der feldsynchrone Ablaufplanung	8
1.4 Wesentliche Aufgaben von ML-Algorithmen	18
1.5 Lernarten von ML-Algorithmen	19

Abbildungsverzeichnis

1.1 Vereinfachte Darstellung des Referenzmodells <i>Virtuelle Fabrik</i>	3
1.2 Erlaubter Bereich eines linearen Optimierungsmodells	11
1.3 Erzeugung einer Nachbarschaftslösung in einem Permutation Flow Shop	14
1.4 Nutzen von Optimierungsmethoden im Bezug zu einer Rechenzeitobergrenze	15
1.5 Einfaches Beispiel eines tiefen KNN	17
1.6 RL-Prozess	21
1.7 Übersicht der Publikationen im Kontext der Teilfragen	25
3.1 Fokusfelder zur Entwicklung smarter Dispositionssysteme	91
3.2 Zustandsautomat zur Simulation eines asynchronen Stationsprozesses	93
3.3 Prozess der vorausschauenden Umplanung mit dynamischen Neuaufträgen	105
3.4 Informationsfluss des implementierten Scheduling-Frameworks	107

1. Einleitung

1.1 Ausgangslage und Forschungsbedarf

Industrie 4.0 – ein Megatrend. Ziel ist die Transformation von Fertigungsunternehmen zu sich selbst organisierenden intelligenten Fabriken. Dies betrifft nicht nur neuartige Produktionstechnologien, sondern auch übergreifende industrielogistische Prozesse und die Möglichkeiten, diese mithilfe digitaler Technologien automatisch zu disponieren. (BMBF 2020) Folglich ist die Produktionsplanung und -steuerung (PPS) eine bedeutende Forschungs- und Anwendungsdomäne im Kontext von Industrie 4.0.

Mit zunehmendem technischen Fortschritt hat sich in jüngerer Zeit ein wachsendes weltweites Interesse an dem Thema herauskristallisiert. Eine technologiebasierte *smarte Disposition* von Fertigungsprozessen ist sowohl von praktischer als auch akademischer Relevanz. Dabei handelt es sich um ein multidisziplinäres Feld in der Schnittstelle von Betriebswirtschaft, Logistik, Informatik, Statistik, Machine Learning (ML), Operations Research und Ingenieurwissenschaften. (Serrano-Ruiz, Mula und Poler 2021) Je nach Komplexität des Anwendungsfalls wird eine solche smarte Disposition erst durch moderne Spitzentechnologien, performante Algorithmen und konsistente Datenbestände möglich (Schmidtke et al. 2018; Sarker 2021; Lavin et al. 2022). Wie in den vier vorgestellten Publikationen (siehe Kapitel 2) ausführlicher dargelegt, ist die ganzheitliche und vorausschauende PPS somit kein triviales Unterfangen. Insbesondere dann nicht, wenn die Planung echtzeitnah unter Berücksichtigung flexibler Faktoren durchgeführt werden soll.

Für eine umfassende PPS auf Basis mathematischer Optimierungsmodelle existieren auf dem Softwaremarkt sogenannte Advanced-Planning-and-Scheduling-Systeme (APS-Systeme). Nach der Analyse repräsentativer wissenschaftlicher Beiträge (Lupeikiene et al. 2014; Fachini, Esposito und Camargo 2018; Hsu, L.-C. Wang und Chu 2018; Eriksson, Carlsson und Olsson 2022; Stüve et al. 2022; Vieira, Deschamps und Valle 2021; L.-C. Wang et al. 2021; M. Li et al. 2021; Oluyisola et al. 2021; Schlecht, Köbler und de Guio 2021), lassen sich folgende Erkenntnisse zusammenfassen: APS-Systeme sind teuer in der Einführung und kompliziert in der Bedienung. Die Integration von APS-Systemen in eine bestehende Prozesslandschaft setzt ein hohes Digitalisierungslevel voraus. Vor allem die Synchronisation mit bestehenden

Informationssystemen wie Produktionsleitsystemen und zugehörigen Datenbeständen gestaltet sich aufwändig. APS-Systeme sind tendenziell wenig benutzerfreundlich, schulungs- und wartungsintensiv. Die implementierten Planungsmodelle weisen eine Vielzahl pflegeintensiver Parameter auf und sind oftmals schwer nachvollziehbar, was zu Misstrauen bei den Anwendern führt. Ferner sind APS-Systeme limitiert in der vorausschauenden Planung unter Unsicherheit und üblicherweise unzureichend in der Lage, ad hoc auf unerwartete Prozessdynamiken oder Zustandsänderungen zu reagieren. Zudem werden in den implementierten Planungsmodellen spezielle reale Produktionsformen und Branchenspezifika nicht ausreichend berücksichtigt. Vor allem für kleine und mittelständische Unternehmen mit hochflexiblen Prozessen sind dies entscheidende Hürden für den Einsatz von APS-Systemen.

Seit einigen Jahren beschäftigt sich ein Teil der Fachwelt mit der Digitalisierung und präzisen Virtualisierung von Produktionsumgebungen mitsamt ihren Auftrags- und Materialflüssen. Speziell geht es unter anderem um die Verzahnung der physischen Produktionsumgebung mit ihrer virtuellen Repräsentanz, was auch als virtuelle Fabrik (Tao und M. Zhang 2017) oder digitaler Steuerungszwilling (Herlyn und Zadek 2020) bezeichnet wird (siehe Abbildung 1.1). Die virtuelle Fabrik ist mit der physischen Umgebung über ein Daten-Repositorium (gemeinsame Datenbasis) verbunden und eröffnet weitreichende Analyse- und Optimierungsmöglichkeiten, z.B. durch computergestützte Was-wäre-wenn-Experimente. Das Repositorium kapselt relevante Realwelt-Entitäten (Produktionsaufträge, Maschinen usw.) sowie aktuelle Zustände über diese (Live-Daten). Die virtuelle Fabrik modelliert das Verhalten der realen Fabrik und unterliegt ihren Regeln und Nebenbedingungen. Zur Veranschaulichung sei an dieser Stelle ein umfassendes Simulationsmodell angeführt, welches einen realen Produktionsprozess präzise imitiert und zugleich mit Live-Daten (z.B. aus einem Produktionsleitsystem) gespeist wird. So ist es mit einer ganzheitlichen Implementierung beispielsweise möglich, realistische Leistungskennzahlen zu erfassen und auszuwerten sowie anhand derer reale Abläufe zu optimieren und Bestände zu senken. Der Trend geht dahin, dass die virtuelle Fabrik nicht nur als ein Abbild der Realität fungiert, sondern darüber hinaus mit der physischen Welt wechselwirkt, indem sie diese maschinell interpretiert und steuert. Beispielsweise wird eine optimierte Auftragsreihenfolge und Maschinenbelegungsplanung in das Produktionsleitsystem zurück übertragen und darüber hinaus in der Durchführung automatisch überwacht. (Neto et al. 2021; Tao und M. Zhang 2017; Herlyn und Zadek 2020; E. Yildiz, Møller und Bilberg 2021) Kurzum: Die virtuelle Fabrik emanzipiert sich vom reinen Analyseinstrument hin zum feldsynchronen smarten Dispositionssystem, welches vorausschauend plant und proaktiv in den Geschäftsprozess eingreifen kann. Sie ist demnach eine geeignete Plattform für eine automatisierte PPS.

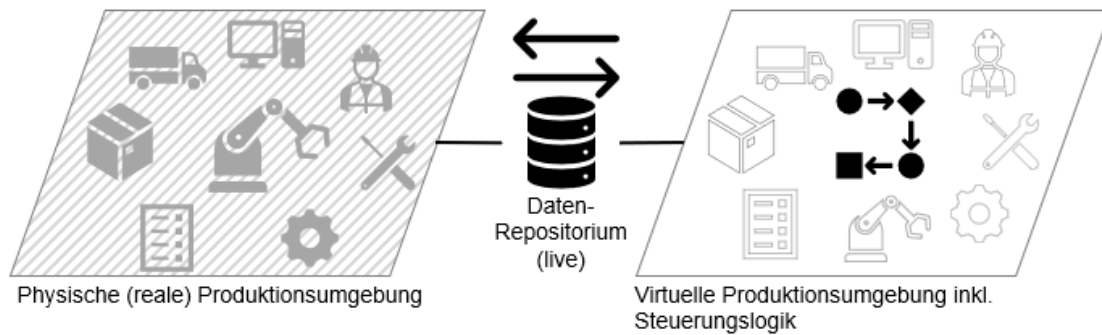


Abbildung 1.1: Vereinfachte Darstellung des Referenzmodells *Virtuelle Fabrik* in Anlehnung an Tao und M. Zhang (2017) und Herlyn und Zadek (2020). Die reale Umgebung wird mitsamt ihren aktuellen Objekten, Abläufen, Regeln, Zielen und Nebenbedingungen virtualisiert. Auf Basis dessen können z.B. Computeralgorithmen zur Analyse und Optimierung der Materialflüsse eingesetzt werden.

In der aktuellen wissenschaftlichen Literatur existieren bereits Ansätze, wie APS-Systeme mit dem Paradigma der virtuellen Fabrik vereint werden können (siehe z.B. Y. Wang und Z. Wu 2020; L.-C. Wang et al. 2021). Jedoch lassen sich, wie bereits dargelegt, grundsätzliche funktionale Defizite beobachten: Es bestehen nach wie vor dringliche Forschungsbedarfe zur Entwicklung realistischer Optimierungsmodelle sowie effizienter Optimierungsmethoden für eine ganzheitliche Ablaufplanung (Schlenkrich und Parragh 2023; Grumbach, Badr et al. 2023). In diesem Kontext sind herkömmliche Verfahren der robusten Planung, bei der unbestimmte Ereignisse und Schwankungen gezielt einkalkuliert werden, entweder sehr rechenintensiv oder sehr ungenau (Grumbach, A. Müller et al. 2022; Grumbach, A. Müller et al. 2023). In der nachhaltigen Planung komplexer Fertigungsprozesse spielt zudem der Faktor *Mensch* eine entscheidende Rolle. Je nach Domäne lassen sich 60-90% der gravierenden Störungen auf Ursachen menschlichen Handelns zurückführen (Kumar, S. Gupta und Gunda 2020). So fand die Optimierung unter Berücksichtigung humanzentrierter Faktoren bisher wenig Beachtung in der Literatur, was auch angesichts des zunehmenden Fachkräftemangels eine bedeutende Einschränkung darstellt (Destouet et al. 2023; Vollenkemper et al. 2023). Vor allem jedoch befasst sich ein Großteil der Studien mit eher abstrakten Optimierungsmodellen, welche eine reale Produktionsumgebung nur rudimentär abbilden. Infolgedessen sind viele Arbeiten algorithmische Benchmarkstudien: Ohne einen Fokus auf praktisch verwertbare Artefakte werden in der Tendenz stattdessen immer bessere Verfahren für abstrakte Probleme entwickelt. Dies führt zu einem systemischen Defizit, da bedeutende praktische Herausforderungen bezüglich komplexer Nebenbedingungen vernachlässigt werden. (Da Col und Teppan 2022; Mohan, Lanka und A. N. Rao 2019; J. Zhang et al. 2017; Romero-Silva, Santos und Hurtado-Hernández 2022)

Mit Blick auf die Industrie des Landes wird dieser Mangel im aktuellen Stand der

Technik noch deutlicher: Deutschland ist das Land mit den meisten mittelständischen Weltmarktführern. Es gibt mehr als 1.500 mittelständische Betriebe, die international in einem Marktsegment führend sind. Sie legen einen großen Wert auf hohe Qualität sowie schnelle Reaktionsfähigkeit und sie fokussieren oftmals eine humanzentrierte und kundenspezifische Produktion. Deshalb sind flexible Fertigungsorganisationen wie die Werkstattfertigung am häufigsten in den Unternehmen vorzufinden. Zur Sicherstellung der operativen Exzellenz, wird die Prozessoptimierung als wichtigstes Verbesserungsinstrument in der Produktion und Logistik erachtet. (Schmieder 2018) Unter der Annahme, dass Menschen nach wie vor eine entscheidende Rolle in Fertigungsprozessen spielen werden (Schmidtke et al. 2018), sind in diesem Zusammenhang eine robuste Planbarkeit dynamischer Abläufe sowie eine rasche Reaktionsfähigkeit wesentliche Anforderungen an moderne Logistiksysteme (Monostori 2018; Jafari et al. 2022).

Sofern die Hürden im Zusammenhang mit aktuellen APS-Systemen sowie die Unzulänglichkeiten auf Modell- und Verfahrensebene nicht effektiv korrigiert werden, sind die bestehenden Lösungen in mittelständischen Unternehmen mit flexibler Produktion nur sehr eingeschränkt einsatztauglich. Im Einzelnen führt der Mangel an geeigneten Lösungen zu ungenauen Planungen, ständigen manuellen Neu- bzw. Umplanungen, erhöhten Planungsaufwänden und damit zu ineffizienten Prozessen, Stress, Qualitätseinbußen und zwangsläufig erhöhten Kosten. Dies wiederum bedroht den Ressourcenvorteil und stellt im Zweifelsfall ein kritisches Wettbewerbsrisiko für die Unternehmen dar (Schmidtke et al. 2018). Zusammenfassend müssen demnach zunächst zwei essentielle Defizite adressiert werden, die bei der Entwicklung von smarten Dispositionssystemen bzw. APS-Systemen der nächsten Generation unbedingt berücksichtigt werden müssen:

1. In der Literatur befindliche Optimierungsmodelle bilden die Praxis in flexiblen humanzentrierten Produktionsumgebungen nur unzureichend ab. Es mangelt an einer ganzheitlichen Berücksichtigung von Abläufen, Nebenbedingungen und Zielsetzungen, wie sie in vielen Fertigungsunternehmen zu beobachten sind.
2. Verfahren der robusten Optimierung unter Unsicherheit sind ineffizient. Zur Ermittlung guter Lösungen sind auch in Näherungsverfahren viele ressourcenintensive Berechnungen erforderlich. Dies ist ein entscheidender Nachteil für eine feldsynchrone und echtzeitnahe Planung.

1.2 Begriffsbestimmungen und Konzeptualisierung

Nachfolgend werden wesentliche Grundlagen und zentrale Begriffe des Themas dieser Arbeit eingeführt und in einen übergeordneten Zusammenhang gebracht. Hiermit

werden zwei Ziele verfolgt:

1. Die Leser sollen eine angemessene Einführung erhalten, die die Nachvollziehbarkeit der präsentierten Publikationen gewährleistet. Aufgrund der Komplexität des interdisziplinären Themas können im Rahmen dieser Mantelschrift jedoch nicht alle theoretischen Grundlagen gänzlich in ihrer Tiefe behandelt werden. Für eine umfassendere Einarbeitung sei an dieser Stelle auf die in diesem Abschnitt referenzierte Fachliteratur verwiesen.
2. Zur Festlegung einer präzisen wissenschaftlichen Zielsetzung (siehe Abschnitt 1.3) müssen die Konzepte dieser Arbeit wohldefiniert werden.

1.2.1 Ablaufplanung dynamischer Fertigungsprozesse

Fertigungsprozess

Die Fertigung ist neben der Werkstoffbeschaffung, dem Werkstofftransport und der Lagerung von Werkstoffen ein Teilbereich der Produktion und somit Teil der betrieblichen Leistungserstellung. In Fertigungsprozessen werden die bereitgestellten Werkstoffe mit weiteren Produktionsfaktoren (Ressourcen) zielorientiert kombiniert und in Güter transformiert. Kombinierbare Produktionsfaktoren lassen sich grundlegend in Elementarfaktoren und dispositive Faktoren unterteilen. Erstere umfassen verbrauchbare Werkstoffe (z.B. Rohmaterialien oder Hilfsstoffe), gebrauchbare Betriebsmittel (z.B. Maschinen) und direkt wertschöpfende menschliche Arbeit. Zweitere sind für die Planung, Steuerung und Kontrolle eines effizienten Fertigungsprozesses erforderlich. So kann je nach Auffassung auch der Produktionsfaktor *Information* als wesentlicher Bestandteil der Planungsgrundlage hinzugezählt werden. (Lück 2004a; Wöhe, Döring und Brösel 2020)

Daraus abgeleitet sei im Kontext dieser Arbeit der **Fertigungsprozess** folgendermaßen definiert:

Organisiertes Vorgehen (im Sinne von zielorientierter Planung, Steuerung, Durchführung und Kontrolle) des Gebrauchs und Verbrauchs von Ressourcen zur Erzeugung neuer Güter.

Dynamischer Fertigungsprozess

Die Abläufe realer Fertigungsprozesse lassen sich aufgrund von Unsicherheiten wie schwankenden Bearbeitungszeiten und stochastischen Ereignissen wie Störungen nur sehr selten genau vorherbestimmen (Davenport, Gefflot und Beck 2001). Dies gilt besonders in hochflexiblen und humanzentrierten Umgebungen mit niedrigen Automatisierungsgraden. Darüber hinaus weisen Fertigungsprozesse oftmals mehrere alterna-

tive Materialflüsse bzw. flexibel zuweisbare Ressourcen auf. So kann es beispielsweise nicht nur möglich sein, einem Prozessschritt verschiedene Maschinen, Mitarbeiter (Kress, D. Müller und Nossack 2018) oder Rohmaterialien zuzuweisen, sondern auch Fertigungsprozesse situativ an andere Produktionsstandorte auszulagern (Mahmoodjanloo et al. 2020). Weiterhin kann eine ursprünglich vorgesehene Ressource zum Zeitpunkt der Ausführung nicht mehr verfügbar sein, z.B. durch Krankheitsfall des zugewiesenen Mitarbeiters. In diesem Fall ist eine Umplanung auf eine alternative Ressource erforderlich.

Daraus abgeleitet sei im Kontext dieser Arbeit der **dynamische Fertigungsprozess** folgendermaßen definiert:

Ein Fertigungsprozess, der (a) mit flexibel allozierbaren Ressourcen geplant und/oder gesteuert werden kann und/oder (b) dessen Durchführung hinsichtlich zeitlicher oder allokativer Unsicherheiten nicht exakt determiniert werden kann.

Ablaufplanung dynamischer Fertigungsprozesse

Die Planung von Fertigungsprozessen ist eine Teilaufgabe der PPS. Die PPS ist ein umfassendes Feld innerhalb der Betriebswirtschaftslehre, dessen vollständige Darlegung den Rahmen dieser Arbeit überschreiten würde. Jedoch existieren aus logistischer Sicht im Wesentlichen zwei Paradigmen, wie die PPS organisiert werden kann:

1. Dezentral organisierte Materialflüsse (Pull-Prinzip)
2. Zentral organisierte Materialflüsse (Push-Prinzip)

Bei der dezentralen Organisation liegt eine auftragsbezogene Produktion mit selbstorganisierten Regelkreisen sowie gegensätzlichen Material- und Informationsflüssen vor. Dies bedeutet, dass die Produktionsaufträge an die letzte Arbeitsstation in der Produktionskette gegeben werden. Wenn die Station die erforderlichen Materialien oder Zwischenerzeugnisse nicht verfügbar hat bzw. eine Mindestlagergrenze im Fertigungsprozess erreicht wird, wird eine Bestellung (Produktions- und ggf. Transportauftrag) an die vorgeschaltete Arbeitsstation gegeben (=rückwärtiger Informationsfluss). Bei der reinen zentralen Organisation existieren hingegen keine selbstorganisierten Regelkreise. Stattdessen werden die Produktionsaufträge in einem zentralisierten Planungssystem hinsichtlich ihrer Reihenfolge und Ressourcenzuweisung eingeplant, wonach sich die Arbeitsstationen dann richten müssen. (Hausladen 2020; Nicolai, Schotten und Much 1999)

Ein Instrument zur Einordnung und Untergliederung der PPS ist das *Aachener PPS-Modell*, welches sich auf drei Kernaufgaben stützt:

Tabelle 1.1: Kernaufgaben des Aachener PPS-Modells nach Schotten (1998)

Kernaufgabe	Fragestellung
Produktionsprogrammplanung	Welche Erzeugnisse müssen in welcher Periode in welchen Mengen produziert werden?
Produktionsbedarfsplanung	Welche Ressourcen werden dafür in welchen Mengen benötigt?
Fremdbezugs- und Eigenfertigungsplanung und -steuerung	Welche Produktionsschritte werden wann mit welchen Ressourcen ausgeführt?

Gemäß der in dieser Arbeit angesiedelten Definition von Fertigungsprozessen, bezieht sich die Planung von Fertigungsprozessen primär auf die dritte Kernaufgabe und dabei insbesondere auf die Eigenfertigungsplanung. Diese lässt sich weiter in folgende Teilschritte unterteilen:

Tabelle 1.2: Teilschritte zur Planung von Fertigungsprozessen nach Schotten (1998) und Meudt, Wonnemann und Metternich (2017)

Teilschritt	Fragestellung
Termin- und Kapazitätsplanung	Welche Produktionsaufträge müssen zu welchen Terminen abgeschlossen werden und welche Gebrauchsmittel werden für die Produktion zugewiesen?
Ablaufplanung	In welcher Reihenfolge werden die Produktionsaufträge mithilfe der zugewiesenen Gebrauchsmittel realisiert?
Auftragsfreigabe	Wann wird ein Auftrag für die Umsetzung tatsächlich freigegeben?
Auftragsüberwachung	Wie können Aufträge in Umsetzung auf Basis von Rückmeldungen kontrolliert und gesteuert werden?

Hierbei lassen sich die Termin- und Kapazitätsplanung sowie die Ablaufplanung der Planungsebene zurechnen. Die Auftragsfreigabe und -überwachung gehören demnach zur Steuerungsebene. Es sei angemerkt, dass die Anordnung dieser Teilschritte in der Literatur nicht explizit festgelegt ist, da Unternehmen dies individuell auf ihre speziellen Anforderungen abstimmen. Z.B. kann es in der Praxis sinnvoll sein, die Auftragsfreigabe entweder vor oder nach der Ablaufplanung anzusiedeln. Darüber hinaus existiert eine Vielzahl an Synonymen zu den Teilschritten. (Meudt, Wonnemann und Metternich 2017)

Daraus abgeleitet sei im Kontext dieser Arbeit die **Ablaufplanung dynamischer Fertigungsprozesse** folgendermaßen definiert:

Termin- und kapazitätsorientierte Ablaufplanung nach dem zentralen Paradigma, um laufende und bevorstehende dynamische Fertigungsprozesse

möglichst optimal im Sinne einer Zielvorstellung zu realisieren.

1.2.2 Anforderungen an eine feldsynchrone Ablaufplanung

Feldsynchronität sei im Kontext der Ablaufplanung definiert als die Abstimmung der eingesetzten Planungsverfahren mit den realen Fertigungsprozessen, mit dem Ziel, die Prozesse ganzheitlich und in Echtzeit zu überwachen und möglichst optimal zu steuern (Ghaleb, Zolfagharinia und Taghipour 2020). Demnach muss einerseits das *Feld* in Bezug auf das Planungsmodell umfassend abgebildet werden. Denn nur, wenn die entscheidenden Aspekte der realen Welt berücksichtigt werden, kann ein belastbarer Plan erzeugt werden. Andererseits müssen die Planungsvorgänge echtzeitnah mit den Realweltprozessen gekoppelt werden, da zu lange Abstimmungszeiten den Nutzen des Verfahrens negativ beeinflussen können (siehe Abschnitt 1.2.3). Dies gilt insbesondere für perspektivische smarte Technologien, die eine Ablaufplanung autonom durchführen können, indem sie sowohl vorausschauend als auch reaktiv operieren (siehe Abschnitt 1.1). Derartige Technologien sind im Kern immaterieller Natur, weswegen zweifelsfrei von Softwareprodukten gesprochen werden kann. Die Qualität von Softwareprodukten kann mittels der Norm ISO/IEC 25000 anhand funktionaler sowie nicht-funktionaler Kriterien gemessen werden (ISO 2005). Unter Berücksichtigung dieser Kriterien sowie den in Abschnitt 1.1 definierten Anforderungen und Verbesserungspotenzialen, können folgende wesentliche Qualitätsmerkmale bestimmt werden:

Tabelle 1.3: Qualitätsmerkmale für Verfahren der feldsynchrone Ablaufplanung. Deduktiv abgeleitet aus der ISO/IEC 25000 und nach Behrendt et al. (2019)

Kriterium	Bezug zur feldsynchrone Ablaufplanung
Effizienz	Die Verfahren müssen hinsichtlich ihrer benötigten Rechenzeit und des Ressourcenverbrauchs, bezogen auf ihren Nutzen und Einsatzzweck, effizient sein (siehe auch Abschnitt 1.2.3).
Reliabilität	Die Verfahren müssen zuverlässig und stabil ausführbar sein sowie valide Produktionspläne erzeugen.
Sicherheit	Die Verfahren müssen die Sicherheit und Integrität sensibler geschäftskritischer und personenbezogener Daten gewährleisten.

Übertragbarkeit und Skalierbarkeit	Die Verfahren müssen eine hohe Kompatibilität und Portierbarkeit in Bezug auf verschiedene IT-Infrastrukturen, Fertigungsorganisationen und unterschiedlich umfangreiche Planungsszenarien aufweisen.
Änderbarkeit	Die Verfahren sind anpassungsfähig und können hinsichtlich neuer Anforderungen weiterentwickelt und gewartet werden. Zudem ist die Implementierung verständlich strukturiert, flexibel modifizierbar und testbar.
Benutzbarkeit	Durch die transparente Gestaltung der Verfahren können sie benutzerfreundlich bedient und ausgewertet werden. Dies fördert eine hohe Gebrauchstauglichkeit, Verständlichkeit und Vertrauenswürdigkeit.
Funktionalität (holistische Planung)	Die Verfahren müssen alle relevanten Parameter, Ziele, Nebenbedingungen und Wechselwirkungen der realen Welt berücksichtigen. Für eine wirklichkeitsnahe Ablaufplanung muss sichergestellt werden, dass die Umsetzung eines Fertigungsauftrages zur richtigen Zeit am richtigen Ort bzw. mit der richtigen Ressourcenzuordnung realistisch eingeplant werden kann (Herlyn und Zadek 2020).
Funktionalität (prädiktive Planung)	Die Verfahren müssen eine vorausschauende und robuste Planung ermöglichen, in der Unsicherheiten und dynamische Ereignisse im Vorfeld berücksichtigt werden. Der erzeugte Plan sollte somit eine Struktur aufweisen, welche möglichst resilient gegenüber stochastischen Effekten ist. (Davenport, Gefflot und Beck 2001; Leon, V. Jorge, S. Wu und Storer 1994)
Funktionalität (adaptive Planung)	Die Verfahren müssen in der Lage sein, zeitnah auf ungeplante Ereignisse im Geschäftsprozess zu reagieren, die Effekte zu antizipieren und den Plan rasch in geeigneter Weise anzupassen (D. Gupta, Maravelias und Wassick 2016).

Die Qualitätsmerkmale können auf zwei Hauptaspekte komprimiert werden: die Prozessintegrität und die Modellqualität. Über die Prozessintegrität wird operationalisiert, wie gut sich die Verfahren in die bestehenden Geschäftsvorgänge sowie in die damit verbundene IT- und Unternehmensarchitektur implementieren lassen. Die

Modellqualität fokussiert hingegen primär funktionale Kriterien der Optimierungsverfahren. (Stüve et al. 2022) Je holistischer, adaptiver und prädiktiver die Verfahren planen können, desto ausgereifter ist ihre Funktionsweise im Kontext smarter bzw. autonomer Dispositionssysteme. Unter Berücksichtigung der vorangegangenen Definitionen stützt sich eine technologiebasierte **feldsynchrone Ablaufplanung dynamischer Fertigungsprozesse** somit auf gezielt orchestrierte datengetriebene Analyse- und Optimierungsmethoden, welche eine hohe Prozessintegrität sowie eine hohe Modellqualität aufweisen.

1.2.3 Effiziente Planung mit Optimierungsmethoden

Dieser Abschnitt beleuchtet wesentliche Herausforderungen bei der Entwicklung effizienter Optimierungsmethoden, die für eine feldsynchrone Planung erforderlich sind.

Grundlagen und Grenzen der exakten diskreten Optimierung

Damit Fertigungsprozesse, explizit bezogen auf ein oder mehrere Ziele, (nahezu) optimal geplant werden können (siehe Abschnitt 1.2.1), sind Optimierungsmodelle und -methoden erforderlich. Aus mathematischer Sicht des Operations Research bestehen Optimierungsmodelle aus einer Menge an Zielvorstellungen und aus einer Menge an Nebenbedingungen (Restriktionen). Die mit dem Modell einhergehende Optimierungsaufgabe besteht darin, aus einem durch die Restriktionen beschränkten Lösungsraum (erlaubter Bereich) sowie unter Berücksichtigung von Zielfunktionen, die bestmöglichen zulässigen Lösungen auszuwählen (siehe Abbildung 1.2). Zur Ausführung der Optimierungsaufgabe - und damit zur Lösung des Optimierungsproblems - werden Optimierungsmethoden (syn. -verfahren) wie Suchalgorithmen verwendet. (Lück 2004b)

Optimierungsmodelle der Ablaufplanung lassen oftmals sehr viele kombinatorische Möglichkeiten zu, wie Ressourcen und Auftragsreihenfolgen eingeplant werden können. Im Falle von sehr schwer lösbaren Optimierungsproblemen wird in der Komplexitätstheorie von sogenannten NP-vollständigen Problemen gesprochen. Dabei handelt es sich um Probleme, bei denen die Vergrößerung der Eingabe (z.B. zu verplanende Aufträge und Ressourcen) dazu führt, dass die benötigte Rechenzeit zur exakten Lösung des Problems exponentiell zunimmt. Das ist darauf zurückzuführen, dass Optimierungsmodelle der Ablaufplanung vor allem diskreter Natur sind und viele Variablen existieren, die nur ganzzahlige Werte annehmen können. (Acker 2011) So kann eine Maschinenbelegungsvariable beispielsweise lediglich binäre Werte annehmen (1 : Maschine belegt, 0 : Maschine nicht belegt), nicht jedoch etwa 0,7. Dies führt bei einer großen Eingabe zu einem erlaubten Bereich sehr hoher Dimensionalität

und zugleich geringer Dichte an zulässigen Lösungen aufgrund der Diskretisierung des Raumes. Der erlaubte Bereich muss dann im Zweifelsfall sehr lange durchsucht werden, bis eine optimale zulässige Lösung gefunden wird.

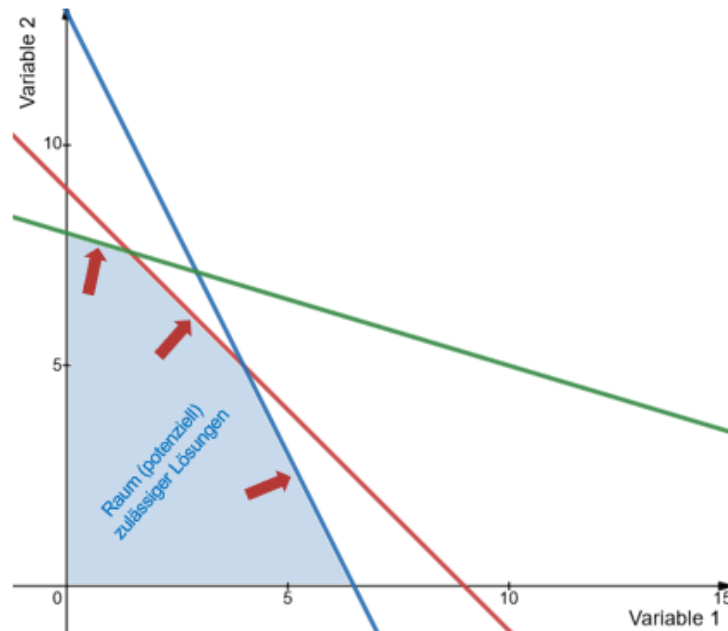


Abbildung 1.2: Erlaubter Bereich (Beschränkungspolyeder) eines linearen Optimierungsmodells (eigene Abbildung). Dargestellt werden drei Restriktionen (siehe Linearfunktionen), angewendet auf zwei Entscheidungsvariablen (siehe Abszisse und Ordinate). Der Bereich zulässiger Lösungen liegt innerhalb des Beschränkungspolyeders (blau-gefüllter Bereich). Der Rand des Polyeders ist mit roten Pfeilen markiert. Die Zielfunktion ist nicht eingezeichnet.

Zur Veranschaulichung der Komplexität sei ein Job Shop Scheduling Problem (JSSP) angeführt. Hierbei handelt es sich um ein Ablaufplanungsproblem für eine Werkstattfertigung, bei der n Aufträge ($I = \{1, \dots, n\}$) auf m Maschinen ($J = \{1, \dots, m\}$) verplant werden müssen. Dabei gelten folgende Restriktionen:

- Jeder Auftrag $i \in I$ hat m nacheinander abzuarbeitende Operationen, von denen jede auf einer verschiedenen Maschine umgesetzt werden muss ($O_{i,1}, \dots, O_{i,m}$);
- Die Reihenfolge, in der die Maschinen besucht werden müssen, kann für jeden Auftrag verschieden sein;
- Die Bearbeitungszeit p jeder Operation ist eine natürliche Zahl ($p_{i,j} \in \mathbb{N}_{>0}, \forall i \in I, \forall j \in J$);
- Operationen eines Auftrags können erst starten, wenn eine etwaige Vorgängeroperation des Auftrags abgeschlossen ist;
- Operationen können in der Bearbeitung nicht unterbrochen werden;
- Jede Maschine kann nur eine Operation pro Zeitpunkt ausführen;

- Maschinen sind ununterbrochen verfügbar (sofern sie nicht mit Aufträgen belegt sind).

Das Optimierungsziel für dieses diskrete Problem sei in diesem Fall die Minimierung der Zykluszeit, also die Minimierung des spätesten Operationsendzeitpunktes ($\min C_{max} = \min \max\{C_{i,j} : i \in I, j \in J\}$). Bei mehr als drei Aufträgen und mehr als fünf Maschinen herrscht eine Laufzeitkomplexität von $O(2^n \times L^n)$ vor, wobei L die Gesamtbearbeitungszeit des längsten Auftrags ist ($L = \max\{\sum_{j \in J} p_{i,j} : i \in I\}$). (Brucker, Sotskov und Werner 2007) Dies führt im Falle von $n = 20$ Aufträgen und bei einem umfangreichsten Auftrag mit $L = 4$ Zeiteinheiten bereits zu bis zu $1,15 \times 10^{18}$ Rechenschritten. So würde ein exhaustives Suchverfahren, welches jede Lösung ausrechnet und pro Berechnung eine Millisekunde benötigt, etwa 36 Millionen Jahre zur Identifizierung der optimalen Lösung benötigen. Infolgedessen wäre ein solches exaktes Lösungsverfahren nicht praxistauglich.

Grundlagen heuristischer Methoden

Moderne Softwaresysteme zur Lösung diskreter Optimierungsprobleme, sogenannte *Solver*, stellen ökonomischere, exakte Suchverfahren wie z.B. *Branch-and-Bound*-Algorithmen bereit. Jedoch benötigen selbst aktuelle Solver auf Hochleistungsrechnern übermäßig viel Zeit zur ganzheitlichen und globalen Optimierung umfangreicher Echtweltprobleme (Grumbach, Badr et al. 2023). Folglich sind exakte Optimierungsmethoden in der Praxis häufig nicht einsetzbar, weshalb viele Unternehmen auf Näherungsverfahren (Heuristiken) zurückgreifen. Ziel von Heuristiken ist es, auf systematische Weise, verhältnismäßig gute Lösungen in einer kurzen Rechenzeit zu finden. Aufgrund der Nicht-Durchführbarkeit eines Konvergenzbeweises können Heuristiken nicht garantieren, die beste Lösung (global optimale Lösung) zu finden. Demnach sind sie darauf beschränkt, lokale Optima im Lösungsraum zu identifizieren. Neben *einfachen* Heuristiken, wie z.B. Prioritätsregelverfahren oder der Netzplantechnik, gibt es sogenannte Verbesserungsverfahren, sowie Verfahren, die auf einer unvollständigen oder relaxierten exakten Suche unter Aufhebung der Ganzzahligkeitsbedingungen basieren. Erstgenannte Verfahren sind in der Praxis weitverbreitet, kommen ohne Optimierungsmodelle im weiteren Sinne aus, sind einfach zu implementieren sowie rasch in der Erzeugung von Lösungen. Jedoch ist die Qualität dieser Lösungen zumeist eher gering. (Acker 2011; Voß 2008)

Hingegen werden in der aktuellen wissenschaftlichen Literatur Verbesserungsverfahren wie Metaheuristiken intensiv behandelt. Dies sind mehrstufige Suchverfahren zur gezielten Erzeugung von Lösungen. Sie setzen Lernstrategien ein, um den Lösungsraum explorativ (erkundend) und exploitativ (erbeutend) zu durchsuchen, und um auf diese Weise effizient gute lokale Optima zu entdecken. Zwei wichtige Gattungen

sind populations- und trajektorienbasierte Methoden, die oftmals Naturphänomene modellieren. Populationsbasierte Verfahren basieren auf mehreren Lösungen, die im Suchprozess gezielt verändert werden (z.B. evolutionäre Strategien oder Schwarmalgorithmen). Trajektorienbasierte Methoden basieren hingegen auf einer einzelnen Startlösung, die im Laufe des Verfahrens sukzessive geändert wird (z.B. Tabusuche). Die meisten Metaheuristiken basieren zudem auf probabilistischen Konzepten, was bedeutet, dass Nachbarschaftslösungen durch zufällige Änderungen erzeugt werden. Das führt dazu, dass dasselbe Verfahren mit derselben Eingabe bei mehrmaligem Durchlauf zu unterschiedlichen Ergebnissen führen kann. (Acker 2011; Osman und Kelly 1996; Vofß 2008)

Besonders geeignet zur feldsynchronen Optimierung umfangreicher und unsicherheitsbehafteter Prozesse sind Simheuristiken (Negri et al. 2020). Simheuristiken integrieren ein Simulationsmodell (z.B. eine Materialflusssimulation einer Fertigungsumgebung) in eine übergeordnete Suchstrategie. So kann das Simulationsmodell dazu eingesetzt werden, die Qualität der von der Suchstrategie erzeugten Lösungen zu evaluieren (Fitnessberechnung). Das Simulationsmodell kann im Laufe der Zeit angepasst oder erweitert werden, ohne dass die implementierte Suchstrategie zwangsläufig mitangepasst werden muss. Darüber hinaus sind Simulationsmodelle im Vergleich zu umfangreichen mathematischen Formalisierungen intuitiver zu entwickeln. (Klemmt et al. 2009; Juan et al. 2015; Quintero-Araujo et al. 2016) Schlussfolgernd fügen sich effiziente Simheuristiken gut in das Paradigma sowie Einsatzspektrum der virtuellen Fabrik ein (siehe Abschnitt 1.1).

Beispiel einer Tabusuche zur Optimierung der Ablaufplanung

Für eine bessere Verständlichkeit sei nachfolgend die Tabusuche für das Permutation Flow Shop Scheduling Problem anhand des Konzepts von Ben-Daya und Al-Fawzan (1998) skizziert. Das Problem ist eine Spezialisierung des bereits eingeführten JSSP und modelliert ein Fertigungssystem mit einem einheitlichen und gekoppelten Materialfluss. Dies bedeutet, dass die Maschinen pro Auftrag stets in der gleichen Reihenfolge besucht werden und dass sich die Aufträge nicht gegenseitig in der Linie *überholen* können. Die Reihenfolgeentscheidung kann folglich nur an der ersten Maschine getroffen werden.

Die Tabusuche ist eine trajektorienbasierte Metaheuristik und sucht auf Basis einer initialen Startlösung, für welche Schritt für Schritt Nachbarschaftslösungen erzeugt und evaluiert werden, eine finale Lösung mit dem bestmöglichen lokalen Optimum. Abbildung 1.3 veranschaulicht dieses Prinzip: Die Startlösung wird in diesem Fall als Reihenfolgevektor repräsentiert. Die Elemente des Vektors repräsentieren jeweils einen Produktionsauftrag und der Vektor ist in der Reihenfolge sortiert, wie die

Aufträge in die Linie eingelastet werden. In der Evaluation des Vektors wird zunächst der Produktionsplan erzeugt, welcher z.B. als Gantt-Diagramm dargestellt werden kann. Anhand dessen lassen sich Metriken ermitteln, die dann in die Ziel- bzw. Fitnessfunktion (z.B. $\min C_{max}$) eingesetzt werden. Die Evaluation umfasst folglich die Fitnessbewertung eines Reihenfolgevektors. Durch Manipulation des Vektors (z.B. Tausch von zwei Vektorpositionen) wird eine Nachbarlösung im Raum der zulässigen Lösungen besucht. Diese kann nach der Evaluation eine andere Fitness zeigen.

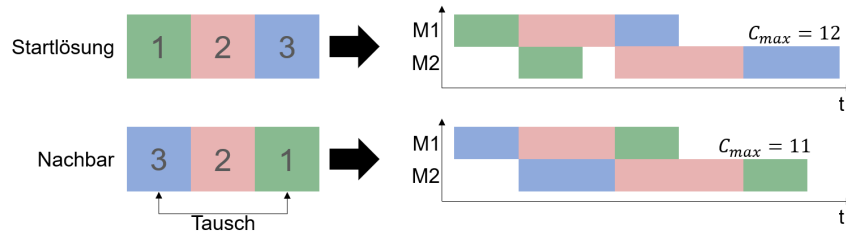


Abbildung 1.3: Erzeugung einer Nachbarschaftslösung in einem Permutation Flow Shop mit drei Produktionsaufträgen 1, 2, 3 und zwei Maschinen M1, M2 (eigene Darstellung). Die erzeugte Nachbarlösung hat aufgrund der geringeren Zykluszeit $C_{max} = 11$ eine bessere Fitness als die Startlösung.

Die Suchstrategie der Tabusuche modelliert ein Kurzzeitgedächtnis, in welchem bereits evaluierte Nachbarn vorgemerkt, aber auch wieder vergessen werden können. Dies zielt darauf ab, doppelte Evaluierungen für eine gewisse Zeit zu vermeiden und spart somit Rechenzeit. Der Vorgang des Vergessens unterstützt wiederum eine bessere Exploration, indem verhindert wird, in einem lokalen Optimum zu verhaften. Hat ein bereits gemerkter Nachbar eine verhältnismäßig geringe Distanz zu einem sehr guten lokalen Optimum, erhöht die erneute Berücksichtigung dieses Nachbarn die Wahrscheinlichkeit, dieses zu erreichen.

Im Wesentlichen kann die Tabusuche gemäß der Autoren folgenden Ablauf haben:

1. Starte mit der ersten Iteration $i \leftarrow 1$ und einem leeren Gedächtnis $M \leftarrow \emptyset$;
2. Erzeuge eine Startlösung x (z.B. mithilfe von Prioritätsregeln), die zugleich die aktuelle Lösung ist;
3. Erzeuge für x eine Menge N von n Nachbarn. Jeder Nachbar $c \in N$ ist zunächst eine Kopie von x ($c \leftarrow x$), für die dann zwei zufällige Positionen im Reihenfolgevektor vertauscht werden $c \leftarrow swap(c)$;
4. Verwerfe in N alle Nachbarn, die aktuell im Gedächtnis M sind $N \leftarrow N \setminus M$;
5. Evaluere alle übrigen Nachbarn und nehme sie ins Gedächtnis auf $M \leftarrow M \cup N$;
6. Selektiere die fitteste Lösung y aus allen $c \in N$;
7. Wenn y fitter ist als die aktuelle Lösung x , überschreibe x mit y als neue aktuelle Lösung $x \leftarrow y$;

8. Andernfalls lösche das Gedächtnis $M \leftarrow \emptyset$;
9. Inkrementiere die Iteration $i \leftarrow i + 1$;
10. Stoppe die Suche wenn die maximale Anzahl an Iterationen durchgeführt wurde. Andernfalls wiederhole ab Schritt 3.

Implikationen für eine effiziente Planung

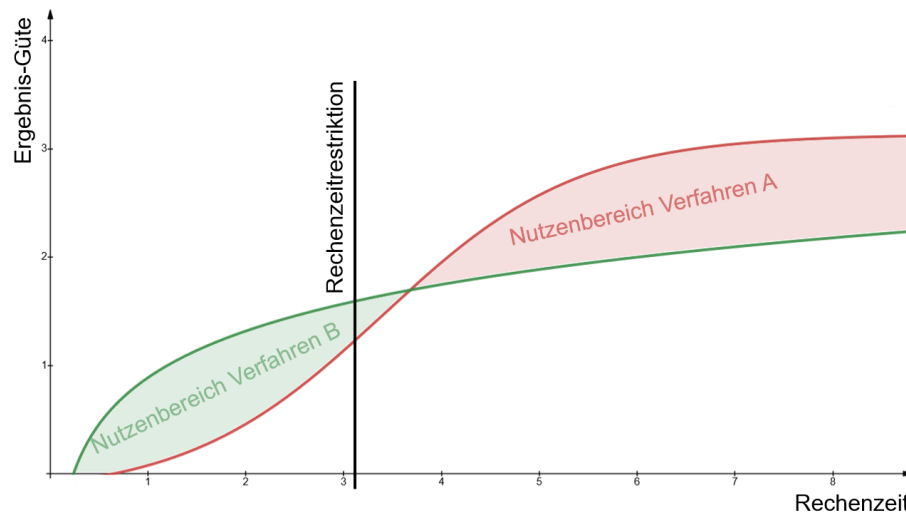


Abbildung 1.4: Nutzen von Optimierungsmethoden im Bezug zu einer Rechenzeitobergrenze (eigene Darstellung). Obwohl Verfahren A ein besseres Ergebnis als Verfahren B erzielen kann, benötigt es dazu mehr Rechenzeit. Aufgrund einer Zeitobergrenze (Rechenzeitrestriktion) ist Verfahren A dennoch unwirtschaftlich, da Verfahren B in kürzerer Zeit vergleichsweise bessere Ergebnisse erzielt.

Die Planung kann in der Praxis verschiedenen Anforderungen unterliegen. So ist es unter anderem denkbar, dass der Planungsvorgang bzw. die eingesetzte Optimierungsmethode einer Zeitobergrenze unterliegt. Beispielsweise ist es in einer zeitkritischen Situation nicht wirtschaftlich, ein Verfahren mit einer langen Rechenzeit anzuwenden. Erfordert ein Verfahren mehr Zeit als maximal beabsichtigt, kann es zu Verzögerungen im Prozess kommen, die wiederum die logistische Gesamtleistung beeinträchtigen und weitreichende Kosten erzeugen können. Demnach kann der Grenznutzen eines solchen Verfahrens rapide abnehmen, wenn eine Zeitrestriktion überschritten wird. Wird andererseits ein Verfahren gewählt, welches zwar eine angemessene Rechenzeit aber einen verhältnismäßig schlechten Plan hervorbringt, können Opportunitätskosten aufgrund ineffizienter Prozesse entstehen. Folglich müssen Verfahren unter Berücksichtigung einer Ressourcen- oder Rechenzeitobergrenze bzw. unter Berücksichtigung einer damit verbundenen Nutzenfunktion so ausgewählt werden, dass sie eine möglichst optimale Balance zwischen Ergebnisqualität und adäquatem Lösungsaufwand erzielen (siehe Abbildung 1.4). In anderen Worten ist die effiziente feldsynchrone Planung mit der Herausforderung verbunden, eine Methode

zu bestimmen, die einen Plan mit dem größtmöglichen situativen Nutzen verspricht.

1.2.4 Techniken des ML

In diesem Kapitel werden grundlegende Konzepte des ML vorgestellt. Ziel ist es, ein Verständnis der verschiedenen ML-Paradigmen sowie deren Anwendungen und Implementierungsmöglichkeiten zu vermitteln.

Theoretische Einführung und Definition von ML

Die Wissenschaft beschäftigt sich seit den 1950er Jahren intensiv mit dem Thema *Künstliche Intelligenz*. Hierbei wird erforscht, wie Maschinen rational und im Sinne einer menschenähnlichen Intelligenz ganzheitlich denken und handeln können. Eine wichtige Teildisziplin ist dabei das ML. (Russell und Norvig 2009) Dieses fußt auf zahlreichen Konzepten der (Inferenz-)Statistik, Wahrscheinlichkeitstheorie, mathematischen Optimierung, Informatik, Kybernetik, Neurowissenschaft, Psychologie und weiteren Disziplinen. In der technischen Umsetzung werden ML-Verfahren in Form von Computeralgorithmen implementiert, die eine große Anwendungsvielfalt in zahlreichen Domänen aufweisen. Im Allgemeinen sind diese Algorithmen in der Lage, Probleme zu lösen, Erfahrungen zu sammeln und auf Basis dessen ihre Problemlösefähigkeit selbst zu verbessern. Dies bedeutet, dass die Vorgehensweise der Problemlösung nicht explizit durch einen Programmierer vorgegeben werden muss, sondern durch die Algorithmen selbst gelernt, optimiert und in ihren Erfahrungsschatz aufgenommen wird. Da es sich um Computerprogramme handelt, nutzen sie eingegebene Daten für die Erfahrungsgewinnung und zur Lösung der Probleme. Die Erfahrung kann dabei explizit (symbolisch) gespeichert werden oder implizit mittels logischer Strukturen wie künstliche neuronale Netze (KNN) repräsentiert werden. (Domingos 2012; Mitchell 1997; Mjolsness und DeCoste 2001)

Wesentliche Funktionsweise von KNN

KNN sind mathematische Modelle, die an die Funktionsweise eines biologischen Gehirns angelehnt sind. Sie werden mithilfe von Eingabedaten trainiert, wobei sie ihre Funktionsweise selbst anhand von Gewichtungsfaktoren anpassen. Sie bestehen aus miteinander verbundenen Neuronen, die in Schichten aufgeteilt sind. Dabei gibt es klassischerweise eine Eingabe- und eine Ausgabeschicht. Jedes Neuron der Eingabeschicht repräsentiert dabei ein Datenfeld (Merkmal) des Eingabe-Datensatzes (Merkmalsvektor). Das KNN nähert die Lösung des Problems an, welche mittels der Neuronen in der Ausgabeschicht abgebildet wird. Dafür ist es erforderlich, dass jedes Neuron eine eigene Aktivierungsfunktion repräsentiert, die mittels aller eingehenden neuronalen Verbindungen (Reizsignale) sowie zugehörigen anpassbaren Gewichten

bestimmt, wie stark das Neuron und damit die ausgehenden Verbindungen aktiviert werden (Reaktion). Der Grad der Aktivierung wirkt sich dementsprechend wieder als Reizsignal auf das Verhalten aller nachgeschalteten Neuronen aus. Das heute vielfach eingesetzte tiefe ML (Deep Learning) ist dadurch definiert, dass es mehrschichtige KNN verwendet. Im Gegensatz zu *flachen* KNN, die nur über eine Ein- und Ausgabeschicht verfügen, besitzen tiefe KNN eine oder mehrere verborgene und miteinander verbundene Neuronenschichten zwischen der Ein- und Ausgabeschicht. Dies führt zu einer besseren Abstrahierung komplexer Zusammenhänge und damit letztlich zu einer umfassenderen Repräsentationsfähigkeit der gelernten Erfahrungen. Tiefe KNN mit nur einer verborgenen Schicht aus Neuronen mit nicht-linearen Aktivierungsfunktionen können bereits als universelle Funktionsapproximatoren eingesetzt werden. Dies bedeutet, dass die Lösung jeder mathematischen Funktion $f(x_1, \dots, x_n)$ erfolgreich angenähert werden kann. (Aggarwal 2018a; Goodfellow, Bengio und Courville 2018)

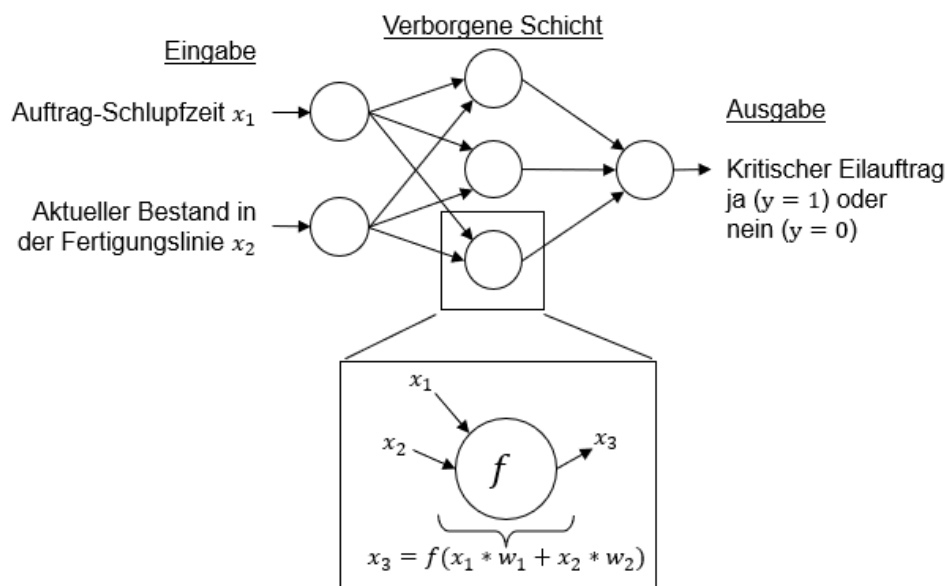


Abbildung 1.5: Einfaches Beispiel eines tiefen KNN (eigene Abbildung).

Abbildung 1.5 zeigt ein vereinfachtes Beispiel eines tiefen KNN für eine produktionslogistische Entscheidung: Das KNN entscheidet, ob ein Fertigungsauftrag als kritischer Eilauftrag klassifiziert werden soll. Auf Basis der Schlupfzeit (verbleibende Pufferzeit bis zum Soll-Fertigstellungstermin) und des insgesamt Bestandes (Anzahl Aufträge im System) wird eine entsprechende binäre Klassifizierung vorgenommen. Zur weiteren Veranschaulichung: Das untere Neuron der verborgenen Schicht hat zwei eingehende Verbindungen aus der Eingabeschicht und eine ausgehende Verbindung zur Ausgabeschicht. Die Aktivierungsfunktion berechnet den Ausgabewert x_3 auf Basis eines zusammengesetzten Arguments (summierte und mit w_1, w_2 gewichtete Eingaben x_1, x_2). Die Gewichte w_1, w_2 sind dabei die Variablen des Modells, die während des

Trainingsprozesses bei der Fehlerrückführung (*Backpropagation*) angepasst werden. Ziel dieser Optimierungsmethode ist es, die Gewichte so festzulegen, sodass die Problemlösefähigkeit des KNN maximiert wird (Aggarwal 2018a).

Einordnung von ML-Verfahren

ML-Verfahren können zunächst anhand der Art ihrer Problemlösefähigkeit bzw. ihrer Aufgabe eingeordnet werden. Hierbei existiert in der Literatur keine eindeutige Einteilung und Untergliederung, da einige Aufgaben nicht trennscharf separierbar sind. Deshalb werden nachfolgend einige wichtige Aufgaben skizziert, die häufig in der Praxis und Theorie aufzufinden sind:

Tabelle 1.4: Wesentliche Aufgaben von ML-Algorithmen gemäß Beekmann und Chamoni (2006), Kulin et al. (2021) und Sarker (2021)

Aufgabe	Erläuterung	Beispiel
Regression	Zuordnung eines Merkmalsvektors zu einer oder zu mehreren stetigen abhängigen Variablen.	Schätzung der realistischen Bearbeitungszeit eines Arbeitsgangs auf Basis des zugewiesenen Mitarbeiters, der Materialqualität, der Tageszeit usw.
Klassifizierung	Zuordnung eines Merkmalsvektors zu einer diskreten Zielklasse. Die möglichen Zielklassen stehen im Vorfeld fest.	Bestimmung eines Produktionauftrags als a) kritischer Eilauftrag oder b) weniger dringender Auftrag.
Clustering	Zuordnung einer Menge an Merkmalsvektoren in ähnliche, im Vorfeld unbekannte und während des Trainings zu bildende Klassen. Das Clustering kann z.B. zur Merkmalsreduktion eingesetzt werden.	Automatische Reduktion eines hochdimensionalen Artikelstamms auf wesentliche differenzierbare Artikeltypen.
Assoziationsanalyse	Erkennung von Abhängigkeiten zwischen Merkmalen und Ableitung von Wenn-dann-Regeln.	Extraktion von Kausaleffekten, die erklären, unter welchen Umständen ein Fertigstellungstermin wahrscheinlich überschritten wird.

Anomalie- erkennung	Erkennung abnormaler Strukturen, Inkonsistenzen oder Ausreißer im Zusammenhang mit einem Merkmalsvektor. Die Anomalieerkennung baut auf der Klassifizierung oder dem Clustering auf und kann z.B. dafür eingesetzt werden, fehlerhafte Datensätze aus dem Training auszuschließen.	Exkludierung fehlerhafter Buchungsdaten vor dem Training eines Regressionsmodells.
------------------------	--	--

Darüber hinaus können ML-Verfahren danach eingeordnet werden, wie sie lernen bzw. welches Feedback sie im Training erhalten:

Tabelle 1.5: Lernarten von ML-Algorithmen nach Kulin et al. (2021) und Sarker (2021)

Aufgabe	Erläuterung
Überwachtes Lernen (<i>Supervised Learning</i>)	Der Algorithmus lernt Schlussfolgerungen anhand eines vordefinierten und markierten (<i>gelabelten</i>) Datensatzes. Dieser enthält eine Menge von Merkmalsvektoren sowie die zugehörigen Zielwerte (<i>Labels</i>). Hiermit lernt der Algorithmus Hypothesen und die Fähigkeit, inferent Vorhersagen abzuleiten, welches Label zu welcher Eingabe passt (Beispiel: Klassifizierung oder Regression).
Unüberwachtes Lernen (<i>Unsupervised Learning</i>)	Der Algorithmus deckt Strukturen in einem nicht-gelabelten Datensatz auf. Zielwerte sind demnach nicht bekannt und werden nicht erschlossen (Beispiel: Clustering).

Teilüberwachtes Lernen (<i>Semi-supervised Learning</i>)	Dies ist eine Kombination aus den vorherigen beiden Ansätzen, welche häufig eingesetzt wird, wenn eine kleine Menge gelabelter und eine größere Menge nicht-gelabelter Daten verfügbar ist. Mit dem unüberwachten Lernen wird versucht, Strukturen und Beziehungen zwischen den Merkmalen und Zielwerten aufzudecken, um danach künstliche Labels zu erzeugen.
---	--

Bestärkendes Lernen (<i>Reinforcement Learning</i> , kurz: <i>RL</i>)	Hierbei lernen virtuelle Agenten eine Strategie, um sich optimal in einer Umgebung zu verhalten.
---	--

Weiterhin können Modelle danach eingeordnet werden, an welchen Zeitpunkten und in welchem Umfang das Training stattfindet. Offline-Lerner sind einerseits Modelle, die einmalig mit einer Datenmenge trainiert werden und deren Parameter sowie Repräsentationen nach dem Training statisch bleiben. Soll das Modell geändert oder erweitert werden, muss es vollständig neu trainiert werden. Online-Lerner trainieren hingegen sukzessive während ihres Einsatzes, wenn neue Datensätze eintreffen. Zudem gibt es aktive Lerner, die nicht auf Basis aller eingehenden Merkmalsvektoren lernen, sondern proaktiv eine repräsentative Teilmenge für ein effizientes Training auswählen. (Kulin et al. 2021; Sarker 2021)

Grundlagen des RL

RL ist ein wichtiges ML-Paradigma, welches auf verhaltenspsychologischen Konzepten wie der operanten Konditionierung nach Skinner (1937) basiert und zur Lösung von Optimierungsproblemen eingesetzt werden kann. Nach François-Lavet et al. (2018) und Morales und Zaragoza (2012) interagiert hierbei ein virtueller Agent mit einer Umgebung und wird anhand von Belohnungssignalen dazu gebracht, zweckdienliche Handlungen zu erlernen. Dabei abstrahiert der Agent mittels *Trial-&-Error* eine Strategie, welche Handlung in welcher Situation wahrscheinlich am besten geeignet ist, um größtmögliche Belohnungen bzw. geringstmögliche Bestrafungen zu erhalten. Wie im Falle klassischer Metaheuristiken (siehe Abschnitt 1.2.3), sind die Trial-and-Error-Prozesse nicht exhaustiv, sondern unterliegen einer Suchstrategie der systematischen Exploration und Exploitation. Das RL-Problem kann als zeitdiskreter, sequenzieller Markov-Entscheidungsprozess (Markov Decision Process, kurz: MDP) formuliert werden (siehe Abbildung 1.6). Im MDP kann die Umgebung pro Zeitpunkt einen konkreten Zustand aus einem endlichen Zustandsraum ($s \in S$) annehmen.

Der Agent beobachtet zu jedem Zeitpunkt t einen wahrnehmbaren Ausschnitt des jeweiligen Zustands $\omega_t \subset s_t$ und wählt auf Basis dessen entsprechende Aktion a_t aus einem endlichen Aktionsraum aus ($a \in A$). Der Einfachheit halber werden s und ω fortan gleichgesetzt. Die Aktion kann je nach Absicht bzw. Modus des Agenten unterschiedlich ausgewählt werden. So kann z.B. während des Trainings eine Aktion gewählt werden, die möglicherweise risikoreich ist, aber dafür der Exploration dienlich ist. Gemäß der sogenannten Markov-Eigenschaft wählt der Agent die Aktion a_t jedoch stets rein auf Grundlage des aktuellen Zustandes s_t aus und ignoriert eine etwaige Zustandshistorie. Nach Ausführung der Aktion in der Umgebung, erhält er ein Belohnungs-/Bestrafungssignal r_t sowie den neuen Zustand der Umgebung s_{t+1} . Die Wahrscheinlichkeit des Überganges eines Zustands s_t in einen neuen Zustand s_{t+1} unter Anwendung der Aktion a_t wird über die Transitionsfunktion $P(s_{t+1}|s_t, a_t)$ bestimmt. Im modellbasierten RL wird die Transitionsfunktion explizit vom Agenten gelernt, sodass er die Wechselwirkungen der Umgebung internalisiert. Im modellfreien RL wird die Funktion nicht gelernt. Stattdessen lernt der Agent lediglich, welche Aktion in welchem Zustand die größtmögliche Belohnung verspricht. Der genannte Ablauf wird so lange wiederholt, bis die Umgebung in ihrem Endzustand terminiert. Mithilfe (tiefer) KNN als Funktionsapproximatoren wird die erlernte Strategie, wie auf Basis eines eingegebenen Zustandes s eine geeignete Aktion a ausgewählt wird, gespeichert. s ist folglich der Merkmalsvektor, der auf der Eingabeschicht repräsentiert werden muss, wobei a mittels der Neuronen der Ausgabeschicht abzubilden ist. Die Aktion kann dabei diskret (Klassifizierungsproblem) oder stetig (Regressionsproblem) sein. (François-Lavet et al. 2018; Morales und Zaragoza 2012).

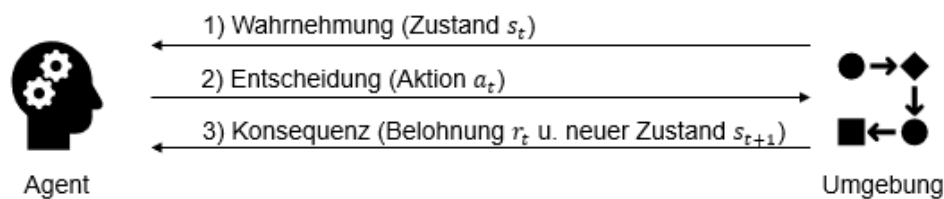


Abbildung 1.6: RL-Prozess (eigene Abbildung, angelehnt an François-Lavet et al. (2018)).

RL ist aufgrund der Einbettung in MDPs komplexer als überwachtes oder unüberwachtes Lernen. Beispielsweise ist es unter Berücksichtigung der Markov-Eigenschaft für den Agenten schwierig zu lernen, wie historische Aktionen die Höhe einer aktuellen Belohnung beeinflussen (*Credit Assignment Problem*). Weiterhin muss stets zwischen Exploration und Exploitation abgewogen werden. Wenn der Agent viel erkundet, lernt er zwar die Zusammenhänge der Umgebung besser zu verstehen und erschließt neue lokale Optima. Jedoch wird in diesem Fall die Ausbeutung lokaler Optima zur Erreichung höherer Belohnungen vernachlässigt. In hochkomplexen Umgebungen kann es zudem eine sehr große Menge verschiedener möglicher Zustände geben, auf

die ein Agent nach dem Training adäquat reagieren muss. Es ist in einigen Fällen unmöglich, alle Zustände im Training zu berücksichtigen, sodass der Agent möglichst gut abstrahieren muss. In diesem Zusammenhang ist es wichtig, dass der Agent auch in der realen Umgebung gute Entscheidungen treffen kann (*Reality Gap Problem*). Somit ist es erforderlich, dass einerseits genügend zugreifbare Daten vorhanden sind und andererseits die virtuelle Umgebung die Realität so wirklichkeitsnah wie möglich abbildet. (Aggarwal 2018b; François-Lavet et al. 2018)

In den vergangenen Jahren wurden bereits zahlreiche Untersuchungen zur Anwendung von RL auf deterministische sowie stochastische Produktionsplanungsprobleme durchgeführt (Kayhan und G. Yildiz 2021; Real Torres et al. 2022). Aufgrund der Separierung von Agent und Umgebung kann RL als Simheuristik umgesetzt werden, was in komplexen Planungssituationen sehr vorteilhaft ist. Beispielsweise kann ein Agent mithilfe einer umfassenden Materialflusssimulation (Umgebung) auf eine realistische Art und Weise lernen, welche Prioritätsregeln für welche Maschine, zu welchem Zeitpunkt und unter welchen Bedingungen am geeignetsten sind (Sakr et al. 2021). So ist es z.B. in der einen Situation ratsam, die Aufträge aufsteigend nach ihrem Soll-Liefertermin abzuarbeiten, wohingegen es in einer anderen Situation sinnvoll ist, nach kürzester Bearbeitungszeit zu sortieren. Weitere Einsatzmöglichkeiten werden in den Literaturreviews der vorgestellten Publikationen erörtert. Im Gegensatz zu laufzeitintensiven Metaheuristiken ist ein geeignet trainierter RL-Agent in der Lage, vergleichbar gute Ablaufpläne in wenigen Sekunden zu erzeugen, was ein Vorteil für die Feldsynchronität ist (Lang 2023b).

Entwicklung und Training von ML-Verfahren

Insbesondere Verfahren des überwachten und bestärkenden Lernens werden in der Praxis üblicherweise in einer mehrschrittigen Vorgehensweise entwickelt. In der Implementierung mündet dies in einer sogenannten ML-Pipeline, welche die erforderlichen (Vor-)Verarbeitungsschritte automatisiert. Nachdem das durch das ML-Verfahren zu lösende Problem definiert wurde, müssen Rohdaten für das Training gesammelt, bereitgestellt und aufbereitet werden. Ziel ist es, auf Basis dessen ein ML-Modell zu entwickeln, welches das definierte Problem so allgemeingültig und so genau wie möglich lösen kann. In anderen Worten: Nach dem Training muss das Modell so generalisiert sein, dass es präzise Vorhersagen auf Basis eines Merkmalsvektors geben kann. Zur Berechnung hochqualitativer Ausgaben, muss die Struktur der Eingabe ebenso hochqualitativ sein. Demnach ist es in der Aufbereitungsphase wichtig, eine gute Struktur des Merkmalsvektors zu definieren (*Feature Engineering*), Merkmale zu reduzieren, zu transformieren, zu normalisieren, hinsichtlich Ausreißern, Duplikaten und Inkonsistenzen zu bereinigen sowie eine i.d.R. möglichst gleichverteilte Menge an

Trainingsdaten bereitzustellen. Die Auswahl wesentlicher Merkmale ist entscheidend, da die Komplexität der Generalisierung exponentiell mit der Dimensionalität des Merkmalsvektors zunehmen kann. Im nächsten Schritt wird der für die Aufgabe geeignete Algorithmus ausgewählt, trainiert und evaluiert. Ferner wird eine taugliche Grundeinstellung des Algorithmus gesucht (z.B. zur Strukturdefinition des KNN), was wiederum mit einer Suchstrategie durchgeführt werden kann (*Hyperparameter Tuning*). Ziel ist es, eine Parametrisierung zu finden, bei der das Modell am besten generalisiert. In der Regel wird das durch statistische Verlust- oder Fehlerfunktionen gemessen, welche die Vorhersagegüte quantifizieren. Bei der Evaluation wird das mit Trainingsdaten trainierte Modell anhand einer Menge separater Testdaten ebenfalls anhand dieser Funktionen bewertet. Dies dient der Qualitätssicherung und soll sicherstellen, dass der Algorithmus sich nicht auf die Struktur der Trainingsdaten überangepasst hat (*Overfitting*). Über die Evaluation wird demnach sichergestellt, dass die Konzepte sowohl richtig als auch präzise gelernt wurden. Anders ausgedrückt: Sowohl die Varianz der Vorhersagen als auch etwaige Bias müssen minimiert werden. Erst nach erfolgreicher Evaluation wird das Modell operativ eingesetzt, überwacht und ggf. nachtrainiert. (Aggarwal 2018a; Domingos 2012; Kulin et al. 2021)

1.3 Zielsetzung und Forschungsdesign

Aufgrund des Mangels an realistischen Optimierungsmodellen und effizienten Optimierungsmethoden für smarte Dispositionssysteme (siehe Abschnitt 1.1), sowie auf Basis der dargelegten Begriffsdefinition und Konzeptualisierung des Themas (siehe Abschnitt 1.2), kann dieser Arbeit folgende Ausrichtung zugrunde gelegt werden:

Leitfrage: *Wie können digitale Technologien und insbesondere Techniken des ML im Zusammenhang mit realistischen Optimierungsmodellen und effizienten Optimierungsmethoden eingesetzt werden, um die feldsynchrone Ablaufplanung dynamischer Fertigungsprozesse zu verbessern?*

Die Arbeit verfolgt das Ziel, Artefakte (Modelle und Methoden) zu erarbeiten, die sich in das Framework der virtuellen Fabrik implementieren lassen und erfolgreich in dynamischen Produktionsumgebungen eingesetzt werden können. Die Artefakte müssen wissenschaftlichen Standards entsprechen sowie neuartig, allgemeingültig und übertragbar sein. Weiterhin ist es wichtig, die praktische und wissenschaftliche Relevanz der Artefakte sicherzustellen und den Entwicklungsprozess systematisch mit methodischer Strenge durchzuführen. (Robey und Markus 1998) Infolge der vielfältigen Einsatzmöglichkeiten von ML in der PPS behandeln die vorgestellten Publikationen jeweils ausgewählte Aspekte, die sich der Leitfrage unterordnen. Demnach unterliegt die Arbeit einer Reihe von Limitationen (siehe Abschnitt 3.2), aus denen

sich weitere Forschungs- und Entwicklungsbedarfe ableiten lassen. Die Mantelschrift nimmt hierbei die Rolle eines Rahmenwerkes für die vier Publikationen ein. Aufgrund der Vielseitigkeit des Themas konsolidiert und diskutiert sie die Ergebnisse sowie die wissenschaftlichen und praktischen Implikationen auf einer eher breiteren Ebene.

Teilfragen

Unter Berücksichtigung der hergeleiteten Qualitätskriterien für eine feldsynchrone Ablaufplanung (siehe Abschnitt 1.2.2) liegen die Schwerpunkte der Untersuchungen im Bereich Modellqualität und Prozessintegrität. Hinsichtlich der Modellqualität liegt der Forschungsschwerpunkt auf Verbesserungsmöglichkeiten für Verfahren der robusten Optimierung sowie für Verfahren der deterministischen Optimierung realistischer Prozesse. In Bezug auf die Prozessintegrität wird vorrangig untersucht, wie die Effizienz solcher Verfahren mithilfe von ML verbessert werden kann. Zusammenfassend behandeln die Publikationen folgende Teilfragen, welche erfolgreich in den Publikationen beantwortet werden konnten (siehe auch Abbildung 1.7):

1. Welche Anforderungen existieren an realistische Optimierungsmodelle für dynamische humanzentrierte Fertigungsprozesse mit komplexen Materialflüssen und wie können Simulationsmodelle möglichst allgemeingültig erzeugt werden?
2. Wie kann RL im Kontext einer Simheuristik dazu beitragen, die Lösung komplexer Optimierungsprobleme zu verbessern?
3. Wie kann ML eingesetzt werden, um rechenintensive stochastische Simulationen zu substituieren?
4. Wie kann RL eingesetzt werden, um robuste Produktionspläne zur Abfederung von Unsicherheiten zu erzeugen?

Vorgehensweise

Zur Beantwortung der Teilfragen und zur Erfüllung des Ziels einer wissenschaftlichen Artefaktentwicklung wurden in den Publikationen vorrangig quantitative und konstruktionsorientierte Forschungsdesigns gewählt. Pro Publikation wurde zunächst eine konzeptionelle und formale deduktive Analyse durchgeführt. Auf diese Weise konnten die Modelle und Methoden unter Berücksichtigung der aktuellen wissenschaftlichen Literatur sowie praktischer Anforderungen diskutiert, begründet und entwickelt werden. Daran anknüpfend wurden experimentelle Simulationsstudien durchgeführt, in welchen die entwickelten Methoden unter verschiedenen Aspekten evaluiert wurden. (Wilde und Hess 2007) So wurden beispielsweise verschiedene Algorithmenkonfigurationen in einem Benchmark miteinander verglichen, um Rückschlüsse auf unabhängige Variablen wie z.B. Hyperparameter zu ziehen. Unter Einbezug von synthetischen

Testdaten und unternehmensbezogenen Realweltdaten konnten die entwickelten Software-Prototypen evaluiert werden.

Die im Detail adressierten Forschungslücken sowie die eingesetzten Methoden werden in den einzelnen Publikationen vorgestellt. Dies bezieht sich auch auf die konkreten Ziele, (erwarteten) Ergebnisse sowie die eingesetzten Modellierungstechniken, Algorithmen, Technologien, Versuchsaufbauten und Mess-/Auswertungsinstrumente. Die verwendeten Daten und Implementierungen sind in Teilen öffentlich zugänglich (siehe Anhang *Forschungsdaten*).

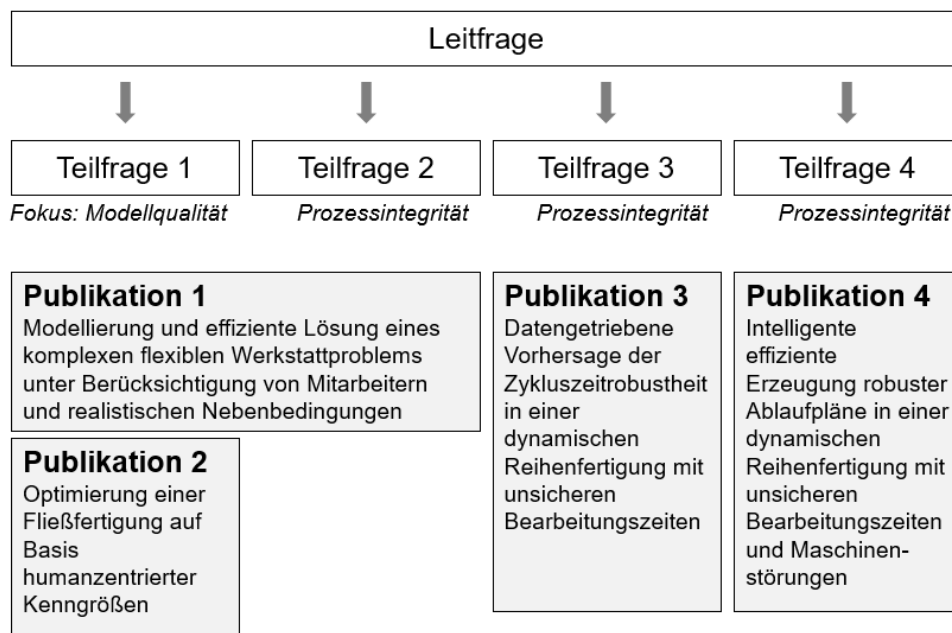


Abbildung 1.7: Übersicht der Publikationen im Kontext der Teilfragen

2. Publikationen

2.1 Publikation 1: A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling

Status

Veröffentlicht

Bibliographische Angabe

Grumbach, Felix, Nour Eldin Alaa Badr, Pascal Reusch und Sebastian Trojahn (2023). „A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling“. In: *IEEE Access* 11, S. 68760-68775. DOI: 10.1109/access.2023.3292548.

Received 4 June 2023, accepted 29 June 2023, date of publication 5 July 2023, date of current version 12 July 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3292548



RESEARCH ARTICLE

A Memetic Algorithm With Reinforcement Learning for Sociotechnical Production Scheduling

FELIX GRUMBACH¹, NOUR ELDIN ALAA BADR¹, PASCAL REUSCH¹,
AND SEBASTIAN TROJAHN²

¹Center for Applied Data Science (CfADS), Bielefeld University of Applied Sciences and Arts, 33330 Gütersloh, Germany

²Department of Economics, Anhalt University of Applied Sciences, 06406 Bernburg, Germany

Corresponding author: Felix Grumbach (felix.grumbach@hsbi.de)

This work was supported in part by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant 490988677, and in part by the Hochschule Bielefeld —University of Applied Sciences and Arts.

ABSTRACT The following interdisciplinary article presents a memetic algorithm with deep reinforcement learning (DRL) for solving practically oriented dual resource constrained flexible job shop scheduling problems (DRC-FJSSP). From research projects in industry, we recognize the need to consider flexible machines, flexible human workers, worker capabilities, setup and processing operations, material arrival times, complex job paths with parallel tasks for bill of material (BOM) manufacturing, sequence-dependent setup times and (partially) automated tasks in human-machine-collaboration. In recent years, there has been extensive research on metaheuristics and DRL techniques but focused on simple scheduling environments. However, there are few approaches combining metaheuristics and DRL to generate schedules more reliably and efficiently. In this paper, we first formulate a DRC-FJSSP to map complex industry requirements beyond traditional job shop models. Then, we propose a scheduling framework integrating a discrete event simulation (DES) for schedule evaluation, considering parallel computing and multicriteria optimization. Here, a memetic algorithm is enriched with DRL to improve sequencing and assignment decisions. Through numerical experiments with real-world production data, we confirm that the framework generates feasible schedules efficiently and reliably for a balanced optimization of makespan (MS) and total tardiness (TT). Utilizing DRL instead of random metaheuristic operations leads to better results in fewer algorithm iterations and outperforms traditional approaches in such complex environments.

INDEX TERMS Discrete event simulation, genetic algorithm, job scheduling, production planning, reinforcement learning, simheuristics.

I. INTRODUCTION

A. TRADITIONAL JOB SCHEDULING MODELS

Job scheduling is a traditional optimization problem that involves arranging activities in a specific order (sequencing), assigning them to resources with limited capacity, and optimizing one or more objectives [1]. This problem and its various subproblems, methods for solving them efficiently as well as applications, are still of significant scientific interest in several fields. One important subproblem is the job shop

scheduling problem (JSSP). Here, a set of jobs is scheduled on a set of machines. Each job consists of a sequence of operations and each operation must be processed on a specific machine for a specific processing time. The machine order can be different for each job, a machine can only process one operation at a time and operations can only start when the predecessor operation is finished. An operation must be processed exactly once and cannot be interrupted. Common optimization objectives for JSSPs are the minimization of MS, TT, flow time or any kind of manufacturing costs such as energy prices. Depending on the objective, the JSSP is an NP-hard sequencing problem [2]. The Flow Shop Scheduling

The associate editor coordinating the review of this manuscript and approving it for publication was Kuo-Ching Ying¹.

Problem (FSSP) is also a classic optimization problem. It is a specialization of the JSSP, with the difference that all jobs visit the machines in the same order [1]. On the other hand, the flexible JSSP (FJSSP) is a generalization of the JSSP: An operation can be processed on a set of alternative machines, but must be assigned to one machine, which can also affect the processing time [3]. Thus, the FJSSP is not only a job sequencing problem but also a resource assignment (synonym: allocation) problem. If two resource types can be scheduled (e.g. machines and human workers) in a flexible manner, then it is called dual resource constrained FJSSP (DRC-FJSSP) [4].

B. MOTIVATION AND SCHEDULING REQUIREMENTS IN PRACTICE

The study was motivated by a lack of efficient and holistic scheduling models for flexible and human-centered production and assembly processes, which are common in manufacturing companies. Considering requirements from our industrial partners, this is particularly relevant for sociotechnical, discontinuous manufacturing environments with high-mix low-volume products. The following aspects must be taken into account: Setup and processing operations can be performed by human workers with different capabilities, where the worker training level affects the operation feasibility and duration. In common practice, setup and processing operations are sometimes carried out by different workers. For example, a trained specialist sets up a machine and an assistant then operates it. Furthermore, workstations can have different characteristics: Some stations require setup operations (e.g. machinery and equipment) and others do not (e.g. assembly workplaces). The setup duration depends not only on the worker's capability but also on the current setup status (sequence-dependent setup time). For example, a machine must be extensively retooled if two very different components are sequentially processed. On the other hand, if two very similar components are manufactured one after the other on the machine, the setup only involves a small change of the machine's setting. In some cases, machines can perform work (partially) autonomously. These are, for example, computer-aided machine tools or industrial robots. In series production, it can often be observed that a worker interacts with several machines at the same time and only loads or re-adjust the machines. Some workstations have multiple slots that can be worked on at the same time (e.g. assembly workbenches). There are also company-external stations for outsourced work processes (extended workbenches). In this context, transport logistics tasks are also required. Moreover, sequential jobs in conventional models such as JSSP or FJSSP are insufficient for practical requirements on more complex structures. This circumstance becomes transparent in the course of the BOM explosion: Manufacturing multi-part products require highly structured jobs with dependencies and the possibility of parallelization. Given this fact, jobs should be mapped as directed acyclic graphs (DAG), which is

comparable with the resource-constrained project scheduling problem (RCPSP) nature [5].

C. METAHEURISTICS

Due to the ever more complex scheduling models, various heuristic approaches have been presented in the past years. This especially includes metaheuristics with population-based methods such as genetic algorithms (GA), methods with a single starting solution (trajectory methods) such as simulated annealing (SA) or hybrid methods such as memetic algorithms. A major benefit of using population-based metaheuristics like GA is the ability to obtain reliable results [6]. Another advantage of metaheuristics is that they can be implemented loosely coupled [7]. As a consequence, changes to the environment or to the evaluation function do not necessarily require changes to the optimization algorithm operations. Moreover, this can be combined with the paradigm of simheuristics, where a simulation model is integrated into an optimization method to generate or analyze solutions [8]. Simulation models are well suited to depicting complex scenarios and are therefore open to the expansion of further features and constraints [9]. Compared to mathematical models, they can be analyzed and developed more intuitively.

D. DRL

In recent years, DRL has increasingly attracted the interest of researchers. DRL is a machine learning technique for solving optimization problems with deep neural networks (DNN). Based on conditioning, a DRL agent learns a policy and how to act in a virtual environment in such a way to maximize rewards. The method can be summarized as a Markov Decision process: A DRL agent receives a time discrete environment state. Afterward, the agent has a set of actions available, that can be performed. Executing an action manipulates the environment state and leads to a direct or delayed distribution of a reward or punishment defined by a reward function. The DNN is used as a function approximator to predict promising actions for respective environment states [10], [11], [12]. DRL achieves high-quality results with very short computing times, provided the model is well trained [13]. However, under- or overfitted models lead to poor performance [14], [15]. Especially in realistic environments with demand and capacity fluctuations, novel situations, large product ranges and changing production goals, this can lead to suboptimal results. This may be one reason that most authors of pure DRL approaches focussing on simple environments such as JSSP or FJSSP.

E. CONTRIBUTION OF THIS WORK

From the current literature, we note a lack of holistic models combined with efficient algorithms for sociotechnical production scheduling evaluated with real-world data (see Section II for the complete definition of the research gap). To fulfill this gap, the paper provides the following contributions:

- 1) A mathematical model that maps the domain constraints and requirements mentioned (see Section III).
- 2) An efficient scheduling framework, where a DRL agent is injected into a GA to improve sequencing and assignment decisions (see Section IV). In this way, we combine the mentioned advantages of metaheuristics and DRL to obtain better and more reliable results.

With the scheduling framework, the following practically relevant aspects are considered:

- A generic and customizable DES is used by the optimization algorithm to generate and evaluate schedules.
- The framework supports parallel computing and can therefore be used scalably in a cloud.
- Objectives can be flexibly defined based on simulation metrics. In this work, MS and TT are exemplary balanced.

In Section V, numerical experiments including benchmarks with self-generated and real-world data are carried out and discussed. It was confirmed that DRL improves the performance of a memetic algorithm by obtaining better results in fewer iterations, which is promising for reactive real-time use.

II. RELATED WORK

The following section gives an overview of recent studies in the area of job scheduling models for flexible manufacturing processes and highlights research gaps. In particular, representative metaheuristics, simheuristics and DRL approaches for FJSSP and DRC-FJSSP have been considered.

A. SOLVING FJSSP

Winklehner and Hauder [16] reviewed the previous scientific literature and concluded that no previous work considered release dates, deadlines and sequence-dependent setup times in FJSSP, especially in real-world problems. Thenarasu et al. [17] proposed a solution for FJSSP using composite dispatching rules and multi-criteria decision making based priority rules integrated with DES. They tested the solution on real-world data. Zhang et al. [18] proposed a GA to solve an extended FJSSP with multiple sequence-dependencies for setup and transportation times. Moreover, three objectives have been considered: MS minimization, total setup time minimization, and transportation time minimization. Zhu and Zhou [19] examined solving a FJSSP with job precedence for MS minimization. In this way, complex job paths, as they arise in the production of BOM components, could be modeled. Depending on the structure of the input data, it is also possible to schedule parallel operations. A multi-micro-swarm leadership hierarchy-based optimization algorithm has been carried out to solve this problem. Lunardi and Voos [20] considered parallel operations within DAG job paths. The authors presented a GA and a firefly algorithm for MS minimization.

Defersha and Rooyani [21] evaluated the effects of using parallel computation on large FJSSP problem instances. They

imposed sequence-dependent setup times and used a GA to minimize the MS. It was shown that parallel computing significantly improves the computing time for population-based methods. Furthermore, Huang and Yang [22] offered important insights into using hybrid algorithms such as GA and SA for improving the results. Here, a multicriteria objective was used to balance MS, workload of critical machines, and total workloads of machines. Sequence-dependent transportation times were taken into account.

B. SOLVING DRC-FJSSP

In recent years, DRC-FJSSP has been considered more frequently to schedule additional resources. Wu et al. [23] considered human workers and their learning ability over time. They combined a GA with variable neighborhood search for MS minimization. Similar to this approach, Du et al. [24] presented a dual resource-constrained flexible flow shop scheduling problem with human workers and the consideration of human fatigue. They proposed a GA-based hybrid metaheuristic to optimize the MS. Defersha et al. [25] considered detached setup operations, thus they proposed DRC-FJSSP with machines and setup operators constraints. Furthermore, they took into account sequence-dependent setup times and workload balancing. They developed an SA algorithm to solve the problem. Liu et al. [26] investigated multi-worker assignments and parallel team scheduling for a product assembly line scheduling problem. A simultaneous optimization of MS and imbalance degree of team workload is taken into account, which has been applied to real-world data. Berti et al. [27] proposed a model, where the age of workers has an affect on their experience level, physical capacity and rest allowance. Furthermore, they demonstrated the importance of balancing intensive work and resting times, taking into account the age of workers. Dispatching rules were used for the scheduling approach: The shortest processing time (SPT) rule was applied for non-scheduled breaks to get a lower MS, while the longest processing time (LPT) rule was applied for scheduled breaks. Recently, there has been an increased emphasis on solving DRC-FJSSP with more complexity, such as multiple objectives and sequence-dependencies. Zhang and Jie [28] attempted to solve a DRC-FJSSP with multiple objectives by balancing MS and production costs (minimizing stocking of raw materials, machines, worker costs and tardiness penalties). Andrade-Pineda et al. [29] dealt with machine-related skills and suitabilities of workers. Moreover, dynamic due dates and multiple objectives such as MS minimization and weighted tardiness minimization are taken into account. Kress et al. [30] presented a DRC-FJSSP with flexible human workers in which sequence-dependent setup times are taken into account. Here, the machine setup state has an effect on the duration of the next setup operation.

C. APPLYING DRL

In the recent past, some scholars examined how DRL can be applied to scheduling problems such as the FJSSP:

Lang et al. [31] evaluated the effects of integrated DRL agents for assignment and sequencing decisions with a DES. The DES was used to execute the agent's decisions and to observe objectives such as MS or TT. Numerical experiments showed that DRL delivers high-quality results in near real time. Khuntiyaporn et al. [32] focused on solving FJSSP using DRL. They simulated each machine into two sets of machines with different processing times and costs to solve a single objective such as MS minimization. Popper et al. [33] applied DRL in FJSSP for multicriteria optimization. Multiple DRL agents were used to minimize MS, delayed products, lateness and energy costs. In order to examine a FJSSP with dynamic disturbances, such as new job insertion and machine disturbance, they tested their approach with experiments in a simulation. Luo et al. [34] proposed a hierarchical multiagent DRL algorithm named hierarchical multi-agent proximal policy optimization to select the best fitting dispatching rules for sequencing and assignment decisions dynamically. The algorithm aims to optimize total weighted tardiness, average machine utilization and variance of machine workload as multiple objectives. Zhou et al. [35] proposed a DRL smart scheduler that handles unexpected events to solve JSSP, besides they considered multicriteria optimization by optimizing composite reward functions. Du et al. [36] considered solving a FJSSP with setup operations, machine processing and transportation times to minimize both MS and total electricity costs. The problem was solved by applying an estimation of distribution algorithm for exploration while DQN was applied for better exploration. Zhu et al. [37] introduced a multi-agent DRL algorithm that solves DRC-FJSSP in real-time. Three agents are trained to perform three decision-making tasks: process planning, job sequencing, and machine selection. Moreover, different dispatching rules are set for each decision-making task to fulfill the requirements of the action space in the DRL algorithm. Furthermore, they considered DAG for defining the precedence relationships of operations.

D. COMBINING DRL AND METAHEURISTICS

Recent studies have also combined metaheuristic algorithms with DRL. Seyyedabbasi et al. [38] compared three metaheuristics models combined with DRL. They concluded that combining reinforcement learning with metaheuristics leads to a more stabilized performance in exploration and exploitation when compared to sole metaheuristic models. Thus, the model aims to discover additional areas and avoids being trapped in local optima, which enables the model to find better solutions. Kosanoglu et al. [39] combined a metaheuristic approach with DRL. They combined SA with a double deep Q network. In their approach, the initial solution for the SA model is not generated randomly but obtained from the best solution provided by the DRL algorithm. The best solution obtained from the SA model is provided as an initial solution for the DRL algorithm. They deduced that this hybrid algorithm obtained the optimal solution and is capable of

TABLE 1. Related work matrix of the reviewed literature. For a better overview, the studies are clustered according to their content. Includes the contribution of the current paper. (MRS: Multi resource scheduling, MCO: Multi criteria optimization, SD: Sequence-dependencies, PO: Parallel operations (=complex BOM job paths), Sim: Simulation/Simheuristic, RWD: Real-world data).

Study	MRS	MCO	SD	PO	Sim	DRL	RWD
[21]			•				
[18]			•				
[22]		•	•				
[25]		•	•				
[16]			•				•
[17]		•			•		•
[26]	•	•					•
[30]	•		•				•
[23]	•						
[27]	•						
[24]	•						
[28]	•	•					
[29]	•	•		•			
[37]	•			•		•	
[31]					•	•	
[32]		•			•	•	
[33]		•	•		•	•	
[36]		•	•			•	
[34]		•				•	
[35]		•				•	
[20]				•			
[19]				•			
<i>Contribution of this paper</i>	•	•	•	•	•	•	•

outperforming other metaheuristic algorithms without combining with DRL.

E. RESEARCH GAP

Based on the literature reviewed, a research gap can be identified: There is a lack of holistic mathematical models and approaches to generate schedules, that fully provide the introduced requirements [36], [40], [41]. Several recent studies have already examined and modeled partial aspects, but not combined them. Most authors used population-based or hybrid methods, which may indicate the good suitability of this class of algorithms. However, DRL-based approaches are still primarily restricted to simple environments. Further work is required to investigate the applicability of DRL, especially in the context of DRC-FJSSP and how to reliably generate schedules in this way. It should also be emphasized, that DES models and the evaluation with real-world data have been rarely utilized. Especially, due to the highly complex reality, modern process simulation tools allow more open and simple modeling compared to pure mathematical models. We consider it very relevant to separate a simulation from the optimization procedure as much as possible to allow for customizing and extensions. Moreover, we could only identify Defersha and Rooyani [21], who optimized the computing time through parallel computing. Computational efficiency is an important requirement for a time-sensitive reactive deployment with fast re-scheduling. Table 1 classifies the studies investigated and shows a clear gap in this context.

TABLE 2. Composed MILP notation.

Notation	Description
I	Set of all job indices
T_i	Set of all tasks indices of a job i
X	Set of all setup and processing operations (see Eq. 1)
K	Set of all station indices
W	Set of all worker indices
L	Matrix over all operation combinations ($L = X \times X$)
L_k	Matrix over all operation combinations, that can be processes on station k
$r_{i,j}$	Release time (raw material arrival time) of a task (i, j) . This just refers to the processing operation. Setup operations can start earlier.
$f_{i,i'}$	1, if job i succeeds job i' . Else 0
c_i	Job due date. Only possible for jobs without successor jobs.
l_i	Last task of a job (index)
$M_{i,j}$	Set of alternative stations for a task
$N_{i,j,s,k}$	Set of alternative workers for a processing or setup operation (i, j) on a station k (defined by capability matrix)
$O_{i,j,s}$	Set of alternative workers, that can execute a processing or setup operation (i, j, s)
P_k	Set of task tuples (i, j) that can be processed on station k
$Q_{i,j,s,w}$	Set of stations, where a worker w can execute a specific operation (i, j, s)
$d_{i,j,s,k,w}$	Setup or processing operation (i, j, s) duration by worker w on station k
$s_{i,j,i',j',k}$	Sequence-dependency factor $\in [-1, 0.5]$ to calculate the setup time, if task (i, j) precedes (i', j') directly on station k .
$u_{i,j,k}$	Automation degree $\in [0, 1]$ of a processing operation (i, j)
w_1, w_2	Objective weights $\in [0, 1]$ to balance MS (w_1) and TT (w_2) optimization
$\alpha_{i,j,s}$	Start time of an operation (variable)
$\beta_{i,j,s}$	Completion time an operation (variable)
σ_i	Tardiness of job i (variable)
$\gamma, \delta, \lambda, \psi$	Binary assignment and sequencing variables (see text)

III. PROBLEM FORMULATION

Considering the practical requirements introduced, we formulate the following composed mixed integer linear program (MILP) that goes beyond traditional scheduling models. It corresponds to a DRC-FJSSP with flexible stations, infinite station buffers, flexible human workers, different worker capabilities with an effect on setup and processing times, DAG job paths for BOM manufacturing, (partially) automated tasks, task release times for raw material arrivals, sequence-dependent setup times and early (left-shifted) setup operations. With early setup operations machines can be set up before materials are available, which are required for the processing operation. The MILP enables a simultaneous optimization of machine assignments, worker assignments and task sequences. Furthermore, setup operations with sequence-dependent setup times are also taken into account. This enables the optimization of short-term lot sizes by aggregating processing tasks with the same or similar setup operations on a station. To the best of our knowledge, this combination of constraints has not yet been modeled in the scientific literature. However, this is relevant to map the circumstances in medium-sized sociotechnical production processes as in the case with our industrial partners.

To fully depict the complex DAG paths, the model has a three-level job hierarchy: A job (denoted by i , 1st level) has several tasks (j , 2nd level) and a task has a setup and

processing operation (s , 3rd level). This is defined over the set X in Eq. 1, which contains all operation triples.

$$X = \left[(i, j, s) \mid i \in I, j \in T_i, s = \begin{cases} 1, & \text{if it is a setup operation} \\ 0, & \text{else} \end{cases} \right] \quad (1)$$

Table 2 provides the model parameters and the time variables $\alpha \in \mathbb{R}_{\geq 0}, \beta \in \mathbb{R}_{\geq 0}, \sigma \in \mathbb{R}_{\geq 0}$. In addition, four binary variables $\gamma, \delta, \lambda, \psi$ are used to assign and sequence the tasks. We know that the number of variables can be reduced, especially when the problem is modeled as a mixed integer non-linear program (MINLP). However, for better comprehension and readability of the constraints, we rely on the auxiliary variables in this section. γ is an assignment variable that determines if a task is allocated to a specific station from a set of alternative stations. The choice of station can have an impact on the processing or setup time. For example, a modern machine could be set up faster and also process faster.

$$\gamma_{i,j,k} = \begin{cases} 1, & \text{if the task } (i, j) \text{ is processed on station } k \\ 0, & \text{else} \end{cases} \quad (2)$$

Depending on which worker from a set of alternative workers carries out a setup or processing operation, this can also have an effect on processing or setup times. For example, an experienced operator could work faster than an apprentice. δ is a binary assignment variable to determine the worker to perform an operation on a station. With $d_{i,j,s,k,w}$, the required time to perform this operation can thus be derived.

$$\delta_{i,j,s,k,w} = \begin{cases} 1, & \text{if the operation } (i, j, s) \text{ is processed} \\ & \text{on station } k \text{ by worker } w \\ 0, & \text{else} \end{cases} \quad (3)$$

λ is a sequencing variable to define the structure of the task DAG (see Sect. IV-B). In contrast to most traditional models, an operation can have any number of predecessor and successor operations, which comes closer to realistic circumstances.

$$\lambda_{i,j,s,i',j',s'} = \begin{cases} 1, & \text{if the operation } (i, j, s) \\ & \text{precedes operation } (i', j', s') \\ 0, & \text{else} \end{cases} \quad (4)$$

ψ is a sequencing variable to determine the sequence-dependent setup time of an operation, taking into account the setup state of a machine and the worker assigned.

$$\psi_{i,j,s,i',j',s',k,w} = \begin{cases} 1, & \text{if operation } (i, j, s) \\ & \text{precedes operation } (i', j', s') \\ & \text{directly on station } k \text{ and if } (i', j', s') \\ & \text{is done by worker } w \\ 0, & \text{else} \end{cases} \quad (5)$$

Two objective functions are defined to minimize MS and TT synchronously:

$$\min C_{max} = \max_{(i,j,s) \in X} \{ \beta_{i,j,s} \} \quad (6)$$

$$\min T = \sum_{i \in I \mid c_i \text{ is set}} \sigma_i \quad (7)$$

The following constraints ensure that the tardiness is calculated correctly and always positive:

$$\sigma_i \geq \beta_{i,l_i,0} - c_i \quad \forall i \in I \mid c_i \text{ is set} \quad (8)$$

$$\sigma_i \geq 0 \quad \forall i \in I \mid c_i \text{ is set} \quad (9)$$

It must be ensured that processing operations can only start, when the assigned raw material is available:

$$\alpha_{i,j,0} \geq r_{i,j} \quad \forall i \in I, \forall j \in T_i \quad (10)$$

Moreover, it must be ensured that all processing and setup operations have the correct duration, considering the human worker assigned (Eq. 11) as well as the station and sequence-dependency (Eq. 12):

$$\beta_{i,j,0} = \sum_{k \in M_{i,j}} \sum_{w \in N_{i,j,0,k}} \delta_{i,j,0,k,w} \times d_{i,j,0,k,w} + \alpha_{i,j,0} \quad \forall i \in I, \forall j \in T_i \quad (11)$$

$$\begin{aligned} \beta_{i,j,1} = & \sum_{k \in M_{i,j}} \sum_{w \in N_{i,j,1,k}} \delta_{i,j,1,k,w} \times d_{i,j,1,k,w} \\ & + \sum_{k \in M_{i,j}} \sum_{w \in N_{i,j,1,k}} \sum_{i' \in I} \sum_{j' \in T_{i'}} \psi_{i',j',0,i,j,1,k,w} \\ & \times d_{i,j,1,k,w} \times s_{i',j',i,j,k} + \alpha_{i,j,1} \quad \forall i \in I, \forall j \in T_i \quad (12) \end{aligned}$$

The following constraints guarantee that all operations are assigned to valid stations (Eq. 13) and workers (Eq. 14). This includes the station, the setup worker and the processing worker.

$$\sum_{k \in M_{i,j}} \gamma_{i,j,k} = 1 \quad \forall i \in I, \forall j \in T_i \quad (13)$$

$$2\gamma_{i,j,k} = \sum_{w \in N_{i,j,1,k}} \delta_{i,j,1,k,w} + \sum_{w \in N_{i,j,0,k}} \delta_{i,j,0,k,w} \quad \forall i \in I, \forall j \in T_i, \forall k \in M_{i,j} \quad (14)$$

Furthermore, a human worker can only be utilized to a maximum of 100% at a time. Therefore, a worker can handle several partially automated operations in parallel. This is a classic case in practice when a machine operator monitors several machine tools at a time:

$$\begin{aligned} & \lambda_{i,j,s,i',j',s'} + \lambda_{i',j',s',i,j,s} + 1 \\ & - \sum_{k \in Q_{i',j',s',w}} \delta_{i',j',s',k,w} \times \begin{cases} u_{i',j',k}, & \text{if } s' = 0 \\ 1, & \text{else} \end{cases} \\ & \geq \sum_{k \in Q_{i,j,s,w}} \delta_{i,j,s,k,w} \times \begin{cases} u_{i,j,k}, & \text{if } s = 0 \\ 1, & \text{else} \end{cases} \\ & \quad \forall (i, j, s, i', j', s') \in L, \forall w \in O_{i,j,s} \cap O_{i',j',s'} \quad (15) \end{aligned}$$

Since stations can have multiple workplaces (slots), it must be ensured that a station can process only one operation per slot at a time:

$$\begin{aligned} \gamma_{i,j,k} \leq & \lambda_{i,j,s,i',j',s} + \lambda_{i',j',s',i,j,s} - \gamma_{i',j',k} + 1 \\ & \quad \forall (i, j, s, i', j', s') \in L, \forall k \in M_{i,j} \cap M_{i',j'} \quad (16) \end{aligned}$$

Eq. 17 ensures that a processing operation cannot start before the corresponding setup operation is completed. Moreover, processing operations of a job cannot overlap in time (Eq. 18). The processing operations of a job cannot start before all predecessor jobs have been completed (Eq. 19).

$$\alpha_{i,j,0} \geq \beta_{i,j,1} \quad \forall i \in I, \forall j \in T_i \quad (17)$$

$$\alpha_{i,j,0} \geq \beta_{i,j-1,0} \quad \forall i \in I, \forall j \in T_i \mid j > 1 \quad (18)$$

$$\alpha_{i,1,0} \geq \beta_{i',l_i,0} \quad \forall i, i' \in I \mid f_{i,i'} = 1 \quad (19)$$

A setup operation must be followed by the corresponding processing operation on the station. However, they do not have to follow one another immediately, so that left-shifted setups are possible even before the job starts or the material is available.

$$\begin{aligned} \gamma_{i,j,k} = & \sum_{w \in N_{i,j,0,k}} \psi_{i,j,1,i,j,0,k,w} \\ & \quad \forall i \in I, \forall j \in T_i, \forall k \in M_{i,j} \quad (20) \end{aligned}$$

The best found solution π is afterward selected from the Pareto front PF :

$$\pi = \min_{\pi' \in PF} Z_{\pi'} \quad (21)$$

Z is the linearly scalarized value of a solution taking into account freely definable weights w_1, w_2 :

$$Z = w_1 \times (C_{max})_{normalized} + w_2 \times T_{normalized} \quad (22)$$

IV. PROPOSED SCHEDULING FRAMEWORK

In this section, framework components are described successively.

A. SCHEDULING FRAMEWORK OVERVIEW

The framework provides a memetic algorithm integrating DRL and a DES to evaluate the schedules under the consideration of complex constraints. Figure 1 displays the main building blocks and their relationship within the scheduling dataflow. The memetic algorithm is based on a GA to control the overall optimization process and to provide a breadth-first search for assignment and sequencing decisions. In addition, a SA procedure is used to further exploit solutions within the GA search strategy (see Section IV-D). In order to evaluate the solutions created by the memetic algorithm operators, the framework integrates a process-based DES (see Section IV-E). Moreover, by integrating a properly trained DRL agent to the DES, sequencing and assignment decisions were further improved while simulating (see Section IV-F). To enhance performance in a scalable manner, the computations of the algorithm can be distributed across a freely

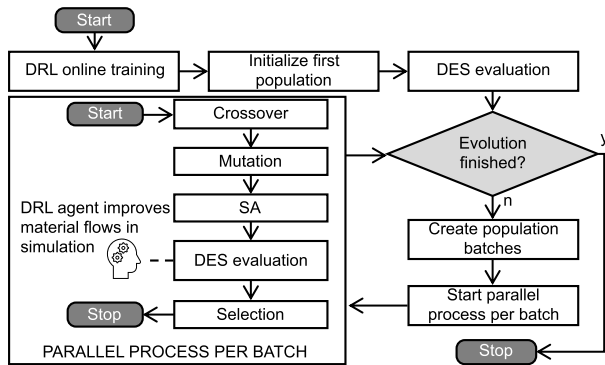


FIGURE 1. Scheduling framework components and dataflow (highlevel representation).

definable number of parallel processes within the computational environment.

The dataflow proceeds as follows: First, the DNN for the DRL agent is (re-)trained optionally to learn the specifics of the input data (see Section V-B). Then, an initial population of genomes with random assignment and sequencing decisions is created (see Section IV-C). The genomes are handed over to the simulation, improved by DRL and evaluated with regard to their fitness. The algorithm is then executed for a finite number of generations or evaluations (stopping criterion). In each generation, the current base population is partitioned into a number of batches according to the number of parallel computing processes. In this way, it is possible to distribute the computationally intensive simulation workload that takes place in each algorithm generation for each genome. Subsequently, the processes are spawned, each iteratively processing the assigned batch step by step. Within each parallel computing process, parent genomes from the batch are crossed over to a given probability. In addition, parent and child genomes can also mutate by a given probability. Afterward, the fittest individual per generation can be further improved by SA for a given probability. At this point it should be mentioned that SA can be exchanged with other trajectory optimization methods. After the simulation-based evaluation, the fittest solutions are selected, from which the next base population emerges. This process repeats until the stopping criterion is reached.

B. GENOME ENCODING

Due to the architecture of the memetic algorithm, we use a newly developed value encoding technique for the individuals, where a genome consists of two subgenomes to determine the schedule to be evaluated. As shown in Figure 2, a genome is splitted into a resource allocation subgenome and a dispatching subgenome. The resource allocation subgenome consists of resource allocation genes (represented as columns in the figure). Every gene corresponds to a task referring a unique task index and defines the assigned station and workers for the setup and processing operations, which are

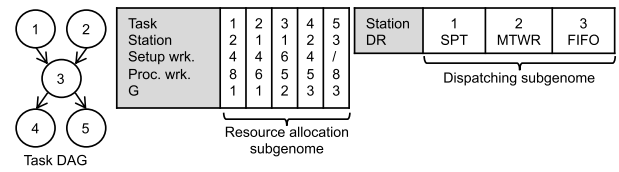


FIGURE 2. Value encoded genome for DAG job paths and to assign setup and processing workers to various stations. (wrk: worker, proc: Processing, DR: Dispatching Rule, SPT: Shortest Processing Time, MTWR: Most Total Work Remaining, FIFO: First In First Out).

represented by unique indices $k \in K$, $w \in W$ (see Table 2). Furthermore, task predecessor/successor relationships are defined by topology groups. A topology group $G \in \mathbb{N}_{>0}$ defines the positional group of a task in a topologically sorted task DAG. This is a required modeling technique to depict complex job paths for BOM manufacturing. The subset of task vertices from all tasks V without incoming edges is located in the first group G_1 . The subsequent groups $G_{2...n}$ can be determined recursively:

$$G_i = \{v \in V \setminus V^* \mid \forall p \in P_v : p \in V^*\} \quad (23)$$

V^* is the set of all visited vertices. P_v is the set of all direct predecessor vertices of a vertice v .

As an example from Figure 2, task 3 (topology group 2) is a successor to tasks of topology group 1 (tasks 1 and 2). Following the same concept, it is also a predecessor of all tasks from topology group 3 (tasks 4 and 5). The dispatching subgenome defines selected dispatching rules for stations, where each station is represented by a separate dispatching gene. Enabled by this value encoding technique, flow production systems can also be depicted in this way by fixing the respective station with FIFO. Therefore, this genome coding allows generic applicability to (permutation) flow shops, (flexible) job shops and dual resource flexible job shops according to the formulated model. During DES-based evaluation of genomes via schedule generation (see Section IV-E), either the preset dispatching subgenome is used, or a DRL agent selects the dispatching rules dynamically for each station (see Section IV-F).

C. INITIALIZATION

To create a first population as a suitable starting point for the algorithm, a specific initialization technique is implemented with the consideration of the proposed genome encoding. First, a random task DAG is generated for each individual, which ensures a valid topological order of the tasks taking into account the predecessor and successor constraints for BOM production (see Section IV-B). At this stage, the DAG is still independent of any resource allocations. Furthermore, the sequence in which the tasks are processed on a station is also not determined at this stage. Afterward, one gene is created in the resource allocation subgenome, for each node of the DAG. Thereafter, a station and one worker for the processing operation and another for the setup operation are selected from a set of available alternative resources

while taking the capability matrix into account (see Table 2). In this case, resources are not randomly selected, rather they are selected in a way that provides an evenly distributed workload. In the resource allocation subgenome, a random dispatching rule is assigned to each station. This determines the sequences of how the tasks are processed on the station. Further rules can be selected in addition to the rules listed in Section IV-F. The detailed procedure can be found in the publicly available source code.

D. OPERATORS

The following operations are sequentially applied per subgenome couple: First, a Job Order Crossover (JOX) operation (see [42]) is applied to the resource allocation subgenome. We implemented the operator as follows:

- 1) Randomly select two parent genomes to be crossed.
- 2) Randomly select 1 to $n - 1$ genes from the first parent, where n is the number of all genes.
- 3) Copy these genes to the same positions in the offspring genome.
- 4) Select the genes of remaining tasks from the second parent.
- 5) Copy these genes to the offspring genome in the order in which they appear in the parent.

The dispatching subgenome is inherited from the first parent. Then the mutation is performed, which includes two different options. With a probability of 50%, a resource flip is conducted for a random gene in the resource allocation subgenome. Here, an assigned station, setup worker and processing worker is replaced by a random valid alternative as defined by $M_{i,j}$ and $N_{i,j,s,k}$ (see Table 2). If the resource flip is not conducted, a dispatching rule flip is conducted for a randomly selected gene in the dispatching subgenome. Here, an assigned dispatching rule is replaced by a random valid alternative. In addition to crossover and mutation, a depth-first search is also possible as an additional operator. Within a trajectory method such as SA, a neighborhood search is carried out for the genome. The method receives the fittest genome of a generation and attempts to further optimize it step by step using the mutation operation.

E. DES MODEL FOR EVALUATION

To decode a genome and calculate its fitness, a DES with several event-based processes is utilized. These are implemented using the *SimPy* framework, which is a lightweight and efficient Python framework to develop DES models with the help of asynchronous generator functions (see [43]). Each station specified in the dispatching subgenome spawns its own station process (see Figure 3). Initially, when no task has yet been assigned, the station is in state *Free*. Then the station sorts its wait queue and pulls an available task (*Dispatching*). All tasks assigned according to the resource allocation subgenome are part of the station queue. For sorting, a dispatching rule is applied, as it is either defined in the dispatching subgenome or as the agent predicts it

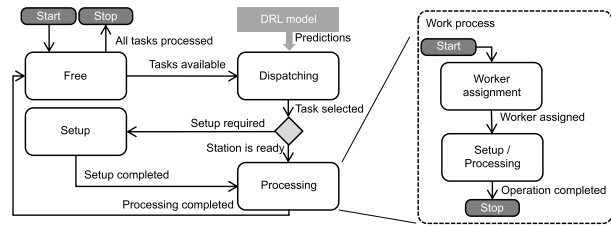


FIGURE 3. State machine for a station simulation process (simplified representation).

situationally (see Sec. IV-F). The next step is to decide whether the station needs to be set up or not. In both cases, a separate work process is started (*Setup* or *Processing*). Setup operations can already start independently of the job sequence and the readiness of the associated processing operation. A sequence-dependent set-up time is also taken into account. The worker defined in the assignment gene is awaited in the work process. After the processing operation is completed, the station changes back to the initial state of *Free*. When no more outstanding tasks are available, the station process ends. Different logistic metrics are measured during a simulation run. This includes, for example, MS, TT, flow time or wait times. A free selectable set of normalized metrics can be used in a weighted objective function to determine the fitness.

F. DRL IMPLEMENTATION

The DRL agent is injected into the simulation model and improves the genome during the decoding process. First, the DRL agent is responsible to improve sequencing decisions by dynamically sorting wait queues in front of each station according to an appropriate dispatching rule. Second, assignment decisions are improved by assigning tasks to another station or another worker. A dual-discrete action space was implemented for this purpose. It consists of two subsets AS_1, AS_2 from which one action is selected. AS_1 provides a set of selectable dispatching rules such as Slack Time Rule (STR). AS_2 offers actions to change task assignments after the dispatching rule has been applied. For example, the agent then recognizes, that two jobs with an early due date are competing for processing, the task with the second highest priority is shifted to another station to resolve the bottleneck (station flip *SF*). With a worker flip (*WF*), the task with the highest priority is shifted to another worker who has the capability to process the task. As a third option, the assignment is not changed (no flip *NF*). The state space was specified to be as generic as possible by applying wait queue metrics such as work in progress (WIP) and throughput rates from the law of [44] (see Appendix A).

$$AS_1 = \{SPT, LPT, MTWR, STR\} \quad (24)$$

$$AS_2 = \{SF, WF, NF\} \quad (25)$$

The reward is divided into an intermediate reward *IR* and a final reward *FR* at the episode end. The final reward is

calculated according to the overall multicriteria objective value achieved by the agent $fit_{achieved}$, which is subtracted from the sample label fit_{label} and multiplied by a large number considering the number of episode steps s . This ensures that the final reward is given a high priority. The IR is calculated after each step and is used for short-term conditioning by improving important key figures such as mean throughput rate or mean slack time remaining (see Appendix A).

$$FR = (fit_{label} - fit_{achieved}) \times 20s^2 \quad (26)$$

The DRL model was implemented in the Stable-Baselines3 framework (see [45]), which provides a standardized programming interface to implement DRL algorithms with PyTorch. Here, we implemented a proximal policy optimization (PPO) algorithm, which is a well-known and performant policy gradient algorithm with actor-critic DNN [46], that has often been used for various cases and benchmarks in recent literature. With some exceptions, such as the network architecture, the learning rate or the discount factor, we used the standard hyperparameters of the Stable-Baselines3 PPO implementation (see Appendix B). The training process and DRL injection to the memetic algorithm are described in Section V.

V. COMPUTATIONAL STUDY

In this section, the proposed scheduling framework is evaluated and discussed within numerical experiments. Since it is an applied research project and our industrial partner required equally weighted objectives that do not prioritize one metric, MS and TT have equal weights in our investigation. Thus, all experiments were performed for objective weights $w_1 = 0.5$ (MS) and $w_2 = 0.5$ (TT). Self-generated benchmark data and real-world data from our partner have been used in the experiments. The following questions were answered:

- 1) How runtime intensive is an exact method? (see Section V-A)
- 2) How can suitable test data be generated and what is the structure of the real-world data? (see Section V-A)
- 3) How successful is a DRL agent in learning a policy? (see Section V-B)
- 4) How good are the results of the framework compared to traditional methods? (see Section V-C)
- 5) Which runtime effects can arise with parallelization and what are the conclusions for a reactive use of the framework? (see Section V-D)

A. DATA GENERATION AND COMPLEXITY ESTIMATION

This section relates to the first two questions. First, the computational intensity of the exact method has been tested. We used an IBM CPLEX V12.8 solver in a high-performance environment (see Appendix B). It could be observed that the problem instance structure had a large impact on the computing time. Even with small inputs (e.g. 5 jobs), small variations could lead to an exorbitant increase in computing time. The complexity has increased especially when the job

TABLE 3. Benchmark datasets used (GBRT01-02: Synthetic data, GBRT03-08: Real-world data from our industry partner). GBRT03-08 global optimum could not be computed in a reasonable time. The global optimum is represented as MS/TT (both objectives considered synchronously with equal weights).

GBRT	Jobs/tasks	Stations/workers	Global optimum
01	6/14	2/2	728/83
02	3/10	6/3	325/75
03	112/251	20/24	?
04	61/89	13/23	?
05	61/144	12/23	?
06	94/188	20/24	?
07	73/143	15/24	?
08	84/168	15/23	?

structure or the resources have been varied. Consequently, the calculation process was aborted after several hours and it can be concluded that an exact method is not suitable to be utilized in practice. Even in a high-performance environment, the NP complexity is too high for solving large inputs with many jobs or resources. Since the JSSP is already NP-hard [2], an even greater complexity can be conjectured for the proposed problem.

Due to the novelty of the developed MILP and the need to assess different heuristic methods, it was necessary to generate dedicated test data with a suitable level of complexity. The test data was then validated using the CPLEX solver: Datasets that are too easy (too fast) to solve, as well as datasets that are too difficult to solve, were excluded. A calculation time of approximately 10 minutes to find the global optimum ensured that very efficient heuristics are challenged, while maintaining the feasibility of obtaining global optima or good local optima in a reasonable amount of time. Thus, this time bound represents a suitable balance between sufficient difficulty and solvability for the heuristics.

Two synthetic datasets with different focuses were generated: The first dataset *GBRT01* has a greater complexity in the job paths, but fewer assignable resources. On the other hand, the second dataset *GBRT02* has a low job structure but more operations and more allocable resources (see Figure 4). A multicriterial global optimum was found for both datasets (see Table 3). Furthermore, we used feasibly solvable real-world datasets *GBRT03-08* based on typical planning situations from our industrial partner. Nevertheless, due to the job and resource structure, less complexity is given. For example, the worker skills are clearly defined by a capability matrix and not every worker can work at every station. In addition, there are not that many flexible stations. With the real-world data, one time unit in the scheduling method corresponds to 30 minutes in reality. However, a global optimum could not be computed. The exact computing time is unclear.

B. DRL TRAINING

This section relates to question 3: It was examined how the DRL agent behaves during training and how it learns a policy in order to make situationally appropriate sequencing or

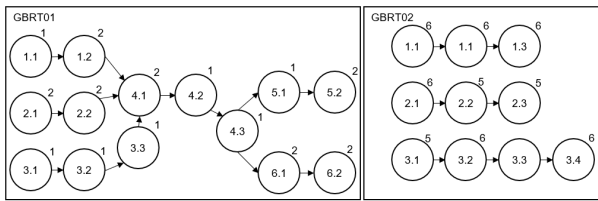


FIGURE 4. GBRT01-02 job paths visualized as DAGs. GBRT01 consists of 6 highly structured jobs with 2-3 tasks each. Job 1 task 1 (1.1) can be assigned to one machine, while the successor task 1.2 can be flexibly assigned to two machines. GBRT02 consists of 3 jobs with 3-4 tasks each but with a lower DAG structure. However, there are more flexible stations to choose from. GBRT03-05 cannot be displayed due to the large number of planning objects. More details about the data structures can be seen in the public code.

assignment decisions. The training process had the following steps: First, a starting solution for an input problem instance was provided as a sample. It can be generated, for example, using a shortened form of the memetic algorithm (e.g. 10 generations). Due to the specialization in one problem instance, the training could be carried out in seconds to a few minutes. Since we do not claim to develop a general agent for all kinds of inputs, temporary overfitting to a specific scenario was accepted as an inherent aspect of this online learning method. In the subsequent training, the DES was used as the environment with which the agent interacts. The agent starts the simulation and waits until a decision has to be made. Looking at Figure 3, a decision situation arises when a station's queue is to be sorted. In this case, the simulation pauses and the current state is given to the agent.

Based on the state, the agent now determines a dispatching rule and if a resource flip has to be carried out, which activates the simulation again. The DRL hyperparameters used can be found in Appendix B. Figure 5 shows the results obtained from a PPO training processes using the example of GBRT03. During training, a clear trend of decreasing loss and increasing reward emerged. In the end, both metrics eventually converged, which is an indicator of successful training and reaching a stable policy. The overall negative end reward is a consequence of exploration that also took place late in the training. Due to the quadratic function for calculating the final reward, the agent receives high penalties in a few cases. Nevertheless, the figures indicate that the DRL agent has learned an appropriate policy to apply sequencing and assignment decisions for the DRC-FJSSP. The next section analyzes how the agent can be injected into the higher-level metaheuristic to improve its results. This also includes conclusions about how the learned policy is applied.

C. ALGORITHM BENCHMARK

In this experiment, the scheduling framework was analyzed in terms of scheduling performance (see question 4). The schedules generated with the conventional approaches were compared with those of the GASA+RL approach, where the trained actor critic DNN is injected to the memetic algorithm.

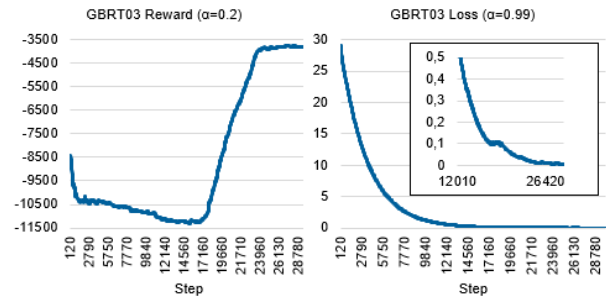


FIGURE 5. PPO agent loss and reward exemplary for GBRT03 (30,000 training steps). Similar patterns could be observed for GBRT01 and GBRT02.

1) EXPERIMENT SETUP

For representative benchmarks, we used a modern GA with adaptive hyperparameters as a basis [47]. The hyperparameters can be found in Appendix B. For GASA, we utilized an implementation of SA with restart strategy (SARS) based on Yu et al. [48]. Moreover, standalone SARS, the well known Tabu Search (TS) method and the dispatching rules *Most Total Work Remaining (MTWR)* and *Slack Time Rule (STR)* were tested. When using the dispatching rules for sequence decisions, the tasks were distributed evenly over the resources. Due to various hardware-related runtime effects such as efficiency advantages through parallelization, the metaheuristics were compared based on their actual complexity. For better comparability, the number of operations for creating and evaluating neighborhood solutions was used instead of the physical runtime. All metaheuristics have a runtime complexity of $O(n)$ where n is the number of operations and evaluations until the stopping criterion is reached. $n = 500$ was set for all algorithms so that their logical performance could be compared. However, Sect. V-D deals with further runtime optimization that is possible through parallelization. An agent was trained for each of GBRT01, GBRT02, and GBRT03. No separate agent was trained for the additional real-world datasets. Instead, the agent from GBRT03 was used, as only the input changed and not the fundamental scenario. Consequently, the experiments for GBRT04-08 also examine the generalizability of the agent.

2) RESULTS AND DISCUSSION

Table 4 and Table 5 show the summary statistics for multicriteria objectives achieved considering mean values and variability. The results indicate that dispatching rules have a poor performance but are easy and quick to compute. Nevertheless, the rules are suitable for generating an initial feasible solution for subsequent search methods. Utilizing the trajectory methods TS and SARS resulted in significantly better outcomes, with SA marginally outperforming TS. Moreover, even better local optima were discovered using population-based approaches (GA, GASA, GASA+RL). This can be explained by the fact that population-based methods have a better ability to explore the breadth of the solution

TABLE 4. Achieved MS and TT mean values for datasets GBRT01-08 by the different heuristics STR, MTWR, TS, SARS, GA, GASA and GASA+RL. The proposed approach GASA+RL outperforms traditional methods in most metrics. All metaheuristics have the same logical runtime (500 operations and evaluations to create neighbor solutions). The physical runtime may vary depending on the dataset and due to various hardware effects (approx. 30-150 seconds).

Dataset (GBRT)	MS / TT						
	STR	MTWR	TS	SARS	GA	GASA	GASA+RL
01	1076.3 / 436.2	1091.8 / 519.9	774.9 / 191.7	777.5 / 179.9	748.7 / 190.8	735.1 / 180.8	730.0 / 172.2
02	438.6 / 346.6	434.0 / 333.3	340.1 / 113.8	336.2 / 112.8	336.8 / 92.3	335.0 / 89.1	333.0 / 83.7
03	66.9 / 15017.7	231.1 / 16449.4	203.6 / 11299.9	196.2 / 10723.0	188.1 / 10753.1	186.3 / 10538.7	189.8 / 10199.9
04	53.6 / 3209.6	52.95 / 3336.9	52.25 / 2727.5	52.35 / 2761.6	52.0 / 2742.8	52.0 / 2740.9	52.0 / 2715.7
05	107.6 / 4852.8	112.2 / 5378.4	93.6 / 4000.5	93.5 / 4014.9	94.3 / 4030.5	93.9 / 4002.8	93.6 / 3947.35
06	241.2 / 9612.8	212.9 / 10704.7	179.5 / 7268.9	175.7 / 7380.3	175.2 / 7212.8	175.1 / 7282.7	174.6 / 7020.1
07	145.9 / 6358.2	139.7 / 6894.8	122.9 / 4678.2	117.3 / 4769.1	115.4 / 4746.3	114.7 / 4695.9	114.2 / 4668.1
08	141.8 / 8040.4	135.4 / 8716.6	115.8 / 6303.7	115.7 / 6288.5	115.0 / 6244.0	115.5 / 6169.9	114.7 / 6162.8

TABLE 5. Achieved MS and TT standard deviations (addition to Table 4).

Dataset (GBRT)	MS / TT						
	STR	MTWR	TS	SARS	GA	GASA	GASA+RL
01	95.9 / 89.5	91.1 / 95.1	44.9 / 43.6	45.8 / 24.6	19.2 / 19.3	19.6 / 23.6	17.2 / 17.8
02	57.7 / 120.2	56.6 / 125.8	8.4 / 31.7	9.2 / 40.5	8.8 / 11.7	6.9 / 12.7	5.0 / 5.3
03	41.5 / 891.2	33.4 / 1142.5	19.3 / 902.1	10.3 / 814.2	2.3 / 288.7	2.7 / 320.6	2.5 / 332.5
04	3.5 / 55.3	1.9 / 54.9	0.9 / 47.5	0.7 / 51.2	0.0 / 32.6	0.0 / 28.5	0.0 / 12.3
05	6.6 / 152.1	11.0 / 222.9	1.7 / 166.0	1.6 / 170.3	2.4 / 127.1	1.8 / 126.8	1.6 / 69.9
06	40.1 / 643.2	37.4 / 805.8	16.6 / 426.7	4.8 / 299.3	2.7 / 336.7	2.5 / 401.4	1.9 / 290.7
07	15.7 / 450.9	19.5 / 528.6	15.9 / 271.5	7.5 / 146.8	4.5 / 147.3	5.5 / 113.9	4.5 / 191.1
08	14.3 / 364.9	14.4 / 406.5	2.5 / 427.7	3.4 / 262.5	2.2 / 270.5	2.7 / 293.6	1.3 / 227.0

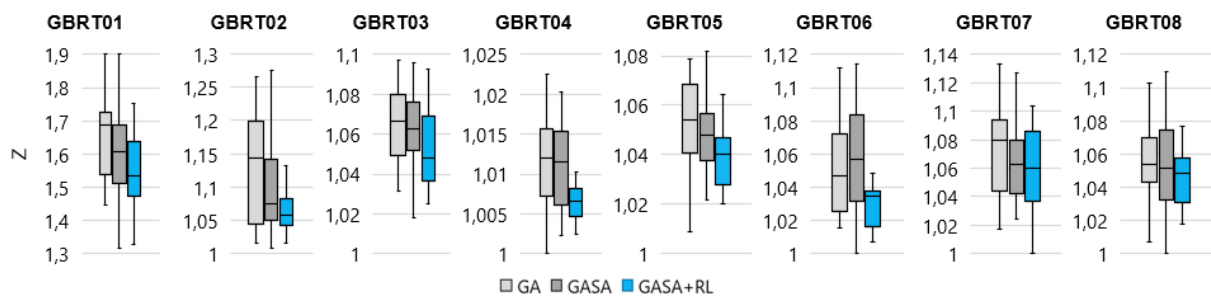


FIGURE 6. Result comparison between GA, GASA and GASA+RL for GBRT01-08 considering the achieved multicriterial objective value Z (see Eq. 22). On average, GASA+RL generates the best schedules and clearly outperforms GA and GASA.

space. Consequently, due to the many possible combinations in the optimization model and input data as well as by the high standard deviations, the solution space appears to consist of many local optima.

For a better overview of the population-based methods, Figure 6 displays related statistical metrics in form of a box chart considering the scalarized objective value Z (see Eq. 22). The first remarkable aspect of the results is the different performances on GBRT01 and GBRT02. While for GBRT02 consistently good fitness values near to the global optimum were achieved, the pattern did not appear for GBRT01. Even the best results still had a distance of more than 30% to the global optimum. Although relatively good MS values were achieved, this resulted in a deterioration in TT. Overall, these results indicate that the proposed memetic algorithm is well suited to schedule production processes with many flexible resources. With complex job paths, it is difficult to find good results quickly. This applies

in particular to the TT minimization. A possible explanation for this issue might be that setup operations can be left shifted and start before the processing operation or even before the job is ready. In the case of non-sequential job paths with many predecessor/successor relationships, few resources and sequence-dependent setup times, it can thus be suggested that it is a difficult decision when to set up a machine in advance.

What is also striking about the table and figures is the dominance of GASA over GA and GASA+RL over GASA in most cases. The average superiority of GASA over GA was to be expected since similar outcomes had already been shown in many other studies. However, this confirms the validity of the memetic algorithm implemented and the hyperparameters chosen. Interestingly, the GASA results had a comparable high standard deviation, which is especially observable in the TT case. A possible explanation for this might be that GASA sometimes finds better schedules due to greater exploitation

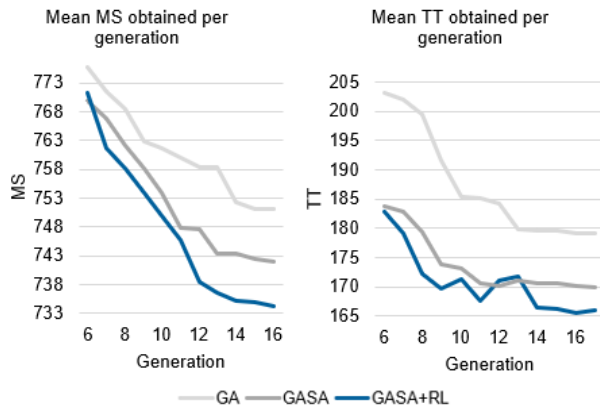


FIGURE 7. Comparison between GA, GASA and GASA+RL in early iterations considering MS and TT. GASA+RL outperforms conventional approaches and identifies better local optima early in the process (exemplary for GBRT01).

through SARS, which is also reflected in the minimum values (see outliers in the box chart).

The dominance of GASA+RL is the most interesting result. Regarding the Z score visualized in the box chart, better average, maximum, Q1 and Q3 values could be achieved in most cases. Especially, the model's predictions may prevent random upward outliers and result in comparatively low Q3 values. This is an important finding and an indicator of a good reliability with relatively short runtimes. Looking at Table 4, it is noteworthy that GASA+RL achieves significantly better TT values for the real-world data (GBRT03-08), but not so in the MS case. This result may be explained by the fact that the agent prioritized the TT optimization due to possible better robustness and predictability while learning the policy. This effect could not be observed in the synthetic datasets (GBRT01-02) and is probably due to the structure of the real-world data. For a further insight, Figure 7 shows how the different approaches already differ in early iterations. Across all approaches, the greatest advances are made in about the first 10 generations. Here, GASA+RL already achieves significantly better fitness values than GA and GASA. This indicates that DRL improves the discovery of local optima early in the process.

The experiment confirms that a suitably trained DNN fed into the metaheuristic leads to better decisions than a randomized creation of neighborhood solutions as it is common for metaheuristics. The prerequisite is that the agent has learned a generally applicable policy that is suitable for the dataset. Based on the hypothesis that conventional metaheuristics can be improved in this way, further research should be undertaken to investigate DRL-based hybrid methods and their applicability to other optimization problems. Related to the production process optimization, further work is required to analyze the suitability of multi agent concepts for DRC-FJS. For example, separate agents for sequencing, machine and worker assignment decisions could be examined in terms of their performance and generalizability. In this

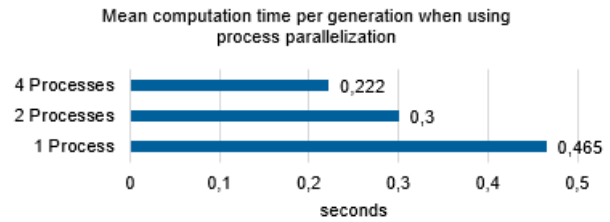


FIGURE 8. Computation time per algorithm iteration without and with parallelization (2 and 4 processes). Parallelization with 4 processes improves the runtime by nearly half the runtime taken with no parallelization for GA, GASA and GASA+RL.

experiment, it could be shown that the neural network trained for GBRT03 also performs well for the other real-world datasets GBRT04-08. In completely different scenarios, however, the agent's performance is poor. Thus, perhaps the most important question is, how DRL agents can internalize a policy as generally as possible for different manufacturing scenarios.

D. PARALLELIZATION EFFECTS

The purpose of this experiment was to investigate the effect of parallelizing the memetic algorithm on multiple computing processes (question 5). With a view to operational practice, a short algorithm runtime is an important requirement. On the one hand, this refers to the reactive use of a planning method where a schedule must be quickly regenerated when unplanned events occur. On the other hand, the method must be efficient so that extensive realistic scenario analyses can be performed dynamically. By choosing a population-based method, we were able to achieve this in a targeted manner. As described in Sect. IV-A, the population of each generation is splitted into equally sized batches, where each batch is processed in a parallel computing process. In this way, it is possible to distribute the operations of the memetic algorithm (crossover, mutation, SA exploitation) as well as the computationally intensive simulation-based evaluation over the computing resources. Figure 8 provides mean values for the runtime per algorithm iteration. There was a significant correlation between mean iteration time and the number of parallel processes: The speed of the procedure can be optimized by parallelization without affecting the resulting quality. In this way, a schedule can be generated in seconds to a few minutes, depending on the input size and hardware. This outcome further supports the idea of providing the proposed algorithm as a scalable cloud-based smart service. In modern managed cloud environments, an abundant number of processes can be dynamically spawned based on demand, allowing a free allocation of computing resources.

E. LIMITATIONS AND IMPLICATIONS

Based on the previous experiments, the essential shortcomings and needs for future works are summarized. The proposed method enables deterministic scheduling without

uncertain parameters or stochastic events. The objectives MS and TT are balanced with equal weights and there is no analysis on how to set the weights or how the algorithm performs with different weights. Through the conception as a simheuristic, the use case for dynamic scheduling or re-scheduling policies can be extended. Worker skills and skill-dependent operation times can be used for workforce planning. However, no further effects of human work are currently taken into account (e.g. preferences for operation types, motivation, mental stress, exhaustion or physical limitations). Short-term lot size planning to minimize setup times is possible. Long-term dynamic lot size planning taking into account primary requirements in subsequent periods is not possible. This study considers an application and not an in-depth investigation of DRL. Due to very different possible inputs, we were not able to find a general DRL hyperparameter setting for all scenarios. This applies in particular to the number of training steps and DNN update intervals. This is not a disadvantage of this study, as we were still able to confirm that DRL can improve the results of a memetic algorithm. However, further research should be undertaken to examine the predictability of appropriate hyperparameters for different production environments considering varying numbers of stations, workers or tasks.

Thus, a central scientific question raised by this study is how DRL can be better generalized to solve varying and complex planning scenarios. In terms of meta-reinforcement learning [49], a highly generic agent may be able to replace metaheuristics, as a pre-trained neural network would have access to existing knowledge and would not need to completely recalculate every instance. This is an crucial potential to save computing time. There is also the question of how proactive planning approaches can be integrated into the scheduling framework, taking uncertainties such as dynamic events into account. In the future, field studies should be carried out to examine the effectiveness of holistic, predictive and reactive scheduling algorithms and how these can be successfully integrated into business processes. Moreover, it would be interesting to assess the effects and feasibility of adding more resource types and real-world processes. This refers to continuous production, inventory levels, special tools, machine buffers, tasks involving multiple workers, dynamic events and customer demands as well as transportation logistics such as the consideration of conveyor belts.

From a managerial perspective, the results of this research support the idea that human-centered scheduling in complex and flexible production processes can be optimized using modern algorithms and technologies such as generic smart services or digital twins. The findings will be of interest to several companies to reduce intransparency, stress and manual effort in terms of reactive scheduling. The deployment of the framework can play a crucial role for a realistic production planning in the domain. An important prerequisite for this is a consistent data basis from operational systems containing

all parameters required by the proposed optimization model.

VI. CONCLUSION

This study set out to develop a novel memetic algorithm enriched with DRL to solve an extensive DRC-FJSSP. The paper's motivation was to address a shortage of prior extensive scheduling models as well as efficient, generic heuristics. The contribution was an algorithm framework capable to map these complex requirements and reliably generating schedules with minimal computing time.

First, a MILP was induced based on practical requirements of human-centered manufacturing processes. Then, a scheduling framework was proposed and tested with real-world data for simultaneous MS and TT minimization. The framework integrates a freely expandable DES and a novel value encoding, which enables both job shop and flow shop organizations. Within a computational study we were able to confirm that a memetic algorithm with DRL outperforms random-based metaheuristics. DRL enables the substitution of a randomized neighborhood creation through targeted sequencing and assignment decisions. As a result, better and more reliable schedules could be generated in fewer iterations. With parallel computing, the algorithm efficiency could be further increased so that the method can be used reactively and in a scalable manner. Under the hardware used and depending of the input dataset, the schedules could be generated in a few seconds to a few minutes.

The developed framework enables our industry partners to plan more holistically compared to conventional methods such as job shop scheduling or critical path method. By considering human workers and their capabilities as well as sequence-dependent setup times, a short-term workforce and lot size planning is synchronized with job scheduling.

APPENDIX A DRL STATE SPACE AND REWARD FUNCTION

Algorithm 1 Intermediate Reward Calculation (Pseudocode Snippet)

```

1: Input :  $C, L, a_1, a_2$  ▷ current and last state features, actions
2:  $r \leftarrow 0$ 
3: if  $a_2 = SF \wedge L_8 = 0$  then
4:    $r \leftarrow -3$ 
5: else if  $a_2 = SF \wedge L_3 > L_5$  then
6:    $r \leftarrow 2$ 
7: else if  $a_2 = WF \wedge \frac{L_6}{L_9} > 1$  then
8:    $r \leftarrow 1$ 
9: end if
10: if  $a_1 = STR \wedge L_{15} < L_{16}$  then
11:    $r \leftarrow r + 1$ 
12: end if
13: if  $C_{12} > L_{12} = 0$  then
14:    $r \leftarrow r + 3$ 
15: end if
16: if  $C_{14} > 0$  then
17:    $r \leftarrow r + 3$ 
18: end if
19: return  $r$ 

```

TABLE 6. DRL state space features (i: Feature index).

i	Description
1	Relative environment time: Current simulation time in relation to the average WIP per station. A relative time greater than 1 is an indicator of wait times and that work is not evenly distributed across resources.
2	Production stage: Average topology group G of available tasks at the station. This feature is intended to describe the position at which the station is located in the production line.
3	Relative station WIP: Duration of all tasks that have not yet been completed (in relation to the overall WIP). This feature provides information on how much work still needs to be done by the station. It is intended to be an indicator of potential congestions.
4	Mean successor station WIP: The average duration of the tasks that have not been completed across all stations that follow directly in the production line. The intention here is to make accumulations and bottlenecks between related stations recognizable.
5	Mean station WIP: Average duration of non-completed tasks over all stations.
6	Relative worker WIP: Duration of all tasks that have not yet been completed for the workers at the station. This is intended to give an indication of whether comparatively much or little human labor is required at the station. Thus, it describes the automation degree.
7	Number of available tasks at the station, that are directly processable (relative to all processable tasks over all stations)
8	Competing tasks (binary encoded): 1, if more than one task is currently processable at the station.
9	Number of processing slots at the station. This feature describes the situational capacity of the station, how many tasks can be processed in parallel and how many workers can work here at the same time.
10	Current station throughput rate. According to Little's Law, this feature should be used in combination with the WIP information to infer waiting times, which are an indicator for the MS [50].
11	Current mean throughput rate of all successor stations
12	Current mean throughput rate over all stations
13	Current throughput rate standard deviation over all stations
14	Minimum job slack time remaining at the station
15	Mean job slack time remaining at the station
16	Mean job slack time remaining over all stations
17	Job slack time remaining standard deviation over all stations

APPENDIX B SETTINGS AND HYPERPARAMETERS FOR THE NUMERICAL EXPERIMENTS

See Tables 7–9.

TABLE 7. GA hyperparameters.

Hyperparameter	Value
Initialization method	Equally distributed resources
	Random dispatching rules
First population size	50 genomes
Selection method	Best
Number of survivors	8
Offspring size	20
Mutation probability	Adaptive (linearly decreasing)
Combination probability	Adaptive (linearly increasing)
Stopping criterion	500 operations and evaluations

TABLE 8. MILP Solver environment.

Solver:	IBM CPLEX V12
OS:	CentOS 7.5.1804
vCPU:	Intel Xeon E5-2630 V4 2.2 GHz with 28 cores assigned
RAM:	64 GB DDR4 2133 MHz

TABLE 9. DRL hyperparameters for the training process and for the actor-critic PPO network.

Hyperparameter	Value
Overall training steps	Dynamic (30,000 on GBRT03)
Network update interval (steps)	Dynamic (10 on GBRT03)
Learning rate	0.0001 (linearly decreasing)
Discount factor	0.999
Activation function	Rectified Linear Unit (leaky)
Network architecture (hidden layers)	2^9 neurons (shared)
	2^7 , 2^6 neurons (value net)
	2^6 neurons (policy net)

ACKNOWLEDGMENT

Author Contribution: Felix Grumbach: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing (original draft), writing (review and editing), visualization, and project administration. Nour Eldin Alaa Badr: Investigation (related work analysis), data curation, writing [original draft (literature review)], and writing [review and editing]. Pascal Reusch: Resources, funding acquisition, project administration, and supervision. Sebastian Trojahn: Supervision.

CODE AVAILABILITY

Open source code is publicly accessible at <https://doi.org/10.17605/OSF.IO/JRVFC>.

REFERENCES

- [1] M. L. Pinedo, *Scheduling*. Cham, Switzerland: Springer, 2012.
- [2] M. R. Garey, D. S. Johnson, and R. Sethi, "The complexity of flowshop and jobshop scheduling," *Math. Oper. Res.*, vol. 1, no. 2, pp. 117–129, May 1976, doi: [10.1287/moor.1.2.117](https://doi.org/10.1287/moor.1.2.117).
- [3] P. Brucker and R. Schlie, "Job-shop scheduling with multi-purpose machines," *Computing*, vol. 45, no. 4, pp. 369–375, Dec. 1990, doi: [10.1007/BF02238804](https://doi.org/10.1007/BF02238804).
- [4] M. Treleven, "A review of the dual resource constrained system research," *IIE Trans.*, vol. 21, no. 3, pp. 279–287, Sep. 1989, doi: [10.1080/07408178908966233](https://doi.org/10.1080/07408178908966233).
- [5] A. A. B. Pritsker, L. J. Waiters, and P. M. Wolfe, "Multiproject scheduling with limited resources: A zero-one programming approach," *Manage. Sci.*, vol. 16, no. 1, pp. 93–108, Sep. 1969, doi: [10.1287/mnsc.16.1.93](https://doi.org/10.1287/mnsc.16.1.93).
- [6] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, no. 5, pp. 8091–8126, Feb. 2021, doi: [10.1007/s11042-020-10139-6](https://doi.org/10.1007/s11042-020-10139-6).
- [7] F. Xhafa, J. A. Gonzalez, K. P. Dahal, and A. Abraham, "A GA(TS) hybrid algorithm for scheduling in computational grids," in *Hybrid Artificial Intelligence Systems* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2009, pp. 285–292, doi: [10.1007/978-3-642-02319-4_34](https://doi.org/10.1007/978-3-642-02319-4_34).
- [8] A. A. Juan, J. Faulin, S. E. Grasman, M. Rabe, and G. Figueira, "A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems," *Oper. Res. Perspect.*, vol. 2, pp. 62–72, Dec. 2015, doi: [10.1016/j.orp.2015.03.001](https://doi.org/10.1016/j.orp.2015.03.001).
- [9] C. L. Quintero-Araújo, A. A. Juan, J. R. Montoya-Torres, and A. Muñoz-Villamizar, "A simheuristic algorithm for horizontal cooperation in urban distribution: Application to a case study in Colombia," in *Proc. Winter Simul. Conf. (WSC)*, Dec. 2016, pp. 2193–2204, doi: [10.1109/WSC.2016.7822261](https://doi.org/10.1109/WSC.2016.7822261).
- [10] E. F. Morales and J. H. Zaragoza, "An introduction to reinforcement learning," in *Decision Theory Models for Applications in Artificial Intelligence*. Hershey, PA, USA: IGI Global, 2012, pp. 63–80, doi: [10.4018/978-1-60960-165-2.ch004](https://doi.org/10.4018/978-1-60960-165-2.ch004).

- [11] K. Arulkumar, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017, doi: [10.1109/MSP.2017.2743240](https://doi.org/10.1109/MSP.2017.2743240).
- [12] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Found. Trends Mach. Learn.*, vol. 11, nos. 3–4, pp. 219–354, 2018, doi: [10.1561/22000000071](https://doi.org/10.1561/22000000071).
- [13] B. M. Kayhan and G. Yildiz, "Reinforcement learning applications to machine scheduling problems: A comprehensive literature review," *J. Intell. Manuf.*, vol. 34, no. 3, pp. 905–929, Mar. 2023, doi: [10.1007/s10845-021-01847-3](https://doi.org/10.1007/s10845-021-01847-3).
- [14] S. Whiteson, B. Tanner, M. E. Taylor, and P. Stone, "Protecting against evaluation overfitting in empirical reinforcement learning," in *Proc. IEEE Symp. Adapt. Dyn. Program. Reinforcement Learn. (ADPRL)*, Apr. 2011, pp. 120–127, doi: [10.1109/ADPRL.2011.5967363](https://doi.org/10.1109/ADPRL.2011.5967363).
- [15] M. Schilling, "Avoid overfitting in deep reinforcement learning: Increasing robustness through decentralized control," in *Artificial Neural Networks and Machine Learning (ICANN) (Lecture Notes in Computer Science)*. Cham, Switzerland: Springer, 2021, pp. 638–649, doi: [10.1007/978-3-030-86380-7_52](https://doi.org/10.1007/978-3-030-86380-7_52).
- [16] P. Winklehner and V. A. Hauder, "Flexible job-shop scheduling with release dates, deadlines and sequence dependent setup times: A real-world case," *Proc. Comput. Sci.*, vol. 200, pp. 1654–1663, Jan. 2022, doi: [10.1016/j.procs.2022.01.366](https://doi.org/10.1016/j.procs.2022.01.366).
- [17] M. Thenarasu, K. Rameshkumar, J. Rousseau, and S. P. Anbuudayasankar, "Development and analysis of priority decision rules using MCDM approach for a flexible job shop scheduling: A simulation study," *Simul. Model. Pract. Theory*, vol. 114, Jan. 2022, Art. no. 102416, doi: [10.1016/j.simpat.2021.102416](https://doi.org/10.1016/j.simpat.2021.102416).
- [18] G. Zhang, Y. Hu, J. Sun, and W. Zhang, "An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints," *Swarm Evol. Comput.*, vol. 54, May 2020, Art. no. 100664, doi: [10.1016/j.swevo.2020.100664](https://doi.org/10.1016/j.swevo.2020.100664).
- [19] Z. Zhu and X. Zhou, "Flexible job-shop scheduling problem with job precedence constraints and interval grey processing time," *Comput. Ind. Eng.*, vol. 149, Nov. 2020, Art. no. 106781, doi: [10.1016/j.cie.2020.106781](https://doi.org/10.1016/j.cie.2020.106781).
- [20] W. T. Lunardi and H. Voos, "An extended flexible job shop scheduling problem with parallel operations," *ACM SIGAPP Appl. Comput. Rev.*, vol. 18, no. 2, pp. 46–56, Jul. 2018, doi: [10.1145/3243064.3243068](https://doi.org/10.1145/3243064.3243068).
- [21] F. M. Defersha and D. Rooyani, "An efficient two-stage genetic algorithm for a flexible job-shop scheduling problem with sequence dependent attached/detached setup, machine release date and lag-time," *Comput. Ind. Eng.*, vol. 147, Sep. 2020, Art. no. 106605, doi: [10.1016/j.cie.2020.106605](https://doi.org/10.1016/j.cie.2020.106605).
- [22] X. Huang and L. Yang, "A hybrid genetic algorithm for multi-objective flexible job shop scheduling problem considering transportation time," *Int. J. Intell. Comput. Cybern.*, vol. 12, no. 2, pp. 154–174, Jun. 2019, doi: [10.1108/ijicc-10-2018-0136](https://doi.org/10.1108/ijicc-10-2018-0136).
- [23] R. Wu, Y. Li, S. Guo, and W. Xu, "Solving the dual-resource constrained flexible job shop scheduling problem with learning effect by a hybrid genetic algorithm," *Adv. Mech. Eng.*, vol. 10, no. 10, pp. 1–14, Oct. 2018, doi: [10.1177/1687814018804096](https://doi.org/10.1177/1687814018804096).
- [24] H. Du, F. Qiao, J. Wang, and H. Lu, "A hybrid metaheuristic algorithm with novel decoding methods for flexible flow shop scheduling considering human fatigue," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2021, pp. 2328–2333, doi: [10.1109/smc52423.2021.9658692](https://doi.org/10.1109/smc52423.2021.9658692).
- [25] F. M. Defersha, D. Obimuyiwa, and A. D. Yimer, "Mathematical model and simulated annealing algorithm for setup operator constrained flexible job shop scheduling problem," *Comput. Ind. Eng.*, vol. 171, Sep. 2022, Art. no. 108487, doi: [10.1016/j.cie.2022.108487](https://doi.org/10.1016/j.cie.2022.108487).
- [26] Z. Liu, J. Liu, C. Zhuang, and F. Wan, "Multi-objective complex product assembly scheduling problem considering parallel team and worker skills," *J. Manuf. Syst.*, vol. 63, pp. 454–470, Apr. 2022, doi: [10.1016/j.jmsy.2022.05.003](https://doi.org/10.1016/j.jmsy.2022.05.003).
- [27] N. Berti, S. Finco, O. Battaia, and X. Delorme, "Ageing workforce effects in dual-resource constrained job-shop scheduling," *Int. J. Prod. Econ.*, vol. 237, Jul. 2021, Art. no. 108151, doi: [10.1016/j.ijpe.2021.108151](https://doi.org/10.1016/j.ijpe.2021.108151).
- [28] J. Zhang and J. Jie, "A multi-objective particle swarm optimization algorithm embedded with maximum fitness function for dual-resources constrained flexible job shop scheduling," in *Proc. 17th Int. Conf. Intell. Comput. (ICIC)*. Shenzhen, China: Springer, Aug. 2021, pp. 725–738, doi: [10.1007/978-3-030-84522-3_59](https://doi.org/10.1007/978-3-030-84522-3_59).
- [29] J. L. Andrade-Pineda, D. Canca, P. L. Gonzalez-R, and M. Calle, "Scheduling a dual-resource flexible job shop with makespan and due date-related criteria," *Ann. Oper. Res.*, vol. 291, nos. 1–2, pp. 5–35, Aug. 2020, doi: [10.1007/s10479-019-03196-0](https://doi.org/10.1007/s10479-019-03196-0).
- [30] D. Kress, D. Müller, and J. Nossack, "A worker constrained flexible job shop scheduling problem with sequence-dependent setup times," *OR Spectr.*, vol. 41, no. 1, pp. 179–217, Mar. 2019, doi: [10.1007/s00291-018-0537-z](https://doi.org/10.1007/s00291-018-0537-z).
- [31] S. Lang, F. Behrendt, N. Lanzerath, T. Reggelin, and M. Müller, "Integration of deep reinforcement learning and discrete-event simulation for real-time scheduling of a flexible job shop production," in *Proc. Winter Simul. Conf. (WSC)*, Dec. 2020, pp. 3057–3068, doi: [10.1109/WSC48552.2020.9383997](https://doi.org/10.1109/WSC48552.2020.9383997).
- [32] T. Khuntiyaporn, P. Songmuang, and W. Limprasert, "The multiple objectives flexible jobshop scheduling using reinforcement learning," in *Proc. 16th Int. Joint Symp. Artif. Intell. Natural Lang. Process. (ISAIR-NLP)*, Dec. 2021, pp. 1–6, doi: [10.1109/ISAIR-NLP54397.2021.9678152](https://doi.org/10.1109/ISAIR-NLP54397.2021.9678152).
- [33] J. Popper, W. Motsch, A. David, T. Petzsche, and M. Ruskowski, "Utilizing multi-agent deep reinforcement learning for flexible job shop scheduling under sustainable viewpoints," in *Proc. Int. Conf. Electr., Comput., Commun. Mechatronics Eng. (ICECCME)*, Oct. 2021, pp. 1–6, doi: [10.1109/ICECCME52200.2021.9590925](https://doi.org/10.1109/ICECCME52200.2021.9590925).
- [34] S. Luo, L. Zhang, and Y. Fan, "Real-time scheduling for dynamic partial-no-wait multiobjective flexible job shop by deep reinforcement learning," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3020–3038, Oct. 2022, doi: [10.1109/TASE.2021.3104716](https://doi.org/10.1109/TASE.2021.3104716).
- [35] T. Zhou, H. Zhu, D. Tang, C. Liu, Q. Cai, W. Shi, and Y. Gui, "Reinforcement learning for online optimization of job-shop scheduling in a smart manufacturing factory," *Adv. Mech. Eng.*, vol. 14, no. 3, pp. 1–19, Mar. 2022, doi: [10.1177/16878132221086120](https://doi.org/10.1177/16878132221086120).
- [36] Y. Du, J. Li, X. Chen, P. Duan, and Q. Pan, "Knowledge-based reinforcement learning and estimation of distribution algorithm for flexible job shop scheduling problem," *IEEE Trans. Emerg. Topics Comput. Intell.*, early access, Feb. 8, 2022, doi: [10.1109/TETCI.2022.3145706](https://doi.org/10.1109/TETCI.2022.3145706).
- [37] X. Zhu, J. Xu, J. Ge, Y. Wang, and Z. Xie, "Multi-task multi-agent reinforcement learning for real-time scheduling of a dual-resource flexible job shop with robots," *Processes*, vol. 11, no. 1, p. 267, Jan. 2023, doi: [10.3390/pr11010267](https://doi.org/10.3390/pr11010267).
- [38] A. Seyyedabbasi, R. Aliyev, F. Kiani, M. U. Gulle, H. Basyildiz, and M. A. Shah, "Hybrid algorithms based on combining reinforcement learning and metaheuristic methods to solve global optimization problems," *Knowledge-Based Syst.*, vol. 223, Jul. 2021, Art. no. 107044, doi: [10.1016/j.knsys.2021.107044](https://doi.org/10.1016/j.knsys.2021.107044).
- [39] F. Kosanoglu, M. Atmis, and H. H. Turan, "A deep reinforcement learning assisted simulated annealing algorithm for a maintenance planning problem," *Ann. Oper. Res.*, pp. 1–32, Mar. 2022. [Online]. Available: <https://link.springer.com/article/10.1007/s10479-022-04612-8#article-info>, doi: [10.1007/s10479-022-04612-8](https://doi.org/10.1007/s10479-022-04612-8).
- [40] M. Dhiflaoui, H. E. Nouri, and O. B. Driss, "Dual-resource constraints in classical and flexible job shop problems: A state-of-the-art review," *Proc. Comput. Sci.*, vol. 126, pp. 1507–1515, Jan. 2018, doi: [10.1016/j.procs.2018.08.123](https://doi.org/10.1016/j.procs.2018.08.123).
- [41] X. Gong, Q. Deng, G. Gong, W. Liu, and Q. Ren, "A memetic algorithm for multi-objective flexible job-shop problem with worker flexibility," *Int. J. Prod. Res.*, vol. 56, no. 7, pp. 2506–2522, Apr. 2018, doi: [10.1080/00207543.2017.1388933](https://doi.org/10.1080/00207543.2017.1388933).
- [42] I. Ono, M. Yamamura, and S. Kobayashi, "A genetic algorithm for job-shop scheduling problems using job-based order crossover," in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1999, pp. 547–552, doi: [10.1109/icc.1996.542658](https://doi.org/10.1109/icc.1996.542658).
- [43] S. Scherfke and O. Lünsdorf, *SimPy: Discrete event simulation for Python*. Accessed Jul. 18, 2022. [Online]. Available: <https://simpy.readthedocs.io>
- [44] J. D. C. Little, "A proof for the queuing formula: $L = \lambda W$," *Oper. Res.*, vol. 9, no. 3, pp. 383–387, Jun. 1961, doi: [10.1287/opre.9.3.383](https://doi.org/10.1287/opre.9.3.383).
- [45] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *J. Mach. Learn. Res.*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: <http://jmlr.org/papers/v22/20-1364.html>
- [46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," 2017, *arXiv:1707.06347*.
- [47] A. Hassanat, K. Almoammadi, E. Alkafaween, E. Abunawas, A. Hammouri, and V. B. S. Prasath, "Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach," *Information*, vol. 10, no. 12, p. 390, Dec. 2019, doi: [10.3390/info10120390](https://doi.org/10.3390/info10120390).

- [48] V. F. Yu, Winarno, A. Maulidin, A. A. N. P. Redi, S.-W. Lin, and C.-L. Yang, "Simulated annealing with restart strategy for the path cover problem with time windows," *Mathematics*, vol. 9, no. 14, p. 1625, Jul. 2021, doi: [10.3390/math9141625](https://doi.org/10.3390/math9141625).
- [49] H. Liu, P. Chen, and Z. Zhao, "Towards a robust meta-reinforcement learning-based scheduling framework for time critical tasks in cloud environments," in *Proc. IEEE 14th Int. Conf. Cloud Comput. (CLOUD)*, Sep. 2021, pp. 637–647, doi: [10.1109/CLOUD53861.2021.00082](https://doi.org/10.1109/CLOUD53861.2021.00082).
- [50] A. K. Kaban, Z. Othman, and D. S. Rohmah, "Comparison of dispatching rules in job-shop scheduling problem using simulation: A case study," *Int. J. Simul. Model.*, vol. 11, no. 3, pp. 129–140, 2012, doi: [10.2507/ijssimm11\(3\)2.201](https://doi.org/10.2507/ijssimm11(3)2.201).



FELIX GRUMBACH received the B.Sc. degree in information systems from the Bielefeld University of Applied Sciences, Bielefeld, Germany, in 2015, and the M.Sc. degree from the University of Hagen, Germany, in 2020. He is currently pursuing the Ph.D. degree with the Doctoral Center for Social, Health and Economic Sciences, Saxony-Anhalt University of Applied Sciences. He is currently a Research Associate at the Center for Applied Data Science, Bielefeld University of

Applied Sciences. His research interest includes the robust and holistic optimization of complex production processes with the help of machine learning techniques.



NOUR ELDIN ALAA BADR received the B.Sc. degree in electrical engineering from the Higher Technological Institute, Cairo, Egypt, in 2016, and the M.Sc. degree in intelligent systems from Bielefeld University, Germany, in 2022. He is currently a Research Associate at the Center For Applied Data Science, Bielefeld University of Applied Sciences. He focuses on researching and developing intelligent production scheduling systems.



PASCAL REUSCH received the Ph.D. degree in economics. He is currently a Professor of production and industrial management at the Bielefeld University of Applied Sciences. He is also a Founding Member of the Center for Applied Data Science (CfADS). He has several years of practical professional experience in the industry.



SEBASTIAN TROJAHN received the Ph.D. degree in engineering. He is currently a Professor of supply chain management, operations management, and digital logistics at the Anhalt University of Applied Sciences, Saxony-Anhalt, Germany. He has several years of practical professional experience in the industry.

...

2.2 Publikation 2: Humanzentrierte Ablaufplanung von Montagelinien

Status

Veröffentlicht

Bibliographische Angabe

Lukas Vollenkemper, Felix Grumbach, Martin Kohlhase und Pascal Reusch (2023). „Humanzentrierte Ablaufplanung von Montagelinien/Human-centered scheduling in assembly lines - Plug and play: Efficient algorithms minimize stress in flow shops“. In: *wt Werkstattstechnik online* 113.04, S. 158–163. DOI: 10.37544/1436-4980-2023-04-58.

Plug and Play – effiziente Algorithmen minimieren Belastung an Transferstraßen

Humanzentrierte Ablaufplanung von Montagelinien

L. Vollenkemper, F. Grumbach, M. Kohlhasse, P. Reusch

Angesichts des zunehmenden Fachkräftemangels sind Gesundheit und Zufriedenheit der Beschäftigten von immer größerer Bedeutung für produzierende Unternehmen. Insbesondere bei variantenreichen Produkten müssen Werker und Werkerinnen komplexe Aufgaben erledigen. Diese Variantenvielfalt ist auch für die Produktionsfeinplanung eine Herausforderung. Die vorgestellte Heuristik hilft in einem realen Anwendungsfall Belastungsspitzen an getakteten Linien zu reduzieren und verbessert so die betriebliche Gesundheit.

STICHWÖRTER

Mensch und Technik, Fertigungsplanung, Software

1 Einleitung

Transferstraßen mit Montagefeldern sind eine verbreitete Form der Produktion. Dabei werden Produkte entlang einer Fertigungsstraße automatisiert, synchronisiert gefördert und von Menschen oder Maschinen bearbeitet. Dieser Organisationstyp der Produktion ist sehr vorteilhaft, da standardisierte Arbeitsflüsse sehr effizient und transparent abgebildet werden können [1]. Andererseits führt die einhergehende Standardisierung zu einer eingeschränkten Flexibilität, was angesichts wachsender Kundenerwartungen eine Herausforderung darstellt.

Um sich weiterhin im Markt behaupten zu können, müssen in vielen Bereichen individualisierbare Produkte oder zumindest ein breites Variantenspektrum angeboten werden [2]. Diese Varianten erfordern fachkundige Beschäftigte, die die Besonderheiten der einzelnen Varianten kennen und umsetzen können. Die Fertigung von individualisierbaren Produkten auf Transferstraßen bringt zudem erhebliche organisatorische Herausforderungen mit sich. Häufig unterscheiden sich einzelne Varianten an den einzelnen Arbeitsplätzen durch ihren Arbeitsaufwand erheblich. So kann an einem Arbeitsplatz für manche Produktkonfigurationen keine Arbeit nötig sein, während das Arbeitspensum bei anderen Produktkonfigurationen nur gerade zu schaffen ist. Dies muss bei der Produktionsplanung und -steuerung berücksichtigt werden.

Trotz der zunehmenden Automatisierung ist menschliche Arbeit in diesem Zusammenhang weiterhin wichtig [3, 4]. Die Belegschaft zu schonen und zu motivieren ist entscheidend, um das kostbare Domänenwissen und die Erfahrung der Beschäftigten lange an das Unternehmen zu binden. Es gilt vor allem Belastungsspitzen für die Beschäftigten vorausschauend zu verhindern.

Human-centered scheduling in assembly lines - Plug and play: Efficient algorithms minimize stress in flow shops

Given a shortage of skilled workers, employee health and satisfaction are of increasing importance to manufacturing companies. Particularly in the case of products with many variants, workers have to perform complex tasks. This diversity of variants is also a challenge for production planning. New planning paradigms help to reduce peaks of stress on flow production and thus contribute to occupational health.

Die klassische Fertigungsplanung, die vor allem Durchsatz und Kosteneffizienz im Blick hatte, muss dazu um die humanzentrierte Perspektive erweitert werden.

Der folgende Beitrag vergleicht anhand eines Praxisbeispiels verschiedene humanzentrierte Zielfunktionen für Werker und Werkerinnen an Transferstraßen und die Lösungsqualität aktueller Metaheuristiken. Der vorgestellte Ansatz ist aufgrund seiner allgemeingültigen Modellierung leicht auf verschiedene Fertigungstypen übertragbar. Darüber hinaus kann der entwickelte Ansatz in bestehende Planungstools, wie etwa Materialflusssimulationen oder ERP-Systeme integriert werden.

2 Problemstellung

Der Anwendungspartner Istringhausen GmbH & Co. KG ist Zulieferer in der Automobilindustrie. Hier werden Sitze nach Kundenwunsch individualisiert hergestellt und Just-in-Sequence an die Hersteller geliefert. Die im Projekt betrachtete Fertigungsanlage besteht aus mehreren identischen Produktionslinien (Transferstraßen). Produkte werden entlang der Linie durch mehrere klar abgegrenzten Arbeitsplätze gefahren, an denen die einzelnen Arbeitsschritte durchgeführt werden. Das Fördersystem fährt dabei in einer konstanten Geschwindigkeit und die Produkte haben alle denselben Abstand zueinander. So steht an jedem Arbeitsplatz für alle Produkte dasselbe Zeitfenster zur Verfügung. Produkte können sich innerhalb der Linie nicht überholen. Geplant werden muss lediglich die Reihenfolge, in der die Produkte aufgegeben werden. Die betrachtete Fertigungsstraße ist in **Bild 1** zu sehen.

Dieses Optimierungsproblem lässt sich in Anlehnung an das Distributed Permutation Flow-Shop Problem (DPFSP) formalisieren, mit dem Unterschied, dass im Falle von Isringhausen die Zeit an den Maschinen für alle Produkte gleich ist. Das DPFSP besteht aus mehreren identischen Produktionslinien. Den Linien werden aus dem Auftragsvorrat eine feste Anzahl von Aufträgen zugewiesen und die optimale Reihenfolge der Aufträge auf der jeweiligen Linie bestimmt [5]. Das Problem, das in dieser Arbeit untersucht werden soll, wird wie folgt charakterisiert.

- Die Produktion besteht aus N identischen Linien, welche ihrerseits aus A Arbeitsstationen bestehen. An jeder Arbeitsstation ist eine Arbeitskraft beschäftigt.
- Die Linien fahren alle mit derselben konstanten Geschwindigkeit, sodass jedes Produkt eine Zeit τ an jeder Arbeitsstation verbringt.
- Es müssen $J \in \mathbb{N}$ individualisierte Produkte produziert werden.
- Für jedes Produkt $j \leq J, j \in \mathbb{N}$ gibt es eine Deadline, bis zu der es produziert sein muss. Die Deadlines werden als Vektor $\mathbf{d} \in \mathbb{N}^J$ festgehalten.
- Zudem werden die Zeiten, die es für die Produktion jedes Produkts an jedem Arbeitsplatz braucht in eine Matrix $\mathbf{M} \in \mathbb{N}^{A \times J}$ geschrieben. Die Zeiten werden im Anwendungsfall durch MTM (Methods Time Measurement) bestimmt. Diese Zeiten werden in Unternehmen häufig verwendet, um Plan- und Vorgabezeiten zu bestimmen. Durch die Individualisierung unterscheidet sich die Arbeitszeit der Produkte an den Arbeitsplätzen.
- Es müssen pro Linie eine Anzahl $k \leq \frac{J}{N}, k \in \mathbb{N}$ Produkte gefertigt werden. Ein möglicher Produktionsplan $\mathbf{P} \in \mathbb{N}^{N \times k}$ kann als Matrix dargestellt werden. Dabei bezeichnet jede Zeile den Produktionsplan für eine Maschine, mit je k Produkten. Der Eintrag p_{ni} für den i -ten Auftrag auf der Linie n bezeichnet dabei ein bestimmtes Produkt. Die Bearbeitungszeit für dieses Produkt an Arbeitsplatz a kann über den entsprechenden Eintrag $m_{p_{ni},n}$ aus \mathbf{M} bestimmt werden.
- Falls der Auftragsvorrat größer als kN ist, wird die überzählige Anzahl an Aufträgen nicht in diesem Batch produziert. Sie können in einem folgenden Batch (dem nächsten Planungszyklus) wieder verplant werden.

DPFSP wurden in den letzten Jahren mit verschiedenen Methoden und mit unterschiedlichen Zielsetzungen gelöst und um weitere Nebenbedingungen ergänzt. Zu den bisher genutzten Metaheuristiken zählen etwa Tabu-Suche [6], Chemical Reaction Optimization [7], Iterated Greedy Search [8] und genetische Algorithmen [9]. Als Nebenbedingungen wurden zum Beispiel begrenzte Zwischenpuffer [10], Rüstzeiten [11] und ablaufende Aufträge [12] untersucht.

In ihrer Übersichtsarbeit identifizieren [13] neun unterschiedliche Zielfunktionen, die bisher für DPFSP-Probleme verwendet wurden. Diese sind die Minimierung der Produktionsspanne, der Gesamtarbeitszeit, der Fließzeit, der Verspätung, der Frühreife, der Anzahl verspäteter Produkte, der maximalen Verspätung und der Gesamtkosten. Beim Ziel der Minimierung der Verspätung werden zwei verschiedene Definitionen von Verspätung betrachtet. Wenn man diese Zielfunktionen betrachtet, wird deutlich, dass alle betriebswirtschaftlich motiviert sind. Die Bedürfnisse der Beschäftigten werden in keinem dieser Ziele berücksichtigt. Daraus ergibt sich ein dringender Handlungsbedarf zur Integration humanzentrierter Aspekte in die Zielfunktionen. So soll im



Bild 1. Eine der betrachteten Linien bei der Firma Isringhausen.
Foto: Isringhausen GmbH

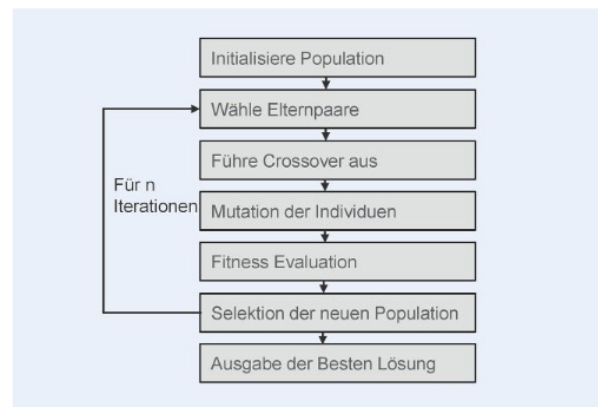


Bild 2. Ablauf eines genetischen Algorithmus. Grafik: FH Bielefeld

Projekt die Belastung der Beschäftigten gesenkt werden ohne betriebswirtschaftliche Aspekte zu vernachlässigen. Gleichzeitig wird die bestehende Forschungslücke adressiert, die sich aus der einseitig betriebswirtschaftlichen Motivation der Zielfunktionen ergibt.

3 Lösungsansatz

Im folgenden Kapitel wird der Lösungsansatz näher erläutert. Zunächst wird der genetische Algorithmus und die Anpassungen an das vorliegende Problem dargestellt. Anschließend werden verschiedene humanzentrierte Zielfunktionen vorgestellt und diskutiert.

3.1 Genetischer Algorithmus

Genetische Algorithmen (GA) sind populationsbasierte Metaheuristiken, die effizient und zuverlässig gute lokale Optima identifizieren können. Durch die Berücksichtigung vieler Lösungen ist die Wahrscheinlichkeit geringer, in schlechten lokalen Optima festzustecken [14]. GA wurden, inspiriert durch den biologischen Prozess der Evolution, durch Selektion und Mutation entwickelt und sind in Bild 2 dargestellt.

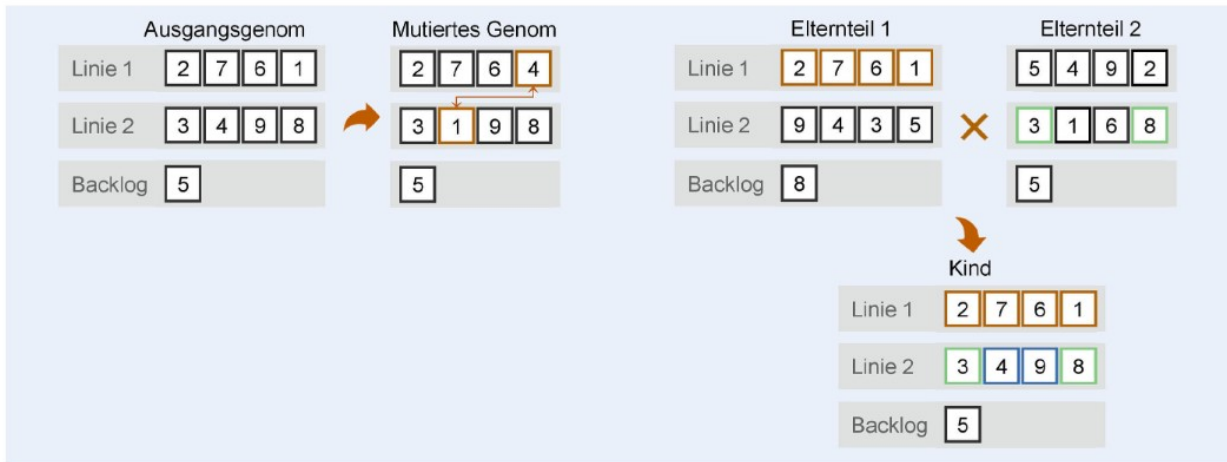


Bild 3. Links: Mutation eines Genoms durch zufälligen Tausch zweier Produkte im Plan. Rechts: Crossover Operation. Orange markierte Produkte wurden von E1 kopiert, grüne aus E2. Blau markierte Produkte mussten neu gesetzt werden. Grafik: FH Bielefeld

Eine Population bezeichnet eine Menge möglichen Lösungen für das Problem. Die einzelnen Lösungen (Individuen) sind dabei in einer einheitlichen Form dargestellt, welche als Chromosom bezeichnet wird. Angelehnt an das biologische Vorbild durchläuft die Population nun immer wieder die Operationen der Paarung (Crossover), Mutation und Selektion. So entsteht in jeder Iteration eine neue Population, deren Individuen das Optimierungsproblem im Durchschnitt besser lösen als die der vorangegangenen. Die Selektion wählt dabei aus Eltern und Kindern eine neue Population. Es gibt verschiedene Selektionsstrategien: So können ausschließlich die Kinder übernommen werden (generational) oder nur die besten Individuen (steady state) [15].

Wie beschrieben müssen für die Implementierung des GA-Operatoren für die Paarung (Crossover) von zwei Individuen und die Mutation eines Individuums definiert werden. Die Mutation tauscht zufällig die Position von zwei Produkten im Plan. Dabei können auch Produkte aus den ungeplanten Produkten in den Plan übernommen werden.

Der Crossover Operation werden zwei Eltern-Individuen übergeben. Zunächst werden zufällig die Hälfte der Linien ausgewählt, deren Plan von Elternteil 1 (E1) übernommen wird. Die übrig gebliebenen Linien werden nach Möglichkeit nach der Reihenfolge in Elternteil 2 (E2) verplant. Dazu wird der Plan von E2 für jede Linie Schritt für Schritt in das Kind kopiert. Für jedes Produkt wird geprüft, ob es durch die aus E1 kopierten Pläne bereits verplant ist. In diesem Fall kann das Produkt nicht im gleichen Slot wie in E2 verplant werden, da es sonst doppelt verplant wäre. Stattdessen wird ein anderes Produkt an diese Stelle gesetzt. Dieses wird aus den Produkten ausgewählt, die im Kind noch nicht verplant sind. Ist das Produkt hingegen noch nicht durch E1 verplant, kann es in das Kind übernommen werden. Die Mutation und Crossover-Operation sind in Bild 3 dargestellt.

3.2 Zielfunktionen

Ein Ziel dieser Arbeit ist es, die oben beschriebene Produktion nicht ausschließlich nach Durchsatz oder Produktionsspanne zu optimieren, sondern die Belastung der Beschäftigten zu minimieren, ohne betriebswirtschaftliche Ziele zu gefährden. Eine Belastung entsteht, wenn die Zeit zur Erledigung der Arbeiten an

einem Produkt fast oder genau so lang ist, wie die Zeit τ die pro Arbeitsplatz zur Verfügung steht. Wenn Beschäftigte häufig hintereinander die Bearbeitung nur ohne Puffer in der vorgegebenen Zeit schaffen, entsteht Stress. Dieser Stress kann quantifiziert werden, indem die MTM-Zeiten für die Produkte aus M mit der zur Verfügung stehenden Zeit verglichen werden. Für einen Produktionsplan P kann die Belastung an Linie n für Arbeitsplatz a durch das i -te Produkt im Plan berechnet werden als:

$$b(n, a, p_{ni}) = m_{p_{ni},a} - \tau \tag{1}$$

Dieser Belastungswert spiegelt die Beanspruchung eines Arbeitsplatzes durch ein bestimmtes Produkt wieder. Sind an einem Produkt p_{ni} an Arbeitsplatz a aufwendige Tätigkeiten notwendig, wird dieser Wert nahe Null sein. Er sollte nie größer als Null sein, da sonst mehr Arbeit in den MTM-Zeiten verplant ist, als durch die Liniengeschwindigkeit zugelassen wird – die Tätigkeit könnte also nicht ausgeführt werden. Muss ein Produkt an einem Arbeitsplatz hingegen nicht bearbeitet werden entspricht der Beanspruchungswert $-\tau$, ist also sehr gering.

Da die Linien mit konstanter Geschwindigkeit laufen, wird irgendwann eine belastende Bearbeitung durch ein besonders aufwändiges Produkt stattfinden müssen. Durch die Wahl der Reihenfolge kann jedoch sichergestellt werden, dass nicht mehrere aufwändige Produkte hintereinander bearbeitet werden müssen. Um die vorangegangenen und die folgenden Produkte in die Bewertung mit einfließen zu lassen wird der gleitende Durchschnitt genutzt. Ein gleitender Durchschnitt mit Ordnung $o > 0 \in \mathbb{N}$ an einer Linie n wird definiert als:

$$MA_o^n(n, a) = \frac{1}{2o} \sum_{i=i-o}^{i=o} b(n, a, p_{ni}) \tag{2}$$

Der Hyperparameter o stellt dabei die Breite des Filters ein. Bei hohen Werten von o wird über mehr Produkte gemittelt, sodass Belastungsspitzen weniger stark ausgeprägt sind. Deshalb werden im Beitrag Werte von 1 oder 2 für o verwendet. Andernfalls würden Änderungen der Reihenfolge den gleitenden Durchschnitt kaum beeinflussen.

Diese Definition führt zu nicht definierten Werten für die ersten und letzten o Einträge des Produktionsplans, da hier keine

o vorangegangenen oder folgenden Werte existieren. In diesem Fall werden statt der vorangegangenen beziehungsweise folgenden Werte Nullen eingesetzt. Ausgehend von dieser reihenfolgeabhängigen Definition der Belastung können verschiedene Zielfunktionen für die Reihenfolgeplanung definiert werden. Die erste, die hier betrachtet werden soll, ist die Minimierung der maximalen Belastung:

$$f_{\min\max}(\mathbf{P}) = \max_{n,a} MA_p^0(n, a) \quad (3)$$

In dieser Formulierung ist das vorrangige Ziel große Belastungsspitzen zu vermeiden. Allerdings wird ausschließlich die größte Belastung betrachtet. Änderungen des Produktionsplanes, welche die Last außerhalb von dieser einen, größten Belastungsspitze betreffen, haben keine Auswirkung. So existieren viele mögliche Produktionspläne mit dem gleichen Zielfunktionswert.

Dies führt zu Plateaus in der Zielfunktion, welche die Heuristiken überwinden müssen. Außerdem sollte eine Reduzierung der Belastung, auch wenn sie nicht die größte Belastungsspitze betrifft, immer mit einem besseren Zielfunktionswert einhergehen. Andernfalls wird eine Zielgröße vernachlässigt, die ohne Zielkonflikt hätte verbessert werden können.

Eine Alternative dazu ist die Minimierung der durchschnittlichen Belastung:

$$f_{\min\text{avg}}(\mathbf{P}) = \frac{1}{NA} \sum_n \sum_a MA_p^0(n, a) \quad (4)$$

Dabei werden alle Belastungswerte berücksichtigt, allerdings ist es möglich, dass einzelne Spitzen in Kauf genommen werden, um an anderer Stelle die Belastung stark zu senken.

Beide oben genannten Funktionen gehen davon aus, dass ausschließlich eine sehr knappe Zeit zur Bearbeitung belastend ist. Allerdings können auch lange Zeiten der Untätigkeit als belastend empfunden werden. In diesem Fall wäre es sinnvoller, die Belastung so zu verteilen, dass sie möglichst konstant ist. Dies kann erreicht werden, indem die Varianz der Belastung minimiert wird.

$$f_{\min\text{var}}(\mathbf{P}) = \text{Var}(MA_p^0(n, a)) \quad (5)$$

Die Belastung der Beschäftigten ist nicht die einzige Zielgröße, die im Anwendungsfall berücksichtigt werden muss. Im Projekt soll deshalb auch gezeigt werden, dass die humanzentrierten Zielfunktionen als Ergänzung zu betriebswirtschaftlichen Zielen genutzt werden können.

Bei Isringhausen müssen zudem die Deadlines der einzelnen Produkte eingehalten werden. Es wird eine Verspätung für jedes Produkt entsprechend dem Produktionsplan berechnet. Dazu wird für jedes Produkt j die Spalte s ermittelt in der das Produkt in \mathbf{P} steht. Die gewichtete Verspätung eines Produkts j , kann dann berechnet werden als:

$$g_j = \exp(ts + sA - d_j) \quad (6)$$

Die Verspätung wird hier mit der Exponentialfunktion gewichtet, um positive (und insbesondere hohe positive) Werte besonders stark zu bestrafen. Negative Verspätungen (pünktliche Produkte), werden durch die Exponentialfunktion ähnlich gewichtet, selbst wenn sie weit vor ihrer Deadline produziert werden. Das

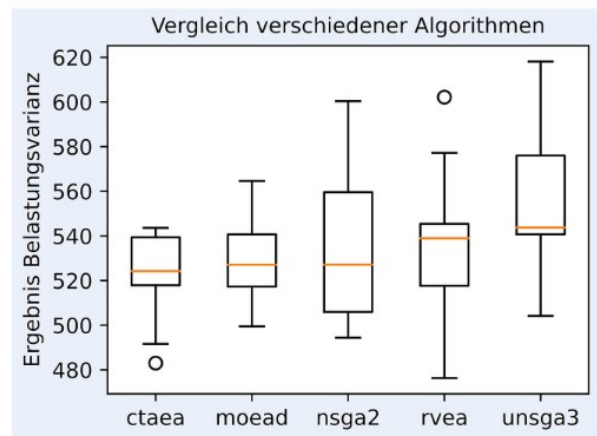


Bild 4. Ergebnisse der fünf Algorithmen bezüglich der Belastungsvarianz (Gleichung 5) auf einer Testinstanz. Grafik: FH Bielefeld

betriebswirtschaftliche Ziel der Pünktlichkeit lässt sich als Minimierung der Summe der einzelnen Verspätungen ausdrücken.

$$f_{\text{Verspätung}}(\mathbf{P}) = \sum_{j=1}^J g_j \quad (7)$$

Gleichung 7 beschreibt ein mögliches betriebswirtschaftliches Ziel. An diesem Beispiel wird im folgenden Kapitel gezeigt, wie die humanzentrierten Zielfunktionen mit den betriebswirtschaftlichen Zielen kombiniert werden können, um beide Perspektiven während der Planung zu berücksichtigen.

4 Ergebnisse

Im folgenden Abschnitt werden genetische Algorithmen genutzt, um die Auftragsreihenfolge zu optimieren. Dazu stehen zehn echte Planungsprobleme aus der Praxis zur Verfügung. Die Optimierung erfolgt durch moderne Varianten des genetischen Algorithmus, die speziell für eine multikriterielle Zielsetzung entwickelt wurden. Hierzu wurden mithilfe der Software-Bibliothek „pymoo“ [16] folgende Verfahren implementiert, welche oftmals in Benchmark-Studien eingesetzt werden und nachweislich gute Näherungslösungen identifizieren können: NSGA 2, MOEAD, RVEA, UNSGA3 und CTAEA.

Es wurde eine multikriterielle Zielfunktion zur Balancierung zweier Ziele mit möglichem Zielkonflikt verwendet. Das erste Ziel ist die Reduktion der Belastung und wird über eine der drei diskutierten Zielfunktionen modelliert (Gleichung 3–5). Dieses Ziel wird mit 80 % gewichtet. Das zweite Ziel ist die Minimierung der Verspätung (Gleichung 7) über alle Aufträge, welches mit 20 % gewichtet wurde. Auf diese Weise wird nicht nur die Belastung in der Linie selbst minimiert (erstes Ziel), sondern auch Stress in der Versandlogistik verhindert. Jede Heuristik wurde zehnmal auf allen Instanzen für 5000 Iterationen getestet. **Bild 4** zeigt die Ergebnisse der fünf verglichenen Algorithmen auf einer Problem Instanz.

Im Median (gelb markiert) ist die Varianz der Belastung für die Beschäftigten bei allen fünf Algorithmen ähnlich. Allerdings zeigen sich Unterschiede in der Varianz der Ergebnisse. CTAEA und MOEAD zeigen weniger Varianz als die übrigen Algorithmen. Die Unterschiede im Median der verschiedenen Algorithmen sind gering. Auf Grundlage dieser Experimente scheint es

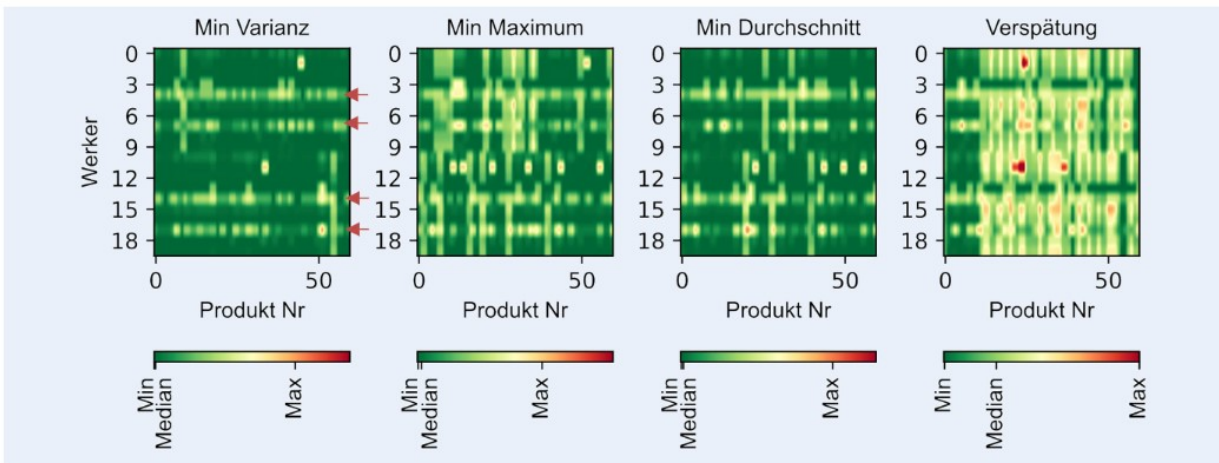


Bild 5. Verteilung der Belastungen (Gleichung 2) nach Optimierung mit einem humanzentrierten Ziel. Im rechten Bild wurde ausschließlich die Verspätung minimiert. Grafik: FH Bielefeld

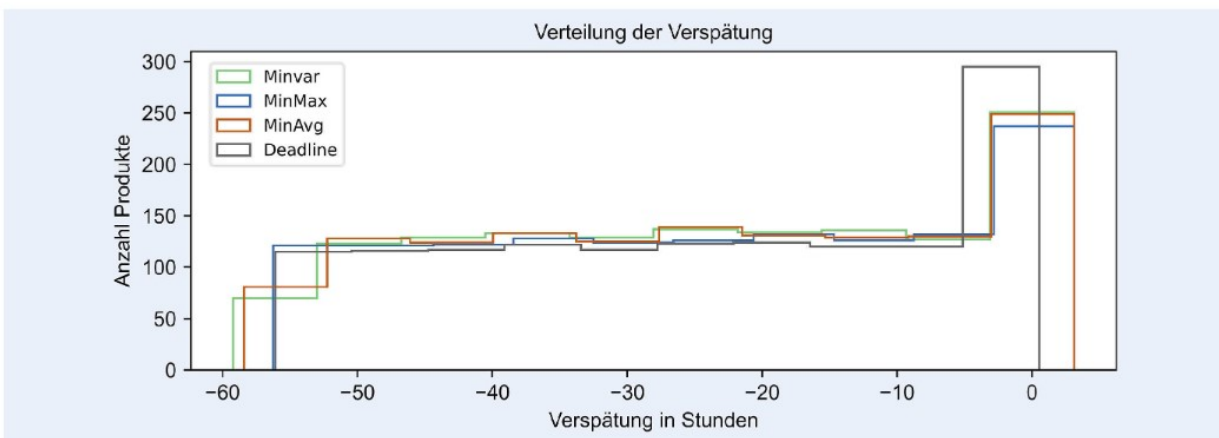


Bild 6. Histogramm der Verspätung mit verschiedenen Optimierungszielen. Grafik: FH Bielefeld

sinnvoll, CTAEA für eine Optimierung der Produktion zu verwenden. Insbesondere im realen Betrieb, wo mehrmals pro Schicht geplant werden soll, ist die geringe Varianz entscheidend, um Fehlplanungen zu vermeiden.

In **Bild 5** sind die Belastungen (gleitender Durchschnitt) für eine Probleminstanz als Heatmap dargestellt.

Jede Zeile markiert dabei einen Arbeitsplatz. In den Spalten stehen die Produkte in der Reihenfolge, in der sie laut Produktionsplan an den Arbeitsplätzen ankommen. Eine hohe Belastung wird rot markiert, grün markiert sind geringe Belastungswerte. Verglichen werden die drei verschiedenen Formulierungen des humanzentrierten Ziels (Gleichung 3–5) auf derselben Probleminstanz. Alle drei Zielfunktionen wurden als multikriterielles Ziel zusammen mit der Pünktlichkeit (Gleichung 7) angegeben. Das humanzentrierte Ziel wurde dabei mit 80 % gewichtet, Pünktlichkeit mit 20 %.

In der rechten Heatmap ist zum Vergleich rein nach dem Pünktlichkeitskriterium geplant worden. Die Lösung wurde in allen Fällen mit CTAEA ermittelt. Vergleicht man die drei Planungen mit humanzentrierten Zielfunktionen gegen das rechte Bild ist in allen eine deutliche Verbesserung der Belastungssituation zu erkennen. In allen Fällen werden große Belastungsspitze für ein-

zelne Werker (rote Werte in der Heatmap rechts) vermieden. Wie zu erwarten, zeigt die Minimierung der maximalen Belastung auch den geringsten maximalen Belastungswert. Die beiden anderen humanzentrierten Zielfunktionen ergeben einen ähnlichen Maximalwert. Der Median der Belastung konnte in allen drei Fällen stark reduziert werden. Vier Arbeitsplätze (Zeilen) wurden mit roten Pfeilen im linken Plot markiert. Es ist deutlich zu sehen, dass diese Arbeitsplätze auch in den anderen humanzentrierten Optimierungen durch hohe regelmäßige Belastungen hervorstechen. Es ist davon auszugehen, dass diese Arbeitsplätze ein Bottleneck bilden. Hier müssen für viele Produkte aufwendige Tätigkeiten durchgeführt werden, sodass die Belastung nur schwer zu senken ist. Sieht man von diesen Zeilen ab, scheint die Minimierung der Varianz die besten Ergebnisse an den anderen Arbeitsplätzen zu liefern. Hier gibt es nur wenige Werte außerhalb der markierten Arbeitsplätze, welche nicht nahe an der optimalen Belastung liegen.

Bild 6 zeigt die Verteilung der Verspätung in den vier Lösungen, die bereits in **Bild 5** gezeigt wurden.

In Grau ist die Planung rein nach Termin zu sehen. Die farbigen Linien zeigen die Ergebnisse unter Berücksichtigung eines humanzentrierten Ziels sowie der Pünktlichkeit. Alle drei human-

zentrierten Planungen lassen zu, dass Produkte deutlich zu spät sind. Eine Planung rein nach Termin zeigt hier erwartungsgemäß eine höhere Liefertreue. Untereinander unterscheiden sich die Planungen mit humanzentrierten Zielen kaum hinsichtlich der Verteilung der Verspätung.

5 Zusammenfassung und Diskussion

Dieser Beitrag behandelt eine kapazitätsorientierte Ablaufplanung von parallelen Transferstraßen unter Berücksichtigung neuartiger humanzentrierter Zielstellung. Vor allem konnten drei humanzentrierte Zielgrößen erarbeitet werden. Die zur Berechnung notwendigen Informationen sind lediglich die Arbeitszeit pro Produkt und MTM-Zeiten. Beide Informationen sind in vielen Unternehmen bereits verfügbar. Somit können die vorgestellten Zielfunktionen ohne großen Mehraufwand in bestehende Planungsheuristiken eingebunden werden. Im Anschluss wurde die Zielfunktion zusammen mit einem betriebswirtschaftlichen Ziel als multikriterielle Zielfunktion verwendet.

Auf Grundlage von echten Probleminstanzen bei der Firma Isringhausen konnte das Potenzial im Anwendungsfall aufgezeigt werden. In den getesteten Probleminstanzen konnten Belastungsspitzen deutlich reduziert werden. So wird ein Mehrwert für die Beschäftigten im Anwendungsfall sichergestellt, ohne dass andere betriebswirtschaftliche Ziele in Mitleidenschaft geraten. Zudem kann für interessierte Unternehmen mit ähnlicher Fragestellung eine klare Handlungsempfehlung bezüglich der Algorithmen ausgesprochen werden. Ein moderner genetischer Algorithmus hat in den Experimenten die kleinste Varianz und im Median eine gute Lösungsqualität gezeigt.

Der Beitrag legt damit das Fundament für die humanzentrierte Produktionsplanung von parallelen Transferstraßen und zeigt wie diese unkompliziert mit modernen Verfahren durchgeführt werden kann. Zur weiteren Validierung des Ansatzes sollten die Effekte des neuen Planungsparadigmas auf den Stress und die Zufriedenheit erhoben und bewertet werden. Die vorgestellte Methodik verlässt sich auf MTM-Zeiten als Indikator für die Stressbewertung. So kann nur Zeitdruck als Stressfaktor erfasst werden, zwischen anstrengenden oder weniger anstrengenden Tätigkeiten kann nicht unterschieden werden.

In weiteren Arbeiten sollte dieses Maß daher um ergonomische Faktoren erweitert werden, um ein differenzierteres Bild der Belastung zu erhalten. Auch sollte die Methode in zukünftigen Studien auf komplexere Szenarien, andere Produktionstypen und auf Basis weiterer Nachhaltigkeitsfaktoren übertragen werden. So kann beispielsweise untersucht werden, wie eine humanzentrierte Optimierung in flexiblen Werkstattorganisationen oder unter Berücksichtigung von Unsicherheiten durchgeführt werden kann. Schwankende Bearbeitungszeiten oder dynamische Ereignisse wie Störungen können in der Planung robuster Realweltprozesse mit menschlicher Beteiligung eine entscheidende Nebenbedingung darstellen. Um dies abzubilden, könnten zukünftige Forschungsarbeiten realistische Simulationsmodelle in die Optimierungsverfahren integrieren. In diesem Zusammenhang ist auch die Balancierung mit weiteren ökologischer Metriken wie zum Beispiel Energieeffizienz zu untersuchen. Relevanz hat auch die Frage, wie anstelle von Stammdaten die Planungsparameter situativ und präzise vorhergesagt werden können, um noch bessere Planungsergebnisse zu erhalten.

Literatur

- [1] Günther, H.-O.: Produktion und Logistik. Berlin: Springer-Verlag 2005
- [2] Boes, A.; Ziegler, A.: Umbruch in der Automobilindustrie. Analyse der Strategien von Schlüsselunternehmen an der Schwelle zur Informationsökonomie. Stand: 2021. Internet: <https://idguzda.de/wp-content/uploads/2021/06/Forschungsreport-Umbruch-in-der-Automobilindustrie.pdf>. Zugriff am 20.04.2023
- [3] Büchler, J.-P. (Hrsg.): Fallstudienkompendium Hidden Champions. Innovationen für den Weltmarkt. Wiesbaden: Springer Fachmedien 2018
- [4] Lindner, D.: KMU im digitalen Wandel. Ergebnisse empirischer Studien zu Arbeit, Führung und Organisation. Wiesbaden: Springer Gabler 2019
- [5] Naderi, B.; Ruiz, R.: The distributed permutation flowshop scheduling problem. *Computers & Operations Research* 37 (2010) 4, pp. 754–768
- [6] Ali, A.; Gajpal, Y.; Elmekawy, T. Y.: Distributed permutation flowshop scheduling problem with total completion time objective. *OPSEARCH* 58 (2021) 2, pp. 425–447
- [7] Bargaoui, H.; Belkahlia Driss, O.; Ghédira, K.: A novel chemical reaction optimization for the distributed permutation flowshop scheduling problem with makespan criterion. *Computers & Industrial Engineering* 111 (2017), pp. 239–250
- [8] Lin, S.-W.; Ying, K.-C.; Huang, C.-Y.: Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm. *International Journal of Production Research* 51 (2013) 16, pp. 5029–5038
- [9] Gao, J.; Chen, R.: A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem. *International Journal of Computational Intelligence Systems* 4 (2011) 4, pp. 497–508
- [10] Zhang, G.; Xing, K.: Differential evolution metaheuristics for distributed limited-buffer flowshop scheduling with makespan criterion. *Computers & Operations Research* 108 (2019), pp. 33–43
- [11] Guo, H.; Sang, H.; Zhang, B. et al.: An effective metaheuristic with a differential flight strategy for the distributed permutation flowshop scheduling problem with sequence-dependent setup times. *Knowledge-Based Systems* 242 (2022), #108328
- [12] Li, W.; Li, J.; Gao, K. et al.: Solving robotic distributed flowshop problem using an improved iterated greedy algorithm. *International Journal of Advanced Robotic Systems* 16 (2019) 5, doi.org/10.1177/1729881419879819
- [13] Perez-Gonzalez, P.; Framinan, J. M.: A review and classification on distributed permutation flowshop scheduling problems. *European Journal of Operational Research* (2023) [in press], doi.org/10.1016/j.ejor.2023.02.001
- [14] Katoch, S.; Chauhan, S. S.; Kumar, V.: A review on genetic algorithm: past, present, and future. *Multimedia tools and applications* 80 (2021) 5, pp. 8091–8126
- [15] Gendreau, M.; Potvin, J.-Y. (eds.): *Handbook of metaheuristics*. New York: Springer Cham 2010, doi.org/10.1007/978-3-319-91086-4
- [16] Blank, J.; Deb, K.: *Pymoo: Multi-Objective Optimization in Python*. *IEEE Access* 8 (2020), pp. 89497–89509



Lukas Vollenkemper 
Foto: Autor

Felix Grumbach 

Martin Kohlhase

Pascal Reusch 

Fachhochschule Bielefeld Center
for Applied Data Science
Schulstr. 10, 33330 Gütersloh
Tel. +49 5241 / 21143-85
lukas.vollenkemper@fh-bielefeld.de
oder felix.grumbach@fh-bielefeld.de
www.hsbi.de/iium/cfads/en

2.3 Publikation 3: Robustness Prediction in Dynamic Production Processes – A new Surrogate Measure based on Regression Machine Learning

Status




Veröffentlicht

Bibliographische Angabe

Felix Grumbach, Anna Müller, Pascal Reusch und Sebastian Trojahn (2023). „Robustness Prediction in Dynamic Production Processes - A New Surrogate Measure Based on Regression Machine Learning“. In: *Processes* 11.4. DOI: 10.3390/pr11041267.

Article

Robustness Prediction in Dynamic Production Processes—A New Surrogate Measure Based on Regression Machine Learning

Felix Grumbach ^{1,*} , Anna Müller ¹ , Pascal Reusch ¹  and Sebastian Trojahn ² 

¹ Center for Applied Data Science (CfADS), Bielefeld University of Applied Sciences, 33330 Gütersloh, Germany

² Department of Economics, Anhalt University of Applied Sciences, 06406 Bernburg, Germany

* Correspondence: felix.grumbach@fh-bielefeld.de (F.G.)

Abstract: This feasibility study utilized regression models to predict makespan robustness in dynamic production processes with uncertain processing times. Previous methods for robustness determination were computationally intensive (Monte Carlo experiments) or inaccurate (surrogate measures). However, calculating robustness efficiently is crucial for field-synchronous scheduling techniques. Regression models with multiple input features considering uncertain processing times on the critical path outperform traditional surrogate measures. Well-trained regression models internalize the behavior of a dynamic simulation and can quickly predict accurate robustness (correlation: $r > 0.98$). The proposed method was successfully applied to a permutation flow shop scheduling problem, balancing makespan and robustness. Integrating regression models into a metaheuristic model, schedules could be generated that have a similar quality to using Monte Carlo experiments. These results suggest that employing machine learning techniques for robustness prediction could be a promising and efficient alternative to traditional approaches. This work is an addition to our previous extensive study about creating robust stable schedules based on deep reinforcement learning and is part of the applied research project, Predictive Scheduling.

Keywords: surrogate measure; regression model; computational cost reduction; uncertainty; stochastic processing times; robust scheduling



Citation: Grumbach, F.; Müller, A.; Reusch, P.; Trojahn, S. Robustness Prediction in Dynamic Production Processes—A New Surrogate Measure Based on Regression Machine Learning. *Processes* **2023**, *11*, 1267. <https://doi.org/10.3390/pr11041267>

Academic Editor: Xiong Luo

Received: 15 March 2023

Revised: 14 April 2023

Accepted: 16 April 2023

Published: 19 April 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

1.1. Production Environments and Schedule Representations

In order to highlight the relevance of an efficient robustness estimation, production environments and the representability of production schedules must be introduced. Production schedules can be generated by mathematical methods for sequencing activities by assigning them to limited resources in order to optimize an objective function [1]. To depict different forms of manufacturing with specific constraints, a variety of optimization models have been developed. Permutation flow shops (PFSs), which can often be found in the industry, are suitable as an introductory example. According to Rossi et al. [2], the following constraints apply in traditional PFSs regardless of uncertainties:

1. Every production job has m operations (activities), each to be processed on a different machine (resource);
2. Within each job, the machines must be visited in a sequential order $1, \dots, m$;
3. Operations of a job can only start when any previous operations of the job have been completed;
4. Each machine can only process one operation at a time;
5. The job order is equal for all machines (jobs cannot overtake each other in the line);
6. All operations must be processed exactly once and non-preemptively.

Common deterministic objectives minimize the makespan ($\min C_{max}$), flow time or total tardiness of jobs. Although the model is comparatively easy to formulate and has

only one decision variable (order of jobs on the first machine), it is a non-deterministic polynomial-time (NP) hard problem [2].

Other well-known production environments are the flow shop (FS), and job shop (JS) as well as their flexible forms. The FS problem is a generalization of the PFS problem without considering constraint 5. Accordingly, jobs in the production line can overtake each other. The JS is another generalization of the FS without considering constraint 2. As a result, the jobs can be based on different material flows with different machine orders. In flexible environments, a machine must be selected from a set of alternatives for each operation. This makes the problems even more difficult to solve, since a resource allocation decision must also be made in addition to the sequencing decision [1].

Regardless of specific constraints and decision variables, a feasible solution to these problems can be represented as a directed acyclic solution graph (DASG) $G(O, \Psi)$ [3], where O is the set of scheduled operations, and Ψ is the set of directed predecessor relationships between the operations. For traditional production scheduling models with one resource type and $m \geq 2$ machines, every operation vertex in the DASG has $\Phi \in \{0, 1, 2\}$ outgoing and incoming edges. Figure 1 shows different exemplary solutions to PFS, FS and JS problems with three jobs and two machines. In addition to the DASG, a Gantt chart is presented at the top to visualize the assigned operations on the timeline. The red arrows highlight the critical path of the schedule. Dashed arrows represent job successors, and solid arrows represent machine successors.

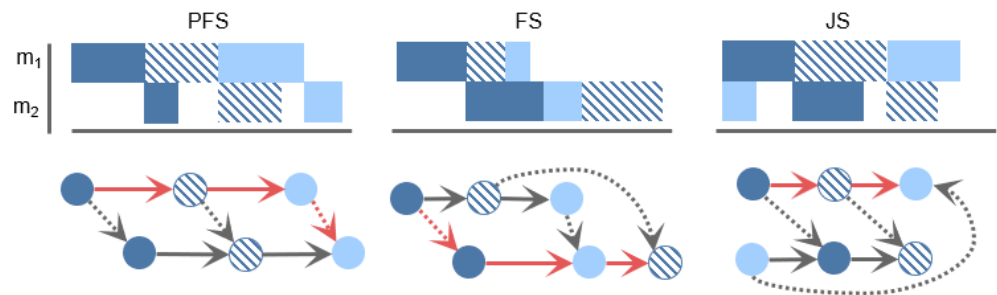


Figure 1. Exemplary feasible solutions to (flexible) PFS, FS and JS problems with three jobs and two machines (Gantt and DASG representation).

1.2. Robustness Measures

Since many real-world production processes are influenced by uncertainties, there has been a substantial amount of research on scheduling in dynamic environments that takes into account factors such as uncertain processing times (UPTs) or dynamic events such as machine failures. In contrast to reactive rescheduling, robust (syn. proactive, predictive) scheduling plays a crucial role in anticipating uncertainties in advance. The aim is to create a robust schedule (RS) in such a way that occurring events change its baseline structure as minimally as possible [4–7].

Consequently, a robust scheduling problem is an extension of its deterministic counterpart considering uncertainties and further objectives for robustness optimization [8]. Generally formulated, a baseline objective (min Z) is balanced with a robustness objective (min R). In addition, a number of constraints $\Theta_1, \dots, \Theta_\eta$ with stochastic parameters are considered:

$$\begin{aligned}
 &\min Z \\
 &\min R \\
 &\text{s.t.} \\
 &\Theta_1, \dots, \Theta_\eta
 \end{aligned}$$

Regarding R , prior studies such as [4,9] established a definition for robustness: A schedule is robust when the performance of a realized robust schedule (RS*) deviates as

minimally as possible from the performance of its baseline RS . The performance deviation refers to a schedule metric Z (e.g., C_{max}) before and after realization:

$$R = RS_Z - RS_Z^*$$

In order to obtain a reliable result, Monte Carlo experiments (*MCEs*) are required to determine uncertainty effects in complex stochastic settings. Alternatively, surrogate measures (*SMs*) can be used to quickly but inaccurately estimate R (see Section 1.3). In connection with positive robustness values ($R > 0 \iff RS_Z > RS_Z^*$), the RS was planned too conservatively. Otherwise, if $R < 0 \iff RS_Z < RS_Z^*$, the RS is too optimistic. Figure 2 visualizes this effect: After realization of a schedule, there is a deviation between the planned and actual makespan ($RS_{C_{max}} < RS_{C_{max}}^*$). Consequently, the RS was planned too optimistically. The figure also shows machine idle times x_1, \dots, x_3 , where an operation is not assigned to the resource. x_2 is a slack time between two operations that can be used to extend the preceding operation without affecting the overall structure of the schedule.

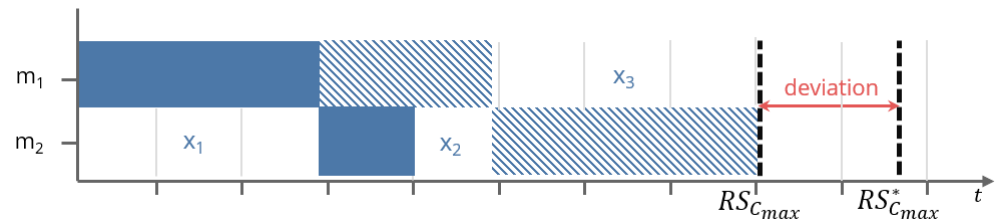


Figure 2. Exemplary schedule with two jobs and two machines (Gantt representation).

In the context of robust scheduling, several authors consider stability S as a second measure. According to most authors, a schedule is stable when the realized operations $o \in RS_O^*$ are terminated as defined a priori [9]. Thus, S can be calculated by the sum of absolute deviations of operation completion times o_e :

$$S = \sum_{i=1}^{|RS_O^*|} |(RS_{O_i})_e - (RS_{O_i}^*)_e|$$

Robustness R and stability S have been identified as conflicting measures [9–12]. Moreover, both measures are in a triangular conflict with several baseline objectives [10,12]. For this reason, it is recommended that robust scheduling methods balance these three dimensions. Complementary to robustness measures, another part of the literature took focus on fuzzy numbers, where different model parameters such as processing times or due dates are represented as fuzzy intervals [13,14]. The practical advantage of using fuzzy numbers is the easy calculability of operation completion times [3] or waiting times [15]. However, the calculation of robustness is likely not trivial in the context of more complex uncertainty models and cannot be performed analytically [16]. This suggests that the robust scheduling literature beyond fuzzy intervals mainly utilized *SMs* or *MCEs*.

1.3. Robust Scheduling Techniques

As this work is supplementary to our prior research, we direct readers to the section “Literature Review” presented in [17], which provides an overview of relevant studies in the field of robust scheduling. The studies have been classified based on the following four dimensions:

- Scheduling stage: (A) RS generation in a second stage based on a given *DASG*. (B) Aggregated with baseline objective optimization.
- Optimization technique: (A) Adding buffer times to critical operations. (B) Creating neighbor solutions, e.g., by swapping operations.
- Evaluation technique: (A) Obtaining robustness by *MCE*. (B) Estimating robustness by *SMs*.

- Robustness objective: (A) Standalone robustness [R]. (B) Robustness and stability [R, S]. (C) Robustness, stability and baseline objective [R, S, Z].

This study pays special attention to the third dimension, the evaluation technique. Either R can be estimated by using SMs (see [5,8,16]) or by performing $MCEs$ (see [4,10,11,18]). On the one hand, $MCEs$ require a lot of computational time but generate reliable results [16]. On the other hand, SMs , which are arithmetic measures to approximate R or S without performing $MCEs$, can be calculated quickly but show an insufficient accuracy [8]. Traditional SMs are total or free slack time per operation, number of critical path operations per machine and average makespan per neighbor RS [19]. Zahid et al. [20] provided a comprehensive survey of surrogate measures utilized both in production and project scheduling. Their objective was to investigate various parameters that can aid the assessment of robustness in the context of tardiness in project scheduling. Their results underline the importance of total slack. More sophisticated SMs are presented in [8,16,21]. Xiao et al. [8] presented two new surrogate measures that both consider the estimated robustness degradation of the critical path as well as of the non-critical path. The authors highlighted the importance of the inspection of the non-critical path, since other studies such as [22] only considered operations on the critical path of a schedule. In a follow-up study [16], the authors analyzed how the expected delay in the schedule realization is propagated downstream the schedule. Both studies share a common set of limitations. Firstly, they necessitate a parameter set by a decision maker. Secondly, the proposed surrogate measures are exclusively applicable for normally distributed processing times. In each study, the authors recommended further research for different distribution types. Yang et al. [21] criticized the fact that many SMs under-utilize the available information contained in a schedule. Thus, the authors utilized a single-hidden layer feedforward network with an extreme learning algorithm to predict the robustness. Three types of information are integrated in the prediction, namely probability of machine breakdown, workload and location of total slack. The proposed method eliminates the need for manual weighting of the three factors. The authors recommended extending this approach to other types of uncertainties, such as uncertain processing times. To the best of our knowledge, this is the only approach that utilizes machine learning (ML) for robustness estimation. In contrast to these highly specific surrogate measures designed for an individual class of problems, Himliche et al. [23] proposed a modeling framework based on stochastic timed automata that enables robustness evaluation based on evaluation and model-checking techniques. The primary benefit of this approach lies in its versatility, as it can be easily adapted to different types of workshops and different types of uncertainties. Conversely, a significant shortcoming is the computation time, which dramatically increases the the size of the workshop.

1.4. Motivation and Contribution of This Work

The study is motivated by lack methods to predict robustness measures, which are efficient and accurate at the same time. According to Section 1.3, robust scheduling methods utilize computationally intensive $MCEs$ to obtain these measures, or SMs , which are either imprecise due to under-utilization of available information or are highly specific and thus not applicable to a wide range of problems. This leads to decisive disadvantages and shortcomings in field-synchronous scheduling methods. Thus, it should be investigated how the advantages of accuracy and computational efficiency can be combined. The contribution of this work is a method that merges the beneficial attributes of the methods outlined in Section 1.3. It combines

- the analysis of disturbance propagation throughout the schedule [16];
- the utilization of different types of information in combination with ML [21];
- and the versatility presented in [23].

The approach utilizes regression models (RM s) to predict the makespan robustness ($R = RS_{C_{max}} - RS_{C_{max}}^*$) based on generic $DASGs$ with gamma distributed $UPTs$. Operations, machines, noisy operations end times, idle and slack times are considered as a priori input features for the RM s. In Section 2, the scope of the method is narrowed down. Moreover,

hypotheses for validating the method are presented. Subsequently, details of the *RMs*, the training data and the implementation are described. According to the hypotheses, the results are presented and discussed in Section 3. It could be shown that *RMs* can predict very accurate robustness scores in a very short computing time and without computationally intensive *MCEs*.

2. Method

In this chapter, we first present the scope and utility of the proposed method and formulate hypotheses for verifying its contribution. Subsequently, we describe mathematical modeling, implementation specifics, and training data related to the *RMs*.

2.1. Scope and Utility

In general, the prediction method is independent of the specific baseline problem, since it is already based on the *DASG* representation of a generated schedule. It can therefore be used for job shops and flow shops with $m > 1$ machines regardless of specific constraints and baseline objectives. Due to the universal calculation of uncertainties and their effect on the critical path (see Section 2.4), the method can also be applied to other types of distributions. However, further uncertainties such as dynamic release times or due dates are not taken into account in this proof-of-concept study.

To illustrate the utility and applicability of the method, Algorithm 1 provides a simplified robust scheduling trajectory search. It is an abstract algorithm to show the basic procedure of many current heuristics to optimize a baseline objective and/or robustness measures. Here, a search method with \bar{n} iterations is conducted, which successively generates neighborhood solutions of a schedule to identify a good local optima considering stochastic and deterministic objectives. When utilizing *MCEs* in the *EVAL* function to obtain the robustness, a large number of simulations is required in complex environments (see Section 2.5). It is even conceivable that the number of *MCEs* \bar{m} is larger than the number of search iterations \bar{n} . This leads to a complexity of $O(\bar{n}\bar{m}) \approx O(\bar{n}^2)$, which can be a crucial bottleneck in reactive real-time environments with extensive simulations. Integrating a *RM* into the *EVAL* function, the number of simulations can be substituted with a singular prediction, which corresponds to a reduced complexity of $O(\bar{n})$. In this way, the same complexity prevails, as is the case with *SMs*.

Algorithm 1 Exemplary local search robust scheduling heuristic (simplified pseudocode)

```

1:  $RS_1 \leftarrow CIS(O)$                                 ▷ Create initial schedule based for a set of operations  $O$  (output: DASG)
2:  $RS_1^* \leftarrow EVAL(RS_1)$                           ▷ Evaluate the initial schedule considering uncertainties
3: for  $i \leftarrow 1, i \leq \bar{n}, i \leftarrow i + 1$  do    ▷ Conduct a local search for  $\bar{n}$  iterations
4:    $RS_2 \leftarrow CNS(RS_1)$                           ▷ Create neighbor solution, e.g., by changing the operation order
5:    $RS_2^* \leftarrow EVAL(RS_2)$                         ▷ Evaluate the neighbor
6:   if  $FIT(RS_2^*) > FIT(RS_1^*)$  then
7:      $RS_1 \leftarrow RS_2$                             ▷ Replace the current schedule if the neighbor has a better fitness
8:   end if
9: end for
10: return  $RS_1^*$ 

```

2.2. Hypotheses and Statistical Measures

In order to concretize the contribution of this study (see Section 1.4), the following hypotheses are formulated and tested in the experiments:

- H1 If *EPTs* are utilized to create an *RS*, and $R \neq 0$, it is caused by deviating idle times between RS^* and *RS*.
- H2 If the *DASG* as well as all related *UPTs* are known, *R* can be predicted with little computational effort (without performing *MCEs*).
- H3 If *RMs* come close to the precision of *MCEs* and significantly outperform *SMs*, robust scheduling problems can be solved more efficiently.

To evaluate the performance of the *RMs*, we utilized a set of commonly used metrics in the *ML* literature. These are Mean square error (*MSE*), root mean square error (*RMSE*),

mean absolute error (*MAE*) and mean absolute percentage error (*MAPE*). The metrics are widely recognized for their effectiveness in evaluating model performance and are often used in combination in many studies [24]. *RMSE* and *MSE* are calculated based on the mean squared distances between predicted and actual variables. These metrics quantify prediction errors and are well suited for determining model accuracy [25]. However, because of the squared distances, they are sensitive to extreme values and are therefore less suitable for analyzing outliers. In contrast, *MAE* and *MAPE* are more suitable in this case, as they measure the absolute distances between the predicted and actual variables. [26]

Moreover, the Pearson coefficient r was used to test the correlation between actual and predicted values. Considering the number and independence of the input features, a coefficient of $0.7 \leq |r| < 1$ indicates a very high positive or negative correlation, while a value of $0.3 \leq |r| < 0.7$ indicates a moderate one with a fuzzy-firm linear rule [27]. The hypotheses H_1, H_2, H_3 are rejected if it is not possible to observe a very high absolute r value. H_3 is also rejected if the absolute r values of traditional *SMs* indicate a very high correlation. In order to ensure a good behavior to outliers, H_2 is rejected if the *RM*s do not obtain a low *MAPE* score ($MAPE < 10\%$) [28].

2.3. Uncertainty Modeling and Operation Parameters

In real-world production environments, *UPTs* are often distributed asymmetrically and can be approximated with Pearson distributions such as the gamma distribution [29]. In our proposed method, gamma distributions with a shape parameter α and a rate parameter $\beta = 1$ in combination with a minimum duration γ were utilized to depict *UPTs*. Each operation $o \in RS_O$ has an *UPT* distribution tuple o_U containing the parameters (α, γ) . The probability density function can be calculated as follows:

$$PDF(x) = \frac{\beta^\alpha}{(\alpha - 1)!} (x - \gamma)^{\alpha-1} e^{-\beta(x-\gamma)}$$

Based on this, the mean processing time $o_E = \frac{\alpha}{\beta} + \gamma$ and the standard deviation $o_\sigma = \sqrt{\frac{\alpha}{\beta^2}}$ are given. Moreover, each operation belongs to a job $o_j, j = \{1, \dots, n\}$ and to a machine $o_k, k = \{1, \dots, m\}$. According to the topological sorting of the schedule's *DASG*, each operation has an index $i = 1, \dots, |O|$. Given the *DASG*, an operation has a determined start time o_s and end time o_e . According to Φ (see Section 1.1), every operation has 0..2 direct predecessor operations $\overleftarrow{o}_j, \overleftarrow{o}_k$ and 0..2 direct successor operations $\overrightarrow{o}_j, \overrightarrow{o}_k$ located within a job j or on a machine k .

2.4. Prediction Features

In order to predict R , the a priori features $\omega_1, \dots, \omega_{11}$ were considered as an input for the *RM*s (see Table 1). In this context, a priori means that they can be derived from the (uncertain) parameters and the structure of the *DASG*. In other words, the features can be collected without *MCEs* or information about the *RS**.

Besides features concerning idle and slack times, so-called overlapping integrals λ of an operation's *PDF* were considered. Figure 3 shows an operation o and its machine successor \overrightarrow{o}_k . The *PDF* of o is depicted in orange. The shape of the *PDF* indicates that there exists a certain probability that operation o will cause a delay of operation \overrightarrow{o}_k . This probability of delay p_d is depicted in blue and can be calculated as follows given that *CDF* is the corresponding cumulative density function to *PDF*:

$$p_d = 1 - CDF(x'), x' = (\overrightarrow{o}_k)_s - o_s - o_\gamma$$

Given the probability of delay p_d , the expected delay E_d can be calculated as follows:

$$E_d = \left(\int_{x'}^{\infty} \frac{PDF'(x)}{p_d} dx \right) - x'$$

Table 1. RM features to predict R.

ω_1	Sum of idle times $\sum_{i=1}^{ RS_x } RS_{x_i}$
ω_2	Number of machines m
ω_3	Number of operations nm
ω_4	Number of overlapping integrals $\lambda \in \Lambda$ (see Figure 3)
ω_5	Standard deviation of all λ
ω_6	Mean λ
ω_7	Mean operation duration $\frac{1}{nm} \sum_{i=1}^{nm} (RS_{O_i})_E$
ω_8	Standard deviation of operation durations
ω_9	Mean free slack relative to nm
ω_{10}	Mean total slack relative to nm
ω_{11}	Total slack standard deviation

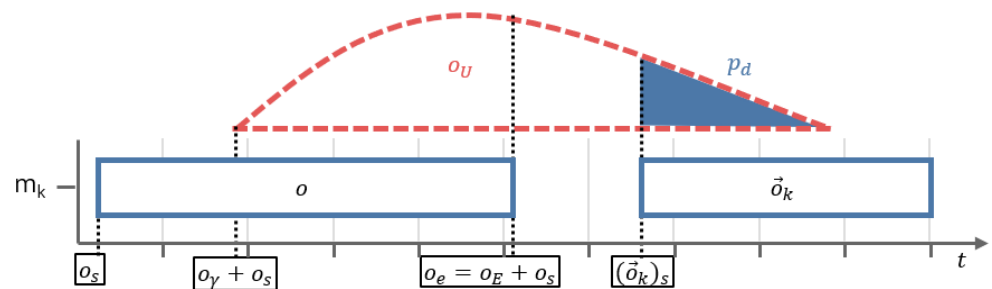


Figure 3. Two subsequent operations on a machine m_k . With a probability of p_d , the machine predecessor operation \vec{o}_k is shifted to the right on the time axis. Expected shift: E_d .

Then, the overlapping integral can be calculated as follows:

$$\lambda_{o, \vec{o}} = p_d * E_d$$

2.5. Test Data Generation

Benchmark studies about production scheduling have traditionally employed well-known test instances such as that proposed by Taillard [30]. However, this paper does not deal with the development of a scheduling algorithm but with the investigation of how robustness can be estimated with RMs regardless of a specific scheduling problem and method. In the robust scheduling literature, many authors such as [8,9,11] have generated custom test data to depict specific uncertainties. Since ML models require a sufficiently large quantity of training data with a broad consideration of different parameter scales (number of machines, jobs, uncertainties), we also generated custom data due to the lack of existing suitable instances.

In order to train and test the RMs, 500 generic RS* samples were created and labeled via MCEs (see Algorithm 2). Per generated DASG, a random number of jobs and machines was chosen (CH_1 function). For each operation, a random UPT was chosen from a set π containing different gamma distributions (CH_2 function). In order to create the DASG, the vector O was split into n equal-sized sectors, each representing a job with its k sequential operations. The job permutation was determined by the sector order $1, \dots, n$.

Algorithm 2 Generate one sample (pseudo code)

1: $n \leftarrow CH_1(3, 5, 7, 10)$	▷ Choose a random number of jobs
2: $m \leftarrow CH_1(2, 3, 5, 7)$	▷ Choose a random number of machines
3: $O \leftarrow CH_2(\pi, nm)$	▷ Choose nm non-unique random UPTs from π
4: $RS \leftarrow GS(O, k)$	▷ Generate RS with EPTs
5: $RS^* \leftarrow MCE(RS, O)$	▷ Evaluate RS* via MCE
6: return CF(RS, RS*)	▷ Collect features (see Section 2.4)

The RS* was then obtained by MCE, where the jobs were processed by DASG topological sorting. Per experiment, the processing time p of each operation was determined

randomly by the distribution $p \sim UPT$. The number of experiments ϵ to be carried out was determined automatically by an adaption of the standard deviation of the means method [31]. A lower precision bound of 2×10^{-3} and a mean $RS_{C_{max}}^*$ standard deviation $\bar{\sigma}$ was considered relative to the RS makespan. The benefit of this approach is that all samples are evaluated with a similar accuracy with respect to R .

$$\epsilon = \left\lceil \sqrt{\frac{\bar{\sigma}(RS_{C_{max}}^*)^{-1}}{2 * 10^{-3}}} \right\rceil$$

2.6. Regression Model Implementation

The following common and widely utilized RM s in the applied ML literature have been tested: linear regression (LR), random forest, support vector machine, gradient boosting and neural (deep feedforward) network. In the processing stage, min–max normalization was applied to all features. Due to the data-generation process (as described in Section 2.5), no further data cleaning was deemed necessary. Then, a univariate feature selection was performed using scikit-learn, resulting in reducing the feature space to the eleven most important features. All models were trained using a four-fold cross validation, with standard scikit-learn hyperparameters used for all models except the neural network. The neural network parameters, including network structure and size, learning rate and learning algorithm, were optimized using grid search.

3. Results

The following sections present and discuss the results from the numerical experiments, addressing the formulated hypotheses.

3.1. Idle Time Deviation

R and idle time deviations between RS and RS^* are certainly related, which confirms H_1 . It is to be assumed that $\forall o \in RS_O^* : o_e - o_s \approx o_E$. If $R \not\approx 0$, then the deviation between RS_Z and RS_Z^* is purely affected by idle times. Thus, the following equation precisely approximates R :

$$R_{C_{max}} \approx \frac{\Delta X}{m} = \frac{1}{m} \left(\sum_{i=1}^{|RS_X|} RS_{x_i} - RS_{x_i}^* \right) \tag{1}$$

It relativizes the sum of idle time deviations ΔX with regard to the number of machines m . A $RMSE < 1\%$ could be obtained, which can be explained by measurement errors from MCE (for more information see Table 2). Overlapping integrals $\lambda > 0$ lead to a leverage effect in schedules with idle times or more than one machine: If the actual operation completion time o_e lies inside an integral, it shifts and delays the successor operation \vec{o}_k or \vec{o}_j , which recursively shifts its direct successors as well [3]. Thus, a cumulative displacement of critical path operations is the reason for $\Delta X \neq 0$. In other words: the more overlaps, the noisier o_e, o_s but not o_E [32], and thus the noisier the idle times between operations. To sum up, this finding suggests to include idle times and overlapping UPT s as features in the RM s. By accounting for these factors, the models may be better able to capture the complex dynamics of the system and provide more accurate predictions.

3.2. Regression Model Validation

H_2 could be confirmed: RM s were able to appropriately predict makespan robustness based on a priori information including overlapping integrals. Table 2 shows the summary statistics of validation results of the implemented ML models. According to the obtained scores, all ML models had low prediction errors and a high robustness to outliers. Thus, it can be inferred that RM s generally appear to be well suited for predicting robustness. Even if a hyperparameter search was conducted for the neural network, LR still had the highest accuracy. One possible explanation is that the data exhibit a linear structure, which

is an advantage for *LR* models. In contrast, neural networks are more suitable for modeling complex nonlinear relationships [33].

Table 2. Comparison of robustness prediction models (results after cross-validation).

Method	MSE	RMSE	MAE	MAPE
Equation (1) (H_1)	0.00004	0.0065	0.0050	0.0095
<i>LR</i>	0.00202	0.0450	0.0363	0.0683
Neural network	0.00283	0.0525	0.0408	0.0877
Gradient boosting	0.00292	0.0539	0.0404	0.0903
Random forest	0.00410	0.0639	0.0481	0.1099
Support vector machine	0.00428	0.0654	0.0536	0.1017

To further examine the *LR* performance, Figure 4 displays a residual analysis. Although there are some minor outliers, the data points show a clear linear direction and a symmetric form with evenly distributed errors. No funnel shape or obvious nonlinear relationship can be observed in the residuals, which is an indicator that there is no heteroscedasticity or other dependencies. If we now turn to the coefficients of the trained model, we observed that ω_{10} (coefficient = 1.12) had the greatest influence on the prediction, which indicates a multicollinearity at first glance. Upon closer analysis, there is a high correlation with ω_{11} (coefficient = -0.69). However, without considering ω_{11} as a feature, the model's predictions noticeably worsen. Therefore, both the mean total slack and its associated standard deviation are important features for prediction. Other important features were ω_4 (coefficient = -0.63) and ω_1 (coefficient = 0.44), which confirms the association between overlapping *UPTs*, idle times and critical path operations. Overall, the analysis suggests that suitable features were selected to predict the makespan robustness. Moreover, the findings suggest that multiple different features are needed to define an accurate *SM*.

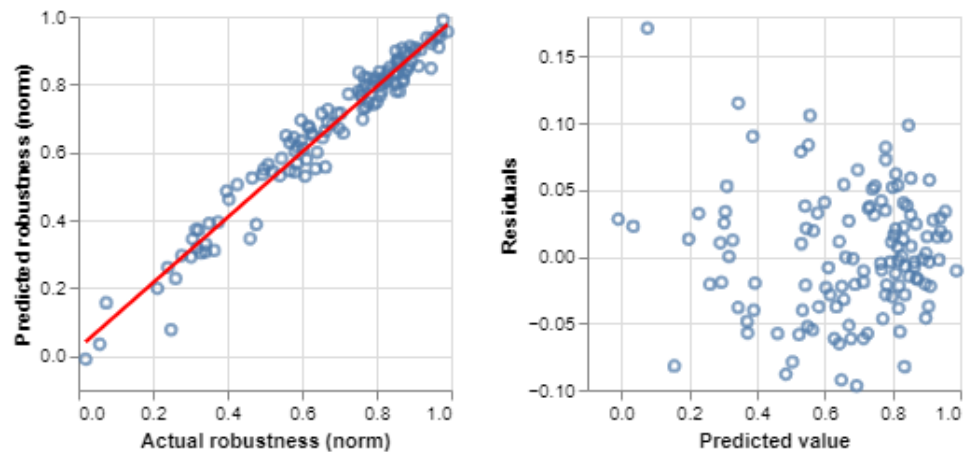


Figure 4. Residual analysis of the *LR* model (best identified *RM*) with 125 test samples.

3.3. Surrogate Measure Benchmark

H_3 could be confirmed in the context of the considered test data: *RM*s obtain very high correlations with actual makespan robustness and clearly outperform *SM*s. The *ML* models were benchmarked against three widely used *SM*s. Traditionally, most surrogate measures are based on slack, and the importance of slack for robustness estimation was also confirmed in [20]. As mentioned in Section 1.3, recent developments in the field of surrogate measures are problem-specific and thus cannot be readily applied to this benchmark. Our first *SM* for robustness was the average slack measure introduced in [4] and utilized by many researchers such as [8,9,12,16,34] up to today. The proposed measure is given by

$$SM_1 = \frac{1}{|RS_o|} \sum_{o \in RS_o} s_o$$

where s_o denotes the total slack of operation o . Total slack is defined as the units of time an operation can be delayed without increasing the makespan. However, since operations on the same path in a schedule share total slack, summation of total slack of all operations could potentially overestimate the capability of the buffer [20]. In [34], the authors proposed the total sum of free slack, which is defined as the units of time an operation can be delayed without delaying the start of the next operation SM . It is calculated as follows:

$$SM_2 = \sum_{o \in RS_o} f_o$$

where f_o denotes the free slack of operation o . The third SM we used was also introduced by [34]. SM_3 is given by the weighted sum of total slack for each operation

$$SM_3 = \sum_{o \in RS_o} \frac{w_{o_k}}{w_{total}} s_o$$

where w_{o_k} describes the workload of the machine operation o that it is processed on, w_{total} describes the total workload of all machines, and s_o is again the total slack of operation o .

Since the introduced SMs do not try to estimate the actual robustness of a schedule, but try to correlate with it, a correlation analysis of the SMs and the predicted robustness values of the ML -algorithms was conducted. As shown in Figure 5, the predictions of the ML algorithms strongly correlate with the actual makespan robustness, which was an expected result given the results from the previous experiment. The SMs , on the other hand, have only a moderate negative correlation with actual robustness. A possible explanation for this outcome is that common SMs tend to focus on a limited number of variables to assess robustness, which may be an overly simplistic assumption. However, in the previous section, it was found that several factors associated with the critical path affect the robustness. This includes not only individual measures of slack times or workloads, but also more detailed features to describe the schedule and its uncertainties as comprehensively as possible. In the next section, the compared methods are applied to a new scenario beyond the provided test data. Here, we investigate how the different approaches affect the actual quality of generated RSs .

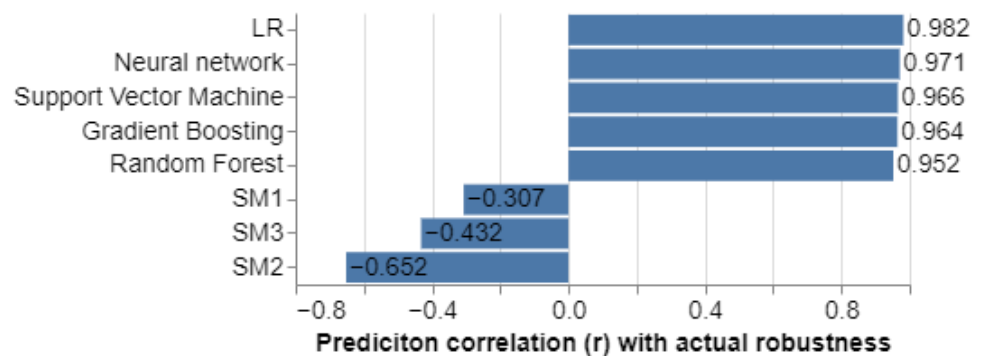


Figure 5. Correlation benchmark of selected RM s and SM s. Key finding: RM s clearly outperform SM s in the approximation of C_{max} robustness.

3.4. Application Study

To investigate the applicability and transferability of the RM s as well as their concrete advantages over SM s and MCE s, this experiment evaluates the proposed approach in the context of a typical robust scheduling procedure. We implemented a simulated annealing with research strategy (SARS) algorithm proposed by [17] to solve a robust PFS scheduling problem considering the constraints formulated in Section 1.1. The algorithm was executed

for 100 iterations, and per iteration, a solution neighbor was generated by swapping the positions of two randomly selected jobs. The following multi-criteria objective function is defined for the synchronous minimization of the expected makespan and the makespan robustness:

$$\begin{aligned} \min C_{max} &= \max_{o \in RS_O} \{ o_e \} \\ \min R_{C_{max}} &= \begin{cases} |RS_{C_{max}} - RS_{C_{max}}^*|, & \text{if MCEs are utilized to determine robustness} \\ |PREDICT(RS)|, & \text{if LR is utilized} \\ -SM_2, & \text{else} \end{cases} \end{aligned} \tag{2}$$

In terms of robustness evaluation, the specific value for Equation (2) can be determined by MCE, SM or by a RM. As SM₂ and LR exhibited the strongest correlations, they were selected as the models to be benchmarked. When selecting the best feasible solution, 80% priority (weight) was given to the robustness objective, and 20% was given to the makespan objective. In order to demonstrate the transferability of the method, we used a well-known test instance (*ta001*) introduced by [30] to evaluate the performance of the three methods. The instance consists of 20 jobs and 5 machines, and we modified it to include *UPTs*. Based on each operation processing time, we selected a random shape parameter $\alpha \in \{1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5\}$ and the scale parameter $\beta = 5$ to generate a gamma distribution (see Section 2.3).

Figure 6 depicts a benchmark of the three methods in terms of the specified problem. In addition to LR and SM₂, MCEs were performed to varying extents. In one case, 200 stochastic simulations were conducted per creation of a neighbor solution, while in the other case, 400 simulations were conducted. Due to the 100 iterations of the SARS procedure, a total of 20,000 and 40,000 simulations had to be performed. For each method, 25 samples were collected to analyze the mean and variability of the obtained values. A closer inspection of the plots confirms the assumption that a traditional SM, while computationally efficient, yields relatively poor results. This applies to both the average values and the variance achieved. Particularly for the robustness values achieved, there are sometimes unacceptable outliers, which explain the moderate correlation observed in experiment 3 (see Section 3.2). On the other hand, MCEs provide significantly better schedules, but this comes at the cost of a large number of simulations. Especially when using extensive material flow simulations with computation times of several seconds per run, this is a crucial and potentially unacceptable hurdle. Another noteworthy observation is the varying robustness spread between 200 and 400 simulations per iteration. Doubling the number of simulations leads to higher precision and greater reliability of the method to minimize actual robustness. However, this is a critical tradeoff in terms of computation time.

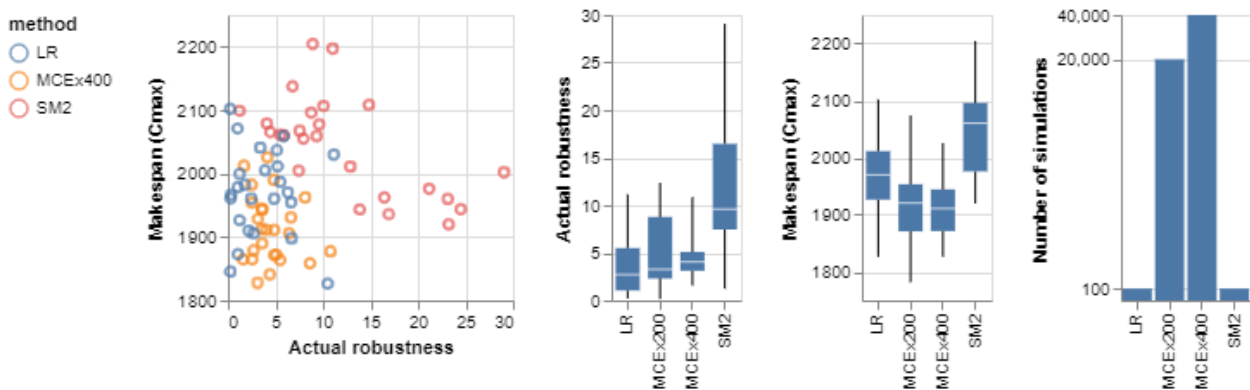


Figure 6. Comparing RSs generated by LR, SM₂ and MCE integrated into a SARS algorithm.

If we now turn to *LR*, the benefits of the method become apparent in the figure. Utilizing *ML* models, it is possible to generate *RSs* that are close in quality to the *MCE*-based schedules with a minimum of computing time. Although the achieved makespan is slightly worse compared to the *MCE*-based method, the average achieved robustness is surprisingly even better. However, this initially unexpected result is understandable with regard to a slight model bias. One possible explanation in this context is that the model provides a rather pessimistic prediction of robustness. On the one hand, this leads to the generation of even more *RSs*. On the other hand, however, it also affects the conflicting objective of minimizing makespan, as displayed in the figure. The same pattern can also be observed between *MCEx200* and *MCEx400*. Using fewer simulations results in a slightly less precise estimated robustness, but in comparison to *MCEx400*, it leads to a slightly better actual robustness and a slightly worse makespan. One possible solution to address this issue in this case is to use two plausible strategies. The first is to further improve the training of the *RM* to reduce the mentioned pessimism. Alternatively, the prioritization of the objectives could be adjusted in favor of minimizing makespan.

Overall, it has been confirmed that *SMs* generate significantly less optimal and less reliable schedules compared to what can be achieved with multivariate *RMs*. Thus, the results of this last experiment complement the results from the previous sections. In this specific scenario, it was possible to reconfirm H_3 from an application-oriented perspective. However, more scenarios must be investigated in future works.

4. Conclusions

This work employed *RMs* to predict makespan robustness for generic production schedules represented by *DASGs* and with the consideration of gamma-distributed *UPTs*. Within a computational study, three hypotheses H_1 , H_2 , H_3 could be confirmed: Overlapping uncertainty distributions on the critical path leads to a cumulative displacement of actual operation end times, which accordingly makes the idle time windows more noisy (see Section 3.1, H_1). This effect needs to be taken into account to construct feature vectors for *ML* models. In this manner, *RMs* and especially *LR* algorithms are suitable to accurately predict robustness with multiple a priori variables from the baseline *DASG* and are known before schedule realization or evaluation (see Section 3.2, H_2). Utilizing these models, well-known *SMs* could be significantly outperformed (see Sections 3.3 and 3.4, H_3). Overall, the models offer a promising alternative to previous approaches such as inaccurate *SMs* and computationally intensive *MCEs*. In particular, they combine the advantages of both approaches: A well-trained model is able to anticipate the output of runtime-intensive simulations with a single prediction. If it is integrated into an optimization procedure, the complexity can be significantly reduced without major loss in solution quality. From a practical point of view, time-critical robust scheduling procedures could be executed in an even more resource-saving manner without the need of stochastic material flow simulations. This is particularly relevant in the context of rescheduling procedures that need to react quickly during daily operations to create a new *RS*. However, further research is necessary to evaluate the external validity of the method in other scenarios. The method must be tested with other scheduling models, uncertainty modelings, robustness or stability measures. Although we assume that our method can be applied to other distribution types, the effect of dynamic events such as machine failures must to be analyzed more closely. Moreover, it would also be interesting to investigate how the method performs on real-world production data. Consequently, the results of this study clearly motivate further research on this topic.

Author Contributions: F.G.: Conceptualization, Methodology, Formal analysis, Investigation, Writing—Original Draft, Writing—Review and Editing. A.M.: Investigation, Formal analysis, Writing—Review and Editing. P.R.: Funding acquisition, Resources, Supervision. S.T.: Supervision. All authors have read and agreed to the published version of the manuscript.

Funding: Open Access funding enabled and organized by Projekt DEAL. This work was supported as part of the joint research project Human-centered Smart Service Lab/Predictive Scheduling (with project number (EFRE-030018) of the European Regional Development Fund), which is funded by the federated state North Rhine-Westphalia, Germany.

Data Availability Statement: Code and data relating to the work presented are fully available at <https://doi.org/10.17605/OSF.IO/T8EZY>.

Acknowledgments: We would like to express our gratitude to the State of North Rhine-Westphalia for providing the necessary third-party funding to support our research project *Predictive Scheduling*. We would also like to thank Miele & Cie. KG, Isringhausen GmbH & Co. KG, Moderne Industrietechnik GmbH & Co. KG, and PerFact Innovation GmbH & Co. KG for their collaboration and for providing us with exciting use cases. Lastly, we are grateful to the Fraunhofer IOSB-INA research institute for their valuable insights on the theory and application of machine learning and mathematical optimization in the context of production and logistics. We appreciate their contributions and look forward to continued collaboration in the future.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

DASG	Directed acyclic solution graph
EPT	Expected processing time
FS	Flow shop
JS	Job shop
LR	Linear regression
MCE	Monte Carlo experiment
ML	Machine learning
MAE	Mean absolute error
MAPE	Mean absolute percentage error
MSE	Mean square error
PFS	Permutation flow shop
RM	Regression model
RMSE	Root mean square error
RS	Robust schedule
RS*	Robust schedule after (MCE) realization
SM	Surrogate measure
UPT	Uncertain processing time

References

- Pinedo, M.L. *Scheduling*, 4th ed.; Springer US: New York, NY, USA, 2012; p. 1. <https://doi.org/10.1007/978-1-4614-2361-4>.
- Rossi, F.L.; Nagano, M.S.; Sagawa, J.K. An effective constructive heuristic for permutation flow shop scheduling problem with total flow time criterion. *Int. J. Adv. Manuf. Technol.* **2016**, *90*, 93–107. <https://doi.org/10.1007/s00170-016-9347-0>.
- Vela, C.R.; Afsar, S.; Palacios, J.J.; Gonzalez-Rodriguez, I.; Puente, J. Evolutionary tabu search for flexible due-date satisfaction in fuzzy job shop scheduling. *Comput. Oper. Res.* **2020**, *119*, 104931. <https://doi.org/10.1016/j.cor.2020.104931>.
- Leon, J.V.; Wu, D.S.; Storer, R.H. Robustness measures and robust scheduling for job shops. *IIE Trans.* **1994**, *26*, 32–43. <https://doi.org/10.1080/07408179408966626>.
- Davenport, A.J.; Gefflot, C.; Beck, J.C. Slack-based Techniques for Robust Schedules. In Proceedings of the Sixth European Conference on Planning (ECP-2001), Toledo, Spain, 12–14 September 2001.
- Ouelhadj, D.; Petrovic, S. A survey of dynamic scheduling in manufacturing systems. *J. Sched.* **2008**, *12*, 417–431. <https://doi.org/10.1007/s10951-008-0090-8>.
- Iglesias-Escudero, M.; Villanueva-Balsera, J.; Ortega-Fernandez, F.; Rodriguez-Montequín, V. Planning and Scheduling with Uncertainty in the Steel Sector: A Review. *Appl. Sci.* **2019**, *9*, 2692. <https://doi.org/10.3390/app9132692>.
- Xiao, S.; Sun, S.; Jin, J.J. Surrogate measures for the robust scheduling of stochastic job shop scheduling problems. *Energies* **2017**, *10*, 543. <https://doi.org/10.3390/en10040543>.
- Goren, S.; Sabuncuoglu, I. Robustness and stability measures for scheduling: Single-machine environment. *IIE Trans.* **2008**, *40*, 66–83. <https://doi.org/10.1080/07408170701283198>.
- Shen, X.N.; Han, Y.; Fu, J.Z. Robustness measures and robust scheduling for multi-objective stochastic flexible job shop scheduling problems. *Soft Comput.* **2016**, *21*, 6531–6554. <https://doi.org/10.1007/s00500-016-2245-4>.

11. Liu, F.; Wang, S.; Hong, Y.; Yue, X. On the Robust and Stable Flowshop Scheduling Under Stochastic and Dynamic Disruptions. *IEEE Trans. Eng. Manag.* **2017**, *64*, 539–553. <https://doi.org/10.1109/tem.2017.2712611>.
12. Sundstrom, N.; Wigstrom, O.; Lennartson, B. Conflict Between Energy, Stability, and Robustness in Production Schedules. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 658–668. <https://doi.org/10.1109/tase.2016.2643621>.
13. Behnamian, J. Survey on fuzzy shop scheduling. *Fuzzy Optim. Decis. Mak.* **2015**, *15*, 331–366. <https://doi.org/10.1007/s10700-015-9225-5>.
14. Delgoshaei, A.; Ariffin, M.K.A.B.M.; Leman, Z.B. An Effective 4-Phased Framework for Scheduling Job-Shop Manufacturing Systems Using Weighted NSGA-II. *Mathematics* **2022**, *10*, 4607. <https://doi.org/10.3390/math10234607>.
15. Goyal, B.; Kaur, S. Flow shop scheduling-especially structured models under fuzzy environment with optimal waiting time of jobs. *Int. J. Des. Eng.* **2022**, *11*, 47. <https://doi.org/10.1504/ijde.2022.127075>.
16. Xiao, S.; Wu, Z.; Dui, H. Resilience-Based Surrogate Robustness Measure and Optimization Method for Robust Job-Shop Scheduling. *Mathematics* **2022**, *10*, 4048. <https://doi.org/10.3390/math10214048>.
17. Grumbach, F.; Müller, A.; Reusch, P.; Trojahn, S. Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning. *J. Intell. Manuf.* **2022**. <https://doi.org/10.1007/s10845-022-02069-x>.
18. Soofi, P.; Yazdani, M.; Amiri, M.; Adibi, M.A. Robust Fuzzy-Stochastic Programming Model and Meta-Heuristic Algorithms for Dual-Resource Constrained Flexible Job-Shop Scheduling Problem Under Machine Breakdown. *IEEE Access* **2021**, *9*, 155740–155762. <https://doi.org/10.1109/access.2021.3126820>.
19. Minguillon, F.E.; Stricker, N. Robust predictive-reactive scheduling and its effect on machine disturbance mitigation. *CIRP Ann.* **2020**, *69*, 401–404. <https://doi.org/10.1016/j.cirp.2020.03.019>.
20. Zahid, T.; Agha, M.H.; Schmidt, T. Investigation of surrogate measures of robustness for project scheduling problems. *Comput. Ind. Eng.* **2019**, *129*, 220–227. <https://doi.org/10.1016/j.cie.2019.01.041>.
21. Yang, Y.; Huang, M.; Wang, Z.Y.; Zhu, Q.B. Robust scheduling based on extreme learning machine for bi-objective flexible job-shop problems with machine breakdowns. *Expert Syst. Appl.* **2020**, *158*, 113545. <https://doi.org/10.1016/j.eswa.2020.113545>.
22. Goren, S.; Sabuncuoglu, I.; Koc, U. Optimization of schedule stability and efficiency under processing time variability and random machine breakdowns in a job shop environment. *Nav. Res. Logist. (NRL)* **2011**, *59*, 26–38. <https://doi.org/10.1002/nav.20488>.
23. Himmiche, S.; Marangé, P.; Aubry, A.; Pétin, J.F. Robustness Evaluation Process for Scheduling under Uncertainties. *Processes* **2023**, *11*, 371. <https://doi.org/10.3390/pr11020371>.
24. Jierula, A.; Wang, S.; OH, T.M.; Wang, P. Study on Accuracy Metrics for Evaluating the Predictions of Damage Locations in Deep Piles Using Artificial Neural Networks with Acoustic Emission Data. *Appl. Sci.* **2021**, *11*, 2314. <https://doi.org/10.3390/app11052314>.
25. Singh, U.; Rizwan, M.; Alaraj, M.; Alsaïdan, I. A Machine Learning-Based Gradient Boosting Regression Approach for Wind Power Production Forecasting: A Step towards Smart Grid Environments. *Energies* **2021**, *14*, 5196. <https://doi.org/10.3390/en14165196>.
26. Chen, C.; Twycross, J.; Garibaldi, J.M. A new accuracy measure based on bounded relative error for time series forecasting. *PLoS ONE* **2017**, *12*, e0174202. <https://doi.org/10.1371/journal.pone.0174202>.
27. Ratner, B. The correlation coefficient: Its values range between +1/−1, or do they? *J. Target. Meas. Anal. Mark.* **2009**, *17*, 139–142. <https://doi.org/10.1057/jt.2009.5>.
28. Montaña, J.J.; Palmer, A.; Sesé, A.; Cajal, B. Using the R-MAPE index as a resistant measure of forecast accuracy. *Psicothema* **2013**, 500–506. <https://doi.org/10.7334/psicothema2013.23>.
29. Yamashiro, H.; Nonaka, H. Estimation of processing time using machine learning and real factory data for optimization of parallel machine scheduling problem. *Oper. Res. Perspect.* **2021**, *8*, 100196. <https://doi.org/10.1016/j.orp.2021.100196>.
30. Taillard, E. Benchmarks for basic scheduling problems. *Eur. J. Oper. Res.* **1993**, *64*, 278–285. [https://doi.org/10.1016/0377-2217\(93\)90182-m](https://doi.org/10.1016/0377-2217(93)90182-m).
31. Jacoboni, C.; Lugli, P. *The Monte Carlo Method for Semiconductor Device Simulation*; Springer: Vienna, Austria, 1989. <https://doi.org/10.1007/978-3-7091-6963-6>.
32. Kuroda, M.; Wang, Z. Fuzzy job shop scheduling. *Int. J. Prod. Econ.* **1996**, *44*, 45–51. [https://doi.org/10.1016/0925-5273\(95\)00091-7](https://doi.org/10.1016/0925-5273(95)00091-7).
33. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016; pp. 164–199. Available online: <http://www.deeplearningbook.org> (accessed on March 15, 2023).
34. Xiong, J.; Xing, L.N.; Chen, Y.W. Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. *Int. J. Prod. Econ.* **2013**, *141*, 112–126. <https://doi.org/10.1016/j.ijpe.2012.04.015>.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

2.4 Publikation 4: Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning

Status

Veröffentlicht

Bibliographische Angabe

Felix Grumbach, Anna Müller, Pascal Reusch und Sebastian Trojahn (2022). „Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning“. In: *Journal of Intelligent Manufacturing*. DOI: 10.1007/s10845-022-02069-x.



Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning

Felix Grumbach¹ · Anna Müller¹ · Pascal Reusch¹ · Sebastian Trojahn² Received: 8 June 2022 / Accepted: 7 December 2022
© The Author(s) 2022

Abstract

This proof-of-concept study provides a novel method for robust-stable scheduling in dynamic flow shops based on deep reinforcement learning (DRL) implemented with OpenAI frameworks. In realistic manufacturing environments, dynamic events endanger baseline schedules, which can require a cost intensive re-scheduling. Extensive research has been done on methods for generating proactive baseline schedules to absorb uncertainties in advance and in balancing the competing metrics of robustness and stability. Recent studies presented exact methods and heuristics based on Monte Carlo experiments (MCE), both of which are very computationally intensive. Furthermore, approaches based on surrogate measures were proposed, which do not explicitly consider uncertainties and robustness metrics. Surprisingly, DRL has not yet been scientifically investigated for generating robust-stable schedules in the proactive stage of production planning. The contribution of this article is a proposal on how DRL can be applied to manipulate operation slack times by stretching or compressing plan durations. The method is demonstrated using different flow shop instances with uncertain processing times, stochastic machine failures and uncertain repair times. Through a computational study, we found that DRL agents achieve about 98% result quality but only take about 2% of the time compared to traditional metaheuristics. This is a promising advantage for the use in real-time environments and supports the idea of improving proactive scheduling methods with machine learning based techniques.

Keywords Dynamic flow shop · Predictive scheduling · Proactive scheduling · Robust scheduling · Reinforcement learning · Simheuristics

Introduction

Job scheduling is a major area of interest within the field of production planning. A production organization that can often be observed in practice is the job shop, where each production job has a set of operations to be processed succes-

sively. Each operation is processed without interruption by a continuously available production resource (e.g. a machine), and each resource can only process one operation at a time. In the job shop scheduling problem, the objective is to define operation sequences in such a way, that an objective function (e.g., makespan or flow time minimization) is optimized. Another common form is the flow shop, where every job has the same processing order regarding the resources. A specialized form of the flow shop is the permutation flow shop, where the sequencing decision can only be made at the first resource (cf. transfer lines) (Naderi & Ruiz, 2010).

However, in realistic manufacturing environments, further planning parameters must be considered, which can be independent of the production organization. Especially dynamic events such as machine failures endanger baseline schedules and can require re-scheduling procedures. This leads to a loss of time, additional manufacturing costs, missing production goals, customer dissatisfaction and stress for employees (Liu et al., 2017; Mailliez et al., 2021; Shen et al.,

✉ Felix Grumbach
felix.grumbach@fh-bielefeld.de

Anna Müller
anna.mueller@fh-bielefeld.de

Pascal Reusch
pascal.reusch@fh-bielefeld.de

Sebastian Trojahn
sebastian.trojahn@hs-anhalt.de

¹ Center for Applied Data Science (CfADS), Bielefeld University of Applied Sciences, Gütersloh, Germany

² Department of Economics, Anhalt University of Applied Sciences, Bernburg, Germany

2016). Dynamic events can be resource-dependent (internal events such as uncertain setup and processing times, machine failures, uncertain material delivery windows or unplanned rework) or customer-related (external events such as new unplanned orders, order cancellations, changed due dates or order priorities) (Rahmani & Heydari, 2014).

For this reason, there is a need for dynamic or stochastic scheduling methods that take into account uncertainties. Proactive scheduling, also called robust or predictive scheduling, plays a central role in the treatment of dynamic events in advance. Instead of a reactive approach based on subsequent or successive re-scheduling, uncertainties are considered *ex ante* while generating baseline schedules. These schedules should be as insensitive as possible to disruptions so that re-scheduling situations are prevented (de Vonder et al., 2007; Goren & Sabuncuoglu, 2008; Jorge Leon et al., 1994; Negri et al., 2020).

As critical indicators for proactive schedules, robustness and stability metrics have been studied widely. Robustness R is defined as the difference between expected performance P_E and real performance P_R (see Eq. 1) (Jorge Leon et al., 1994), whereby the performance corresponds to the main scheduling objective value (MSOV), e.g. makespan. If the difference is positive, the baseline schedule is too conservative. As a result, resources are not used optimally. If the difference is negative, the baseline schedule is too optimistic. This can lead to overload and time pressure. Due to the nature of minimization problems, an absolute robustness value is also used in some studies (see e.g. Al-Behadili et al., 2019; Goren & Sabuncuoglu, 2008). However, it is then not possible to take precise consideration of conservatism and optimism. The stability S indicates how precisely the operations $o \in O$ are planned for the resources. A stability measure, often used in other studies, is defined as the sum of absolute deviations between expected operation completion times c_E and real operation completion times c_R (see Eq. 2) (Al-Behadili et al., 2019; Goren & Sabuncuoglu, 2008; Liu et al., 2017; Rahmani & Heydari, 2014).

$$R = P_E - P_R \quad (1)$$

$$S = \sum_{o \in O} |c_E^o - c_R^o| \quad (2)$$

Depending on the chosen performance indicator, there is a conflict in balancing robustness and stability: Improving one metric can worsen the other (Goren & Sabuncuoglu, 2008). It has been shown that exact robust-stable scheduling is an NP-complete problem even in single machine environments (Bougeret et al., 2019). Recent studies have demonstrated heuristic methods to generate balanced robust-stable schedules. We identified two main paradigms to measure robustness and stability: using MCE or using surrogate measures. There are also two main ways to balance

robustness and stability: by creating neighborhood solutions or by adding slack times to critical operations (see Section “Literature review”). The problem with these approaches is, that they either do not explicitly optimize robustness and stability or require too much computing time. This can be a crucial hurdle in reactive real-time environments.

In recent years, several researchers have identified DRL as an appropriate optimization method for scheduling problems (see e.g. Kardos et al., 2021; Park et al., 2021). DRL is a machine learning technique to solve decision problems with deep artificial neural networks. Through conditioning, an autonomous virtual agent learns a policy on how to interact with a sequentially influenceable environment in order to achieve objectives and maximize the outcome. The iterative process can be formulated as a Markov Decision Process, which can be summarized as follows: The agent observes an environment state S_i and applies action A_i . Based on the action effect, the environment is transferred to a new state S_{i+1} and a reward R_i is distributed to the agent (Morales & Zaragoza, 2012). As function approximators, deep neural networks are used to represent the policy gradually learned and to predict appropriate actions (output) for given state observations (input) (François-Lavet et al., 2018). Fully trained DRL agents achieve a very good compromise between result quality and computing time (Liu et al., 2020). However, after conducting a systematic literature review, we identify a notable lack of scientific approaches investigating, how DRL can be used to optimize and to balance robustness and stability in dynamic flow shops in the proactive stage. The contribution of this study is a proposal on how modern DRL techniques can be implemented to enable a good trade-off between robustness, stability and computing time (see Section “Trade-off between MSO, robustness and stability”).

This article is organized as follows: First, we present related studies with robust-stable scheduling approaches and highlight research gaps (see Section “Literature review”). After that, we show our method of how DRL can be used to dynamically modify operations by stretching or compressing processing times (see Section “Proposed approach”). We then carry out numerical experiments and benchmark different DRL models and metaheuristics in terms of computing time and result quality. In addition, the robustness/stability trade-off, the behavior and the scalability of DRL agents is examined (see Section “Computational study”). Finally, we summarize our results and present future research questions (see Section “Conclusions and future research”).

Literature review

In recent years, a considerable amount of literature has been published on proactive or reactive scheduling in different

problem contexts such as project, production, crew or workforce planning. However, there is a relatively small body of literature that is concerned with pure proactive scheduling in production environments considering robustness and stability. The existing literature was analyzed with respect to the use of slack-based techniques, the use of simulation, the trade-off analysis of robustness and stability and the application of DRL for proactive scheduling problems. An overview of the reviewed papers is given in Table 1.

Dynamic environments and uncertainty modeling

A great deal of previous research has focused on rather simple environments, such as single machine (SM) and parallel machine (PM) environments, flow shops (FS), permutation flow shops (PFS), job shops (JS) or flexible job shops (FJS) (Goren & Sabuncuoglu, 2008; Goren et al., 2011; Liu et al., 2017; Shen et al., 2016; Vieira et al., 2017; Wang et al., 2022; Wu et al., 2020). Few authors have used more complex environments like dual resource JS (DRJS) or dual resource FJS (DRFJS) (Soofi et al., 2021; Xiao et al., 2019). Most researchers considered the minimization of makespan (C_{max}), flow time (F), total tardiness (T) resp. earliness/tardiness (E/T) as single or separate main scheduling objectives (MSO) (Davenport et al., 2001; Goren & Sabuncuoglu, 2008; Gonzalez-Neira et al., 2021; Jorge Leon et al., 1994; Liu et al., 2017; Rahmani & Heydari, 2014). Furthermore, the minimization of energy consumption (EC), setup time (ST), and idle time (IT) has been considered (Moratori et al., 2010; Sundstrom et al., 2017), some of which were taken into account simultaneously (Moratori et al., 2010; Salmasnia et al., 2014).

The greater part of the literature on proactive scheduling focuses on varying processing times (PT) (Juan et al., 2014; Rahmani & Heydari, 2014; Shen et al., 2016) or machine failures (MF) (Al-Behadili et al., 2019) separately or simultaneously (Negri et al., 2020). Gonzalez-Neira et al. (2021) and Vieira et al. (2017) considered varying setup times (ST) in addition to varying processing times. Sundstrom et al. (2017) considered internal delays (ID). It has been noted that external uncertainties (ex.) such as new job arrivals are only considered in the reactive stage (e.g. Moratori et al., 2010). Since we focus on proactive scheduling approaches, we did no further analysis on the considered external uncertainties. In the literature examined, uncertainties were often modeled by probability distributions (PD) (Al-Behadili et al., 2019; Davenport et al., 2001), intervals (Wang et al., 2022) or scenarios (Rahmani & Heydari, 2014; Soofi et al., 2021; Xiao et al., 2019). A scenario can be defined as a specific production situation from a finite set of possible situations that can occur with a certain probability or under certain conditions and is reflected in various parameters (e.g. processing times). Xiong et al. (2013) determined a failure probability (FB) for every

machine considering the machine's busy time and the total workload of all machines. Negri et al. 2020 used real-time sensor data (RTD) to model machine health conditions.

Techniques to generate proactive schedules

One approach to solve a scheduling problem under uncertainty is to explicitly allow the insertion of slack or analyze a schedule's slack (Davenport et al., 2001; Liu et al., 2017; Moratori et al., 2010; Salmasnia et al., 2014; Sundstrom et al., 2017; Xiong et al., 2013). The idea of slack-based techniques is to add additional time to certain activities, so that the schedule can absorb the effects of the uncertainty and avoid re-scheduling (Davenport et al., 2001; Sundstrom et al., 2017). Davenport et al. (2001) introduced two slack-based techniques: time window slack (TWS) and focused time window slack (FTWS). TWS ensures that each operation will have at least a specified amount of slack. FTWS additionally takes into account where along the temporal horizon the operation is scheduled. Interestingly, the authors claimed that later scheduled operations need more slack to improve robustness. Unlike other studies, Hatami et al. (2018) applied slack-based techniques in backward scheduling. Their approach aims at setting a suitable amount of slack at the beginning of the schedule.

Another approach to generate proactive production schedules is by creating neighborhood solutions and choosing the best neighbor found (Al-Behadili et al., 2019; Goren & Sabuncuoglu, 2008; Shen et al., 2016). Neighbors are feasible solutions that have a small distance in the solution space relative to the initial solution (Shen et al., 2016). In practical scheduling heuristics, neighborhoods are generated by swapping operations in their topological order (Goren & Sabuncuoglu, 2008) or by flipping assigned resources in flexible environments. Up to now, a number of studies adopted a simheuristic approach to generate proactive schedules. Simheuristics integrate simulation into a metaheuristic-driven framework (Juan et al., 2015). Thus, they combine the effectiveness of metaheuristics and the simulation's capability of uncertainties. Negri et al. (2020) proposed a simheuristic approach that composes a genetic algorithm and a discrete event simulation (DES). Liu et al. (2017) proposed a hybridized evolutionary multi-objective optimization algorithm, where each phenotype of the population is evaluated by a simulation model. Another simheuristic approach was proposed by Gonzalez-Neira et al. (2021). Their approach can be split up into two phases. In the construction phase, a schedule that optimizes earliness/tardiness, is generated. In the second phase, the local search phase, interchanges between jobs are performed and evaluated through MCE. So far, several studies have used exact approaches for robust-stable scheduling (e.g. Davenport et al., 2001; Goren et al., 2011; Rahmani & Heydari, 2014; Salmasnia et al., 2014; Wu et al., 2020).

Table 1 Related work matrix

Study	Environment	MSO	Uncertainty	Uncertainty modeling	Approach	Slack time	Robustness criterion	Simulation
Al-Behadili et al. (2019)	PFS	C_{max}	MF, ex.	PD	H		R, S	✓
Davenport et al. (2001)	JS	T	MF	PD	E	✓	R	✓
Gonzalez-Neira et al. (2021)	PFS	E/T	PT, ST	PD	H		EV, SD	✓
Goren and Sabuncuoglu (2008)	SM	T, F	MF	PD	H		R, S	
Goren et al. (2011)	JS	C_{max}	MF, PT	PD	E, H		S	
Hatami et al. (2018)	PFS	C_{max}	PT	PD	H		R	✓
Jorge Leon et al. (1994)	JS	C_{max}	PT, MF	PD	H		R	✓
Juan et al. (2014, 2015)	PFS	C_{max}	PT	PD	H		R	✓
Liu et al. (2017, 2020)	PFS	F	MF, ex.	PD	H	✓	R, S	✓
Moratori et al. (2010)	JS	T, ST, IT, F	ex.		H	✓	S	
Negri et al. (2020)	FS	C_{max}	MF, PT	PD, RTD	H		R	✓
Rahmani and Heydari (2014)	FS	C_{max}	PT, ex.	Scen.	E		MMR	
Salmasnia et al. (2014)	SM	$C_{max}, E/T$	PT	PD	E	✓	R	
Shen et al. (2016)	FJS	C_{max}	PT	Scen.	H		R	✓
Soofi et al. (2021)	DRF- JS	C_{max}	MF	Scen.	H		AMS, RMS	✓
Sundstrom et al. (2017)	JS	EC, C_{max}	ID	PD	E	✓	R, S	
Vieira et al. (2017)	JS	C_{max}	ST, PT	PD	H		R	✓
Wang et al. (2020, 2022)	PM	C_{max}	PT	Inter-val	H		MMR	
Wu et al. (2020)	FS	C_{max}	PT	Scen.	E, H		RMS	✓
Xiao et al. (2017, 2019)	DRJS	C_{max}	PT	Scen.	H		R	✓
Xiong et al. (2013)	FJS	C_{max}	MF	FB	H	✓	R	

But due to the high computational time required for exact methods, most researchers propose heuristic approaches (see Al-Behadili et al., 2019; Goren et al., 2011; Soofi et al., 2021; Xiao et al., 2019; Wang et al., 2022).

Techniques to evaluate proactive schedules

Most researchers evaluate proactive schedules by using robustness and stability measures described in Section “Introduction”. Rahmani et al. (2014), who model varying processing times through scenarios, evaluated a schedule by the expected robustness over all scenarios. This approach was also applied by Wang et al. (2022), who called it the min-max regret criterion (MMR). Wu et al. (2020) introduced a metric called robust makespan (RMS). RMS is defined as the maximum makespan of a schedule among all scenarios. Soofi et al. (2021) also use RMS and additionally consider the average makespan (AMS) over all scenarios. Gonzalez-Neira et al. (2021) additionally used two qualitative criteria to evaluate the schedule performance and address robustness through expected value (EV) and standard deviation (SD) of the earliness/tardiness. The greater part of the literature on proactive scheduling has utilised simulation to evaluate a schedule’s robustness or stability, such as Leon et al. (1994), Davenport et al. (2001) and Shen et al. (2016). Alternatively, simulation models can be integrated into the optimization algorithm, as described above.

Since simulation experiments require substantial computational times (Goren & Sabuncuoglu, 2008), some studies used surrogate measures for evaluation, such as Goren and Sabuncuoglu (2008), Goren et al. (2011), Liu et al. (2017) or Sundstrom et al. (2017). Surrogate measures are quickly observable metrics that are assumed to correlate well with stochastically ascertainable robustness or stability. Two well-known examples are the average total slack time (Jorge Leon et al., 1994) or sum of variances on the critical path (Goren et al., 2011). Classically, they are used to estimate how susceptible the critical path is to disturbances such as machine breakdowns or varying processing times. The disadvantage is that they only implicitly describe the actual robustness and stability, which can lead to undesirable side effects such as neglecting non-critical operations. Conclusively, surrogate measures have shown low effectiveness compared to simulation-based metrics (Xiao et al., 2017).

Trade-off between MSO, robustness and stability

A number of studies optimized two or more objectives simultaneously. Al-Behadili et al. (2019) proposed an approach that optimizes makespan as MSO, robustness and stability. In general, a schedule is called stable, if the real operations do not deviate from the planned operations of the baseline schedule. It can be measured by the sum of absolute devia-

tions between expected operation completion times and real operation completion times (see Eq. 2) or by using surrogate measures, such as slack-based methods, which do not require simulation and therefore show more computational efficiency (see e.g. Davenport et al., 2001). But there are only few researchers who analyzed the trade-offs between MSO and robustness or between robustness and stability. Xiong et al. (2013) analyzed the trade-off between MSO and robustness. They developed a multi-objective evolutionary algorithm to minimize the estimated makespan and estimated robustness, that generates a set of feasible, pareto-optimal schedules. They suggested involving managers in the decision-making process by allowing them to choose the final schedule based on domain knowledge. In further research, Shen et al. (2016) also proposed a heuristic approach to optimize two MSOs and the robustness over different scenarios. They highlighted that the MSOs and robustness are seriously conflicted, and no solution can simultaneously optimize all three objectives.

Few studies have analyzed the trade-off between robustness and stability. Goren and Sabuncuoglu (2008) analyze the trade-off in a single-machine environment with random machine failures for three different MSO namely makespan, total flow time and total tardiness. They explored the effect on stability if robustness is optimized and vice versa. They discovered that stability is not considered during robustness optimization and that the schedule’s robustness worsens during stability optimization. Thus, the researchers suggested managing the trade-off by optimizing the weighted linear combination of robustness and stability depending on the MSO: In case of makespan minimization as MSO, practitioners should focus on stability optimization and in case of flow time or tardiness minimization, robustness optimization should be prioritized. Liu et al. (2017) managed the trade-off between robustness and stability by developing an algorithm that efficiently creates a set of pareto-optimal schedules. The visualization of these pareto fronts show, that increasing stability leads to lower robustness and vice versa. The paper focuses on the algorithm’s efficiency and draws no further conclusions about the actual trade-off. Sundstrom et al. (2017) systematically evaluated the triangular trade-off between MSO namely energy consumption, robustness and stability. They also used pareto fronts to analyze the trade-off. Each front depicts robustness and stability values for a constant energy consumption and shows the same results as Liu et al. (2017). Further analysis of the corresponding schedules revealed that slack at the end of the schedule would improve the robustness, while stability is improved by slack throughout the schedule (Sundstrom et al., 2017).

DRL-based methods

Recent studies have been utilized DRL for scheduling with uncertainties. However, the usage of DRL in the field of

production scheduling is limited to the reactive phase of the approaches, such as in (Minguillon & Stricker, 2020; Shahrabi et al., 2017; Wang et al., 2020). However, examples of robust scheduling with DRL in the proactive phase can be found in other domains (Kenworthy et al., 2021; Su et al., 2018). Kenworthy et al. (2021) used a combination of DRL and Integer Programming (IP) to solve an aircraft crew-scheduling problem. The probabilities in the neural network output layer are used to assign coefficient weights to the variables in the IP. This approach enables an IP with fewer variables and constraints that can deal with stochastic flight durations. Their scheduling objective is to maximize the total amount of buffers. A buffer, in this case, is defined as the amount of time between successive flights of a particular pilot. Su et al. (2018) proposed a hybrid teaching-learning-based optimization algorithm that includes DRL techniques in the teaching phase. They apply their algorithm for aircraft carrier flight deck operations with stochastic durations to maximize the probability of completing within the limitative makespan and minimize the weighted sum of expected makespan and the makespan variance.

Research gaps

The following essential research gaps can be summarized from our literature review:

1. There exist a few exact robust-stable scheduling approaches such as (Davenport et al., 2001; Goren et al., 2011; Rahmani & Heydari, 2014; Salmasnia et al., 2014; Wu et al., 2020) and several heuristic approaches such as (Al-Behadili et al., 2019; Goren et al., 2011; Soofi et al., 2021; Xiao et al., 2019; Wang et al., 2022). To the best of our knowledge, there does not exist an approach that uses DRL in the proactive stage of production planning. We also cannot identify a DRL-based approach that directly balances robustness and stability in general. Since a fully trained DRL agent can be computationally efficient, can handle uncertain environments, and has proven good results in proactive scheduling in other domains, the use of DRL for robust-stable production scheduling should be scientifically investigated.
2. Many researchers used simulations to evaluate robustness and stability (Al-Behadili et al., 2019; Davenport et al., 2001; Soofi et al., 2021; Wu et al., 2020; Xiao et al., 2019), since the effectiveness is higher compared to surrogate measures (Xiao et al., 2017). Thus, there exist several approaches that integrate simulations directly into the heuristic search strategies (Gonzalez-Neira et al., 2021; Liu et al., 2017; Negri et al., 2020). These so called simheuristics improve the verification and validation processes (Juan et al., 2015) and thus, lead to better results. But a major disadvantage of simheuristics is their computational effort (Juan et al., 2015). To ensure stochastic precision in practical cases with a large number of planning objects and uncertainties, a correspondingly high number of MCEs must be carried out. Thus, the real-time applicability of simheuristics is rather low and more research is required too minimize the computational effort to generate proactive schedules without losing insights gained by simulation.
3. A few researchers have analyzed a schedule's robustness and its stability simultaneously (Al-Behadili et al., 2019; Goren & Sabuncuoglu, 2008; Liu et al., 2017; Sundstrom et al., 2017). The approaches of Goren and Sabuncuoglu (2008) and Liu et al. (2017) suggested two different ways how to manage the trade-off between robustness and stability. To the best of our knowledge, Sundstrom et al. (2017) are the only researchers, who further analyze, how prioritizing robustness and stability influence the schedule itself. Thus, we suggest further research on how proactive schedules are influenced by the trade-off and additionally, if the trade-off is influenced by the MSO.

Proposed approach

This section describes the core components of our DRL-based method to generate robust-stable schedules proactively. The approach enables a non-iterative method with a metrics-oriented optimization: A DRL agent internalizes the behavior of a stochastic DES and is able to adjust operation processing times without performing computationally intensive MCE step by step. The method is aligned to a dynamic flow shop scenario considering uncertain processing times, machine failure probabilities and uncertain repair times. It is inspired by our industrial partner (see Sections “[Deterministic flow shop problem](#)”, “[Scenario with uncertainties](#)”). The method comprises the following sequential sub-steps: First, an opening optimization procedure is conducted, where a baseline schedule is generated deterministically without considering robustness and stability. Then, stability, robustness and other metrics are measured within DES-based MCE (see Section “[Robustness and stability evaluation](#)”). Based on the stochastic results, a DRL agent modifies the baseline schedule to improve robustness and stability. Here, either operations are extended by additional slack times (*stretching*) or operations are shortened in their plan duration (*compressing*) (see Sections “[Subsequent robustness and stability optimization](#)”, “[DRL design](#)”). Stretching corresponds to conservative planning, while compression corresponds to optimistic planning. For example, the agent can put additional slack times on the critical path of the schedule to stretch operations and catch endangering dynamic events (see Fig. 1).

Fig. 1 Illustrated baseline schedule with 3 machines and 3 jobs (input for the proposed DRL-based approach) and proactive schedule with additional slack times (output from the DRL agent)

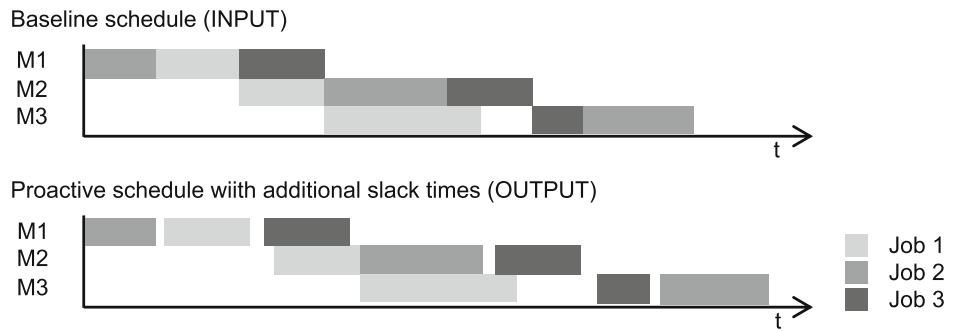


Table 2 Notation for the deterministic baseline problem (flow shop scheduling problem)

Notation	Description
n	Number of jobs
m	Number of subsequent machines
$p_{i,k}$	Processing time of a job i on machine k
\mathbb{N}	Big positive integer number
$\delta_{i,k}$	Start time of a job i on machine k (variable)
$\epsilon_{i,j,k}$	1, if job i precedes job j on machine k , else 0 (variable)

Deterministic flow shop problem

Before describing the dynamic problem with uncertainties and the robustness/stability optimization, the opening procedure’s deterministic baseline problem must be formalized. Inspired by the production organization by our corporate research partner, a flow shop scheduling problem with waiting times and infinite machine buffers is considered. In this setting, every production job has a number of operations equal to the number of machines. As a non-permutation setting, jobs can overtake each other in the production line. As MSO, the operation sequences should be set in such a way that either the makespan or the total flow time without release times is minimized. Due to the alignment as a feasibility study, the problem was simplified to ensure a better transferability of the proposed approach. For example, employees, sequence dependencies or setup operations were not taken into account (Table 2).

The baseline flow shop model for makespan minimization can be formulated as the following mixed integer linear program:

$$\min C_{max} = \max_{1 \leq i \leq n} \{\delta_{i,m} + p_{i,m}\} \tag{3}$$

s.t.

$$\delta_{i,k} + p_{i,k} \leq \delta_{i,k+1} \quad \forall i \in \{1, \dots, n\}, \forall k \in \{1, \dots, m - 1\} \tag{4}$$

$$\epsilon_{i,j,k}(\mathbb{N} + p_{j,k}) + \delta_{i,k} - \delta_{j,k} \geq p_{j,k} \tag{5}$$

$$\forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, n\}, \forall k \in \{1, \dots, m\}$$

$$(1 - \epsilon_{i,j,k})(\mathbb{N} + p_{i,k}) + \delta_{j,k} - \delta_{i,k} \geq p_{i,k} \tag{6}$$

$$\forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, n\}, \forall k \in \{1, \dots, m\}$$

$$\delta_{i,k} \in \mathbb{N} \quad \forall i \in \{1, \dots, n\}, \forall k \in \{1, \dots, m\} \tag{7}$$

$$\epsilon_{i,j,k} \in \{0, 1\} \quad \forall i \in \{1, \dots, n\}, \forall j \in \{1, \dots, n\}, \forall k \in \{1, \dots, m\} \tag{8}$$

Equation 3 represents the objective function for makespan minimization. Equation 4 ensures that an operation of a job can only start, when the preceding operation on the previous machine is completed. Furthermore, wait times between operations are allowed with this modeling. Equations 5, 6 ensure that a machine can only process one operation at a time. For total flow time minimization (without release times), Eq. 3 must be replaced by the following objective function:

$$\min F = \sum_{i=1}^n \delta_{i,m} + p_{i,m} \tag{9}$$

Scenario with uncertainties

To specify a dynamic problem with consideration of uncertainties, the scenario must be explained in more detail: the partner company manufactures custom industrial fittings, where the single production of complete systems involves the non-taktet manufacturing steps (1) machining, (2) welding and (3) assembly. For each product category $PC \in \{1, 2\}$ and manufacturing step k , we assume triangularly distributed uncertain processing times $D_{k,PC}$. Each uncertain processing time is represented as a tuple (a, c, b) , where $[a, b]$ are the interval bounds and c is the mode ($a \leq c \leq b$). In this way, asymmetric distributions can also be defined in a compact

Table 3 Metrics for a single simulation run

Metric	Description
R	Robustness of a schedule measured in a simulation run (see Eq. 1)
S	Stability (see Eq. 2)
C_{max}	Makespan
F	Total flow time
$e_{i,k}$	End time of operation (i, k)
$TS_{i,k}$	Total slack of an operation

manner. The expected value $E(X)$ and the standard deviation $\sqrt[3]{V(X)}$ for a triangular distribution tuple X can be defined as follows:

$$E(X) = \frac{a + b + c}{3} \quad (10)$$

$$\sqrt[3]{V(X)} = \frac{\sqrt[3]{(a-b)^2 + (b-c)^2 + (a-c)^2}}{6} \quad (11)$$

Moreover, each machine has a failure probability $P(F_k)$. As a simplified modeling, a failure can occur once per operation and requires a repair operation with a triangularly distributed machine-specific repair time Q_k . Equation 12 defines the expected processing time of an operation including a probable machine repair time. This value is used for each operation processing time $p_{i,k}$ in the deterministic opening procedure to achieve an optimal or near optimal MSOV under realistic conditions (see Section “[Action space analysis](#)”).

$$E(D_{k,c}) + P(F_k)E(Q_k) \quad (12)$$

Robustness and stability evaluation

After generating a baseline schedule deterministically with expected processing times, robustness and stability must be evaluated under stochastic conditions within DES-based MCE. The DES is used to dynamically simulate a baseline schedule according to the operation sequence set by the opening procedure. Dynamic simulation means that random processing times are set according to the processing time distributions and consideration of downtimes. During simulation, operations are started as soon as the previous operation is completed and the machine is available. The start time of an operation set by the opening procedure can therefore also be undershot. According to the formulated baseline problem (see Section “[Deterministic flow shop problem](#)”), waiting times between operations and jobs are possible. Moreover, station buffer upper bounds are not taken into account. The DES is designed as a process-based simulation in which each resource is designed as its own asynchronous process. In the beginning, a separate process is started for each machine M_1, \dots, M_m . A process runs until all associated operations are completed according to the baseline schedule. In each

Table 4 Summary metrics for all simulation runs

Metric	Description
\bar{R}	Average robustness from all simulation runs
\bar{S}	Average stability
\bar{C}_{max}	Average makespan
\bar{F}	Average total flow time
$\bar{e}_{i,k}$	Average end time of an operation
$\bar{TS}_{i,k}$	Average total slack of an operation
\bar{TS}	Average total slack from all operations

process, the next specified operation is taken from the input buffer as it becomes available. Next, the operation is executed with a randomly set processing time according to the distributions. In addition, a machine failure can occur that must be remedied within the respective repair time. After the operation is completed, it is committed to the next machine’s buffer or final drain. The simulation ends when all processes are finished. In the context of MCE, the simulation is performed multiple times, with dynamic events being triggered randomly in each case. After a simulation run, the metrics described in Table 3 are measured. Considering all simulation experiments, the metrics described in Table 4 can be obtained.

Subsequent robustness and stability optimization

For a better understanding of the proposed DRL-based method, the subsequent dynamic problem is formalized in this section. The optimization model enables stretching and compressing operation durations to absorb dynamic events and to improve robustness and stability simultaneously. To evaluate a proactive schedule candidate PS , the optimization procedure integrates MCE (see Section “[Robustness and stability evaluation](#)”) to calculate an aggregated and normalized robustness/stability value $\Lambda \in \mathbb{R}_{>0}$.

$$\min \Lambda = \frac{|PS_{\bar{R}}|w}{|BS_{\bar{R}}|} + \frac{PS_{\bar{S}}(1-w)}{BS_{\bar{S}}} \quad (13)$$

s.t.

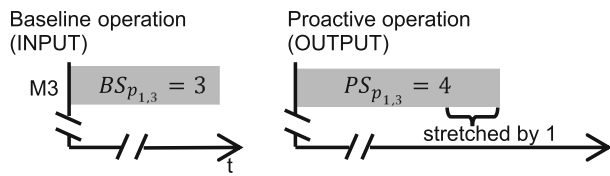


Fig. 2 Stretching an operation before evaluating robustness and stability via MCE. Simplified example with 1 machine and 1 job

$$PS_{p_{i,k}} \in \mathbb{N} \quad \forall i \in \{1, \dots, n\}, \forall k \in \{1, \dots, m\} \quad (14)$$

The lower the Λ value, the better is the linear combination of robustness and stability in comparison to the related baseline schedule BS . Since robustness and stability are competing objectives, a linear weighting is applied using a robustness weight $w \in [0, 1]$ and a stability weight $1 - w$ for balancing (see Eq. 13). Stretching or compressing operations is enabled by manipulating operation plan durations $p_{i,k}$ in a range as it is allowed by technological constraints. Every operation duration $PS_{p_{i,k}}$ from the PS is overwritten with regard to improve robustness and stability (see Eq. 14, Fig. 2). The next section describes how a DRL agent can be implemented to solve this dynamic problem, taking into account the logistical specifics of the baseline problem (see Section “**Deterministic flow shop problem**”).

DRL design

This section describes the general conception of the DRL agent based on the formalized dynamic problem in Section “**Subsequent robustness and stability optimization**”. We identified two state-of-the-art actor critic DRL algorithms for discrete action spaces: Proximal Policy Optimization (PPO) (see OpenAI, 2022c; Schulman et al., 2017) and Advantage Actor-Critic (A2C) (see Mnih et al., 2016; OpenAI, 2022b), which have been realized using an OpenAI Gym environment (see OpenAI, 2022a) in combination with the *Stable-Baselines3* framework (see Stable-Baselines3, 2022). Hyperparameter configuration including neural network architecture and documentation of the training process can be found in Section “**DRL policy learning benchmark (PPO vs. A2C)**”.

Algorithm 1 illustrates the essential training procedure including reward calculation. An episode always includes as many steps as there are operations in the schedule. The first step refers to the first operation on machine M_1 , the last step to the last operation on machine M_m . For each training step, the current state π , the chosen action ζ and the current modified proactive schedule candidate PS are passed as arguments. The state space contains various features that are intended to describe the baseline schedule and the current proactive schedule structure as generally as possible (see Table 5). Figure 3 shows the discrete action space utilized.

It provides three actions applicable for each scheduled operation (i, k) . Depending on the action chosen, the operation processing time $p_{i,k}$ is stretched or compressed to improve robustness and stability.

In the step function essential for the episodic learning, the next operation is first initialized and then modified considering the chosen action, which is also overwritten in the PS candidate. Then, the PS is simulated once in a deterministic manner. In this way, current effects on robustness and stability are measured and updated in a new state space π^* . Based on this, the intermediate reward can be calculated, that is distributed after each step (see Algorithm 2). It is calculated according on how the agent set the processing time per operation. Setting optimistic processing times improves stability, if operations were previously stretched. On the other hand, conservative processing times improve robustness (see Section “**Action space analysis**”). Stability optimization through action 2 is rewarded directly but results in a penalty due to robustness degradation considering w . On the other hand, robustness improvement only results in a reward, which is slightly higher. In the last episode step, the PS is handed over to the MCE to calculate the robustness/stability value Λ (see Eq. 13). Based on this, a final penalty is distributed, where Λ is multiplied by a large number. This ensures that the final penalty has a significantly greater impact than the intermediate rewards. A value greater than 1 awards a higher penalty that increases linearly with degradation. For a value less than 1, the penalty decreases rapidly as the improvement increases. In this way, combined with appropriate discount factors, we were able to make the agent more greedy and reliable in terms of achievable end results. The specified reward and penalty scores are the result of a successive empirical analysis and will not be explained further.

Algorithm 1 DRL training episode (pseudocode snippet)

```

1: function STEP( $\pi$ : state,  $\zeta$ : action,  $PS$ : proactive schedule candidate)
2:    $(i, k) \leftarrow getNextOperation()$ 
3:    $PS \leftarrow modifyOperation(\zeta, i, k, PS)$ 
4:    $\pi^* \leftarrow simulateSchedule(PS)$ 
5:    $rew \leftarrow interReward(\zeta, \pi, \pi^*)$  ▷ See Algorithm 2
6:   if Last scheduled operation then
7:      $rew \leftarrow rew + \begin{cases} (-100\Lambda)^{10}, & \text{if } \Lambda < 1 \\ -125\Lambda, & \text{else} \end{cases}$  ▷ See Eq. 13
8:   end if
9:   return  $\pi^*, rew$ 
10: end function
    
```

Computational study

Numerical experiments were conducted and discussed to answer the following questions: (1) How does the plan dura-

Table 5 DRL observation features with logistical specifics for non-permutation flow shops with waiting times and infinite machine buffers

No.	Description
1	Number of jobs
2	Current episode step
3	Current operation machine k [dummy encoded]
4	Current operation product category c [dummy encoded]
5	Determines, if the total slack of the current operation (i, k) operation is above average ($BS_{\overline{TS}_{i,k}} > BS_{\overline{TS}}$) [binary encoded]
6	Determines, if the current operation is on the critical path ($BS_{\overline{TS}_{i,k}} = 0$) [binary encoded]
7	Number of succeeding jobs on the current machine
8	Sum of stretched and compressed operations on the machine $\left(\sum_{o \in O'} \begin{cases} 0, & \text{if action 1 has been chosen} \\ -1, & \text{if action 2 has been chosen } O' := \text{Machine operations} \\ 1, & \text{else} \end{cases} \right)$
9	Sum of stretched and compressed operations within the job
10	Current effect on robustness ($PS_{MSOV} - BS_{MSOV} + BS_{\overline{R}}$)
11	Current effect on the job end time ($PS_{e_{i,m}} - BS_{e_{i,m}}$)
12	Current effect on the operation end time ($PS_{e_{i,k}} - BS_{e_{i,k}}$)

Algorithm 2 Intermediate reward calculation (pseudocode snippet)

```

1: function INTERREWARD( $\zeta, \pi, \pi^*$ )
2:    $rew \leftarrow 0$ 
3:   if  $|\pi_{10}^*| < |\pi_{10}|$  then ▷ see Table 5
4:      $rew \leftarrow 6w$ 
5:   else
6:      $rew \leftarrow -4w$ 
7:   end if
8:   if  $\zeta_2$  has been selected then
9:      $rew \leftarrow 4(1 - w)$ 
10:  end if
11:  return  $rew$ 
12: end function

```

tion affect the trade-off between robustness and stability and how to define the action space (see Section “[Action space analysis](#)”)? (2) How well do DRL agents learn a policy to generate robust-stable schedules (see Section “[DRL policy learning benchmark \(PPO vs. A2C\)](#)”)? (3) How is the learned policy reflected in the agent’s decision behavior (see Section “[DRL agent behavior analysis](#)”)? (4) How well performs a DRL agent in comparison to a Simulated Annealing with Research Strategy (SARS) approach regarding result quality and computational efficiency (see Section “[DRL performance benchmark \(PPO vs. SARS\)](#)”)? (5) How performs the agent when varying problem size and uncertainty and how are robustness and stability affected (see Section “[Scalability investigation](#)”)? The general experimental setup for all experiments including test data description is documented in Section “[Experimental setup](#)”.

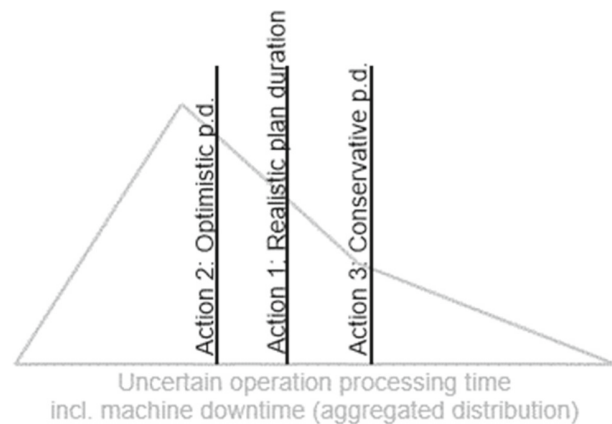


Fig. 3 DRL discrete action space with three possible actions for selecting an operation plan duration within the aggregated uncertainty distribution. The actions have been determined by the experiments in Section “[Action space analysis](#)”. *Action 1* corresponds to plan duration PD_3 ; *Action 2* corresponds to PD_2 ; *Action 3* corresponds to PD_4 (see Appendix Table 9). Illustrated representation not to scale

Experimental setup

The experiments were conducted with the following hardware: GPU: Nvidia Quadro RTX3000 5980MB VRAM; CPU: Intel Core i7-10875H @ 2.3 GHz (16 CPUs); 16384MB RAM. Makespan (C_{max}) and total flow time (F) minimization were considered as separate MSOs. With respect to the scope, robustness and stability have been considered equally in the experiments ($w = 0.5$). In fact, the weight w is an independent variable that has an effect on the planning result depending on the MSO Goren and Sabuncuoglu

(2008). However, results from equal weighting are discussed in relation to the agent performance (see Section “DRL policy learning benchmark (PPO vs. A2C)”). After intensive research, we could not identify suitable benchmark instances which can be used for our approach and uncertainty modeling without further modification. Furthermore, the uncertainties had to be scalable in order to analyze the degree of uncertainty. Consequently, like other authors (see e.g. Al-Behadili et al., 2019; Goren & Sabuncuoglu 2008; Liu et al., 2017), we generated suitable instances. In addition to instances based on the industrial scenario (see Section “Scenario with uncertainties”), we modified a well-known instance proposed by Taillard (1993) for deterministic flow shop scheduling benchmarks.

- **GMRT5x3.** Compact scenario-based test data with 5 jobs and 3 machines.
- **GMRT10x3.** Extended scenario-based test data with 10 jobs.

The GMRT instances were utilized to generally analyze the DRL agent learning performance and behavior (cf. initially mentioned experiment questions 1–4). For training and evaluating the DRL agent, 10 training and 10 test instances were generated per data set and per MSO, each with a random number of product categories. The COIN-OR Branch-and-Cut solver V2.10 has been used to generate optimal deterministic baseline schedules. The uncertainties considered for the subsequent proactive optimization method can be found in Appendix Tables 6, 7. According to experiment question 4, the proposed DRL-based approach has been compared against an iterative metaheuristic. Consequently, the number of experiments must be chosen such that the true Λ value of a neighbor is precisely approximated. It had to be prevented that random outliers are confused with local optima. We used the standard deviation of the means to measure the error of yielded Λ values Jacoboni and Lugli (1989). Here we set an upper limit of about $\sigma_n \approx 0.005$, which corresponds to about $n = 10,000$ experiments and led to reliable results.

In order to indicate the reliability and scalability of the method, the following instances were used:

- **T20x5LV.** Modified Taillard instance with 20 jobs, 5 machines and low uncertainty variance (LV).
- **T20x5HV.** The same instance with a high uncertainty variance (HV).

According to experiment question 5, it was analyzed how the agent handles more extensive environments with varied uncertainties. In this case, the baseline schedules were generated using the Shortest Processing Time (SPT) dispatching rule. Due to the specifics of our scheduling and uncertainty

model, the data was modified as described in Appendix Table 8.

Action space analysis

This analysis addressed the first experiment question and examined the impact of different plan durations (PD) utilizable as $PS_{pi,k}$ in the subsequent robustness and stability optimization method (see Section “Subsequent robustness and stability optimization”). The aim was to investigate and to specify the scope of the DRL action space (see Section “DRL design”), which is important for a proper policy learning. In particular, it was analyzed which PD leads to which consequences in terms of robustness and stability. Moreover, general conclusions about the conflict between robustness and stability could be drawn. Five PD were considered, where PD_1 is a very optimistic, PD_3 a realistic and PD_5 a very conservative time. All PD are within the bounds of the aggregated processing time distribution of an operation (see Appendix Table 9). Appendix Table 10 and Fig. 4 give a statistical overview of the effects.

What stands out in the results are the very similar trade-off patterns and the associated impact of different PD. According to our observations, PD_3 achieved good stability values for C_{max} and F minimization. More optimistic or conservative values worsened the stability in all cases. The more conservative PD_4 , on the other hand, led to better robustness. With PD_5 , even positive robustness was achieved for all baseline schedules. This measurement confirms the conflict in a simultaneous robustness and stability optimization in this case: The more realistic the plan durations, the better the stability. If, on the other hand, the PD are chosen more conservatively, the robustness increases at the expense of stability. This interesting finding may be related to statistical scope per metric. With a single operation, it is most likely that the expected operation duration will occur (scope of stability). However, at the level of the overall schedule (scope of robustness), previously unknown causes lead to overly optimistic planning when expected values are utilized. Future studies should examine these causes in more detail, which would go beyond the scope of this work.

A closer inspection of the figure shows that the robustness values in the F case are significantly more sensitive to the PD chosen and more widely spread, which can also be observed in the different value ranges of the robustness axes. This result can be explained by the fact that the C_{max} objective function only depends on one value (end time of the last operation), whereby the F objective function contains the end time of each job. Therefore, instead of one, several operations must be considered in the F case. This is an indicator that stretching or compressing operation durations has to be done in a targeted manner. In this case, it is not sufficient to just stretch an operation at the end in order to achieve

apparent robustness. Conclusively, it can be confirmed that a corresponding optimization method for balancing robustness and stability can be utilized by choosing PD dynamically per operation and in a targeted manner. Since we locate the best trade-offs between PD_2 and PD_4 , these three PD have been included in the DRL action space. In terms of a feasibility study, this choice of actions may be appropriate. However, future work should examine the applicability of continuous action spaces, where even more targeted values can potentially be set. Subject to the three selectable actions defined, the agent's learning and prediction performance are presented and discussed in the next section. How and why the agent selects which action in which situation is examined in Section "DRL agent behavior analysis".

DRL policy learning benchmark (PPO vs. A2C)

The purpose of this experiment was to evaluate an A2C and a PPO model to compare two modern DRL algorithms regarding their learning and predicting performance (see experiment question 2). Separate models were trained for F minimization and C_{max} minimization as MSO. Each model was trained multiple times with GMRT5x3 and GMRT10x3 training samples. After extensive tests, some *Stable-Baselines3* standard parameters have been modified to improve policy learning (see Appendix Table 11). During successive modification and identification of proper hyperparameters, PPO was significantly more robust to adjustments. In contrast to A2C, PPO has generally shown good performance even with different settings. A2C sometimes delivered bad results, so choosing a proper parameter setting was very time-consuming. A possible explanation for this might be that A2C is more sensitive to the hyperparameters than PPO.

Figure 5 shows an overview of the agents' performances quantified by overall reward obtained during the training process for F and C_{max} minimization as MSO (left graph). In the first training half, all models experienced significant growth, which eventually slowed during the exploitation phase. Independently from the MSO, the learning curves share a similar pattern, where the PPO learning curve has a steeper slope and has significantly higher rewards in the final stages. Another interesting finding is, that both models consistently earn more reward for F minimization in comparison to C_{max} minimization. This result may be explained by the fact that the weight $w = 0.5$ has an effect on the outcome. According to Goren and Sabuncuoglu (2008), greater priority should be given to stability in the context of C_{max} minimization. Furthermore, robustness is more insensitive in this case, which can lead to poorer results with this weighting. In summary, the observed difference in performance is complementary to the authors' recommendation.

Moreover, the obtained predictions (average Λ values) for the fully trained DRL models are visualized in form of a box

chart (right graph). Statistical details are added in Appendix Table 12. For all test samples and for both MSO, PPO generally achieved better average Λ values and lower standard deviations. This also reflects the patterns of rewards achieved in the training processes, with the improvement being more pronounced in the F case than in the C_{max} case. The standard deviation was less than 0.043 in each case, with PPO tending to have slightly less scatter with this metric. Therefore, in some cases, the Λ values deteriorated compared to the baseline schedule. Nevertheless, the average robustness and stability could be improved by the agents, whereby PPO has outperformed A2C. In addition, the PPO model was handier to train and better able to learn policy, because it was more independent of hyperparameters.

The results in this section indicate that DRL is a suitable method for robust-stable scheduling in dynamic manufacturing environments. However, with respect to DRL design and to the reduced and abstracted problem, further investigations must be carried out. First, this study was limited to the manipulation of processing times (stretching or compressing operations). With respect to the large scope, other approaches could not be considered. Consequently, more research is required to analyze and implement other paradigms of robust-stable scheduling, such as generating proactive schedules by creating neighborhood solutions through resource flipping or changing sequences by operation swapping. Second, it must be examined how an agent performs in different environments with different parameters. A related scalability analysis is carried out in Section "Scalability investigation". In advance, the next section moves on to discuss the behavior of the PPO agent and to answer the question: Which action has been chosen in which situation and what are the consequences?

DRL agent behavior analysis

The following analysis examines the DRL agent's decision behavior and draws conclusions about the optimization metrics (see experiment question 3). It was analyzed, which actions based on which observations were chosen by the agent and how did this affect robustness and stability. The fully trained PPO agent was analyzed based on GMRT10x3 instances in the context of F minimization. From 500 episodes, all actions chosen by the agent were compared with the situational state observation.

Looking at Fig. 6, it is apparent that the agent did not choose the actions evenly: Stretching (applying PD_4) was applied significantly more often, whereby the realistic value (PD_3) was hardly retained. This result may be explained by the fact that the baseline schedules generated are always too optimistic. This corresponds to the expectation in accordance with the analysis in Section "Action space analysis": The use of PD_3 always implies a negative robustness. However,

Fig. 4 Different PD and their effect on robustness and stability. Results observed by GMRT5x3 with F and C_{max} minimization as MSO. In both MSO cases, the robustness/stability conflict becomes transparent. Good stability is obtained with realistic expected values (PD_3). Good robustness can be achieved with more conservative values (PD_4)

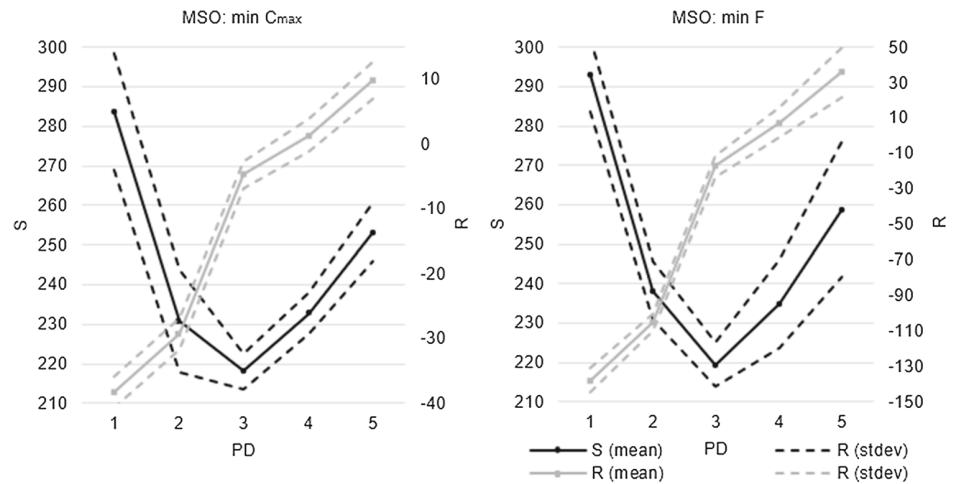
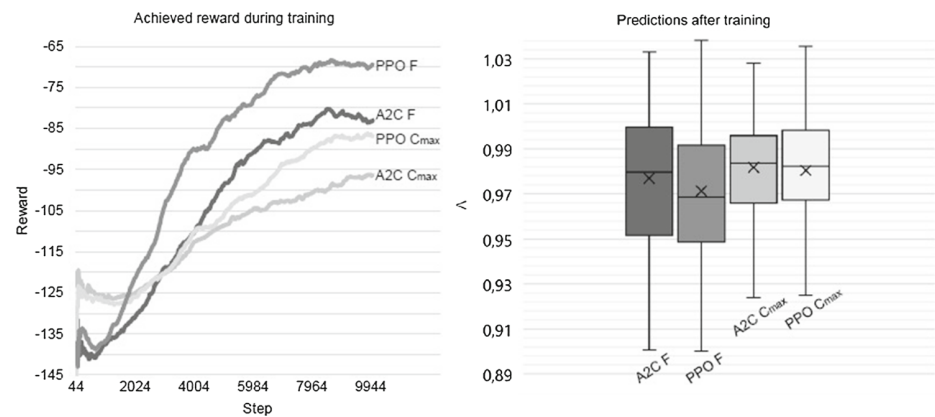


Fig. 5 PPO vs. A2C benchmark for F and C_{max} minimization as MSO ($w = 0.5$). Average rewards achieved during training over time (left graph) and best model’s predictions after training (Δ values) (right graph). All models were trained and tested with GMRT5x3 and GMRT10x3 samples. PPO outperforms A2C in terms of obtained rewards and predictions after training (Δ values)



these results were predictable, since robustness and stability have to be balanced with $w = 0.5$. On the other side, the application of compressing (applying PD_2) is particularly interesting. Overall viewed and in comparison to PD_3 and PD_4 , PD_2 has a negative impact on robustness and stability. Nevertheless, the agent selected compression in about 15% of all cases.

In a further analysis, we examined in which situations the agent tends to compress operations. For this purpose, we have generated a classification decision tree based on state-action tuples. Techniques based on decision trees are a suitable method to (approximately) explain the behavior of DRL agents (Ding et al., 2020). In order to reduce complexity, stretching actions were not taken into account. Figure 7 shows an excerpt of the first decision tree leaves. Even if the Gini coefficients are relatively even, the essential pattern is emerging that be interpreted as follows: If the agent assumes that (1) an operation will end earlier than planned and (2) that the operation tends to be on the critical path, the operation is compressed by the agent. Such a situation occurs especially, when predecessor operations have already been stretched. Thus, compressing the operation compensates the

effect of an “overflow” of the current operation and all successor operations.

At first glance, this behavior seems trivial due to the reward design (see Section “DRL design”), but it may indicate a problem related to the stability metric. The stability metric used in this study and by most authors only considers operation end times (see Eq. 2). And if, in practice, target processing times are shortened in order to cushion other operations, this can lead to stress and pressure. This observation may support the hypothesis that the widely used stability metric is too imprecise. We suggest qualitatively, that stability should be characterized by the fact that re-scheduling procedures can be avoided as often as possible. The more stable the plan, the less often re-scheduling occurs or the less impact re-scheduling causes. In particular, the impact must be suitably quantified.

DRL performance benchmark (PPO vs. SARS)

In this experiment we analyzed the PPO agent performance in comparison to a SARS algorithm roughly based on Yu et al. (2021) (see experiment question 4). The SARS algorithm has been implemented as follows: In every iteration, a random

Fig. 6 Actions chosen by the agent (F minimization as MSO in the context of GMRT10x3). The agent stretched most of the operations and compressed them less often. The default expected value (realistic operation duration) was hardly selected

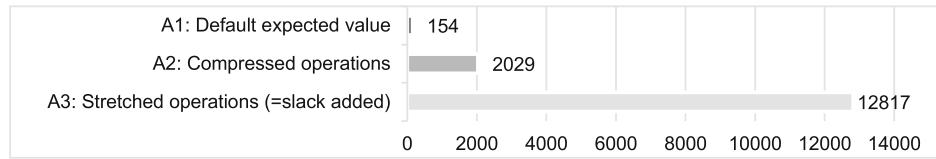


Fig. 7 Decision process whether operations are compressed (CART decision tree). The agent tends to choose compression when a critical path operation may end earlier than originally planned

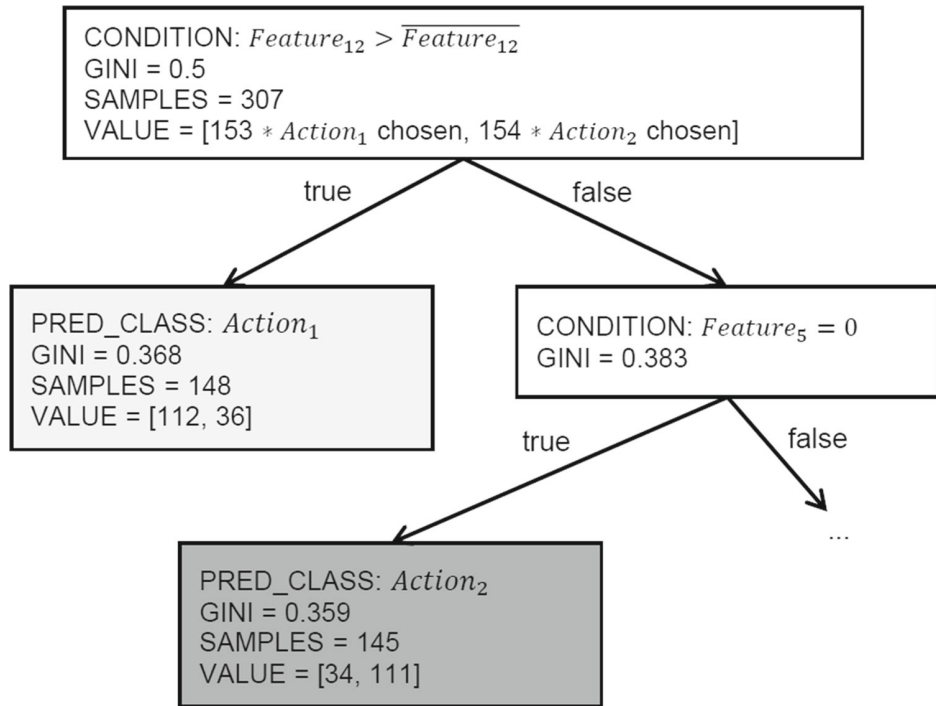
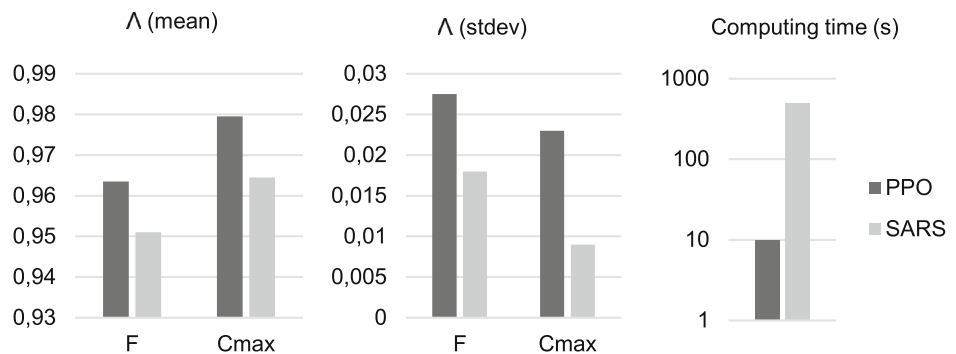


Fig. 8 PPO vs. SARS benchmark for F and C_{max} minimization as MSO ($w = 0.5$) in the context of GMTR5x3 and GMRT10x3. In this analysis, average Λ values achieved, scatter and computing time are compared. The computing time is shown on a logarithmic axis. PPO achieves about 2% worse results than SARS, but only requires about 2% computing time

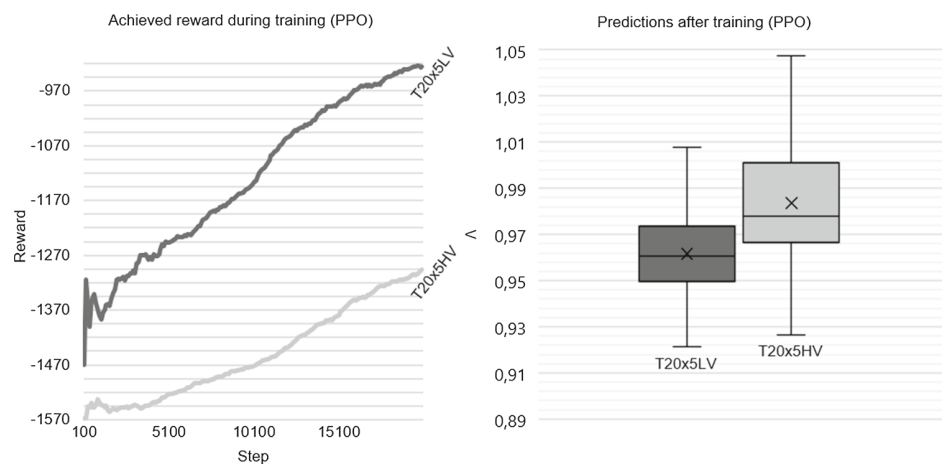


operation is randomly stretched or compressed according to the action space options (see Appendix Algorithm 3). The number of cooling steps was set to 50, which already required a computation time of about 8 minutes. This is due to the fact that the step-by-step evaluation performed 10,000 MCE each time, which significantly increased the computing time. All five steps were checked to see whether an improvement had occurred. If not, the current solution has been reverted to the best solution found. This performance analysis examined the relationship between average Λ value achieved, the related

scatter in the form of standard deviation, and approximate computation time in seconds.

Figure 8 shows the obtained results summarized for GMRT10x3 and GMRT5x3. Statistical details can be found in Appendix Table 13. What stands out in these results is the dominance of the DRL agent in terms of computing time. The DRL Agent required about 10 seconds to generate a proactive schedule and to evaluate via MCE. Without evaluation, the time could be further reduced to less than one second. The average Λ values and scatter obtained by the PPO agent were

Fig. 9 PPO performance for F minimization as MSO ($w = 0.5$) in the context of T20x5LV and T20x5HV data sets. Average rewards achieved during training over time (left graph) and best model's predictions after training (right graph). Lower uncertainty variances (T20x5LV) lead to better performances during and after training



only slightly worse in all cases. This can be an indication that the agent needs even better features or training sequences for better generalization. However, this potential for improvement is not considered in detail in this feasibility study. From the graph, it can be seen that in the case of C_{max} minimization, significantly poorer average results were also achieved with the SARS implementation. This supports the hypothesis that this is an effect of the weight $w = 0.5$ and not attributed to the DRL agent or training process design (see Section “DRL policy learning benchmark (PPO vs. A2C)”). Interestingly, the scattering behaves in the opposite way: Both the agent and the metaheuristic each achieve a better standard deviation in the C_{max} case. This can be explained by the greater variability of the F objective, which has already been observed and discussed in Section “Action space analysis”. The more operations are stretched in terms of conservative, robust planning, the greater the scattering of the robustness achieved in the F case (see Fig. 4).

Based on the previous findings, these results support the idea that DRL is a viable and efficient method to generate proactive schedules, although there is room for improvement. The greatest advantage of the proposed DRL approach lies in the low computing time. After a proper training, it obtains about 98% of the result quality in about 2% of the time compared to traditional methods such as metaheuristics. This can be explained by the fact, that a DRL agent internalized the behaviour of the stochastic simulation model within the training to infer on robustness and stability. Conventional probabilistic methods such as metaheuristics have no way of storing experiences and hypotheses that lead to the situational selection of suitable actions. Instead, it is necessary to evaluate successively, which takes up a lot of computing time due to the complex MCE.

As mentioned in the previous parts, there are some disadvantages or improvement potentials that can be investigated in further studies. The central research question raised in this section is: How is it possible to outperform metaheuristics in

other performance criteria such as mean scores and scatter? In particular, it should be examined whether hyperparameters, training processes, actions, rewards and observations can be further improved. Finally, the scalability of the DRL design is evaluated in the next section.

Scalability investigation

In order to assess the applicability of the proposed approach to other environments, repeated measurements of the learning and prediction performance were conducted (see Fig. 9 and Appendix Table 14). It was analyzed how the agent handles a larger problem input (more machines, more jobs) and how the extent of uncertainties affects robustness and stability. With respect to the large scope, we have focused on F minimization as MSO and PPO as DRL technology. With minor modifications, we were able to make the agent operational for a larger scaled environment. The training duration was increased to 20,000 steps and the final reward was distributed by a factor of ten. This ensured the higher weighting of the final reward compared to the sum of the intermediate rewards.

The charts below illustrate the average reward achieved during training (left graph) and the prediction quality after training completion (right graph). The agents experienced a learning curve in both scenarios (low and high uncertainty) and generated comparatively good proactive schedules after training. What is striking about the training performances is the significantly different reward scores obtained. At higher uncertainty variances (T20x5HV data set), the agent has a significantly lower slope of rewards over time, possibly influenced by the MCE: The greater the impact of uncertainties, the more experiments have to be carried out in order to calculate precise robustness and stability metrics. In addition, it can be assumed that the agent has to explore more during training and also experiences many deterioration effects. Likewise, the fully trained agent generated significantly bet-

ter proactive schedules in the case of low uncertainty. With regard to the Δ values, better means, less scattering and fewer outliers could be achieved with T20x5LV. Interestingly, the results were even better than in the previous GMRT experiments. This is an indicator that the proposed method can also be scaled to larger flow shops. However, an increase in uncertainty results in a loss of obtainable robustness and stability. Conclusionally, the degree of uncertainty has a greater effect than the number of planning objects such as machines or jobs.

A major disadvantage is still, that an agent must be trained for new situations. Future work should analyze how an agent can be better generalized for a varying number of planning objects. It is particularly important to design the reward and the observation space even more universally. Due to the need for more and more robustness, flexibility and responsiveness in innovative logistic systems (Jafari et al., 2022; Monostori, 2018) this also refers to other scheduling domains and associated MSO. In addition to related production organizations such as flexible job shops or more complex environments such as multi-resource shops, robust-stable scheduling can also be used, for example, in route planning, project management or for crew scheduling. From a mathematical point of view, it is also about allocating resources with tasks on the timeline. Consequently, the robustness and stability metrics defined as well as the design of the action space remain universally applicable. However, the observation space responsible for describing the environmental constraints has to be adapted for each specific problem in an extensive feature engineering process. Moreover, the proactive consideration of other types of dynamic events and especially external events such as new job arrivals is of scientific and practical relevance.

Conclusions and future research

The presented research examined, how DRL can be applied in the proactive stage for robust-stable scheduling to absorb uncertainties in advance. For this purpose, a DRL concept was developed, where scheduled operations were stretched or compressed in their time in order to optimize the competing metrics, robustness and stability. The metrics were collected in the course of DES-based MCE for dynamic flow shops, whereby uncertain processing times and machine repair times were given by triangular distributions. The study was primarily set out to analyze the effectiveness, efficiency and scalability of DRL. After extensive numerical experiments, the findings suggest that DRL, and especially PPO, is a viable method to generate proactive schedules in near real-time. PPO comes about 98% close to SARS, but requires only 2% of its computing time after a successful training. This is a great advantage for the application in time-critical reactive

environments. Moreover, it could be shown that the DRL agents can also learn and predict successfully after varying jobs, machines and uncertainties. These findings will be of interest to researchers and practitioners and could be used to develop proactive methods by making them more efficient and intelligent.

The major restriction of this work was proof-of-concept DRL design focusing the proactive stage for F and C_{max} minimization as MSO. In summary, the following wide range of future research questions can be derived from the limitations of this study:

- **Aggregated robust-stable scheduling.** A limitation of the proposed approach is the decomposition of MSOV optimization and proactive planning in two consecutive steps. It could be analyzed which positive and undesirable effects are related to an aggregation of both stages (see Section “Proposed approach”).
- **DRL-based re-scheduling.** This article does not include an analysis of how to combine proactive planning and reactive planning. Further work could analyze and develop a hybrid approach of predictive-reactive scheduling considering DRL on one or both stages (see Section “Proposed approach”).
- **Robustness/stability trade-off.** The conflict between robustness and stability must be formally explained in more detail (see Section “Action space analysis”).
- **Continuous actions.** Future work could improve the agent’s precision by utilizing continuous rather than discrete action spaces (see Section “Action space analysis”).
- **Neighborhood-based scheduling.** It would be interesting how robust-stable neighborhood solutions can be generated using DRL (see Section “DRL policy learning benchmark (PPO vs. A2C)”).
- **Balancing robustness and stability.** As far as we know, general situational rules for weighting robustness and stability with regard to MSO and practical requirements have not yet been established (see Section “DRL policy learning benchmark (PPO vs. A2C)”).
- **Stability metric improvement.** It could be identified that the frequently used stability metric in particular can lead to undesirable effects. Future research should develop methods that focus directly on minimizing re-scheduling situations or associated effects (see Section “DRL agent behavior analysis”).
- **Agent performance improvement.** Further investigations could analyze how DRL can obtain even better end results than traditional metaheuristics and how to reduce the prediction scatter (see Section “DRL performance benchmark (PPO vs. SARS)”).
- **Other scheduling contexts.** It would be interesting to evaluate the applicability of other scheduling models,

MSO or different types of dynamic events (see Section “Scalability investigation”).

- **Agent generalizability.** Further research is required to make the agents more reliable and independent of the environment (see Section “Scalability investigation”).
- **Practical application.** Due to the scope, no evaluation of practical use could be carried out in this work. In this context, future studies should address in-situ simulations, benchmarks and in-depth analysis of robust-stable methods in practical environments.

Author Contributions FG: conceptualization, methodology, formal analysis, investigation, writing—original draft, writing—review and editing; AM: investigation (related work analysis), writing—original draft, writing—review and editing; PR: funding acquisition, resources; ST: supervision.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was supported as part of the joint research project *Human-centered Smart Service Lab/Predictive Scheduling* (with project number [EFRE-030018] of the European Regional Development Fund) which is funded by the federated state North Rhine-Westphalia, Germany. Project description available on <https://www.fh-bielefeld.de/forschung/forschungsprojekte/aktuelle-projekte-fb-3/reusch-predictive-scheduling> (German language).

Code availability Code with test instances and experiment results available on: <https://doi.org/10.17605/OSF.IO/SXM3Q>.

Declarations

Conflict of interest The authors have no conflict of interest to declare that are relevant to the content of this article.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

Appendix

See Tables 6, 7, 8, 9, 10, 11, 12, 13, 14 and Algorithm 3.

Table 6 GMRT uncertain processing times $D_{k,PC}$ for machine k and product category c (triangularly distributed)

Constant	Value
$D_{1,1}$	(149, 150, 155)
$D_{2,1}$	(165, 167, 170)
$D_{3,1}$	(90, 95, 102)
$D_{1,2}$	(165, 168, 172)
$D_{2,2}$	(205, 207, 212)
$D_{3,2}$	(195, 198, 200)

Table 7 GMRT failure probabilities $P(F_k)$ and triangular machine repair time distributions Q_k

Constant	Value
$P(F_1)$	0.05
$P(F_2)$	0.15
$P(F_3)$	0.1
Q_1	(35, 40, 48)
Q_2	(25, 27, 30)
Q_3	(30, 35, 40)

Table 8 Uncertainty modeling for modified Taillard instances (LV: low uncertainty variance, HV: high uncertainty variance)

Uncertain parameter	Value
$D_{i,k}$	LV: $(p_{i,k} * 0.94, p_{i,k}, p_{i,k} * 1.06)$ HV: $(p_{i,k} * 0.88, p_{i,k}, p_{i,k} * 1.12)$
$P(F_k)$	LV: {0.04, 0.06, 0.1} HV: {0.07, 0.09, 0.15}
Q_k	LV: {(10, 15, 20), (8, 16, 24)} HV: {(20, 25, 35), (25, 32, 40)}

Product categories are not considered in comparison to the GMRT instances. Instead, every job has its own triangularly distributed uncertain processing time $D_{i,k}$. On initializing the modified test instances, machine failure probability $P(F_k)$ and repair time Q_k have been selected randomly from the sets defined below

Table 9 Plan duration (PD) parameters for the subsequent method to balance robustness and stability (see Section “Subsequent robustness and stability optimization”)

PD	Processing time expected value	Comment
PD_1	$E(D_{k,PC}) - \frac{\sqrt[3]{V(D_{k,PC})}}{2}$	Very optimistic
PD_2	$E(D_{k,PC})$	Optimistic
PD_3	$E(D_{k,PC}) + P(F_k)E(Q_k)$	Realistic
PD_4	$E(D_{k,PC}) + P(F_k)E(Q_k) + \frac{\sqrt[2]{V(D_{k,PC})}}{4} + P(F_k)\frac{\sqrt[2]{V(Q_k)}}{4}$	Conservative
PD_5	$E(D_{k,PC}) + P(F_k)E(Q_k) + \sqrt[3]{V(D_{k,PC})} + P(F_k)\sqrt[3]{V(Q_k)}$	Very conservative

Basis for the specification of the DRL action space. PD_3 corresponds to the expected value including probable repair time. PD_2 corresponds to the expected value without repair time. Within the bounds of the aggregated distribution, the further values were set even more optimistically or conservatively by subtracting or adding a proportion of the standard deviation

Table 10 Effects of PD on robustness and stability, differentiated according to makespan and total flow time optimization

PD	MSO	Robustness			Stability		
		R	R_σ	R_{min}	S	S_σ	$ R + S$
PD_1	C_{max}	- 38.35	2.38	- 44.64	283.8	14.63	322.15
PD_2	C_{max}	- 29.19	2.39	- 34.83	230.67	12.98	259.86
PD_3	C_{max}	- 4.68	2.1	- 9.43	218.06	4.37	222.74
PD_4	C_{max}	1.35	2.52	- 4.37	232.79	5.34	234.14
PTP_5	C_{max}	9.89	2.84	3.05	253.28	7.26	263.17
PD_1	F	- 137.79	6.93	- 150.99	293.06	9.5	430.85
PD_2	F	- 105.32	4.6	- 112.73	238.22	7.46	343.54
PD_3	F	- 16.97	5.88	- 25.88	219.39	5.59	236.36
PD_4	F	7.56	8.23	- 5.7	234.85	11.15	242.41
PTP_5	F	35.93	14.29	17.12	258.88	17.19	294.81

The data presented here was collected in the context of GMRT5x3

Table 11 DRL hyperparameters for the PPO/A2C benchmark

Hyperparameter	Value
Overall training steps	10^4
Network update interval (steps)	22
Learning rate (linearly decreasing)	10^{-4} (PPO) / 10^{-3} (A2C)
Discount factor	0.99
Activation function	Rectified Linear Unit (ReLU)
Network architecture	- Input layer: min. 12 neurons (depending on the feature encoding) - 1st hidden layer (shared): 2^9 neurons - 2nd hidden layer (value net): $2^7, 2^6$ neurons - 2nd hidden layer (policy net): 2^6 neurons - Output layer: 3 neurons (corresponding to PD_2, PD_3, PD_4)

Table 12 PPO vs. A2C benchmark for F and C_{max} minimization as MSO ($w = 0.5$)

Model	Data	MSO: min F		MSO: min C_{max}	
		Λ	Λ_σ	Λ	Λ_σ
PPO	GMRT5x3	.969	.027	.972	.03
PPO	GMRT10x3	.958	.028	.987	.016
PPO	GMRT5x3 GMRT10x3	.971	.029	.98	.023
A2C	GMRT5x3	.983	.043	.976	.028
A2C	GMRT10x3	.967	.031	.991	.017
A2C	GMRT5x3 GMRT10x3	.977	.035	.982	.026

Predictions by the fully trained DRL models were obtained for GMRT5x3 and GMRT10x3 (separate and mixed). Mean Λ values and standard deviations are presented for each model and MSO

Algorithm 3 SARS candidate creation (pseudocode snippet)

```

1:  $o \leftarrow \text{choice}(O)$ 
2:  $a \leftarrow \text{choice}(A)$ 
    $o_{dur} \leftarrow \text{applyAction}(a)$ 
3:  $\Lambda \leftarrow MCE(O)$ 
4: return  $\Lambda$ 
    
```

▷ Choose a random operation from the set of all scheduled operations O
 ▷ Get random action from action space A
 ▷ Stretch or compress the operation duration
 ▷ Conduct MCE and measure robustness R and stability S

Table 13 SARS results for F and C_{max} minimization as MSO ($w = 0.5$) in the context of GMRT5x3 and GMRT10x3

Method	Data	t	MSO: min F		MSO: min C_{max}	
			$\bar{\Lambda}$	Λ_{σ}	$\bar{\Lambda}$	Λ_{σ}
SARS	GMRT5x3	≈ 500	.955	.017	.948	.01
SARS	GMRT10x3	≈ 500	.947	.019	.981	.008

Table 14 PPO after-training predictions on T20x5LV and T20x5HV instances for F minimization as MSO ($w = 0.5$)

Method	Data	$\bar{\Lambda}$	Λ_{σ}
PPO	T20x5LV	.961	.019
PPO	T20x5HV	.984	.027

The degree of uncertainty has an effect on the achievable robustness and stability

References

Al-Behadili, M., Ouelhadj, D., & Jones, D. (2019). Multi-objective biased randomised iterated greedy for robust permutation flow shop scheduling problem under disturbances. *Journal of the Operational Research Society*, 71(11), 1847–1859. <https://doi.org/10.1080/01605682.2019.1630330>

Bougeret, M., Pessoa, A. A., & Poss, M. (2019). Robust scheduling with budgeted uncertainty. *Discrete Applied Mathematics*, 261, 93–107. <https://doi.org/10.1016/j.dam.2018.07.001>

Davenport, A. J., Gefflot, C. & Beck, J. C. (2001). Slack-based techniques for robust schedules. In *Proceedings of the sixth european conference on planning (ECP-2001)*.

de Vonder, S. V., Demeulemeester, E., & Herroelen, W. (2007). A classification of predictive-reactive project scheduling procedures. *Journal of Scheduling*, 10(3), 195–207. <https://doi.org/10.1007/s10951-007-0011-2>

Ding, Z., Hernandez-Leal, P., Ding, G. W., Li, C., & Huang, R. (2020). [arXiv:2011.07553](https://arxiv.org/abs/2011.07553).

François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, 11(3–4), 219–354. <https://doi.org/10.1561/22000000071>

Gonzalez-Neira, E. M., Montoya-Torres, J. R., & Jimenez, J.-F. (2021). A multicriteria simheuristic approach for solving a stochastic permutation flow shop scheduling problem. *Algorithms*, 14(7), 210. <https://doi.org/10.3390/a14070210>

Goren, S., & Sabuncuoglu, I. (2008). Robustness and stability measures for scheduling: Single-machine environment. *IIE Transactions*, 40(1), 66–83. <https://doi.org/10.1080/07408170701283198>

Goren, S., Sabuncuoglu, I., & Koc, U. (2011). Optimization of schedule stability and efficiency under processing time variability and random machine breakdowns in a job shop environment. *Naval Research Logistics (NRL)*, 59(1), 26–38. <https://doi.org/10.1002/nav.20488>

Hatami, S., Calvet, L., Fernandez-Viagas, V., Framinan, J. M., & Juan, A. A. (2018). A simheuristic algorithm to set up starting times

in the stochastic parallel flowshop problem. *Simulation Modelling Practice and Theory*, 86, 55–71. <https://doi.org/10.1016/j.simpat.2018.04.005>

Jacoboni, C., & Lugli, P. (1989). The Monte Carlo method for semiconductor device simulation. *Springer Vienna*. <https://doi.org/10.1007/978-3-7091-6963-6>

Jafari, H., Ghaderi, H., Malik, M., & Bernardes, E. (2022). The effects of supply chain flexibility on customer responsiveness: The moderating role of innovation orientation. *Production Planning & Control*. <https://doi.org/10.1080/09537287.2022.2028030>

Jorge Leon, V., David Wu, S., & Storer, R. H. (1994). Robustness measures and robust scheduling for job shops. *IIE Transactions*, 26(5), 32–43. <https://doi.org/10.1080/07408179408966626>

Juan, A. A., Barrios, B. B., Vallada, E., Riera, D., & Jorba, J. (2014). A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times. *Simulation Modelling Practice and Theory*, 46, 101–117. <https://doi.org/10.1016/j.simpat.2014.02.005>

Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., & Figueira, G. (2015). A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. *Operations Research Perspectives*, 2, 62–72. <https://doi.org/10.1016/j.orp.2015.03.001>

Kardos, C., Laflamme, C., Gallina, V., & Sihm, W. (2021). Dynamic scheduling in a job-shop production system with reinforcement learning. *Procedia CIRP*, 97, 104–109. <https://doi.org/10.1016/j.procir.2020.05.210>

Kenworthy, L., Nayak, S., Chin, C., & Balakrishnan, H. (2021). *NICE: Robust scheduling through reinforcement learning-guided integer programming*. [arXiv:2109.12171](https://arxiv.org/abs/2109.12171).

Liu, C.-L., Chang, C.-C., & Tseng, C.-J. (2020). Actor-critic deep reinforcement learning for solving job shop scheduling problems. *IEEE Access*, 8, 71752–71762. <https://doi.org/10.1109/access.2020.2987820>

Liu, F., Wang, S., Hong, Y., & Yue, X. (2017). On the robust and stable flowshop scheduling under stochastic and dynamic disruptions. *IEEE Transactions on Engineering Management*, 64(4), 539–553. <https://doi.org/10.1109/tem.2017.2712611>

Mailliez, M., Battaia, O., & Roy, R. N. (2021). Scheduling and rescheduling operations using decision support systems: Insights from emotional influences on decision-making. *Frontiers in Neuroergonomics*. <https://doi.org/10.3389/fnrgo.2021.586532>

- Minguillon, F. E., & Stricker, N. (2020). Robust predictive—Reactive scheduling and its effect on machine disturbance mitigation. *CIRP Annals*, 69(1), 401–404. <https://doi.org/10.1016/j.cirp.2020.03.019>
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). *Asynchronous methods for deep reinforcement learning*. arXiv:1602.01783.
- Monostori, J. (2018). Supply chains robustness: Challenges and opportunities. *Procedia CIRP*, 67, 110–115. <https://doi.org/10.1016/j.procir.2017.12.185>
- Morales, E. F., & Zaragoza, J. H. (2012). An introduction to reinforcement learning. *Decision theory models for applications in artificial intelligence* (pp. 63–80). IGI Global. <https://doi.org/10.4018/978-1-60960-165-2.ch004>
- Moratori, P., Petrovic, S., & Vazquez-Rodriguez, J. A. (2010). Fuzzy approaches for robust job shop rescheduling. In *International conference on fuzzy systems*. IEEE. <https://doi.org/10.1109/fuzzy.2010.5584722>.
- Naderi, B., & Ruiz, R. (2010). The distributed permutation flowshop scheduling problem. *Computers & Operations Research*, 37(4), 754–768. <https://doi.org/10.1016/j.cor.2009.06.019>
- Negri, E., Pandhare, V., Cattaneo, L., Singh, J., Macchi, M., & Lee, J. (2020). Field-synchronized digital twin framework for production scheduling with uncertainty. *Journal of Intelligent Manufacturing*, 32(4), 1207–1228. <https://doi.org/10.1007/s10845-020-01685-9>
- OpenAI. (2022a). *Getting started with gym*. <https://gym.openai.com/docs/>. Accessed May 24, 2022.
- OpenAI. (2022b). *OpenAI baselines: ACKTR & A2C*. <https://openai.com/blog/baselines-acktr-a2c/>. Accessed May 24, 2022.
- OpenAI. (2022c). *Proximal policy optimization*. <https://openai.com/blog/openai-baselines-ppo/>. Accessed May 24, 2022.
- Park, K. T., Jeon, S.-W., & Noh, S. D. (2021). Digital twin application with horizontal coordination for reinforcement-learning-based production control in a re-entrant job shop. *International Journal of Production Research*. <https://doi.org/10.1080/00207543.2021.1884309>
- Rahmani, D., & Heydari, M. (2014). Robust and stable flow shop scheduling with unexpected arrivals of new jobs and uncertain processing times. *Journal of Manufacturing Systems*, 33(1), 84–92. <https://doi.org/10.1016/j.jmsy.2013.03.004>
- Salmasnia, A., Khatami, M., Kazemzadeh, R. B., & Zegordi, S. H. (2014). Bi-objective single machine scheduling problem with stochastic processing times. *TOP*, 23(1), 275–297. <https://doi.org/10.1007/s11750-014-0337-9>
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). *Proximal policy optimization algorithms*. arXiv:1707.06347.
- Shahrabi, J., Adibi, M. A., & Mahootchi, M. (2017). A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. *Computers & Industrial Engineering*, 110, 75–82. <https://doi.org/10.1016/j.cie.2017.05.026>
- Shen, X.-N., Han, Y., & Fu, J.-Z. (2016). Robustness measures and robust scheduling for multi-objective stochastic flexible job shop scheduling problems. *Soft Computing*, 21(21), 6531–6554. <https://doi.org/10.1007/s00500-016-2245-4>
- Soofi, P., Yazdani, M., Amiri, M., & Adibi, M. A. (2021). Robust fuzzy-stochastic programming model and meta-heuristic algorithms for dual-resource constrained flexible job-shop scheduling problem under machine breakdown. *IEEE Access*, 9, 155740–155762. <https://doi.org/10.1109/access.2021.3126820>
- Stable-Baselines3. (2022). *Reliable reinforcement learning implementations*. <https://stable-baselines3.readthedocs.io>. Accessed May 24, 2022.
- Su, X., Han, W., Wu, Y., Zhang, Y., & Liu, J. (2018). A proactive robust scheduling method for aircraft carrier flight deck operations with stochastic durations. *Complexity*, 2018, 1–38. <https://doi.org/10.1155/2018/6932985>
- Sundstrom, N., Wigstrom, O., & Lennartson, B. (2017). Conflict between energy, stability, and robustness in production schedules. *IEEE Transactions on Automation Science and Engineering*, 14(2), 658–668. <https://doi.org/10.1109/tase.2016.2643621>
- Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64(2), 278–285. [https://doi.org/10.1016/0377-2217\(93\)90182-m](https://doi.org/10.1016/0377-2217(93)90182-m)
- Vieira, G. E., Kück, M., Frazzon, E., & Freitag, M. (2017). Evaluating the robustness of production schedules using discrete-event simulation. *IFAC-PapersOnLine*, 50(1), 7953–7958. <https://doi.org/10.1016/j.ifacol.2017.08.896>
- Wang, H., Sarker, B. R., Li, J., & Li, J. (2020). Adaptive scheduling for assembly job shop with uncertain assembly times based on dual q-learning. *International Journal of Production Research*, 59(19), 5867–5883. <https://doi.org/10.1080/00207543.2020.1794075>
- Wang, W., Gao, C., & Shi, L. (2022). Robust optimization on unrelated parallel machine scheduling with setup times. *IEEE Transactions on Automation Science and Engineering*. <https://doi.org/10.1109/tase.2022.3151611>
- Wu, C.-C., Gupta, J. N. D., Cheng, S.-R., Lin, B. M. T., Yip, S.-H., & Lin, W.-C. (2020). Robust scheduling for a two-stage assembly shop with scenario-dependent processing times. *International Journal of Production Research*, 59(17), 5372–5387. <https://doi.org/10.1080/00207543.2020.1778208>
- Xiao, S., Sun, S., & Jin, J. (2017). Surrogate measures for the robust scheduling of stochastic job shop scheduling problems. *Energies*, 10(4), 543. <https://doi.org/10.3390/en10040543>
- Xiao, S., Wu, Z., & Yu, S. (2019). A two-stage assignment strategy for the robust scheduling of dual-resource constrained stochastic job shop scheduling problems. *IFAC-PapersOnLine*, 52(13), 421–426. <https://doi.org/10.1016/j.ifacol.2019.11.092>
- Xiong, J., Ning Xing, L., & Wu Chen, Y. (2013). Robust scheduling for multi-objective flexible job-shop problems with random machine breakdowns. *International Journal of Production Economics*, 141(1), 112–126. <https://doi.org/10.1016/j.ijpe.2012.04.015>
- Yu, V. F., Maulidin, A., Redi, A. A. N. P., Lin, S.-W., & Yang, C.-L. (2021). Simulated annealing with restart strategy for the path cover problem with time windows. *Mathematics*, 9(14), 1625. <https://doi.org/10.3390/math9141625>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

3. Synthese der Ergebnisse

In diesem Kapitel werden zunächst die Ergebnisse und Limitierungen der durchgeführten Studien vorgestellt und mit Bezug zu künftiger Forschung und Anwendung diskutiert. In der Folge wird die Grobarchitektur des entwickelten Scheduling-Frameworks präsentiert, welches als Dispositionssystem in den Industrieprojekten im Zusammenhang mit dem Promotionsvorhaben entwickelt wurde und teilweise bereits in den Geschäftsprozessen der Industriepartner eingesetzt wird. Abschließend werden die wichtigsten Erkenntnisse mit Bezug zur Leitfrage dieser Arbeit zusammengefasst.

3.1 Hauptergebnisse der durchgeführten Studien

In den Publikationen konnten die aktuellen Forschungsbedarfe und deren Relevanz (siehe Abschnitt 1.1) bestätigt und diskutiert werden. Darüber hinaus konnten die Forschungslücken im Sinne der Zielsetzung (siehe Abschnitt 1.3) systematisch bearbeitet werden. Die ersten beiden Publikationen widmen sich der ersten Teilfrage der Arbeit, wie realistische und möglichst allgemeingültig übertragbare Optimierungs- und Simulationsmodelle für eine humanzentrierte Ablaufplanung entwickelt werden können. Zudem wird in der ersten Publikation die zweite Teilfrage adressiert, wie RL die Effizienz des Planungsverfahrens verbessern kann:

Publikation 1: A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling

Anhand eines realen Anwendungsfalls konnte ein erweitertes Dual Resource Constrained Flexible Job Shop Scheduling Problem (DRC-FJSSP) für eine humanzentrierte Werkstattfertigung mit komplexen Materialflüssen formuliert werden. Das Modell ermöglicht eine synchrone Optimierung der Zykluszeit sowie Termintreue und berücksichtigt flexibel zuweisbare Arbeitsstationen und Mitarbeiter, sequenzabhängige Rüstoperationen, Mitarbeiterfähigkeiten, Materialankunftszeitpunkte, teilautomatisierte Operationen für eine Mensch-Maschine-Kollaboration und komplexe Auftragspfade mit parallelen Operationen. Anhand der Parameter des o.g. Modells konnte eine generisch initialisierbare diskrete Ereignissimulation entwickelt werden. Sie stellt Entscheidungsknoten (Schnittstellen) für intelligente Suchverfahren

zur Optimierung von Reihenfolge- und Belegungsentscheidungen bereit. Ein umfassendes populationsbasiertes Suchverfahren hat unter Verwendung des Simulationsmodells in der Evaluierungsfunktion (Simheuristik) umsetzbare Produktionspläne mit einem guten Kompromiss aus Ergebnisqualität und Rechenzeit generiert. Die Integration von RL an den Entscheidungsknoten des Simulationsmodells konnte die Ergebnisse des Suchverfahrens weiter verbessern. Der RL-Agent konnte durch die gezielte, situative Auswahl von Prioritätsregeln sowie durch die Zuweisung alternativer Maschinen und Werker abermals bessere Ergebnisse erzielen. Dieser Effekt konnte sowohl im Zusammenhang mit synthetischen Probleminstanzen als auch für unternehmensbezogene Echtweltdaten beobachtet werden. Durch eine frei skalierbare Parallelisierung der Simulationslogik auf verteilte Rechenressourcen konnte die Effizienz des Lösungsverfahrens weiter verbessert werden. (Grumbach, Badr et al. 2023)

Publikation 2: Humanzentrierte Ablaufplanung von Montagelinien

Im Kontext einer humanzentrierten Ablaufplanung für eine reale Fließproduktion konnten neuartige Zielfunktionen modelliert werden, welche im Einklang mit einer originären betriebswirtschaftlichen Zielsetzung eine gleichmäßige Auslastung der Mitarbeiter herbeiführen. Auf diese Weise soll sichergestellt werden, dass die physische und psychosoziale Beanspruchung der Mitarbeiter reduziert wird. Im Vergleich mit einer klassischen Zielfunktion zur Minimierung verspäteter Fertigstellungszeitpunkte konnten Belastungsspitzen an den Arbeitsplätzen mithilfe moderner Metaheuristiken geglättet werden. In einer Anwendungsstudie wurden unternehmensbezogene Realweltdaten zur Modellierung und Evaluation der Verfahren verwendet. Dennoch wurde der Ansatz so allgemeingültig wie möglich entwickelt, sodass die Zielfunktionen und die Optimierungsmethoden auf andere Produktionsformen übertragen werden können. (Vollenkemper et al. 2023)

In den ersten beiden Publikationen wurde eine deterministische Umgebung angenommen, bei der die Parameter aus der Eingabe (z.B. Sollzeiten) unveränderlich feststehen. Die nächsten beiden Publikationen befassen sich mit stochastischen Umgebungen, in denen Unsicherheiten wie schwankende Bearbeitungszeiten oder dynamische Ereignisse vorherrschen. Die robuste Ablaufplanung ist ein Sonderfall der stochastischen vorausschauenden Ablaufplanung, die zum Ziel hat, eine Struktur für die Pläne zu finden, sodass Unsicherheiten möglichst gut absorbiert werden können und Umplanungserfordernisse minimiert werden (Leon, V. Jorge, S. Wu und Storer 1994; Davenport, Gefflot und Beck 2001; Grumbach, A. Müller et al. 2022). Gemäß der Publikationen 3 und 4 sind zur Bestimmung genauer Robustheitsmetriken viele Simulationen in Form von Monte-Carlo-Experimenten (MCE) erforderlich, weshalb

viele Autoren rasch berechenbare, aber ungenaue Ersatzmetriken (Surrogatmaße, kurz: SM) vorschlagen. Infolgedessen existiert ein Trade-off bei der Auswahl einer der beiden Strategien: schnelle, aber ungenaue Berechnung mit SM vs. genaue aber langsame Berechnung mit MCE. Die nächste Publikation widmet sich deshalb der dritten Teilfrage, wie ML eingesetzt werden kann, um MCE mit komplexen stochastischen Materialflusssimulationen zur Bestimmung von Robustheitsmetriken zu substituieren:

Publikation 3: Robustness Prediction in Dynamic Production Processes – A new Surrogate Measure based on Regression Machine Learning

Im Kontext von dynamischen Fertigungsprozessen mit schwankenden Bearbeitungszeiten können ML-Regressionsmodelle erfolgreich zur Vorhersage der Zykluszeitrobustheit eingesetzt werden. Wie in der Publikation genauer erläutert, bezieht sich die Robustheit auf eine Performancemetrik des Gesamtplans. Z.B. ist ein Plan umso robuster, wenn die geplante Zykluszeit nur sehr wenig von der tatsächlich erreichten abweicht. Durch die Berücksichtigung überlappender Unsicherheitsverteilungen auf dem kritischen Pfad und zusätzlicher strukturbeschreibender Eingabemerkmale konnten bisherige SM zur Annäherung der Robustheit übertroffen werden. Insbesondere ermöglichen lineare Regressionsmodelle eine sehr hohe Vorhersagegüte, wenn sie angemessen trainiert werden. Sie können die Robustheit in Echtzeit annähernd so präzise abschätzen, wie es bisher nur mit rechenintensiven MCE möglich war. Auf diese Weise lässt sich die Laufzeit eines robusten Scheduling-Verfahrens erheblich reduzieren, ohne dass dabei die Ergebnisqualität signifikant beeinträchtigt wird. (Grumbach, A. Müller et al. 2023)

Zur Beantwortung der letzten Teilfrage befasste sich die folgende Publikation mit der Entwicklung eines RL-Ansatzes für die robuste Ablaufplanung, bei der auch Stabilitätsaspekte im Zusammenhang mit einer Zyklus- und Flusszeitoptimierung berücksichtigt werden:

Publikation 4: Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning

Es konnte ein RL-Agent entwickelt werden, welcher innerhalb von Ablaufplänen einer stochastischen Reihenfertigung gezielt Sollzeiten für die Operationen festlegt. Auf diese Weise wird ermöglicht, dass die Robustheit mit der konfliktären Stabilität der erzeugten Pläne effizient ausbalanciert wird. Im Unterschied zur Robustheit werden zur Berechnung der Stabilität alle Operationen und nicht nur eine Performancemetrik des Gesamtplans berücksichtigt. Auf diese Weise kann ermittelt werden, wie präzise die einzelnen Operationen zeitlich eingeplant sind. Im Wesentlichen konnte beobachtet

werden, dass unter Verwendung von Sollzeit-Erwartungswerten eine hohe Stabilität zu Lasten der Robustheit erreicht wird. Für eine höhere Robustheit ist es in vielen Fällen erforderlich, konservativere Zeiten anzunehmen. Anders ausgedrückt: Durch das gezielte Hinzufügen von Pufferzeiten auf dem kritischen Pfad des Plans konnten Effekte wie schwankende Bearbeitungszeiten und Maschinenstörungen *insgesamt* besser absorbiert werden. Auf der Betrachtungsebene der einzelnen Operation kommt es dadurch jedoch zu Ungenauigkeiten und im Zweifelsfall zu Stillstandzeiten. Der RL-Agent ist in der Lage, rasch einen Ausgleich zwischen Robustheit und Stabilität zu finden, indem der kritische Pfad gezielt entzerrt wird. Im Vergleich zu herkömmlichen Metaheuristiken benötigt er für diese Aufgabe nach geeignetem Training lediglich 2% der Rechenzeit, erreicht jedoch etwa 98% der Qualität, was ein sehr guter Kompromiss für eine feldsynchrone Planung ist. (Grumbach, A. Müller et al. 2022)

3.2 Perspektiven künftiger Forschung, Entwicklung und Anwendung

In diesem Abschnitt werden die Schlussfolgerungen der durchgeführten Studien in einen übergeordneten Zusammenhang gestellt sowie wesentliche Limitierungen und sich daraus ergebende Forschungsfragen diskutiert. Insgesamt können hierbei vier Fokusfelder abgeleitet werden, die für die weitere Untersuchung und Entwicklung smarter Dispositionssysteme für eine feldsynchrone Ablaufplanung dynamischer und humanzentrierter Fertigungsprozesse bearbeitet werden müssen (siehe Abbildung 3.1).

Dabei ist es von Bedeutung, virtuelle Fabriken möglichst umfassend, generisch und mit geringem Mitteleinsatz zu erzeugen sowie für die Interaktion mit effizienten Planungs- und Steuerungsverfahren zu öffnen. Zusätzlich bedarf es weiterer Forschung, um die Weiterentwicklung von ML-Techniken voranzutreiben und eine realistische, vorausschauende sowie effiziente Ablaufplanung unter Berücksichtigung humanzentrierter Faktoren zu verbessern. Zur Sicherstellung einer hohen Modellqualität und Prozessintegrität muss außerdem das Fokusfeld *Systementwicklung und Rollout* berücksichtigt werden. Es ist von großer Bedeutung, eine geeignete Softwarearchitektur zu entwickeln, sowie den Prozess der organisierten Einführung und Integration solcher Systeme zu untersuchen und zu standardisieren.

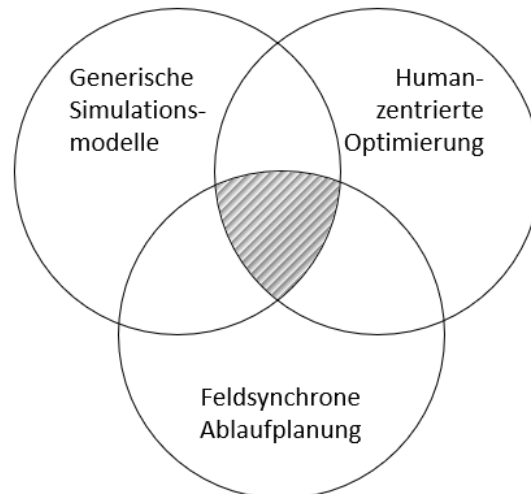


Abbildung 3.1: Fokusfelder zur Entwicklung smarter Dispositionssysteme für eine feldsynchrone Ablaufplanung dynamischer und humanzentrierter Fertigungsprozesse. Im Kontext des Überschneidungsbereichs müssen weitere Konzepte und Methoden zur Verbesserung der Modellqualität und Prozessintegrität erarbeitet werden

3.2.1 Generische Simulationsmodelle

Potenziale und Herausforderungen automatisch-generierbarer Modelle

Im Kontext komplexer Realweltprozesse mit umfangreichen Datensätzen weisen simulationsbasierte Ansätze gegenüber mathematischen Optimierungsmodellen einen Vorteil auf. Einerseits können Simulationsmodelle mit modernen Softwarelösungen auf eine intuitivere Art und Weise entwickelt werden. Dies ist vor allem darauf zurückzuführen, dass die Komplexität der realen Welt nicht in mathematisch wohldefinierte Nebenbedingungen überführt werden muss. Auch wenn mittels mathematischer Modelle die genaueste Formulierung des Planungsproblems erzielt werden kann, ist das Vorgehen sehr aufwändig und erfordert ein tiefes Wissen in der mathematischen Optimierung. Andererseits können Simulationsmodelle auf Basis von Stamm- und Bewegungsdaten eines Produktionssystem (z.B. Manufacturing Execution System) automatisch generiert werden. (Klemmt et al. 2009) Vor allem für kleine und mittlere Unternehmen ist eine (teil-)automatische Modellerzeugung in vielen Situationen vorteilhaft, da die manuelle Datenbeschaffung, Entwicklung und Wartung eines Simulationsmodells ebenfalls mit großen Aufwänden verbunden ist. Dabei ist hervorzuheben, dass allein die Datenbeschaffung durchschnittlich bis zu 50% der Entwicklungszeit beanspruchen kann. Für eine automatische Erzeugung valider Modelle ist es erforderlich, dass eine hohe Datenqualität und reibungslose Datenversorgung aus dem Produktionssystem sichergestellt ist. Vor allem in komplexen Umgebungen, wie der flexiblen Werkstattfertigung, ist die Berücksichtigung umfangreicher Nebenbedingungen eine Herausforderung. Rein datengetriebene Ansätze können zwar Prozessschritte ex post auf Basis von Stamm- und Bewegungsdaten

rekonstruieren, stoßen bei der Identifikation der genauen Steuerungslogik jedoch auf Schwierigkeiten. Insofern verbleibt eine gewisse Unsicherheit, welche Nebenbedingungen auf welche Weise die Prozessabläufe beeinflussen können. Aus diesem Grund können Hybridansätze geeignet sein, welche eine möglichst generische und parametrisierbare Steuerungslogik integrieren und mittels datengetriebener Initialisierung ergänzt werden. (Schlecht, de Guio und Köbler 2023)

Steuerungslogik für eine flexible Werkstattfertigung

Ein solcher Hybridansatz wird in der ersten Publikation (*A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling*) vorgestellt. Ergänzend zu einer generischen Steuerungslogik, welche alle im DRC-FJSSP formulierten Nebenbedingungen abbildet, werden die im Produktionsleitsystem vorhandenen Daten (aktuelle Aufträge, Arbeitspläne usw.) zur Initialisierung des Simulationsmodells verwendet. Auf diese Weise wird die Simulation stets datengetrieben parametrisiert und auf die aktuelle Situation spezialisiert. Mithilfe des Python-Frameworks *Simpy* wurde ein ressourcenorientierter Simulationsansatz (Schlecht, de Guio und Köbler 2023) gewählt, bei dem jede in den Eingabedaten vorhandene Arbeitsstation als asynchroner Prozess simuliert wird. Gemäß Abbildung 3.2 startet jede Station zunächst im Zustand *Station frei*. Sobald Aufträge verfügbar sind, die an dieser Station bearbeitet werden können, wechselt der Prozess in den Zustand *Auftrag zuweisen*. Hierbei ist es nun möglich, die Auftrags-Warteschlange mittels Entscheidungslogik zu organisieren, indem Aufträge z.B. nach ihrer Dringlichkeit sortiert werden oder indem Belegungsentscheidungen angepasst werden. Dies bezieht sich u.a. auf die vorgesehenen Mitarbeiter, kann aber bewirken, dass der Auftrag einer anderen Station zugewiesen wird. Nach der Auswahl und Zuweisung eines Auftrags beginnt der Rüstvorgang und/oder die Umsetzung der aktuellen Operation. Dazu wird ein weiterer *Simpy*-Prozess gestartet, welcher die Mitarbeiter über die Sollzeit zur Realisierung der Operation anfragt und dann blockiert. Weitere Details befinden sich in der Publikation sowie in Abschnitt 3.3.2. Bezogen auf den entwickelten Ansatz existieren zwei wesentliche limitierende Faktoren, die in den nachfolgenden Absätzen diskutiert werden.

Grenzen der entwickelten Steuerungslogik (Limitation 1)

Erstens ist die generische Steuerungslogik auf die Darstellungsgrenzen des in der Studie formulierten DRC-FJSSP begrenzt, was aus ganzheitlicher Planungssicht eine Beschränkung der Modellqualität ist. An der Stelle, wo die Nebenbedingungen des DRC-FJSSP hinsichtlich der Abbildung der realen Welt an seine Grenzen kommen, stößt auch das Simulationsmodell an seine Grenzen. So ist das Simulationsmodell zwar

auf alle Produktionsformen innerhalb dieser Grenzen übertragbar und damit auch in vielen Unternehmen der Domäne einsetzbar. Darüber hinausgehende Anforderungen können jedoch nicht ohne entsprechende Weiterentwicklungen abgebildet werden. Grundsätzlich liegt der Fokus auf einer kapazitätsorientierten Ablaufplanung mit zentral gesteuerten Materialflüssen, wodurch andere Planungsstufen und -paradigmen weitgehend außer Acht gelassen werden. So bleibt es z.B. fraglich, ob die entwickelten Methoden in Fertigungsprozessen eingesetzt werden können, die nach dem Pull-Prinzip organisiert sind. Es besteht jedoch unter Umständen die Möglichkeit, dass sie modifiziert werden könnten, um diesen spezifischen Anforderungen gerecht zu werden. Weiterhin wird mittels sequenzabhängiger Rüstzeiten zwar eine kurzfristige Losgrößenoptimierung ermöglicht, indem gleiche oder ähnliche Produkte zeitlich auf einer Arbeitsstation gebündelt werden. Eine längerfristige Losgrößenoptimierung unter Berücksichtigung von Primärbedarfen ist jedoch nicht mit dem aktuellen Modell möglich. Eine weitere Schwäche des Modells ist der Fokus auf eine diskrete azyklische Produktion ohne Zwischenpuffer. Künftige Arbeiten sollten generische Ansätze entwickeln, sodass auch eine kontinuierliche Produktion, Kreislauftechniken oder Zwischenlager abgebildet werden können. Ferner ist das Modell auf die beiden Ressourcentypen *Mitarbeiter* und *Arbeitsstation* begrenzt. In der Praxis können für eine realistische Planung jedoch weitere Ressourcentypen wie z.B. Werkzeuge, Transporteinheiten oder Lagerorte relevant sein (siehe z.B. Ba, Y. Li und M. Yang 2016; Reddy et al. 2021).

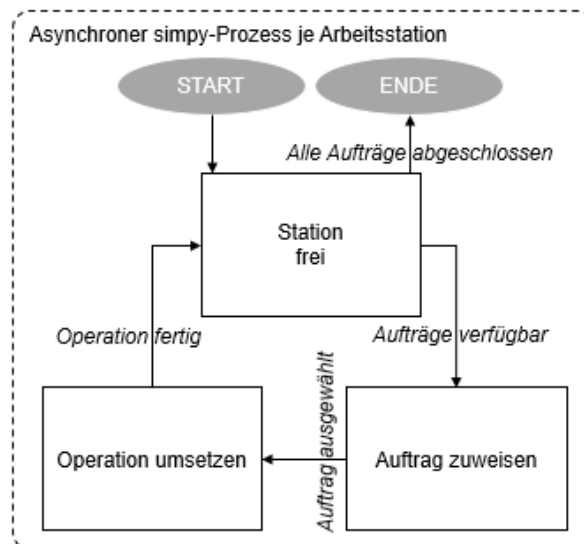


Abbildung 3.2: Zustandsautomat zur Simulation eines asynchronen Stationsprozesses mit drei Zuständen (Kästchen) und bedingten Transitionen (Pfeile). Vereinfachte Darstellung nach Grumbach, Badr et al. (2023)

In diesem Kontext wären nicht nur Literaturreviews über bestehende Modelle (siehe z.B. Da Col und Teppan 2022; Serrano-Ruiz, Mula und Poler 2021), son-

dern ergänzende qualitative Untersuchungen im Rahmen von vergleichenden Fallstudien bzw. Querschnittsstudien erforderlich. Dadurch können praktisch relevante wesentliche Modellaspekte und Anforderungen für bestimmte Shopfloor-Typen induziert werden, die dann als Grundlage für die Entwicklung einer umfassenderen generischen Steuerungslogik dienen. (Romero-Silva, Santos und Hurtado-Hernández 2022) Dabei wäre es auch interessant, das Betrachtungslevel von einer isolierten Simulation der lokalen Produktionsumgebung auf ein Lieferkettennetzwerk bzw. standortübergreifende Produktion abzuheben, sodass im Rahmen der Ablaufplanung z.B. auch externe Arbeitsgänge und Transportvorgänge geplant und analysiert werden können (siehe z.B. Liu et al. 2017).

ML-gestützte Modellerzeugung (Limitation 2)

Zweitens wurden keine Methoden untersucht und entwickelt, wie eine datengetriebene Simulationsmodellerstellung auf Basis von Mustern in Bewegungsdaten umgesetzt werden kann. Einige Autoren wie z.B. Lugaresi et al. (2019), Mesabbah und McKeever (2018) und Mesabbah, Abo-Hamad und McKeever (2019) verwenden Process Mining Techniken um Materialflüsse, Auftragspfade sowie ressourcenspezifische Relationen für eine datengetriebene Simulationsmodellerstellung automatisch zu rekonstruieren. Dabei werden Ereignislog-Dateien mit Zeitstempeln und Objektinformationen systematisch ausgewertet und in ein graphenbasiertes Format (z.B. Petrinetz) extrapoliert. Auf diese Weise ist es möglich, wichtige ergänzende Informationen wie z.B. Förderbandgeschwindigkeiten aufzudecken, die üblicherweise nicht in herkömmlichen Produktionsstammdaten vorzufinden sind. Zukünftige Arbeiten sollten weiter untersuchen, wie solche ergänzenden Process Mining Techniken in datengetriebenen Ansätzen bzw. Hybridansätzen zur automatischen Modellerzeugung eingesetzt werden können, um die Qualität und den Abdeckungsgrad der Modelle zu verbessern. Eine weitere Forschungsfrage in diesem Rahmen ist, welche Anforderungen an die Datenbasis erfüllt werden müssen, damit Verfahren dieser Art praktisch einsetzbar sind. Zudem müssen Reifegrade und Gütekriterien aufgestellt werden, um die Technologiebereitschaft für generische simulationsbasierte Ansätze anhand der vorliegenden Datenbasis zu bestimmen.

Eine zentrale Fragestellung im Rahmen von (teil-)automatisiert erzeugbaren Simulationsmodellen betrifft folglich die präzise Generierung bzw. effiziente und benutzerfreundliche Entwicklung sowie Validierung einer komplexen Steuerungslogik. Denn nur, wenn die Steuerungslogik die tatsächlichen Nebenbedingungen umfassend berücksichtigt, ist eine realistische und belastbare Planung mithilfe des Simulationsmodells möglich. Mit dem Aufkommen von Large Language Models (LLM) in jüngerer Zeit eröffnen sich hierzu einige Möglichkeiten, die genauer untersucht werden

sollten. So sind die Modelle bereits in der Lage, einfache Optimierungsmodelle auf Basis einer Texteingabe zu generieren. Dabei wird der eingegebene Text in logisch zusammenhängende Entitäten überführt, um dann schließlich in Zielfunktionen und Nebenbedingungen konvertiert zu werden. Auf diese Weise wird es perspektivisch auch Nicht-Experten möglich gemacht, Planungsprobleme teilautomatisiert erzeugen zu lassen. (Gangwar und Kani 2022; Ramamonjison et al. 2023) LLM haben derzeit jedoch noch Schwierigkeiten, komplexere Systeme mit vielen Entitäten und umfassenden Relationen zu modellieren (Cámara et al. 2023), wozu auch Steuerungslogiken von Simulationsmodellen gezählt werden können. Trotz alledem erwarten Felten, Raj und Seamans (2023), dass mittel- bis langfristig ein überdurchschnittliches Risiko für die Tätigkeiten von Operations-Research-Analysten besteht, zumindest teilweise durch LLM substituiert zu werden. Untersucht wurden 774 Berufe, wobei OR-Analysten Platz 122 in der Rangfolge nach Risiko einnehmen.

3.2.2 Humanzentrierte Optimierung

In den durchgeführten Studien wurden verschiedene Modellierungs- und Vorhersagetechniken für eine humanzentrierte Ablaufplanung beleuchtet.

Situationsabhängige Sollzeiten

Anhand des formulierten DRC-FJSSP in der ersten Publikation werden zunächst nicht nur Arbeitsstationen, sondern auch Mitarbeiter und zugehörige Kompetenzprofile für eine realistische Ablauf- und Personaleinsatzplanung berücksichtigt. So können Mitarbeiter unterschiedlich ausgeprägte Fähigkeiten aufweisen, Arbeitsstationen zu rüsten oder zu bedienen, was einen Effekt auf die Sollzeit des Arbeitsganges hat. Weiterhin wird verhindert, dass eine Rüst- oder Bearbeitungsoperation einem Mitarbeiter zugewiesen wird, der nicht die dafür erforderliche Fähigkeit hat. Eine damit einhergehende Limitierung ist, dass die Sollzeiten bereits in der Eingabe des Verfahrens determiniert sind. Es werden keine situativen Effekte wie Stress oder Müdigkeit berücksichtigt, die in einem festgelegten Arbeitsprozess anfallen können und einen weiteren Einfluss auf die Sollzeit ausüben (Du et al. 2021). Legt das Verfahren beispielsweise einen Plan fest, bei dem ein Mitarbeiter eine sehr große Arbeitslast hat, indem er z.B. zwischen verschiedenen Stationen oft hin- und herwechselt, wird dennoch die statische Sollzeit als Parameter eingesetzt. Künftig sollte weiter untersucht werden, welche Situationen im Fertigungsprozess auf welche Weise Verlängerungen oder Verkürzungen der Sollzeit bzw. Arbeitsunterbrechungen bedingen (siehe z.B. Du et al. 2021). Zudem wäre es auch denkbar, dass einem Arbeitsgang zeitgleich mehrere, sich gegenseitig unterstützende Mitarbeiter zugewiesen werden und dies auch einen Effekt auf die Dauer oder Belastung hat (siehe z.B. Z. Wang et

al. 2023).

Anknüpfend daran haben wir (A. Müller und Grumbach 2023) untersucht, wie auf Basis verschiedener (impliziter) Belastungsfaktoren realistische Sollzeiten für Arbeitsgänge vorhergesagt werden können. Mithilfe einer großen Menge realer Buchungsdaten aus der Produktion eines Unternehmens mit flexibler Werkstattfertigung konnten dahingehend Regressionsmodelle trainiert werden, welche die Genauigkeit der in den Stammdaten hinterlegten Sollzeiten übertreffen. Dies ist ein Vorteil im Zusammenhang mit großen Artikelstämmen und umfangreichen Arbeitsplänen, da die manuelle Datenpflege durch ML-Methoden zumindest teilweise ersetzt werden kann. Als Eingabemerkmale wurden u.a. die bestehende Sollzeit aus den Stammdaten, die wöchentliche Auslastung des zugewiesenen Mitarbeiters, die Arbeitsstation, die Produktkategorie, aufgetretene Fehler im bisherigen Prozess, die Dringlichkeit des Fertigungsauftrags, die Komplexität der Stückliste, die Stückzahl sowie die aktuelle Wochen- und Tageszeit verwendet. Auf diese Weise konnte situationsbezogen eine 23% präzisere Planzeit gegenüber den vorhandenen Stammdaten geschätzt werden. Interessant ist dabei die Tatsache, dass die in den Stammdaten hinterlegte Sollzeit den größten Einfluss auf die Prognose hat, jedoch zugleich negativ mit dieser korreliert. Dies ist ein Indikator dafür, dass Expertenschätzungen über die erwartete Dauer eine wichtige Variable für Regressionsmodelle darstellen. Zudem geht daraus hervor, dass die zuständigen Arbeitsvorbereiter dazu tendieren, komplexe Arbeitsgänge hinsichtlich der Dauer zu konservativ zu schätzen und vice versa. Hingegen korreliert das wöchentliche Arbeitsvolumen positiv mit der tatsächlich benötigten Zeit: Je mehr der Mitarbeiter in der Periode ausgelastet ist, desto länger benötigt er für die ihm übertragenen Aufgaben im Vergleich zur Sollzeit. Die Ursache kann mutmaßlich auf Erschöpfungseffekte zurückgeführt werden, jedoch auch sehr vielfältig sein und sollte deshalb genauer untersucht werden. Zudem hat die Entwicklung und Reduktion des Merkmalsvektors in einigen Aspekten sehr unternehmensspezifisch stattgefunden. Zukünftige Studien sollten vergleichbare Modelle in anderen Kontexten trainieren, um schließlich die Generalisierbarkeit der Merkmale zu untersuchen.

Schätzung realistischer, unsicherer Sollzeiten während der Simulation

Darüber hinaus wurde im Rahmen der Publikationen kein Konzept entwickelt, wie ein solches Vorhersagemodell zur Schätzung einer realistischen Operationsdauer in ein Optimierungsmodell bzw. -verfahren integriert werden kann. So wäre es beispielsweise interessant, wie situationsabhängige Sollzeiten in einer Simheuristik während der Simulation bestimmt werden können und so die Struktur des dadurch erzeugten Planes beeinflussen. Zukünftige Arbeiten sollten an dieser Stelle ansetzen, um die mitarbeiterbezogene Nachhaltigkeit der erzeugten Ablaufpläne zu verbessern. Ferner

kann es in der Realität zu Unsicherheiten wie schwankenden Bearbeitungszeiten kommen. Im Sinne der robusten Optimierung unter Unsicherheit sollte künftig die Vorhersagbarkeit eines Intervalls bzw. einer Verteilungsfunktion analysiert werden. Beispielsweise könnte im Kontext von normalverteilten Schwankungen ein Multi-Output-Regressor entwickelt werden, welcher sowohl den Mittelwert als auch die mögliche Varianz der unsicheren Bearbeitungszeit vorhersagt. Künftige Studien sollten Modelle und Verfahren entwickeln, welche nicht nur humanzentrierte robuste Ablaufpläne erzeugen, sondern zugleich auch effiziente Umplanungsmechanismen bei schwer antizipierbaren Störungen integrieren. Derartige Störungen liegen z.B. dann vor, wenn ein Mitarbeiter unerwartet nicht bzw. nicht pünktlich zur Arbeit erscheint oder nicht mehr genügend Zeit hat, einen Arbeitsgang fristgerecht abzuschließen (Hassani, Desaulniers und Elhallaoui 2020).

Balancierung humanzentrierter, ökologischer und ökonomischer Ziele

Es nicht nur wichtig, humanzentrierte Aspekte wie realistische Sollzeiten oder geeignete Umplanungsprozeduren in die Verfahren zu integrieren, sondern auch entsprechende Zielfunktionen zu formulieren. In der zweiten Publikation (*Humanzentrierte Ablaufplanung von Montagelinien*) wurde herausgefunden, dass die Minimierung der Belastungsvarianz im Kontext einer getakteten Fließfertigung die beste Glättung von Lastspitzen erzielte, ohne die konfliktäre Termintreue maßgeblich zu beeinflussen. Die Belastungen wurden jedoch lediglich anhand der statischen Sollzeiten ermittelt. Es wurde angenommen, dass viele umfangreiche Arbeitsgänge hintereinander bzw. geringe Pufferzeiten zwischen umfangreichen Arbeitsgängen eine hohe Belastung für den Mitarbeiter verursachen. Andere Faktoren, wie persönliche Präferenzen, physische Belastungen oder Monotonieeffekte wurden in der Modellierung nicht berücksichtigt. Da der industrielle Sektor einen Anteil von einem Drittel am stetig steigenden weltweiten Energiebedarf hat, sollten neben humanzentrierten Faktoren auch ökologische Aspekte, wie die Einsparung von Energie, vermehrt in künftige Ablaufplanungsmodelle integriert werden (Alaouchiche, Ouazene und Yalaoui 2020; Mohan, Lanka, N. A. Rao et al. 2022). So sind weitere Studien erforderlich, die untersuchen, wie verschiedene ökonomische, humanzentrierte sowie ökologische Faktoren vor allem im Zusammenhang mit Unsicherheiten balanciert werden können (Destouet et al. 2023).

Zusammenfassend existieren zum jetzigen Zeitpunkt einige Studien, welche Teilaspekte der humanzentrierten Optimierung behandeln (Destouet et al. 2023). Es konnte jedoch kein ganzheitliches Konzept identifiziert werden, welches für dynamische Fertigungsprozesse in flexiblen Werkstattumgebungen eingesetzt werden kann. Zukünftige Arbeiten sollten Modelle und Verfahren entwickeln, welche zunächst

die erforderlichen realistischen Nebenbedingungen und Planungsmöglichkeiten im Kontext soziotechnischer Produktionsumgebungen zur Verfügung stellen. Darüber hinaus sollten die Verfahren nicht nur effizient und reaktiv umplanen können, sondern Ablaufpläne auch proaktiv unter Berücksichtigung von Unsicherheiten erzeugen. Hierbei ist es erforderlich, realistische und situationsabhängige Sollzeiten gezielt abzuschätzen, damit eine möglichst nachhaltige und robuste Struktur des Ablaufplans gewährleistet wird. Letztlich sollten auch Konzepte entwickelt werden, damit Belastungen einerseits differenziert gemessen und andererseits gemeinsam mit anderen Zielen optimiert werden können. Hier bieten sich zunächst interdisziplinäre In-Situ-Untersuchungen mit geeigneten, repräsentativen Anwendungsfällen an, um die dafür erforderlichen Metriken zu ermitteln. Ebenso sollten Untersuchungen wie z.B. Fallstudien oder Feldexperimente durchgeführt werden, um den tatsächlichen Effekt einer humanzentrierten Planung messen und bewerten zu können.

3.2.3 Feldsynchrone Ablaufplanung

Wie anhand der Publikationen dargelegt, sind ML-Techniken dazu geeignet, die Effizienz von sowohl deterministischen als auch stochastischen Optimierungsverfahren im Bereich der Fließ-, Reihen- und Werkstattfertigung zu verbessern.

ML-gestützte effiziente robuste Optimierung unter Unsicherheit

In der dritten Publikation (*Robustness Prediction in Dynamic Production Processes – A new Surrogate Measure based on Regression Machine Learning*) wurde zunächst eine Methode für das Training von Regressionsmodellen zur präzisen Vorhersage der Zykluszeitrobustheit erarbeitet. Die Ergebnisse legen nahe, dass ML-Modelle im Vergleich zu klassischen SM eine ähnlich schnelle Rechendauer aufweisen, jedoch in der Ergebnisqualität nah an rechenintensive MCE herankommen. Durch die Einbettung eines angemessen trainierten Vorhersagemodells in eine Simheuristik zur Erzeugung robuster Ablaufpläne kann die Rechenzeit etwa um den Faktor der Quadratwurzel hinsichtlich der benötigten Simulationen beschleunigt werden. Dies ist nicht nur ein Effizienzvorteil für eine feldsynchrone Ablaufplanung, sondern spart auch Energiekosten bzw. schont Ressourcen bei der regelmäßigen Erzeugung komplexer Ablaufpläne. Der Versuchsaufbau ist jedoch insofern limitiert, als dass ein abstraktes Optimierungsmodell mit schwankenden Bearbeitungszeiten als einzige Unsicherheitsquelle verwendet wurde. Obwohl in der Arbeit gammaverteilte Bearbeitungszeiten zugrunde gelegt wurden, sollte die Methode für beliebige Verteilungstypen übertragen werden können. Hintergrund ist, dass der Merkmalsvektor zur Vorhersage der Zykluszeitrobustheit so gestaltet wurde, dass sogenannte Überlappungsintegrale eingesetzt werden. Diese können unabhängig von der konkreten Verteilung anhand der austauschbaren

Wahrscheinlichkeitsdichtefunktion und der kumulativen Verteilungsfunktion berechnet werden. Dennoch sollten künftige Studien weitere Unsicherheiten modellieren und die damit verbundene erzielbare Vorhersagegüte evaluieren. Neben schwankenden Zeiten wäre es interessant zu untersuchen, wie sowohl ressourceninduzierte Ereignisse (z.B. Maschinenausfälle) als auch nachfrageinduzierte Ereignisse (z.B. Auftragsstornierungen) (Kilger, Reuter und Stadtler 2014) in den Vorhersagemodellen berücksichtigt werden können. Ferner muss die externe Validität des Ansatzes weiter evaluiert werden, indem er auf Modelle mit umfangreicheren Nebenbedingungen, realistische Szenarien mit Echtweltdaten sowie zusätzliche Robustheits- und Stabilitätsmetriken angewendet wird. Beispielsweise könnte der Ansatz im Zusammenhang mit einem realistischen und stochastischen DRC-FJSSP erprobt werden, in dem neben schwankenden, situationsbedingten Rüst- und Bearbeitungszeiten, sowohl Maschinen- als auch Mitarbeiterausfälle, Auftragsstornierungen, ungeplante Eilaufträge und unsichere Materialankunftszeitpunkte möglich sind. Neben der Zykluszeitrobustheit sollten weitere Metriken wie Stabilität, Flusszeitrobustheit, Termintreue-Robustheit oder humanzentrierte Kenngrößen berücksichtigt werden.

Die entwickelte Methode zur Robustheitsvorhersage kann mit dem Agentenkonzept aus der vierten Publikation (*Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning*) kombiniert werden. Da der RL-Agent zur Erzeugung robuster und stabiler Ablaufpläne noch mithilfe von MCE trainiert wurde, dauert der Trainingsprozess je nach verwendeter Hardware mehrere Minuten. Die MCE sind hierbei der entscheidende Engpass, weil sie nach jeder Trainingsepisode ausgeführt werden müssen, um die Endbelohnung auf Basis der Robustheit und Stabilität zu bestimmen. Mit dem Einsatz der Regressionsmodelle anstelle der MCE kann die Trainingszeit womöglich auf mehrere Sekunden bis wenige Minuten reduziert werden. Diese Vereinigung der beiden Ansätze könnte in einer anknüpfenden Studie getestet werden. Zudem sollten auch weitere Paradigmen hinsichtlich des Agentendesigns erprobt werden. Das Konzept ist bis dato noch insofern beschränkt, als dass ein Ablaufplan bereits feststeht und der Agent in einer zweiten Verbesserungsprozedur die Operationszeiten manipuliert, um den kritischen Pfad zu entzerren. Wie im Fazit der vierten Publikation erläutert, wäre es interessant zu untersuchen, wie ein Agent nicht nur die robuste und stabile Planung übernimmt, sondern auch grundsätzliche Reihenfolge- und Belegungsentscheidungen trifft - also eine deterministische und robuste Planung vereint. Dies könnte beispielsweise nicht nur über das Einfügen von Pufferzeiten, sondern auch durch die Erzeugung von Nachbarlösungen durch den Austausch von Sequenzen oder Ressourcen geschehen. Gemeinsam mit der dritten Publikation ist die vierte Publikation in dieser Hinsicht als Machbarkeitsstudie aufzufassen, wie eine effiziente robuste Ablaufplanung mithilfe von ML-Techniken verbessert werden kann. Auch hier gilt, dass das entwickelte bzw. modifizierte Konzept

auf weitere, komplexere Szenarien übertragen werden muss. Denn nur so kann eine Allgemeingültigkeit des Ansatzes weiter untersucht und plausibilisiert werden.

RL-gestützte deterministische Optimierung in flexiblen Werkstätten

In der ersten Publikation (*A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling*) wird ein RL-Agent für ein DRC-FJSSP vorgestellt, welcher im Zusammenspiel mit einer Simheuristik zur Verbesserung von Reihenfolge- und Belegungsentscheidungen eingesetzt wird. Nach abgeschlossenem Training wird der Agent während der Simulation einer Lösung aktiv, wenn eine Entscheidungssituation an einer Station vorliegt (siehe Abschnitt 3.2.1, Abbildung 3.2, *Auftrag zuweisen*). Auf Basis eines umfassenden Zustandsraumes inkl. Warteschlangen-Kennzahlen, Schlupfzeiten und Lokalitäten der Maschinen im Materialfluss, kann der Agent die ihm aktuell am geeignetsten erscheinende Prioritätsregel für die Arbeitsstation festlegen, Operationen in andere alternative Stationswarteschlangen verschieben sowie zugewiesene Mitarbeiter zur Umsetzung der Operation austauschen. Auf diese Weise findet eine Nachoptimierung des durch die Metaheuristik festgelegten Lösungskandidaten statt, was zu einer rascheren Identifizierung lokaler Optima führt. Im Vergleich zur rein auf Zufall basierenden Metaheuristik sind somit weniger Iterationen erforderlich, um sowohl die Zykluszeit als auch die Termintreue zu verbessern. Darüber hinaus konnte in den meisten Fällen eine geringere Streuung bei den erreichten Ergebnissen beobachtet werden. Dies ist ein Indikator dafür, dass der Algorithmus zuverlässiger globale Optima identifizieren kann. Vorausgesetzt ist, dass der Agent bereits mit ähnlichen Situationen in der entsprechenden Fertigungsumgebung im Training konfrontiert war und dadurch eine geeignete Strategie erlernt hat. So ist ein Agent im Kontext komplexer Auftragspfade und wenigen Maschinen nicht erfolgreich, wenn er im Training stets mit vielen Maschinen und vielen, unterkomplexen Aufträgen geplant hat. Im Kontext eines realen Anwendungsfalles kann das KNN des Agenten jedoch erfolgreich in unbekanntem, aber ähnlichen Probleminstanzen verwendet werden.

Verbesserungsmöglichkeiten für RL-Agenten in der PPS

Da komplexe und dynamische Fertigungsumgebungen selbst in einem einzigen Unternehmen viele unterschiedliche Situationen und Zustände hervorbringen können, ist es relevant zu untersuchen, wie RL-Agenten bzw. die internalisierten Strategien noch besser generalisiert werden können (Gerpott et al. 2022). Dies ist ein entscheidender Faktor zur Einsparung von Rechenzeit und zur Verbesserung der Feldsynchronität, da auf diese Weise nicht für jede neue Probleminstanz auch ein neuer Trainingsprozess durchgeführt werden muss. So wäre es eine Möglichkeit, dass ein universeller, vor-

trainierter RL-Agent für eine neue Situation vergleichsweise kurz nachtrainiert wird und so die zunächst begrenzt effektive Strategie auf das konkrete Problem angepasst wird (siehe Abschnitt 1.2.4, *Einordnung von ML-Verfahren, Online-Lerner*). Je nach Zusammensetzung und Komplexität der Probleminstanz müssen die Hyperparameter jedoch oft spezifisch abgestimmt werden. In den Publikationen 1 und 4 bezieht sich dies insbesondere auf die Episodenlänge und die Aktualisierungsintervalle des KNN. Zukünftige Forschung sollte sich damit befassen, wie geeignete Hyperparameter auf Basis einer Probleminstanz automatisch vorhergesagt werden können, um aufwändiges und manuelles Hyperparameter-Tuning zu vermeiden (siehe z.B. Shu, Meng und Zongben Xu 2021).

Zur Erlernung einer universellen Strategie in einer komplexen und dynamischen Umgebung ist darüber hinaus ein umfangreiches Training mit vielen Probleminstanzen erforderlich (Gil und Lee 2022; Zhe Xu et al. 2021). In anderen Worten: Je umfassender das Szenario und die Möglichkeiten der Ablaufplanung, desto *datenhungriger* ist der Agent bei der Erlernung einer geeigneten Strategie. Deshalb müssen für das Training entweder genügend Echtdaten oder realistische, synthetische Instanzen zur Verfügung gestellt werden. Ersteres kann in der praktischen Anwendung eine große Herausforderung sein, wenn sehr große Datenmengen für ein umfangreiches Training bereitgestellt werden müssen. Aus diesem Grund ist es sinnvoll, qualitativ hochwertige synthetische Instanzen auf eine flexible Weise in beliebigen Mengen zu erzeugen. In der Literatur gibt es zum einen klassische, etablierte Benchmarkinstanzen, die hauptsächlich entwickelt wurden, um die Leistung exakter Verfahren und Heuristiken für abstrakte Optimierungsprobleme wie das JSSP zu evaluieren (siehe z.B. Taillard 1993). Bekannte Datensätze sind jedoch in ihrer Quantität begrenzt und müssen zur Abbildung spezifischer Nebenbedingungen modifiziert werden. Zum anderen schlagen einige Autoren problemspezifische Methoden vor, wie realistische Datenstrukturen und Parameter auf Basis des (Pseudo-)Zufalls festgelegt werden können (siehe z.B. Harrison et al. 2022; Rigo et al. 2023). Derartige Methoden zur Erzeugung realistischer, synthetischer Instanzen unterliegen der Einschränkung, dass sie auf das jeweilige Optimierungsproblem oder Szenario ausgerichtet sind und in einer tendenziell aufwändigen, ingenieurmäßigen Vorgehensweise entwickelt werden müssen.

Künftige Arbeiten sollten untersuchen, wie generische, datengetriebene Ansätze diese Aufgabe übernehmen können. Figueira und Vaz (2022) legen nahe, dass Generative Adversarial Networks (GAN) für diesen Anwendungsfall weiter untersucht und entwickelt werden sollten. GAN sind ML-Modelle, die zur Erzeugung künstlicher, authentischer Datenobjekte (z.B. Fotos) eingesetzt werden können, indem sie sich an der Verteilung von realen Datenpunkten orientieren. Dabei ist der sogenannte

Generator für die Erzeugung des künstlichen Datenobjektes zuständig, welches dann vom sogenannten Diskriminator als realistisch oder unrealistisch klassifiziert wird. Im Laufe des Trainings werden diese beiden konkurrierenden Komponenten stets besser: Der Generator lernt, immer realistischere Objekte zu erzeugen, die vom Diskriminator als solche klassifiziert werden, wohingegen der Diskriminator immer besser in der Erkennung synthetischer Objekte wird. Die Autoren betonen, dass ML-Modelle im Vergleich zu regelbasierten Verfahren zwar zunächst aufwändiger zu entwickeln sind, aber dafür hochwertigere synthetische Daten generieren können. Jedoch liegt im Kontext von GAN der wissenschaftliche Schwerpunkt bis heute in der Erzeugung von Bilddaten. Zukünftige Forschung sollte sich vermehrt der Erzeugung von Daten in tabellarischer Form widmen, wozu auch Probleminstanzen von Ablaufplanungsproblemen gezählt werden können. In der Praxis könnte ein allgemeingültiger und übertragbarer ML-Ansatz nützlich sein, um RL-Agenten für komplexe Szenarien zu trainieren. Auf Basis einer Menge Echtdaten könnten mittels GAN viele authentische Trainingsinstanzen erzeugt werden.

Eine weitere zentrale Beschränkung ist, dass der Agent auf den Umfang des jeweiligen Optimierungsproblems begrenzt ist, wie es auch bei der Steuerungslogik des Simulationsmodells der Fall ist (siehe Abschnitt 3.2.1). Darüber hinaus ist der MDP hinsichtlich des Zustandsraumes und der Belohnungssignale speziell für die Optimierung vordefinierter Metriken (z.B. Zykluszeit oder Termintreue) entwickelt worden. Wenn andere Ziele oder andere Zielgewichtungen berücksichtigt werden sollen, muss der festgelegte MDP dahingehend modifiziert werden. In künftigen Arbeiten sollten noch allgemeingültigere und flexiblere Konzepte entwickelt werden, um weitere Zielfunktionen, deren freie Zusammensetzung bzw. die Priorisierung beliebiger konfliktärer Ziele festlegen zu können. Dazu gehören perspektivisch nicht nur übliche betriebswirtschaftliche Ziele wie z.B. Auslastung oder Durchsatz, sondern auch nachhaltige und humanzentrierte Größen (siehe Abschnitt 3.2.2). Ein möglicher Ansatz hierfür könnten Neugier-getriebene bzw. intrinsisch motivierte RL-Agenten sein. Nach Colas et al. (2022) entwickeln sogenannte autotelische Agenten ihre Strategie nicht nur auf Basis externer Ziele und Belohnungssignale sowie vordefinierter Aktions- bzw. Zustandsräume. Vielmehr verfolgen sie das Ziel bzw. setzen sie sich eigenständig Belohnungssignale, wenn sie erfolgreich explorieren, neue Situationen aufdecken und neue nützliche Fähigkeiten lernen, wozu z.B. die selbstständige, zweckmäßige Erweiterung des Zustands- oder Aktionsraumes gehört. So wird beispielsweise die Neugier des Agenten *geweckt*, wenn mit der Transitionsfunktion $P(s_{t+1}|s_t, a_t)$ (siehe Abschnitt 1.2.4) ein hoher Vorhersagefehler einhergeht. In diesem Fall kann der Agent durch entsprechende Belohnungen motiviert sein, den unerwarteten Zustandsübergang genauer zu untersuchen und ggf. neue Aktionsmöglichkeiten zu erlernen. Künftige Forschung in der Schnittstelle von PPS (Ablaufplanung) und

RL sollte untersuchen, wie Agenten auf Basis neuer, externer Ziele eigenständig Selbstbelohnungsanreize und geeignete Strategien erlernen können, ohne dass eine auf das Hauptziel zugeschnittene Belohnungsfunktion konstruiert werden muss.

Orchestration feldsynchroner und adaptiver Optimierungsverfahren

Im Hinblick auf ein in der Praxis einsetzbares Gesamtsystem sollten zukünftige Arbeiten die Weiterentwicklung und Vereinigung der vorgestellten Konzepte und Verfahren anstreben. Gemäß des in dieser Arbeit untersuchten Instrumentariums erstreckt sich dies aus methodischer Sicht vor allem auf folgende Bereiche:

1. Datengetriebene, (teil-)automatisierte Erzeugung von diskreten Simulationsmodellen zur Abbildung realistischer Auftrags- und Materialflüsse.
2. Überbrückung langer Simulationszeiten durch ML-basierte SM zur Bestimmung von Robustheitsmetriken.
3. Entwicklung generischer RL-Agenten mit kurzen, optimierten Trainingsprozessen für eine deterministische und stochastische Optimierung, die sich flexibel auf neue Situationen anpassen können.
4. Freie Definierbarkeit und Gewichtung multipler Optimierungsziele, wozu neben betriebswirtschaftlichen Größen und Robustheitsmetriken auch humanzentrierte und Nachhaltigkeitsfaktoren gehören.
5. Integration adaptiver bzw. reaktiver Methoden für effiziente Umplanungsprozeduren.

Gegenstand dieser publikationsbasierten Arbeit waren die ersten vier Bereiche, wohingegen die reaktive Planung nur in ihren Grundzügen diskutiert wurde (siehe z.B. Publikation 1). In der weiteren Forschung und Entwicklung orchestrierter Methoden ist es jedoch ratsam, größeren Fokus auf diesen fünften Punkt zu legen. Nach D. Gupta, Maravelias und Wassick (2016) ist eine systematische Umplanung dann erforderlich, wenn die initiale Struktur des Ablaufplanes kaum oder gar nicht mehr umsetzbar ist, nachdem unerwartete und/oder schwer abfederbare Ereignisse eingetreten sind. Die Autoren analysierten einige einschlägige Studien zu dem Thema und identifizierten wesentliche Herausforderungen und Methoden. Insbesondere muss identifiziert werden, in welcher Situation eine ereignisbasierte Umplanung überhaupt notwendig ist, was z.B. über Schwellwerte erlaubter Abweichungen der Solltermine definiert werden kann. Zudem muss ermittelt werden, welche Operationen vom eingetretenen Ereignis hinsichtlich der notwendigen Umplanung betroffen sind und wie ein Plan mit möglichst minimalen Eingriffen *repariert* werden kann. Dabei kann es vorteilhaft sein, Strategien zu entwickeln, welche die Auswirkungen der Ereignisse auf explizit oder implizit betroffene Operationen minimieren. Beispielsweise könnte

direkt ein Ersatz für eine ausgefallene Ressource festgelegt werden, anstatt dass auf die Ressource gewartet wird und sich die Operationen dadurch aufstauen. Je nach Anwendungsfall und vorherrschenden Anforderungen stellt sich darüber hinaus die Frage, ob eher eine vorausschauende und robuste oder eine reaktive Planung forciert werden sollte. Die ausschließlich robuste Planung geht mit dem Risiko einher, dass die erzeugten Pläne in ihren Pfaden ausgedehnt werden und damit eine konservative Struktur aufweisen. Dies kann die Ressourcenauslastung mindern und führt zu Opportunitätskosten. Eine rein rollierende, reaktive Planung kann jedoch nach sich ziehen, dass die Pläne oft geändert werden müssen und auf diese Weise Unruhe bzw. Planungsunsicherheit durch nur kurzfristig gültige Pläne entsteht. Infolgedessen wird ein aggregierter Online-Scheduling-Ansatz vorgeschlagen, der nicht nur vorausschauend plant und bei eingetretenen gravierenden Ereignissen reaktiv umplant, sondern auch in regelmäßigen periodischen Abständen geringfügige Planverbesserungen vornimmt. Die Implementierung des Ansatzes ist jedoch immer auf den Anwendungsfall abzustimmen, was ein aufwendiges Vorgehen ist. (D. Gupta, Maravelias und Wassick 2016)

Aktuelle Untersuchungen widmen sich auch der Frage, wie eine vorausschauende Umplanung (*Predictive ReScheduling*) umgesetzt werden kann, sodass spezifische Ereignisse bereits antizipiert werden, bevor sie eintreten. So haben beispielsweise Y. Yang et al. (2023) ein Framework für eine vorausschauende Umplanungsprozedur mit hypothetischen Neuaufträgen erarbeitet. Das Framework greift auf ein Live-Datenrepositorium einer virtuellen Fabrik zu und kann in einer flexiblen Werkstattumgebung eingesetzt werden. Abbildung 3.3 veranschaulicht die grundlegende Prozedur: Nachdem ein Basisplan erzeugt und freigegeben wurde, greift das Verfahren, beispielsweise nach einem definierten Intervall, auf die Live-Daten der virtuellen Fabrik zu. Hierbei werden datengetriebene Hypothesen für mögliche Ereignisszenarien erzeugt, auf Basis dessen dann vorrätige, präventive Pläne erzeugt werden. Das Verfahren wartet für eine gewisse Zeit, und überprüft dann, ob eines der Szenarien eingetroffen ist. Ist dies der Fall, wird der entsprechende präventive Plan unmittelbar und ohne weitere erforderliche Rechenzeit verfügbar gemacht. Trifft kein neuer Auftrag ein, wird die Prozedur wiederholt.

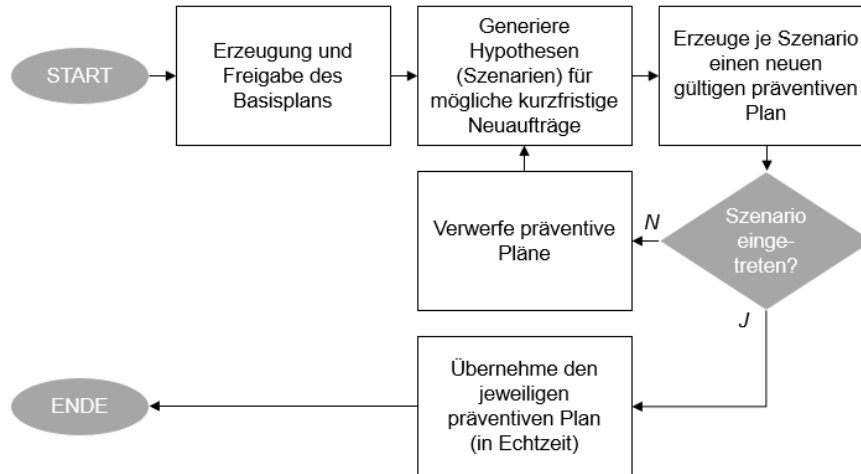


Abbildung 3.3: Prozess der vorausschauenden Umplanung mit dynamischen Neuaufträgen nach Y. Yang et al. (2023)

Im Kontext eines holistischen smarten Dispositionssystems müssen weitere Ereignistypen und Unsicherheiten für eine reaktive bzw. kontextbewusste, vorausschauende reaktive Ablaufplanung berücksichtigt werden. Dies kann sich im Sinne einer humanzentrierten Optimierung auch auf die Früherkennung mitarbeiterbedingter Ereignisse beziehen. So wäre es interessant zu untersuchen, wie z.B. Streamdaten (Ton- oder Motiondaten) zur rechtzeitigen Erkennung von Belastungen eingesetzt werden können. Auf diese Weise könnten Fehler oder Prozessverzögerungen frühzeitig erkannt werden, was eine Echtzeit-Umplanung ermöglicht.

3.2.4 Systementwicklung und Rollout

Im Zusammenhang mit herkömmlichen APS-Systemen existieren maßgebliche Hürden bei der nutzerorientierten Einführung und Anwendung (siehe Abschnitt 1.1). Besonders für kleine und mittelständische Unternehmen ist es entscheidend, die Flexibilität der Lösung, das Vertrauen in die Lösung, die Förderung durch relevante Stakeholder, das Kosten-Nutzen-Verhältnis, sowie die Servicequalität zu optimieren (Gui et al. 2021). Aus praktischer und wirtschaftlicher Perspektive ist es daher wichtig, eine ganzheitliche Systemintegration mit minimalem Mitteleinsatz (*Plug & Play*) zu ermöglichen und eine gute Prozessintegrität sicherzustellen. Daraus lässt sich schlussfolgern, dass sowohl flexible Systemarchitekturen als auch anwenderorientierte und individuell anpassbare Herangehensweisen zur Einführung smarter Dispositionssysteme entwickelt sowie erprobt werden müssen. So existieren bereits iterative Vorgehensmodelle zur systematischen Einführung von RL-basierten Ablaufplanungsmethoden (siehe z.B. Lang 2023a) oder virtuellen Fabriken (siehe z.B. Behrendt et al. 2019). Nichtsdestotrotz wurden diese Aspekte in der vorliegenden Arbeit nicht in der Tiefe behandelt, was eine entscheidende Limitation darstellt.

So sollten zukünftige Arbeiten in Form konkreter Fallstudien oder Erhebungen die wesentlichen Erfolgsfaktoren für einschlägige Architekturen und Vorgehensmodelle weiter beleuchten, um daraufhin allgemeingültige und praxistaugliche Konzepte abzuleiten. Wenngleich funktionale und nicht-funktionale Kriterien zur Sicherstellung einer hohen Modellqualität und Prozessintegrität erarbeitet werden konnten (siehe Abschnitt 1.2.2), muss auch hier weitere Forschung erfolgen. Bei einem Fokus auf eine humanzentrierte Optimierung mitsamt datengetriebener Methoden, welche auf sensible personenbezogene Daten angewendet werden, darf dabei auch der Datenschutz nicht vernachlässigt werden.

3.3 Entwickeltes Scheduling-Framework

In diesem Abschnitt wird das prototypische Scheduling-Framework, welches in den Verbundprojekten mit den Industriepartnern in einer agilen Vorgehensweise entwickelt wurde, in seinen Grundzügen eingeführt. Als Ergänzung zur theoretischen Ausarbeitung wird damit auf der Vorschlagsebene eine grobe Systemarchitektur für eine feldsynchrone Ablaufplanung vorgestellt.

3.3.1 Architektur und Anwendungskontext

Es wurde ein Aufbau gewählt, um eine Simheuristik sowie zugeschaltete ML-Algorithmen mit einem Produktionsleitsystem (z.B. *Enterprise-Resource-Planning-System* oder *Manufacturing-Execution-System*) zu verbinden (siehe z.B. Nahhas, Krist und Turowski 2021). Das Scheduling-Framework kann demnach als ein externes Zusatzmodul für das führende System betrachtet werden. So kann es beispielsweise in Form eines Clouddienstes mit diesem verbunden werden. Es stellt eine Datenschnittstelle bereit, sodass die für die Planung relevanten Quelldaten in das Framework übertragen werden können. Prozessual an eine Materiabedarfsplanung aus dem Leitsystem anknüpfend, ist die Funktionalität auf eine termin- und kapazitätsorientierte Ablaufplanung nach dem zentralen Planungsparadigma beschränkt. So werden in dieser Hinsicht beispielsweise keine Stücklisten aufgelöst, Lagerbestände reserviert oder Bestellungen erzeugt. Es wird hingegen vorausgesetzt, dass die zu verplanenden Produktionsaufträge und deren Abhängigkeiten sowie die vorgesehenen Ressourcen und frühestmöglichen Startzeitpunkte der Arbeitsgänge bereits festgelegt wurden. Die im Framework befindlichen Algorithmen ermöglichen keine umfassende Losgrößenplanung unter Berücksichtigung von Primärbedarfen. Es wird jedoch eine kurzfristige Losgrößenoptimierung ermöglicht, indem sequenzabhängige Rüstzeiten zwischen den Arbeitsgängen minimiert werden können. Darüber hinaus wird eine grundlegende Personaleinsatzplanung ermöglicht, indem Mitarbeiter gemäß ihrer

Anwesenheitszeiträume und Fähigkeiten flexibel zugewiesen werden können. So kann anhand einer Fähigkeitsmatrix abgeleitet werden, welche Mitarbeiter welche Arbeitsgänge und Rüstoperationen umsetzen können. Darüber hinaus kann sich der Grad der Fähigkeit in der Bearbeitungszeit widerspiegeln. Insgesamt stellt das Framework kein fertiges Softwareprodukt dar, welches mit einer breiten Funktionalität eines ausgereiften APS-Systems vergleichbar ist. Es bedient vorrangig den Zweck, die entwickelten Methoden in den realen Geschäftsprozessen unter Berücksichtigung realer Produktionsdaten zu erproben. Aus wissenschaftlicher Perspektive und im Sinne der Zielsetzung dieser Arbeit sollen hiermit der Anwendungsbezug sowie Möglichkeiten der Integration der Algorithmen in eine virtuelle Fabrik aufgezeigt werden.

3.3.2 Workflow der Datenverarbeitung

Abbildung 3.4 zeigt eine Übersicht der grundlegenden Teilschritte der Datenverarbeitung im Framework, welche in den nachfolgenden Unterabschnitten erläutert werden.

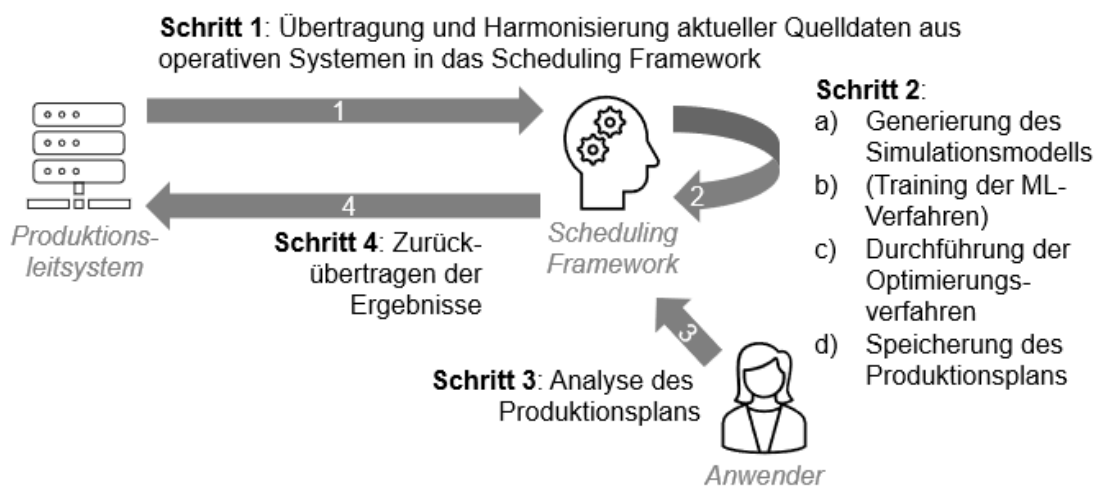


Abbildung 3.4: Informationsfluss des prototypisch implementierten Scheduling-Frameworks. Diese Architektur wurde im EFRE-Projekt *Humanzentriertes Smart Service Lab / Predictive Scheduling* entwickelt.

Übertragung der Quelldaten (Schritt 1)

Im ersten Schritt werden die erforderlichen Stamm- und Bewegungsdaten aus dem Leitsystem in das Datenmodell des Frameworks repliziert. Für die Übertragung wird eine generische Datenschnittstelle für die wesentlichen Entitäten zur Verfügung gestellt, um den Status der Produktionsumgebung möglichst konsistent abzubilden. Dazu gehören u.a. Betriebsmittel- und Mitarbeiterstammdaten, Schichtpläne und Verfügbarkeitsinformationen, Rüstzeit- und Fähigkeitsmatrizen, Artikelstammdaten und Arbeitspläne, zu verplanende Fertigungsaufträge und Arbeitsgänge sowie offene

Materiallieferungen. So ist es mit dem Datenmodell beispielsweise möglich, beliebig komplexe Auftragspfade und Arbeitsgänge abzubilden, sofern sie als gerichteter azyklischer Graph modelliert werden können. Hierdurch können bestimmte Arbeitsgänge erst starten, wenn andere Arbeitsgänge zur Herstellung erforderlicher Werkstoffe abgeschlossen sind. Ebenso werden Materiallieferungen durch externe Lieferanten berücksichtigt, in dem Arbeitsgänge erst ab dem definierten Soll-Ankunftszeitpunkt starten können. Im Anhang *Scheduling-Framework* befindet sich das zentrale relationale Datenmodell, welches die unter anderem genannten Entitäten bereitstellt. Auf diese Weise werden die für die Analyse- und Optimierungsmethoden erforderlichen Eingaben wohlstrukturiert gespeichert. In anderen Worten: Das Datenmodell standardisiert jene Datenstrukturen, die zur Generierung virtueller Fabriken erforderlich sind. Es hat ein Leitsystem-unabhängiges Format, weshalb es mit diversen austauschbaren Quellsystemen kompatibel ist. (Schlecht, Köbler und de Guio 2021) Die Datenübertragung kann zeitbasiert in regelmäßigen Intervallen oder ereignisbasiert durch den Anwender durchgeführt werden. In Zukunft könnten Methoden integriert werden, die situativ eine Übertragung anstoßen, wenn eine Neu- oder Umplanung mithilfe kontextbewusster Algorithmen als notwendig erachtet wird.

Erstellung des Ablaufplans (Schritt 2)

Nachdem die Quelldaten übertragen wurden, werden im zweiten Schritt die Analyse- und Optimierungsmethoden ausgeführt. Das Framework ermöglicht hierbei eine flexible mehrfache Zielsetzung mittels Gewichten. So kann jede erhebbare Kennzahl (z.B. Termintreue oder Zykluszeit) Teil der Zielfunktion bzw. Fitnessberechnung sein. Es ist jedoch erforderlich, dass etwaige bestehende ML-Modelle nach der Änderung der Zielfunktion neu trainiert werden müssen. Des Weiteren erlaubt das Framework die flexible Parallelisierung von Rechenprozessen, was eine freie Skalierbarkeit der Anwendung gewährleistet. Das ereignisdiskrete Simulationsmodell wird datengetrieben anhand der übertragenen Planungsobjekte (Maschinen, Mitarbeiter, Arbeitsgänge usw.) initialisiert und emuliert die Nebenbedingungen der realen Welt so umfassend wie nötig (siehe auch Abschnitt 3.2.1). In einem weiteren Schritt können dann die in dieser Arbeit vorgestellten ML-Modelle mithilfe des Simulationsmodells trainiert werden. So ist es möglich, die Modelle von Grund auf neu zu trainieren (Offline Learning) oder bestehende Modelle nachzutrainieren (Online Learning). Idealerweise sind die ML-Modelle jedoch so gut generalisiert, dass sie nur kurz oder gar nicht nachtrainiert werden müssen. Auf diese Weise können im Kontext robuster Optimierungsmethoden ML-Modelle dazu genutzt werden, rechenintensive MCE durch schnelle Vorhersagen zu ersetzen. In deterministischen Optimierungsverfahren mit Erwartungswerten kann das Simulationsmodell als Evaluationsfunktion in eine Metaheuristik eingebettet werden, um Kennzahlen eines Ablaufplans für die Fitness-

berechnung zu erheben. Zudem kann ein RL-Agent die festgelegten Materialflüsse innerhalb eines Simulationslaufes situativ und gezielt verändern, um noch bessere Reihenfolge- und Belegungsentscheidungen zu erzielen. Der generierte Ablaufplan wird schließlich im Datenmodell des Frameworks gespeichert.

Auswertung und Verwertung der Ergebnisse (Schritte 3 und 4)

Nachdem die Ablaufplanung durch die Algorithmen vorgenommen wurde und die Ergebnisse persistiert wurden, können sie einerseits über eine Benutzerschnittstelle analysiert werden und andererseits zurück in das Leitsystem übertragen werden. Als Benutzerschnittstelle wurde ein Dashboard mit *Microsoft PowerBI*® erstellt, welches die Auswertung der Ablaufpläne aus verschiedenen Perspektiven ermöglicht. Im Anhang *Scheduling-Framework* befinden sich einige exemplarische Screenshots des Dashboards. So ist es erstens möglich, aggregierte Kennzahlen für den Gesamtplan einzusehen, wie z.B. die Gesamtauslastung oder die Pünktlichkeit. Weiterhin ermöglichen verschiedene Detailanalysen einen genaueren Einblick in die terminierten Arbeitsgänge und zugewiesenen Ressourcen, was mehr Transparenz schaffen soll. So können Warnungen und Hinweise über potenzielle Komplikationen abgerufen werden. Dazu gehören zum Beispiel unzureichende Bedarfsdecker, die den termingerechten Start eines konkreten Arbeitsgangs gefährden. Außerdem ist es möglich, Maschinenbelegungen oder Mitarbeiterereinteilungen als Gantt-Diagramm oder Liste auszugeben. Hier können auch Filtereinstellungen gesetzt werden, um z.B. nur bestimmte Zeiträume, Maschinen oder Operationstypen anzuzeigen. Ebenso können zugehörige Auslastungsdiagramme von Maschinen und Mitarbeitern abgerufen werden. Zur Rückübertragung der Pläne in das Leitsystem kann lesend auf das Datenmodell des Frameworks zugegriffen werden. Auf diese Weise können z.B. die Soll-Termine und Soll-Zuweisungen ausgelesen und dann im Leitsystem zur operativen Durchführung des Plans überschrieben werden.

3.4 Schlussbemerkungen

Motivation und Beitrag

Diese Arbeit behandelte die Entwicklung realistischer Ablaufplanungsmodelle und effizienter Optimierungsmethoden zur feldsynchronen Planung dynamischer und humanzentrierter Fertigungsprozesse, welche in der aktuellen wissenschaftlichen Literatur im Sinne eines Gesamtkonzeptes noch unzureichend betrachtet werden (siehe Abschnitt 1.1). Schwerpunktmäßig wurde untersucht, wie ML-Techniken zur datengetriebenen Analyse und Entscheidungsfindung eingesetzt werden können, um die Prozessintegrität und die Modellqualität holistischer und vorausschauender

Dispositionssysteme zu verbessern (siehe Abschnitt 1.2.2). Die dieser Arbeit zugrunde gelegte Leitfrage (siehe Abschnitt 1.3) konnte im Kontext der abgeleiteten vier Teilfragen durch die vorgestellten Publikationen (siehe Kapitel 2) beantwortet werden, wenngleich noch ein beträchtlicher Bedarf an weitere Untersuchungen besteht.

Erkenntnisse

Besonders in Bezug auf die Recheneffizienz eröffnet ML vielseitige Verbesserungspotenziale. So sind Regressionsmodelle geeignet, Robustheitsmetriken vorherzusagen, ohne dass umfangreiche stochastische Experimente durchgeführt werden müssen (siehe Publikation 3). Darauf aufbauend können RL-Agenten robuste und stabile Ablaufpläne in kurzer Rechenzeit generieren, indem sie den kritischen Pfad des Plans gezielt entzerren (siehe Publikation 4). Ferner stoßen rein mathematische Modelle auf die Grenzen der Darstellbarkeit, wenn umfassende Nebenbedingungen realer Produktionsumgebungen abgebildet werden müssen. Aus diesem Grund bieten sich vor allem diskrete Ereignissimulationen an, die generisch und datengetrieben initialisiert werden sowie auch spezifisch angepasst werden können. Im Kontext solch komplexer Umgebungen können RL-Agenten bei der Erzeugung kapazitätsorientierter Ablaufpläne unterstützen (siehe Publikation 1). Durch die Integration eines angemessen trainierten RL-Agenten in eine Simheuristik können gute lokale Optima in weniger Iterationen identifiziert werden. Des Weiteren konnten Zielfunktionen erarbeitet werden, die eine humanzentrierte Optimierung ermöglichen, indem Belastungsspitzen auf eine allgemeingültige Weise geglättet werden können (siehe Publikation 2). Auch hinsichtlich der Datenqualität eröffnen sich einige Potenziale durch den Einsatz von ML-Verfahren. So können realistische Soll-Bearbeitungszeiten für die Ablaufpläne vorhergesagt werden, was wiederum Aufwände in der manuellen Zeitschätzung und Datenpflege reduziert. Die Bereitstellung möglichst genauer Planungsparameter wie Sollzeiten, Unsicherheitsverteilungen und dynamische Ereignisse sind entscheidende Faktoren für eine belastbare Planung.

Implikationen

Indes konnte festgestellt werden, dass die Arbeit aufgrund der Komplexität des Themas einer Reihe an Limitationen unterliegt und somit noch umfassende Forschungs- und Entwicklungsbedarfe bestehen (siehe Abschnitt 3.2). Hierbei konnten vier Fokusfelder abgeleitet werden, die von zukünftigen Arbeiten behandelt werden müssen: (1) Generische Simulationsmodelle, (2) Humanzentrierte Optimierung, (3) Feldsynchrone Ablaufplanung sowie (4) Systementwicklung und Rollout. Obwohl ein prototypisches Scheduling-Framework vorgestellt wurde, welches mit einem Produktionsleitsystem verbunden werden kann (siehe Abschnitt 3.3), müssen generische und flexible

Architekturen und Vorgehensmodelle in einer fundierteren Weise konstruiert und erprobt werden. Ebenso müssen die vorgestellten Analyse- und Optimierungsmethoden weiterentwickelt, vereint und im Kontext umfangreicherer Fertigungsumgebungen evaluiert werden. Hierbei ist es auch entscheidend, reaktive bzw. vorausschauend reaktive Umplanungsprozeduren zu berücksichtigen, um die Ablaufpläne nach schwer abfederbaren Ereignissen effizient zu aktualisieren. Besonders im Hinblick auf ML-Techniken liegt die Herausforderung dabei, möglichst generalisierbare als auch anpassungsfähige Modelle bereitzustellen. Dies bezieht sich vor allem auf RL-Agenten, die innerhalb komplexer Umgebungen mit realistischen Nebenbedingungen und unterschiedlichen Zielsetzungen robuste Ablaufpläne erzeugen. Dazu gehören auch humanzentrierte sowie ökologische Aspekte und Zielfunktionen, die im Kontext von Ablaufplanungsmodellen intensiver beforscht werden sollten.

Abschließend kann beurteilt werden, dass das Thema nicht allumfassend in wenigen Studien behandelt werden kann. Es existiert sowohl ein starker Bezug zur praktischen Anwendung als auch zur Grundlagenforschung im Bereich PPS, Operations Research, Angewandte Informatik und ML. Aus diesem Grund kann in der vorliegenden Mantelschrift kein abgeschlossenes Gesamtkonzept präsentiert werden. Stattdessen werden Stoßrichtungen für ein umfassendes, interdisziplinäres Forschungsprogramm aufgezeigt.

Literaturverzeichnis

Die in der Einleitung und Synthese der Ergebnisse verwendeten Quellen sind im Folgenden aufgelistet. Jene Quellen, die ausschließlich in den Publikationen zitiert wurden, befinden sich im jeweiligen Literaturverzeichnis der Publikation.

- Acker, Isabel Jasmin (2011). „Lösungsverfahren für Job-Shop-Probleme“. In: *Methoden zur mehrstufigen Ablaufplanung in der Halbleiterindustrie*. Gabler, S. 91–162. DOI: 10.1007/978-3-8349-6731-2_5.
- Aggarwal, Charu C. (2018a). *Neural Networks and Deep Learning*. Springer International Publishing, S. 1–27. DOI: 10.1007/978-3-319-94463-0.
- (2018b). *Neural Networks and Deep Learning*. Springer International Publishing, S. 373–416. DOI: 10.1007/978-3-319-94463-0.
- Alaouchiche, Yasmine, Yassine Ouazene und Farouk Yalaoui (2020). „Economic and Energetic Performance Evaluation of Unreliable Production Lines: An Integrated Analytical Approach“. In: *IEEE Access* 8, S. 185330–185345. DOI: 10.1109/access.2020.3029761.
- Ba, L., Y. Li und M. S. Yang (2016). „Modelling and Simulation of a Multi-Resource Flexible Job-Shop Scheduling“. In: *International Journal of Simulation Modelling* 15.1, S. 157–169. DOI: 10.2507/ijssimm15(1)co3.
- Beekmann, Frank und Peter Chamoni (2006). „Verfahren des Data Mining“. In: *Analytische Informationssysteme*. Springer Berlin Heidelberg, S. 263–282. DOI: 10.1007/3-540-33752-0_13.
- Behrendt, Fabian, Niels Schmidtke, Elke Glistau und Margarete Wagner (2019). „Der Intelligente Logistikkaum: Neue Gestaltungsformen im Kontext der digitalen Transformation“. In: *industrie 4.0 Management* 2019.4, S. 35–38. DOI: 10.30844/i40m_19-4_s35-38.
- Ben-Daya, M. und M. Al-Fawzan (1998). „A tabu search approach for the flow shop scheduling problem“. In: *European Journal of Operational Research* 109.1, S. 88–95. DOI: 10.1016/s0377-2217(97)00136-7.
- BMBF (2020). *Industrie 4.0: Innovationen im Zeitalter der Digitalisierung*. Bundesministerium für Bildung und Forschung, S. 7–11.

- Brucker, Peter, Yu N. Sotskov und Frank Werner (2007). „Complexity of shop-scheduling problems with fixed number of jobs: a survey“. In: *Mathematical Methods of Operations Research* 65.3, S. 461–481. DOI: 10.1007/s00186-006-0127-8.
- Cámara, Javier, Javier Troya, Lola Burgueño und Antonio Vallecillo (2023). „On the assessment of generative AI in modeling tasks: an experience report with ChatGPT and UML“. In: *Software and Systems Modeling* 22.3, S. 781–793. DOI: 10.1007/s10270-023-01105-5.
- Colas, Cédric, Tristan Karch, Olivier Sigaud und Pierre-Yves Oudeyer (2022). „Autotelic Agents with Intrinsically Motivated Goal-Conditioned Reinforcement Learning: A Short Survey“. In: *Journal of Artificial Intelligence Research* 74, S. 1159–1199. DOI: 10.1613/jair.1.13554.
- Da Col, Giacomo und Erich C. Teppan (2022). „Industrial-size job shop scheduling with constraint programming“. In: *Operations Research Perspectives* 9, S. 100249. DOI: 10.1016/j.orp.2022.100249.
- Davenport, Andrew J., Christophe Gefflot und J. Christopher Beck (2001). „Slack-based Techniques for Robust Schedules“. In: *Proceedings of the Sixth European Conference on Planning (ECP-2001)*. Toledo, Spain. ISBN: 978-1577356295.
- Destouet, Candice, Houda Tlahig, Belgacem Bettayeb und Bélahcène Mazari (2023). „Flexible job shop scheduling problem under Industry 5.0: A survey on human reintegration, environmental consideration and resilience improvement“. In: *Journal of Manufacturing Systems* 67, S. 155–173. DOI: 10.1016/j.jmsy.2023.01.004.
- Domingos, Pedro (2012). „A few useful things to know about machine learning“. In: *Communications of the ACM* 55.10, S. 78–87. DOI: 10.1145/2347736.2347755.
- Du, Hangming, Fei Qiao, Junkai Wang und Hong Lu (2021). „A Hybrid Metaheuristic Algorithm with Novel Decoding Methods for Flexible Flow Shop Scheduling Considering Human Fatigue“. In: *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. Melbourne, Australia: IEEE. DOI: 10.1109/smc52423.2021.9658692.
- Eriksson, Kristina M., Linnéa Carlsson und Anna Karin Olsson (2022). „To digitalize or not? Navigating and merging human- and technology perspectives in production planning and control“. In: *The International Journal of Advanced Manufacturing Technology* 122.11-12, S. 4365–4373. DOI: 10.1007/s00170-022-09874-x.
- Fachini, Ramon Faganello, Kleber Francisco Esposto und Victor Claudio Bento Camargo (2018). „A framework for development of advanced planning and scheduling (APS) systems in glass container industry“. In: *Journal of Manufacturing Technology Management* 29.3, S. 570–587. DOI: 10.1108/jmtm-06-2017-0126.
- Felten, Edward W., Manav Raj und Robert Seamans (2023). „How will Language Modelers like ChatGPT Affect Occupations and Industries?“ In: *SSRN Electronic Journal*. DOI: 10.2139/ssrn.4375268.

- Figueira, Alvaro und Bruno Vaz (2022). „Survey on Synthetic Data Generation, Evaluation Methods and GANs“. In: *Mathematics* 10.15, S. 2733. DOI: 10.3390/math10152733.
- François-Lavet, Vincent, Peter Henderson, Riashat Islam, Marc G. Bellemare und Joelle Pineau (2018). „An Introduction to Deep Reinforcement Learning“. In: *Foundations and Trends® in Machine Learning* 11.3-4, S. 219–354. DOI: 10.1561/22000000071.
- Gangwar, Neeraj und Nickvash Kani (2022). „Highlighting Named Entities in Input for Auto-Formulation of Optimization Problems“. In: DOI: 10.48550/ARXIV.2212.13201.
- Gerpott, Falk T., Sebastian Lang, Tobias Reggelin, Hartmut Zadek, Poti Chaopaisarn und Sakgasem Ramingwong (2022). „Integration of the A2C Algorithm for Production Scheduling in a Two-Stage Hybrid Flow Shop Environment“. In: *Procedia Computer Science* 200, S. 585–594. DOI: 10.1016/j.procs.2022.01.256.
- Ghaleb, Mageed, Hossein Zolfagharinia und Sharareh Taghipour (Nov. 2020). „Real-time production scheduling in the Industry-4.0 context: Addressing uncertainties in job arrivals and machine breakdowns“. In: *Computers & Operations Research* 123, S. 105031. DOI: 10.1016/j.cor.2020.105031.
- Gil, Chang-Bae und Jee-Hyong Lee (2022). „Deep Reinforcement Learning Approach for Material Scheduling Considering High-Dimensional Environment of Hybrid Flow-Shop Problem“. In: *Applied Sciences* 12.18, S. 9332. DOI: 10.3390/app12189332.
- Goodfellow, Ian, Yoshua Bengio und Aaron Courville (2018). *Deep Learning. Das umfassende Handbuch*. de. 1. Aufl. mitp Professional. Frechen, Germany: MITP, S. 1–12. ISBN: 978-3958457003.
- Grumbach, Felix, Nour Eldin Alaa Badr, Pascal Reusch und Sebastian Trojahn (2023). „A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling“. In: *IEEE Access* 11, S. 68760–68775. DOI: 10.1109/access.2023.3292548.
- Grumbach, Felix, Anna Müller, Pascal Reusch und Sebastian Trojahn (2022). „Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning“. In: *Journal of Intelligent Manufacturing*. DOI: 10.1007/s10845-022-02069-x.
- (2023). „Robustness Prediction in Dynamic Production Processes - A New Surrogate Measure Based on Regression Machine Learning“. In: *Processes* 11.4. DOI: 10.3390/pr11041267.
- Gui, Anderes, Yudi Fernando, Muhammad Shabir Shaharudin, Mazita Mokhtar, I Gusti Made Karmawan und - Suryanto (2021). „Drivers of Cloud Computing Adoption in Small Medium Enterprises of Indonesia Creative Industry“. In: *JOIV* :

- International Journal on Informatics Visualization* 5.1, S. 69–75. DOI: 10.30630/joiv.5.1.461.
- Gupta, Dhruv, Christos T. Maravelias und John M. Wassick (2016). „From rescheduling to online scheduling“. In: *Chemical Engineering Research and Design* 116, S. 83–97. DOI: 10.1016/j.cherd.2016.10.035.
- Harrison, Kyle Robert, Saber M. Elsayed, Ivan L. Garanovich, Terence Weir, Sharon G. Boswell und Ruhul A. Sarker (2022). „Generating datasets for the project portfolio selection and scheduling problem“. In: *Data in Brief* 42, S. 108208. DOI: 10.1016/j.dib.2022.108208.
- Hassani, Rachid, Guy Desaulniers und Issmail Elhallaoui (2020). „Real-time personnel re-scheduling after a minor disruption in the retail industry“. In: *Computers & Operations Research* 120, S. 104952. DOI: 10.1016/j.cor.2020.104952.
- Hausladen, Iris (2020). „Prozesse und Anwendungen einer IT-gestützten Logistik“. In: *IT-gestützte Logistik*. Springer Fachmedien Wiesbaden, S. 95–250. DOI: 10.1007/978-3-658-31260-2_4.
- Herlyn, Wilmljakob Johannes und Hartmut Zadek (2020). „Der Digitale Steuerungs-Zwilling – Dynamische Auftrags- und Materialflusststeuerung auf Basis des Konzepts eines Digitalen Steuerungs-Zwillings“. In: *Zeitschrift für wirtschaftlichen Fabrikbetrieb* 115.s1, S. 70–73. DOI: doi:10.3139/104.112338.
- Hsu, Tzu-Han, Li-Chih Wang und Pei-Chun Chu (2018). „Development of a Cloud-based Advanced Planning and Scheduling System“. In: *Procedia Manufacturing* 17, S. 427–434. DOI: 10.1016/j.promfg.2018.10.066.
- ISO (2005). „IEC 25000 Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—Guide to SQuaRE“. In: *International Organization for Standardization*.
- Jafari, Hamid, Hadi Ghaderi, Mohsin Malik und Ednilson Bernardes (2022). „The effects of supply chain flexibility on customer responsiveness: the moderating role of innovation orientation“. In: *Production Planning & Control*, S. 1–19. DOI: 10.1080/09537287.2022.2028030.
- Juan, Angel A., Javier Faulin, Scott E. Grasman, Markus Rabe und Gonçalo Figueira (2015). „A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems“. In: *Operations Research Perspectives* 2, S. 62–72. DOI: 10.1016/j.orp.2015.03.001.
- Kayhan, Behice Meltem und Gokalp Yildiz (2021). „Reinforcement learning applications to machine scheduling problems: a comprehensive literature review“. In: *Journal of Intelligent Manufacturing* 34.3, S. 905–929. DOI: 10.1007/s10845-021-01847-3.
- Kilger, Christoph, Boris Reuter und Hartmut Stadtler (2014). „Collaborative Planning“. In: *Supply Chain Management and Advanced Planning: Concepts, Models,*

- Software, and Case Studies*. Hrsg. von Hartmut Stadtler, Christoph Kilger und Herbert Meyr. 5. Aufl. Springer Berlin Heidelberg, S. 257–277. DOI: 10.1007/978-3-642-55309-7_14.
- Klemmt, Andreas, Sven Horn, Gerald Weigert und Klaus-Jürgen Wolter (2009). „Simulation-based optimization vs. mathematical programming: A hybrid approach for optimizing scheduling problems“. In: *Robotics and Computer-Integrated Manufacturing* 25.6, S. 917–925. DOI: 10.1016/j.rcim.2009.04.012.
- Kress, Dominik, David Müller und Jenny Nossack (2018). „A worker constrained flexible job shop scheduling problem with sequence-dependent setup times“. In: *OR Spectrum* 41.1, S. 179–217. DOI: 10.1007/s00291-018-0537-z.
- Kulin, Merima, Tarik Kazaz, Eli De Poorter und Ingrid Moerman (2021). „A Survey on Machine Learning-Based Performance Improvement of Wireless Networks: PHY, MAC and Network Layer“. In: *Electronics* 10.3, S. 318. DOI: 10.3390/electronics10030318.
- Kumar, Pramod, Suprakash Gupta und Yuga Raju Gunda (2020). „Estimation of human error rate in underground coal mines through retrospective analysis of mining accident reports and some error reduction strategies“. In: *Safety Science* 123, S. 104555. DOI: 10.1016/j.ssci.2019.104555.
- Lang, Sebastian (2023a). „Eine Methode zum Einsatz von bestärkenden Lernverfahren für die Produktionsablaufplanung“. In: *Methoden des bestärkenden Lernens für die Produktionsablaufplanung*. Springer Fachmedien Wiesbaden, S. 105–195. DOI: 10.1007/978-3-658-41751-2_5.
- (2023b). „Schlussbetrachtung“. In: *Methoden des bestärkenden Lernens für die Produktionsablaufplanung*. Springer Fachmedien Wiesbaden, S. 259–266. DOI: 10.1007/978-3-658-41751-2_7.
- Lavin, Alexander et al. (2022). „Technology readiness levels for machine learning systems“. In: *Nature Communications* 13.1. DOI: 10.1038/s41467-022-33128-9.
- Leon, V. Jorge, S. David Wu und Robert H. Storer (1994). „Robustness Measures and Robust Scheduling for Job Shops“. In: *IIE Transactions* 26.5, S. 32–43. DOI: 10.1080/07408179408966626.
- Li, Mingxing, Ray Y. Zhong, Ting Qu und George Q. Huang (2021). „Spatial-temporal out-of-order execution for advanced planning and scheduling in cyber-physical factories“. In: *Journal of Intelligent Manufacturing* 33.5, S. 1355–1372. DOI: 10.1007/s10845-020-01727-2.
- Liu, Changchun, Xi Xiang, Li Zheng und Jing Ma (2017). „An integrated model for multi-resource constrained scheduling problem considering multi-product and resource-sharing“. In: *International Journal of Production Research* 56.19, S. 6491–6511. DOI: 10.1080/00207543.2017.1363428.

- Lück, Wolfgang (2004a). *Lexikon der Betriebswirtschaft*. 6. Aufl. München: Oldenbourg Wissenschaftsverlag, S. 539–540. DOI: 10.1515/9783486815627.
- (2004b). *Lexikon der Betriebswirtschaft*. 6. Aufl. München: Oldenbourg Wissenschaftsverlag, S. 492. DOI: 10.1515/9783486815627.
- Lugaresi, Giovanni, Marco Zanotti, Diego Tarasconi und Andrea Matta (2019). „Manufacturing Systems Mining: Generation of Real-Time Discrete Event Simulation Models“. In: *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. Bari, Italy: IEEE. DOI: 10.1109/smc.2019.8914025.
- Lupeikiene, Audrone, Gintautas Dzemyda, Ferenc Kiss und Albertas Caplinskas (2014). „Advanced Planning and Scheduling Systems: Modeling and Implementation Challenges“. In: *Informatika* 25.4, S. 581–616. DOI: 10.15388/informatika.2014.31.
- Mahmoodjanloo, Mehdi, Reza Tavakkoli-Moghaddam, Armand Baboli und Ali Bozorgi-Amiri (2020). „Dynamic Distributed Job-Shop Scheduling Problem Consisting of Reconfigurable Machine Tools“. In: *IFIP Advances in Information and Communication Technology*. Springer International Publishing, S. 460–468. DOI: 10.1007/978-3-030-57997-5_53.
- Mesabbah, Mohammed, Waleed Abo-Hamad und Susan McKeever (2019). „A Hybrid Process Mining Framework for Automated Simulation Modelling for Healthcare“. In: *2019 Winter Simulation Conference (WSC)*. National Harbor, Maryland, USA: IEEE. DOI: 10.1109/wsc40007.2019.9004800.
- Mesabbah, Mohammed und Susan McKeever (2018). „Presenting a hybrid processing mining framework for automated simulation model generation“. In: *2018 Winter Simulation Conference (WSC)*. Gothenburg, Sweden: IEEE. DOI: 10.1109/wsc.2018.8632467.
- Meudt, Tobias, Andreas Wonnemann und Joachim Metternich (2017). *Produktionsplanung und -steuerung (PPS) – ein Überblick der Literatur der unterschiedlichen Einteilung von PPS-Konzepten*. <https://tuprints.ulb.tu-darmstadt.de/id/eprint/6654>.
- Mitchell, Thomas (1997). *Machine Learning*. McGraw-Hill series in computer science. New York, NY: McGraw-Hill Professional, S. 2–4. ISBN: 978-0070428072.
- Mjolsness, Eric und Dennis DeCoste (2001). „Machine Learning for Science: State of the Art and Future Prospects“. In: *Science* 293.5537, S. 2051–2055. DOI: 10.1126/science.293.5537.2051.
- Mohan, Jatoth, Krishnanand Lanka und A. Neelakanteswara Rao (2019). „A Review of Dynamic Job Shop Scheduling Techniques“. In: *Procedia Manufacturing* 30, S. 34–39. DOI: 10.1016/j.promfg.2019.02.006.
- Mohan, Jatoth, Krishnanand Lanka, Neelakanteswara A. Rao und Vijaya Kumar Manupati (2022). „Sustainable Flexible Job Shop Scheduling: A Systematic Litera-

- ture Review“. In: *Advances in Manufacturing Processes, Intelligent Methods and Systems in Production Engineering*. Springer International Publishing, S. 227–246. DOI: 10.1007/978-3-030-90532-3_18.
- Monostori, Judit (2018). „Supply chains robustness: Challenges and opportunities“. In: *Procedia CIRP* 67, S. 110–115. DOI: 10.1016/j.procir.2017.12.185.
- Morales, Eduardo F. und Julio H. Zaragoza (2012). „An Introduction to Reinforcement Learning“. In: *Decision Theory Models for Applications in Artificial Intelligence*. IGI Global, S. 63–80. DOI: 10.4018/978-1-60960-165-2.ch004.
- Müller, Anna und Felix Grumbach (2023). „Predicting processing times in high mix low volume job shops“. In: *16th International Doctoral Student Workshop on Logistics, Supply Chain and Production Management*. Magdeburg, Deutschland: Universitätsbibliothek Otto-von-Guericke-Universität. DOI: 10.25673/103491.
- Nahas, Abdulrahman, Marco Krist und Klaus Turowski (2021). „An adaptive scheduling framework for solving multi-objective hybrid flow shop scheduling problems“. In: *Proceedings of the Annual Hawaii International Conference on System Sciences*. Hawaii, USA: Hawaii International Conference on System Sciences. DOI: 10.24251/hicss.2021.199.
- Negri, Elisa, Vibhor Pandhare, Laura Cattaneo, Jaskaran Singh, Marco Macchi und Jay Lee (2020). „Field-synchronized Digital Twin framework for production scheduling with uncertainty“. In: *Journal of Intelligent Manufacturing* 32.4, S. 1207–1228. DOI: 10.1007/s10845-020-01685-9.
- Neto, Anis Assad, Elias Ribeiro da Silva, Fernando Deschamps und Edson Pinheiro de Lima (2021). „Digital twins in manufacturing: An assessment of key features“. In: *Procedia CIRP* 97, S. 178–183. DOI: 10.1016/j.procir.2020.05.222.
- Nicolai, Harald, Martin Schotten und Detlef Much (1999). „Aufgaben“. In: *Produktionsplanung und -steuerung*. Springer Berlin Heidelberg, S. 29–74. DOI: 10.1007/978-3-662-09472-3_3.
- Oluyisola, Olumide Emmanuel, Swapnil Bhalla, Fabio Sgarbossa und Jan Ola Strandhagen (2021). „Designing and developing smart production planning and control systems in the industry 4.0 era: a methodology and case study“. In: *Journal of Intelligent Manufacturing* 33.1, S. 311–332. DOI: 10.1007/s10845-021-01808-w.
- Osman, Ibrahim H. und James P. Kelly (1996). „Meta-Heuristics: An Overview“. In: *Meta-Heuristics*. Springer US, S. 1–21. DOI: 10.1007/978-1-4613-1361-8_1.
- Quintero-Araujo, Carlos L., Angel A. Juan, Jairo R. Montoya-Torres und Andres Munoz-Villamizar (2016). „A simheuristic algorithm for Horizontal Cooperation in urban distribution: Application to a case study in COLOMBIA“. In: *2016 Winter Simulation Conference (WSC)*. Arlington, Virginia, USA: IEEE. DOI: 10.1109/wsc.2016.7822261.

- Ramamonjison, Rindranirina et al. (2023). „NL4Opt Competition: Formulating Optimization Problems Based on Their Natural Language Descriptions“. In: DOI: 10.48550/ARXIV.2303.08233.
- Real Torres, Alejandro del, Doru Stefan Andreiana, Álvaro Ojeda Roldán, Alfonso Hernández Bustos und Luis Enrique Acevedo Galicia (2022). „A Review of Deep Reinforcement Learning Approaches for Smart Manufacturing in Industry 4.0 and 5.0 Framework“. In: *Applied Sciences* 12.23, S. 12377. DOI: 10.3390/app122312377.
- Reddy, N. Sivarami, D.V. Ramamurthy, M. Padma Lalitha und K. Prahlada Rao (2021). „Integrated simultaneous scheduling of machines, automated guided vehicles and tools in multi machine flexible manufacturing system using symbiotic organisms search algorithm“. In: *Journal of Industrial and Production Engineering* 39.4, S. 317–339. DOI: 10.1080/21681015.2021.1991014.
- Rigo, Cezar Antônio, Edemar Morsch Filho, Laio Oriel Seman, Luís Loures und Valderi Reis Quietinho Leithardt (2023). „Instance and Data Generation for the Offline Nanosatellite Task Scheduling Problem“. In: *Data* 8.3, S. 62. DOI: 10.3390/data8030062.
- Robey, Daniel und M. Lynne Markus (1998). „Beyond Rigor and Relevance“. In: *Information Resources Management Journal* 11.1, S. 7–16. DOI: 10.4018/irmj.1998010101.
- Romero-Silva, Rodrigo, Javier Santos und Margarita Hurtado-Hernández (2022). „A conceptual framework of the applicability of production scheduling from a contingency theory approach: addressing the theory-practice gap“. In: *Production Planning & Control*, S. 1–21. DOI: 10.1080/09537287.2022.2076627.
- Russell, Stuart und Peter Norvig (2009). *Artificial intelligence*. 3. Aufl. Upper Saddle River, New Jersey, USA: Pearson, S. 1–33. ISBN: 978-0136042594.
- Sakr, Ahmed H., Ayman Aboelhassan, Soumaya Yacout und Samuel Bassetto (2021). „Simulation and deep reinforcement learning for adaptive dispatching in semiconductor manufacturing systems“. In: *Journal of Intelligent Manufacturing* 34.3, S. 1311–1324. DOI: 10.1007/s10845-021-01851-7.
- Sarker, Iqbal H. (2021). „Machine Learning: Algorithms, Real-World Applications and Research Directions“. In: *SN Computer Science* 2.3. DOI: 10.1007/s42979-021-00592-x.
- Schlecht, Michael, Roland de Guio und Jürgen Köbler (2023). „Automated generation of simulation model in context of industry 4.0“. In: *International Journal of Modelling and Simulation*, S. 1–13. DOI: 10.1080/02286203.2023.2206075.
- Schlecht, Michael, Jürgen Köbler und Roland de Guio (2021). „Flexibles Referenzmodell zur Planung und Optimierung der Produktion – Generierung digitaler Fabrikmodelle mit dem digitalen Zwilling“. In: *Industrie 4.0 Management* 2021.5, S. 53–56. DOI: 10.30844/i40m_21-5_s53-56.

- Schlenkrich, Manuel und Sophie N. Parragh (2023). „Solving large scale industrial production scheduling problems with complex constraints: an overview of the state-of-the-art“. In: *Procedia Computer Science* 217, S. 1028–1037. DOI: 10.1016/j.procs.2022.12.301.
- Schmidtke, Niels, Fabian Behrendt, Lisa Thater und Sascha Meixner (2018). „Technical potentials and challenges within internal logistics 4.0“. In: *2018 4th International Conference on Logistics Operations Management (GOL)*. Le Havre, France: IEEE. DOI: 10.1109/go1.2018.8378072.
- Schmieder, Matthias (2018). „Hidden Champions Benchmarking“. In: *Fallstudienkompendium Hidden Champions: Innovationen für den Weltmarkt*. Hrsg. von Jan-Philipp Büchler. Wiesbaden: Springer Fachmedien Wiesbaden, S. 197–222. ISBN: 978-3-658-17829-1. DOI: 10.1007/978-3-658-17829-1_12.
- Schotten, Martin (1998). „Aachener PPS-Modell“. In: *Produktionsplanung und -steuerung*. Springer Berlin Heidelberg, S. 9–56. DOI: 10.1007/978-3-662-09474-7_2.
- Serrano-Ruiz, Julio C., Josefa Mula und Raúl Poler (2021). „Smart manufacturing scheduling: A literature review“. In: *Journal of Manufacturing Systems* 61, S. 265–287. DOI: 10.1016/j.jmsy.2021.09.011.
- Shu, Jun, Deyu Meng und Zongben Xu (2021). „Learning an Explicit Hyperparameter Prediction Function Conditioned on Tasks“. In: DOI: 10.48550/ARXIV.2107.02378.
- Skinner, Burrhus F (1937). „Two types of conditioned reflex: A reply to Konorski and Miller“. In: *The Journal of General Psychology* 16.1, S. 272–279.
- Stüve, David, Robert van der Meer, Mouhamad Shaker Ali Agha und Matthias Lütke Entrup (2022). „A systematic literature review of modelling approaches and implementation of enabling software for supply chain planning in the food industry“. In: *Production & Manufacturing Research* 10.1, S. 470–493. DOI: 10.1080/21693277.2022.2091057.
- Taillard, Eric (1993). „Benchmarks for basic scheduling problems“. In: *European Journal of Operational Research* 64.2, S. 278–285. DOI: 10.1016/0377-2217(93)90182-m.
- Tao, Fei und Meng Zhang (2017). „Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing“. In: *IEEE Access* 5, S. 20418–20427. DOI: 10.1109/access.2017.2756069.
- Vieira, Jaison, Fernando Deschamps und Pablo Deivid Valle (2021). „Advanced Planning and Scheduling (APS) Systems: A Systematic Literature Review“. In: *Advances in Transdisciplinary Engineering*. IOS Press. DOI: 10.3233/atde210118.
- Vollenkemper, Lukas, Felix Grumbach, Martin Kohlhase und Pascal Reusch (2023). „Humanzentrierte Ablaufplanung von Montagelinien/Human-centered scheduling

- in assembly lines - Plug and play: Efficient algorithms minimize stress in flow shops“. In: *wt Werkstattstechnik online* 113.04, S. 158–163. DOI: 10.37544/1436-4980-2023-04-58.
- Voß, Stefan (2008). „Metaheuristics“. In: *Encyclopedia of Optimization*. Springer US, S. 2061–2075. DOI: 10.1007/978-0-387-74759-0_367.
- Wang, Li-Chih, Chun-Chih Chen, Jen-Li Liu und Pei-Chun Chu (2021). „Framework and deployment of a cloud-based advanced planning and scheduling system“. In: *Robotics and Computer-Integrated Manufacturing* 70, S. 102088. DOI: 10.1016/j.rcim.2020.102088.
- Wang, Yunrui und Zhengli Wu (2020). „Model construction of planning and scheduling system based on digital twin“. In: *The International Journal of Advanced Manufacturing Technology* 109.7-8, S. 2189–2203. DOI: 10.1007/s00170-020-05779-9.
- Wang, Zhen, Qianwang Deng, Like Zhang, Haiqiu Li und Fengyuan Li (2023). „Joint optimization of integrated mixed maintenance and distributed two-stage hybrid flow-shop production for multi-site maintenance requirements“. In: *Expert Systems with Applications* 215, S. 119422. DOI: 10.1016/j.eswa.2022.119422.
- Wilde, Thomas und Thomas Hess (2007). „Forschungsmethoden der Wirtschaftsinformatik“. In: *Wirtschaftsinformatik* 49.4, S. 280–287. DOI: 10.1007/s11576-007-0064-z.
- Wöhe, Günter, Ulrich Döring und Gerrit Brösel (2020). *Einführung in die Allgemeine Betriebswirtschaftslehre*. 27. Aufl. Vahlen, S. 263–338. ISBN: 978-3-8006-6300-2.
- Xu, Zhe, Bo Wu, Aditya Ojha, Daniel Neider und Ufuk Topcu (2021). „Active Finite Reward Automaton Inference and Reinforcement Learning Using Queries and Counterexamples“. In: *Lecture Notes in Computer Science*. Springer International Publishing, S. 115–135. DOI: 10.1007/978-3-030-84060-0_8.
- Yang, Yanfang, Miao Yang, Nabil Anwer, Benoit Eynard, Liang Shu und Jinhua Xiao (2023). „A novel digital twin-assisted prediction approach for optimum rescheduling in high-efficient flexible production workshops“. In: *Computers & Industrial Engineering* 182, S. 109398. DOI: 10.1016/j.cie.2023.109398.
- Yildiz, Emre, Charles Møller und Arne Bilberg (2021). „Demonstration and evaluation of a digital twin-based virtual factory“. In: *The International Journal of Advanced Manufacturing Technology* 114.1-2, S. 185–203. DOI: 10.1007/s00170-021-06825-w.
- Zhang, Jian, Guofu Ding, Yisheng Zou, Shengfeng Qin und Jianlin Fu (2017). „Review of job shop scheduling research and its new perspectives under Industry 4.0“. In: *Journal of Intelligent Manufacturing* 30.4, S. 1809–1830. DOI: 10.1007/s10845-017-1350-2.

Anhang

Liste aller Veröffentlichungen

Vorträge und Präsentationen

- Felix Grumbach und Pascal Reusch (2021): Data-driven generation of digital twin models for predictive-reactive job shop scheduling. OR Konferenz 2021, Bern, Schweiz
- Felix Grumbach und Pascal Reusch (2021): Problem-oriented initialization of discrete simulation models for a socio-technical shopfloor scheduling. IFORS 2021, Seoul, Südkorea
- Pascal Reusch, Felix Grumbach und Johannes Nordmann (2021): Predictive Scheduling - Ein humanzentrierter Ansatz zur Pascal der Produktionssteuerung durch einen KI-basierten Cloudservice. Fachkongress Digitale Innovationen der Fachhochschule Bielefeld, Bielefeld, Deutschland
- Felix Grumbach (2022): Datengetriebene robuste Optimierung flexibler Produktionsprozesse (Posterpräsentation). Forschungskolloquium am Promotionszentrum SGW, Stendal, Deutschland

Begutachtete Zeitschriftenartikel

- Felix Grumbach, Anna Müller, Pascal Reusch und Sebastian Trojahn (2022): Robust-stable scheduling in dynamic flow shops based on deep reinforcement learning. Journal of Intelligent Manufacturing. DOI: 10.1007/s10845-022-02069-x
- Lukas Vollenkemper, Felix Grumbach, Martin Kohlhase und Pascal Reusch (2023): Humanzentrierte Ablaufplanung von Montagelinien. wt Werkstattstechnik online 113.04. S. 158-163. DOI: 10.37544/1436-4980-2023-04-58
- Felix Grumbach, Anna Müller, Pascal Reusch und Sebastian Trojahn (2023): Robustness Prediction in Dynamic Production Processes - A new Surrogate Measure based on Regression Machine Learning. Processes 11.4. DOI: 10.3390/

pr11041267

- Felix Grumbach, Nour E. A. Badr, Pascal Reusch und Sebastian Trojahn (2023): A Memetic Algorithm with Reinforcement Learning for Sociotechnical Production Scheduling. IEEE Access 11. DOI: 10.1109/ACCESS.2023.3292548

Artikel in Kongressbänden

- Anna Müller und Felix Grumbach (2023): Predicting Processing Times in High Mix Low Volume Job Shops. 16th International Doctoral Student Workshop on Logistics, Supply Chain and Production Management, Madgeburg, Deutschland. DOI: 10.25673/103491

Forschungsdaten

Die erstellten Quelltexte und verwendeten Testdaten befinden sich zum Teil in persistenten, online-verfügbaren Repositorien:

Publikation	Material (URL)
1) Grumbach, Badr et al. 2023	doi.org/10.17605/OSF.IO/JRVFC
2) Vollenkemper et al. 2023	<i>nicht öffentlich</i>
3) Grumbach, A. Müller et al. 2023	doi.org/10.17605/OSF.IO/T8EZY
4) Grumbach, A. Müller et al. 2022	doi.org/10.17605/OSF.IO/SXM3Q

Eingesetzte Technologien und Hilfsmittel

Die in dieser Arbeit vorgestellten Publikationen beinhalten empirische Simulationstudien mit numerischen Experimenten. Die bei der Implementierung eingesetzten Softwarebibliotheken befinden sich in der nachfolgenden Tabelle:

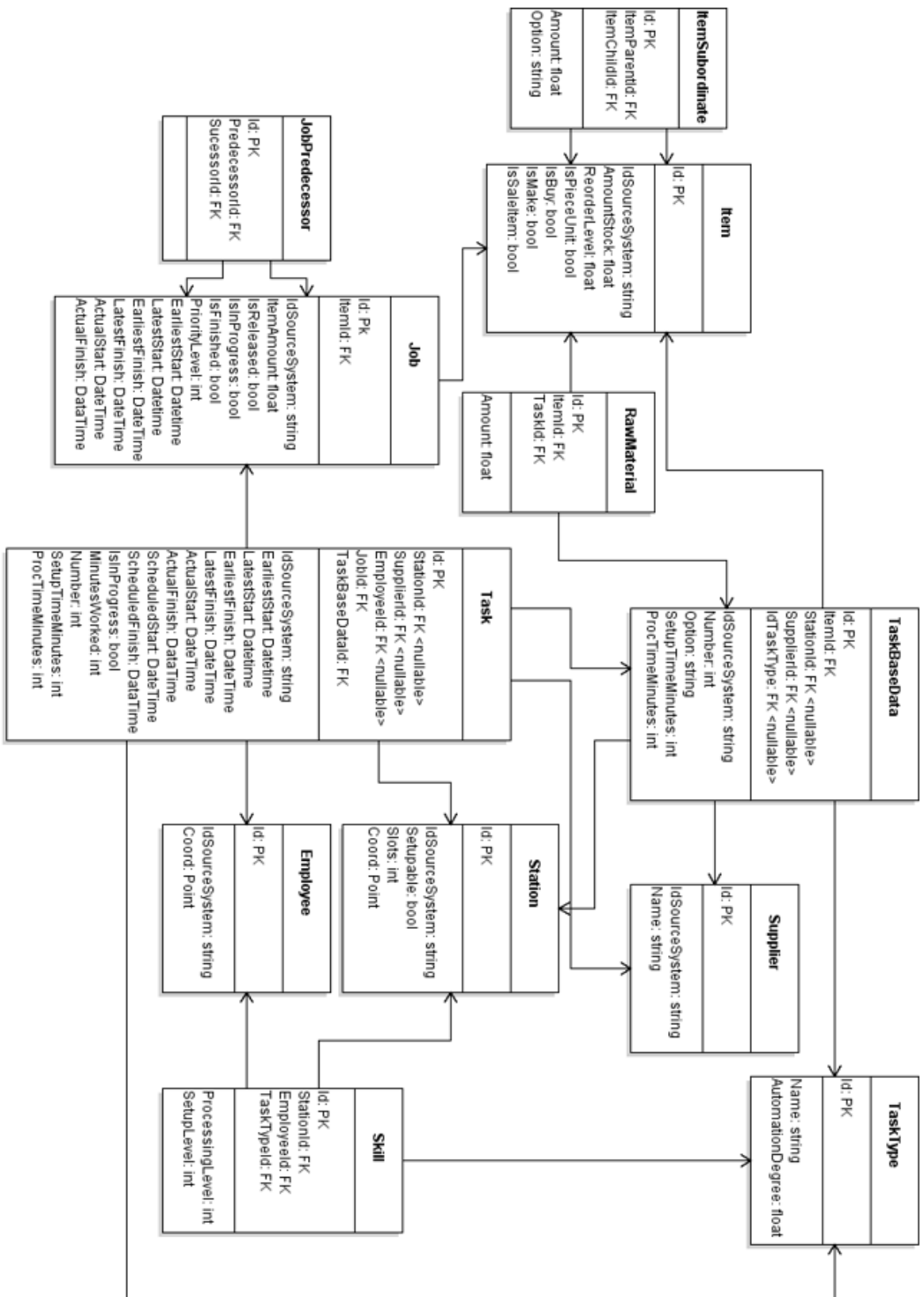
Python-Bibliothek	Verwendung
DEAP v1.3	Implementierung von genetischen Algorithmen
Gym v0.17	Implementierung von MDP für RL-Agenten
IBM CPLEX v12.8	Solver zur Lösung gemischt-ganzzahliger Programme
numpy v1.23	Numerische Berechnungen und Datenmodellierung

numba v0.56	Rechenzeitoptimierung durch Just-in-Time-Kompilierung
Pandas v1.3	Datenmodellierung
PyTorch v1.12	Implementierung von KNN
PuLP v2.4	Implementierung gemischt-ganzzahliger Programme
pymoo v0.6	Implementierung populationsbasierter Metaheuristiken
Python v3.9+	Programmiersprache zur Entwicklung der Softwareprototypen
scikit Learn v1.2	Datenanalyse und Entwicklung von ML-Verfahren
scipy v1.10	Numerische Berechnungen
simpy v4.0	Entwicklung diskreter Ereignissimulationen
Stable-Baselines3 v1.3	Implementierung und Training von RL-Agenten
Visual Studio Code v1.55+	Integrierte Entwicklungsumgebung

Zur Erstellung der Publikationen sowie der Mantelschrift wurden ML-basierte Werkzeuge angewendet. Zur Verbesserung sprachlicher Formulierungen und Ungenauigkeiten (Grammatik-, Rechtschreib-, Kasus- und Tempusfehler) sowie zur Übersetzung wurde ChatGPT 3.5 verwendet. Zudem wurde DeepL (2021-2023) als Übersetzungswerkzeug eingesetzt. Die Werkzeuge wurden lediglich für Lektoratszwecke und nicht zur Generierung neuer Inhalte verwendet.

Scheduling-Framework

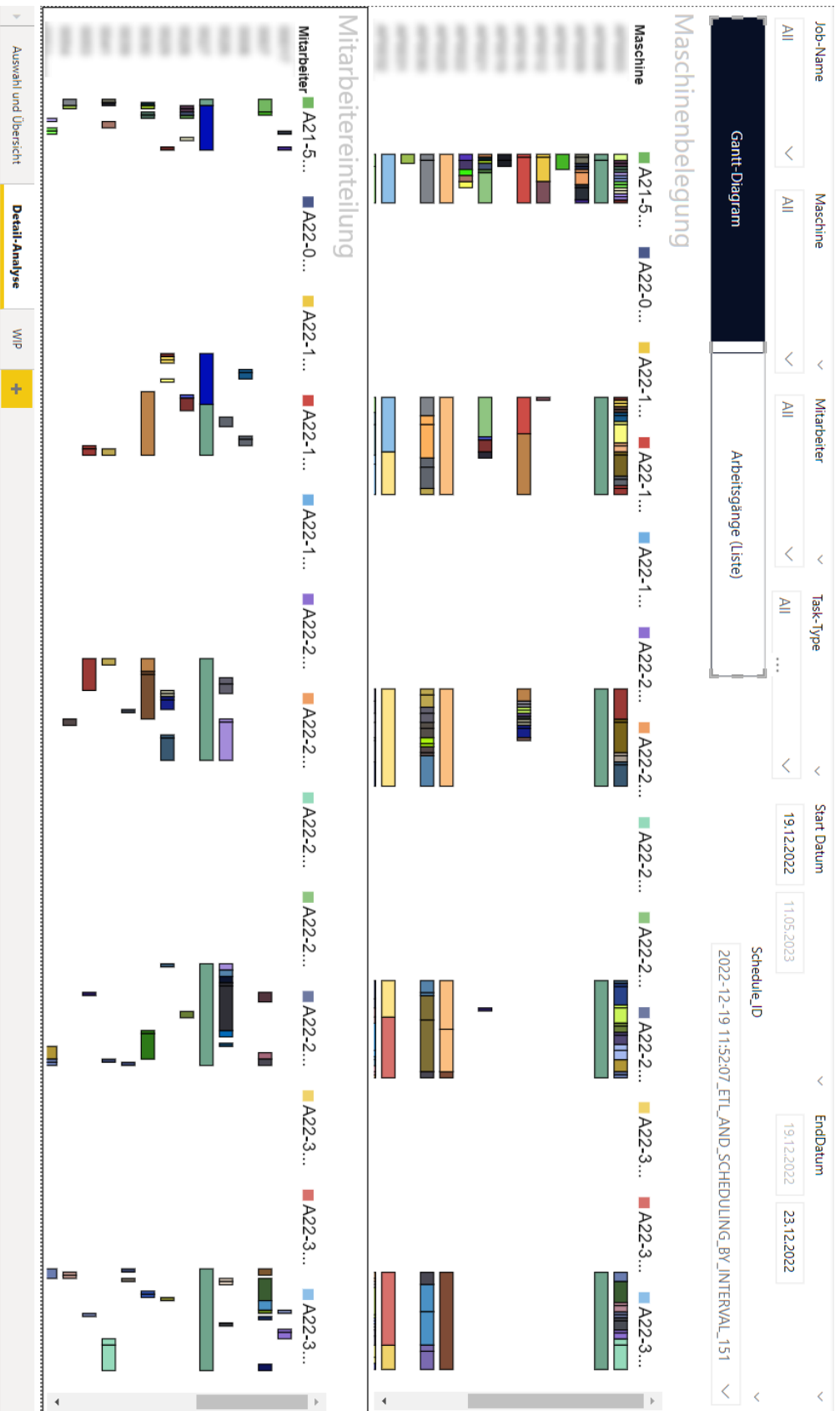
Auf den nachfolgenden Seiten werden Teile des entwickelten Scheduling-Frameworks (Datenmodell und Benutzerschnittstelle) vorgestellt. Das Datenmodell wurde als MySQL-Datenbank implementiert. Die Benutzerschnittstelle wurde mit *Microsoft PowerBI*[®] umgesetzt.



Zentrales relationales Datenmodell des prototypisch implementierten Scheduling-Frameworks für eine Fertigungsprozessplanung mit komplexen Stücklisten, Mitarbeitern, Kompetenzmatrizen, Rüstvorgängen, externen Werkbanken, Arbeitsstationen mit mehreren Arbeitsplätzen und Automatisierungsgraden. Das Datenmodell ist eine Grundlage für eine automatische Generierung und Initialisierung einer prozessbasierten diskreten Ereignissimulation. Unsicherheiten, externe Ereignisse, historische Buchungsdaten und Sequenzabhängigkeiten sind befinden sich in weiteren Tabellen, die in dieser Ansicht nicht vorhanden sind.



Dasboard (*Microsoft PowerBI*®) - Work in Progress (WIP). In dieser Registerkarte kann die Auslastung von Maschinen und Mitarbeitern im Detail (z.B. nach Wochentag oder in tabellarischer Form) sowie unter Berücksichtigung verschiedener Filter (z.B. von-bis, bestimmte Fertigungsaufträge, Operationstyp usw.) analysiert werden.



Dashboard (*Microsoft Power BI*®) - Detail-Analyse. In dieser Registerkarte können die Produktionspläne als Gantt-Diagramm oder als Liste ausgegeben werden. Dabei kann sowohl die Maschinenbelegung als auch die Mitarbeiter-einteilung unter verschiedenen Filtereinstellungen ausgewertet werden. So können z.B. der Auswertungszeitraum spezifiziert oder nur bestimmte Maschinen oder Fertigungsaufträge berücksichtigt werden.

Journal-Rankings

Folgende Metriken wurden am 30. Juni 2023 auf www.scopus.com abgerufen:

Journal of Intelligent Manufacturing

Scopus coverage years: from 1990 to Present

Publisher: Springer Nature

ISSN: 0956-5515 E-ISSN: 1572-8145

Subject area: [Engineering: Industrial and Manufacturing Engineering](#) [Computer Science: Artificial Intelligence](#) [Computer Science: Software](#)

Source type: Journal

CiteScore 2022
17.3

SJR 2022
2.160

SNIP 2022
2.910

IEEE Access

Open Access 

Scopus coverage years: from 2013 to Present

Publisher: IEEE

ISSN: 2169-3536

Subject area: [Engineering: General Engineering](#) [Computer Science: General Computer Science](#) [Materials Science: General Materials Science](#)

Source type: Journal

CiteScore 2022
9.0

SJR 2022
0.926

SNIP 2022
1.422

Processes

Open Access 

Scopus coverage years: from 2013 to Present

Publisher: Multidisciplinary Digital Publishing Institute (MDPI)

E-ISSN: 2227-9717

Subject area: [Chemical Engineering: Chemical Engineering \(miscellaneous\)](#) [Chemical Engineering: Process Chemistry and Technology](#)
[Chemical Engineering: Bioengineering](#)

Source type: Journal

CiteScore 2022
4.7

SJR 2022
0.529

SNIP 2022
0.979

WT Werkstattstechnik

Scopus coverage years: from 2012 to Present

Publisher: VDI Fachmedien GmBbH & Co.

E-ISSN: 1436-4980

Subject area: [Engineering: Automotive Engineering](#) [Engineering: Control and Systems Engineering](#)

Source type: Journal

CiteScore 2022
0.3

SJR 2022
0.151

SNIP 2022
0.171

Eidesstattliche Erklärung

Hiermit erkläre ich, die vorliegende Dissertation selbstständig und ohne unzulässige fremde Hilfe angefertigt zu haben. Ich habe keine anderen als die angeführten Quellen und Hilfsmittel benutzt und sämtliche Textstellen, die wörtlich oder sinngemäß aus veröffentlichten oder unveröffentlichten Schriften entnommen wurden, und alle Angaben, die auf mündlichen Auskünften beruhen, als solche kenntlich gemacht. Ebenfalls sind alle von anderen Personen bereitgestellten Materialien oder erbrachten Dienstleistungen als solche gekennzeichnet. Die Grundsätze guter wissenschaftlicher Praxis wurden eingehalten.

Ort, Datum

Unterschrift Promovend