Matthias Bode

Design eines Multiskalen-Algorithmus zur reaktiven Prozess- und Ablaufplanung

Leibniz Universität Hannover

Design eines Multiskalen-Algorithmus zur reaktiven Prozess- und Ablaufplanung

Von der Fakultät für Bauingenieurwesen und Geodäsie

der Gottfried Wilhelm Leibniz Universität Hannover

zur Erlangung des Grades

Doktor der Ingenieurwissenschaften
(Dr.-Ing)

genehmigte Dissertation von:
Matthias Bode, M.Sc.

2016

Referent: apl. Prof. Dr.-Ing. habil. Volker Berkhahn
Korreferent: Prof. Dr.-Ing. André Borrmann, TU München
Tag der Promotion: 26.02.2016

Abstract

The industries of manufacturing engineering and logistic have seen in recent years a revolution that is being discussed in the trade press under the keyword *Industry 4.0*. Based on new technologies in IT and automation, especially in the machine-to-machine communication, the technical infrastructure is becoming much smarter by techniques of self-tuning, self-configuration and cognition. The technical infrastructure therefore supports people in their increasing complex work tasks significantly better (see Jasperneite (2012)).

By the digitalization *decision support systems* have become more important as they are able to give concrete recommendations based on the current situation. Due to the progressing interconnection between machines it has become possible to better understand the actual conditions and constraints. One of the key areas of application of decision support systems in production and logistics has become the *process planning* and the *scheduling*.

Hence, a framework which is intended to optimize the process planning and the scheduling has to have the capability to consider this new information and to react in an adequate way. For this purpose, a multiscale algorithm for *Flexible Job-Shop-Problems* with execution time windows is being developed in this thesis. An example based on two levels which can be supplemented by intermediate layers is being presented.

On a macroscopic level, which serves as the supervisory authority and the feasibility check, the long-term planning is roughly optimized. The process planning and the scheduling, which are treated widely separated in literature, are considered on macroscopic scale as one unit, optimized by a co-evolutionary algorithm. The coding of the population that describes the scheduling follows the *Least-Commitment Strategy* that allows a flexible description of the schedule. This indirect coding by priority rules is transferred to the process planning, while within an iterative process valid routes for the jobs are getting selected. Furthermore, this form of coding allows a reusability of the existing solutions in a later iteration. In order to generate robust schedules on the macroscopic level, the suitability of a method proposed by Masmoudi u. Haït (2013) based on the *fuzzy set theory* is discussed. On a microscopic level the operations are scheduled optimally, considering the exact constraints by using a *Rolling Horizon algorithm*.

The algorithm is being tested by using a complex application example. In the present case the operation on a container terminal for combined transport is considered and

optimized by exemplary scenarios. For highly resource-constrained problems the method proposed by Masmoudi u. Haït (2013) is proven to be less suitable to consider uncertainties at the macroscopic level. Looking at the entire problem a comparative significant improvement of the examined optimization variables can be achieved in comparison with *Greedy algorithms*, whose results were validated against real data already.

Keywords: multiscale algorithm, integrated process planning and scheduling, reactive scheduling, co-evolutionary algorithm, fuzzy set theory

Zusammenfassung

Die Fertigungstechnik und die Logistik erleben in den letzten Jahren eine Revolution, die unter dem Schlagwort „Industrie 4.0“ in vielen Fachzeitschriften diskutiert wird. Basierend auf neuen Technologien in der IT und der Automatisierungstechnik, vor allem in der Maschine-zu-Maschine-Kommunikation, soll die technische Infrastruktur durch Verfahren der Selbstoptimierung, Selbstkonfiguration, Selbstdiagnose und Kognition intelligenter werden und die Menschen bei ihrer zunehmend komplexeren Arbeit besser unterstützen (siehe Jasperneite (2012)).

Durch die Digitalisierung sind Entscheidungsunterstützungssysteme (englisch: *decision support system*) wichtiger geworden, da sie konkrete Handlungsempfehlungen anhand der momentanen Situation geben können. Durch die voranschreitende Vernetzung ist es möglich geworden, die tatsächlichen Umstände und Nebenbedingungen besser nachzuvollziehen. Ein entscheidendes Einsatzgebiet von Entscheidungsunterstützungssysteme in der Produktion und Logistik sind die Prozess- und die Ablaufplanung.

Ein Framework, das zur Optimierung des Prozess- und Ablaufplans dienen soll, muss folglich die Möglichkeit haben, diese neu gewonnenen Informationen zu berücksichtigen und darauf zu reagieren. Hierzu wird in dieser Arbeit ein Multiskalen-Algorithmus für *Flexible-Job-Shop*-Probleme mit Ausführungszeitfenstern entwickelt. Dabei wird exemplarisch von zwei Ebenen ausgegangen, die durch Zwischenebenen ergänzt werden können.

Auf einer makroskopischen Ebene, die als Kontrollinstanz und zur Machbarkeitsüberprüfung dient, wird die langfristige Planung überschlägig optimiert. Die Prozess- und die Ablaufplanung, die in der Literatur weitgehend einzeln behandelt werden, werden auf makroskopischer Ebene als Einheit betrachtet und mithilfe eines koevolutionären Algorithmus optimiert. Die Codierung der Population, die die Ablaufplanung beschreibt, folgt hierbei der *Least-Commitment*-Strategie, die eine möglichst flexible Beschreibung des Ablaufplans ermöglicht. Diese indirekte Codierung mittels Prioritätsregeln wird auf die Prozessplanung übertragen, wobei innerhalb eines iterativen Verfahrens gültige Routen für die Jobs ausgewählt werden. Weiterhin ermöglicht diese Form der Codierung eine Wiederverwendbarkeit der Lösungen in einem späteren Iterationsschritt. Zur Generierung von robusten Ablaufplänen auf makroskopischer Ebene wird die Eignung eines Verfahrens nach Masmoudi u. Haït (2013), das auf der *Fuzzy-Set*-Theorie basiert, diskutiert. Auf einer mikroskopischen Ebene werden projektbegleitend im Rahmen

eines *Rolling-Horizon*-Algorithmus die Operationen unter Berücksichtigung von exakten Nebenbedingungen optimal eingeplant.

Der Algorithmus wird im Rahmen eines komplexen Anwendungsbeispiels getestet. Dabei wird der Betrieb auf einem Containerterminal des kombinierten Verkehrs betrachtet und anhand exemplarischer Szenarien eine Optimierung vorgenommen. Bei stark ressourcenbeschränkten Problemen stellt sich das Verfahren nach Masmoudi u. Haït (2013) zur Berücksichtigung der Unsicherheiten auf makroskopischer Ebene als weniger geeignet heraus. Bei der Betrachtung des gesamten Problems kann im Vergleich mit *Greedy*-Verfahren, deren Ergebnisse bereits gegen Realdaten validiert wurden, eine signifikante Verbesserung der untersuchten Gütekriterien erreicht werden.

Schlüsselwörter: Multiskalen-Algorithmus, Integrierte Prozess- und Ablaufplanung, reaktive Ablaufplanung, Co-evolutionäre Algorithmen, Fuzzy-Set-Theorie.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

1. Einleitung

Eines der größten Gebiete der heutigen Problemstellungen in Wirtschaft und Wissenschaft ist das Gebiet der Optimierung. Die Optimierung umfasst dabei das Verbessern von bestimmten Abläufen, Prozessen oder Verfahren. Das spiegelt sich auch in den Forschungsarbeiten der Naturwissenschaften, der Ingenieurwissenschaften und der Wirtschaftswissenschaften wider. Mathematisch gesehen sucht man die beste Lösung eines Optimierungsproblems. Um zu entscheiden, welche Lösung als beste angesehen werden kann, müssen die einzelnen Kandidaten vergleichbar sein und eine klare Definition von Güte haben.

Im Rahmen der Produktion und Logistik sind die Prozess- und Ablaufplanung von essentieller Bedeutung. Bei der Prozessplanung wird für einen Prozess die Sequenz von Operationen bestimmt, die für seine Fertigstellung nötig sind. Die Ablaufplanung beschäftigt sich mit der Fragestellung, zu welchem Zeitpunkt die zuvor in der Prozessplanung bestimmten Operationen ausgeführt werden sollen. Zumeist werden in der Literatur beide Fragestellungen separat behandelt. Für die gesamtheitliche Optimierung ist es allerdings wichtig, die Wechselwirkungen zwischen beiden Teilbereichen zu berücksichtigen.

Erschwerend kommt bei realen Problemen hinzu, dass diese Unsicherheiten ausgesetzt sind. Zumeist werden Aufwandswerte oder Operationsdauer lediglich geschätzt, wenn mehr Informationen zur Verfügung stehen, werden mittlere Operationsdauern verwendet. Die Problematik besteht darin, eine Prozessplanung sowie eine Ablaufplanung zu erstellen, die trotz dieser Unsicherheiten verwendbar sind.

1.1. Stand der Wissenschaft und Technik

Ablaufplanprobleme sind seit Jahrzehnten Bestandteil der Forschung in verschiedensten Wissenschaftsbereichen. Zur Optimierung von Ablaufplänen werden zumeist Heuristiken eingesetzt. Die Startzeiten der einzuplanenden Operationen werden hierbei nicht direkt als Vektor repräsentiert, da der Lösungsraum dadurch extrem groß werden würde und es schwierig ist, die Gültigkeit der Lösungen zu gewährleisten. Deshalb werden *Schedule-Generation-Scheme* (SGS) eingesetzt, die als Eingabe eine permutationscodierte Aktivitätsliste benötigen, die vorgibt, in welcher Reihenfolge versucht wird,

die Operationen einzuplanen. Alternativ dazu setzte beispielsweise Ozdamar (1999) Prioritätsregeln ein, die die Operationsreihenfolge indirekt vorgeben.

Der SGS wird zur Lösung des Optimierungsproblems zumeist in eine Metaheuristik wie beispielsweise einem Genetischen Algorithmus (Mattfeld (1996), Ozdamar (1999), Lin u. Chong (2015)), Simmulated Annealing (Osman u. Potts (1989), Van Laarhoven u. a. (1992)) oder der Tabu-Search (Dell'Amico u. Trubian (1993), Nowicki u. Smutnicki (1996)) eingebettet.

Um eine gewisse Flexibilität bei der Ablaufplanung zu ermöglichen, wird in vielen Veröffentlichungen ein *Multi-Mode-Resource-Constrained-Project-Scheduling-Problem* (MM-RCPSP) betrachtet, bei dem eine Operation in verschiedenen Moden ausgeführt werden kann (vgl. Brucker u. Neyer (1998), Mori u. Tseng (1997), Alcaraz u. a. (2003)). Erst im letzten Jahrzehnt wurden Verfahren vorgestellt, die auf einer kombinierten Betrachtung von Prozess- und Ablaufplanung basieren. So zeigte Qassim (2012) das Problem am Beispiel der Schifffertigung und bezeichnet es als *Integrated-Process-Planning-and-Scheduling-Problem* (IPPSP). Shao u. a. (2009) stellten einen modifizierten genetischen Algorithmus vor, um Probleme dieser Art zu lösen. Maravelias u. Sung (2009) diskutierten weitere Ansätze für die kombinierte Betrachtung der Produkt- und der Ablaufplanung. Sie konzentrieren sich hierbei auf die Planung der Bereitstellung von Ressourcen im Rahmen des *Supply-Chain-Management*. Auf einer übergeordneten Ebene wird über die Produktion von Rohmaterial unter Berücksichtigung von Lagerkosten entschieden, wohingegen auf niedrigster Ebene die Verteilung des Endproduktes im Vordergrund steht.

Die meisten realen Projekte und somit deren Prozess- und Ablaufplanung unterliegen vielen äußeren Einflüssen. Bedingt durch diese Einflüsse können Änderungen der Nebenbedingungen auftreten und die ursprünglich bestimmten Lösungen ungültig werden. Eine der Schlüsselanforderungen an ein Optimierungsverfahren ist somit die Berücksichtigung dieser Unsicherheiten in den Nebenbedingungen. Li u. Ierapetritou (2008) und Herroelen u. Leus (2005) bieten einen ausführlichen Überblick über die eingesetzten Verfahren. Grundsätzlich unterscheiden sie in *reaktive Ablaufplanung*, *stochastische Ablaufplanung*, *robuste Ablaufplanung* und Methoden der *Fuzzy-Set*-Theorie bei der Ablaufplanung.

Im Bereich der reaktiven Ablaufplanung verwendeten Li u. Ierapetritou (2010) und Ovacik u. Uzsoy (1995) einen *Rolling-Horizon*-Algorithmus, der die Generierung eines Ablaufplans unter Berücksichtigung möglicher zukünftiger Ereignisse innerhalb eines kleinen Zeitfensters vom aktuellen Betrachtungspunkt ausgehend bestimmt. Dieses Zeitfenster wird projektbegleitend iterativ verschoben. Stefansson u. a. (2006) entwickelteN, von dieser Kernidee ausgehend, einen Multiskalen-Algorithmus für die Ablaufplanung in der pharmazeutischen Industrie. Sie führen drei Hierarchieebenen ein, wobei auf der obersten Ebene eine Langzeitplanung vorgenommen wird, die die Bereitstellung von Ressourcen für etwaige Bestellungen vornimmt. Auf der mittleren Ebene sind konkrete Bestellungen eingegangen und der auf oberster Ebene entwickelte Ablaufplan wird

an die geänderten Bedingungen angepasst. Auf der untersten Ebene wird eine genaue Optimierung der Produktionsschritte vorgenommen. Die gesamte Planung auf den einzelnen Ebenen wird im Rahmen eines *Rolling-Horizon*-Verfahrens vorgenommen.

Methoden der Stochastik können nur konsequent angewandt werden, wenn die Eingangsgrößen zufälliger Natur sind und genügend Informationen zur Modellierung der Wahrscheinlichkeitsdichtefunktion vorhanden sind. Daher sind in den letzten Jahrzehnten nicht-probabilistische Methoden, wie die Intervallanalyse, die Möglichkeitstheorie sowie die Fuzzy-Set-Theorie entwickelt worden. Fortemps (1997) entwickelte grundlegende Ideen zum Einsatz der *Fuzzy-Set*-Theorie bei der Ablaufplanung und optimierte einen Ablaufplan mittels *Simulated Annealing* in Hinblick auf dessen unscharfe Projektdauer. Dubois u. a. (2003) erweiterten diese Ansätze um die Möglichkeit, flexible Nebenbedingungen bei der Ablaufplanung zu berücksichtigen, bei denen Präferenzen abgebildet werden, beispielsweise ein wünschenswerter Fälligkeitstermin. Hierbei optimieren sie den Grad der Erfüllung dieser Präferenzen. Hapke u. a. (1999) entwickelten einen paralleln SGS, der unsichere Verfügbarkeitszeitpunkte und Dauern von Operationen berücksichtigt. Allerdings wird dabei die Auslastung innerhalb der Ressourcen deterministisch angenommen. Masmoudi u. Haït (2013) entwickelten eine auf die Possibilitätstheorie gestützte Modellierung der Auslastung und erhalten unscharfe Auslastungsprofile, die sie im Rahmen eines Genetischen Algorithmus glätten.

Bei der kombinierten Betrachtung der Prozess- und der Ablaufplanung handelt es sich um eine komplexe Problemstellung, die sich zwar in Teilprobleme aufteilen lässt, deren Wechselwirkungen allerdings nicht vernachlässigt werden dürfen. Bei vergleichbaren Fragestellungen zeigte sich, dass der Einsatz von Koevolutionären Algorithmen vielversprechend ist. Für die jeweiligen Teilprobleme werden einzelne Teil- oder Unterpopulationen erzeugt, auf denen eigenständige Genetische Algorithmen angewandt werden. Zumeist wird lediglich bei der Bestimmung der Fitness die Interaktion zwischen den Teilproblemen berücksichtigt. Grundlegende Arbeiten wie die von Potter u. De Jong (1994) verwendeten Koevolutionäre Algorithmen zur Optimierung von Funktionen mehrerer Veränderlichen. Für jede Variable einer zu untersuchenden Funktion erzeugen sie eine Subpopulation und wenden einen Genetischen Algorithmus auf diese an. Cai u. Peng (2002) verwendeten Koevoultionäre Algorithmen zur Pfad-Planung von mobilen Robotern. Die Roboter transportieren Güter zwischen verschiedenen ausgewählten Punkten im Untersuchungsgebiet. Dabei agieren die einzelnen Roboter als Agenten im Rahmen eines Agentensystems und suchen sich zunächst ihren besten Weg. Im Rahmen der Problemstellung sollen möglichst wenige dieser Punkte von verschiedenen Robotern mehrmals angefahren werden. Um die Bewegungen abzustimmen, wenden sie einen Koevolutionären Algorithmus an, wobei die Bewegungen jedes einzelnen Agenten einer eigenen Subpopulation entsprechen. Ladjici u. a. (2014) setzten Koevolutionäre Algorithmen zur Bestimmung von Strategien für Lieferanten im Strommarkt ein. Dabei wird die Interaktion zwischen Agenten im Spot- und im Terminmarkt optimiert.

1.2. Zielsetzung

Gerade im Bauingenieurwesen lassen sich aufgrund der vielen äußeren Faktoren und Einflüsse Problemstellungen finden, die großen Unsicherheiten unterliegen, beispielsweise die Bauablaufplanung. Die betrachteten Projekte erstrecken sich häufig über lange Zeiträume und erfordern somit eine langfristige Planung. Kurzfristig müssen weiterhin konkrete Ablaufplanungen auf detaillierter Ebene vorgenommen werden, wie beispielsweise die Koordination der zur Verfügung stehenden Ressourcen vor Ort.

Ebenso ist eine Planung auf mehreren Betrachtungsebenen auch aus Sicht einzelner Unternehmen wichtig, die mehrere Projekte bearbeiten. Eine Baufirma muss langfristig ihren Ressourcenbedarf planen und kurzfristig Arbeitskolonnen zu den einzelnen Baustellen entsenden und deren konkrete Arbeitsschritte koordinieren. Ähnliche Problematiken sind aus diversen anderen Bereichen bekannt, beispielsweise bei der Optimierung des Betriebs eines Hochlagers oder bei der Optimierung von Lieferketten.

Um langfristige Planungen vorzunehmen, wird ein robustes Mittel benötigt, welches überschlägig Aussagen über die Machbarkeit, Auslastungen und den Ressourcenbedarf ermöglicht. Projektbegleitend ist es wünschenswert, auf Basis der langfristigen Planung die konkrete Umsetzung innerhalb eines kleinen Zeitfensters vom momentanen Zeitpunkt aus zu optimieren. Um diesen Anforderungen gerecht zu werden, wird im Rahmen dieser Arbeit ein Multiskalen-Algorithmus entwickelt, der das Problem der Prozess- und Ablaufplanung als Einheit betrachtet. Begleitend zum betrachteten Problem wird die langfristige Planung an die aktuellen Zustände angepasst.

Auf makroskopischer Ebene wird eine robuste Ablauf- und Prozessplanung durchgeführt. Im Rahmen eines *Rolling-Horizon*-Algorithmus wird für kleinere Zeitfenster auf mikroskopischer Ebene eine exakte Ablaufplanung durchgeführt. Zur Optimierung wird ein Koevolutionärer genetischer Algorithmus auf makroskopischer Ebene eingesetzt, um die Prozess- und Ablaufplanung zu koppeln. Um bereits gefundene gute Lösungen zu einem späteren Zeitpunkt des Projekts, an dem sich Nebenbedingungen geändert haben könnten, als vielversprechende Startlösung verwenden zu können, wird eine prioritätsregelbasierte Codierung verwendet, die für die Entscheidungspunkte Prioritätsregeln festlegt. Diese in der Ablaufplanung erprobte Codierung wird auf die Prozessplanung übertragen.

Die von Masmoudi u. Haït (2013) entwickelte Auslastungsbeschreibung auf Basis der *Fuzzy-Set*-Theorie wird weiterentwickelt und deren Eignung zur makroskopischen Optimierung im Rahmen des Multiskalen-Algorithmus diskutiert.

Anhand eines komplexen Anwendungsbeispiels wird der Multiskalen-Algorithmus getestet. Hierbei wird der Betrieb auf einem Containerterminal betrachtet.

2. Grundlagen

Im folgenden Abschnitt werden die Grundlagen, die zur Umsetzung des Multiskalen-Algorithmus benötigt werden, vorgestellt. Dazu zählen zunächst die Grundbegriffe der Ablaufplanung. Zur Lösung von Optimierungsproblemen bei der Ablaufplanung lassen sich Genetische Algorithmen verwenden, deren grundsätzlicher Aufbau erklärt wird. Zur Beschreibung von Unsicherheiten werden grundlegende Verfahren und deren Anwendung auf Ablaufplanprobleme diskutiert.

2.1. Ablaufplanung

Unter Zeitplanerstellung oder Ablaufplanerstellung versteht man das Erzeugen eines Ablaufplanes, der Operationen ausführende Ressourcen und eine Startzeit zuordnet. Häufig wird auch im deutschen Sprachgebraucht der Begriff *Scheduling* verwendet. Im Folgenden wird auf die einzelnen Varianten von Ablaufplan-Problemen eingegangen und diese mathematisch formuliert. Mathematische Strukturen, die zur Beschreibung des Problems benötigt werden, werden definiert.

2.1.1. Optimierungsprobleme

In der Mathematik versteht man unter dem Begriff Optimierung das Bestimmen optimaler Parameter eines komplexen Systems aus den Naturwissenschaften, den Ingenieurwissenschaften oder den Wirtschaftswissenschaften. Das mathematische Modell des Systems beschreibt eine Menge von zulässigen Lösungen des Problems aus denen diejenige Lösung gesucht wird, die zuvor bestimmte Kriterien bestmöglich erfüllt.

Definition 1: Optimierungsproblem

Im Rahmen eines Optimierungsproblems wird unter der Menge von möglichen Lösungen Ω auf Basis einer Zielfunktion $f : \Omega \to \mathbb{R}$ diejenige Lösung $x \in \Omega$ gesucht, deren Bewertung durch die Zielfunktion $f(x)$ einen möglichst großen oder aber einen möglichst kleinen Wert annimmt.

Ein solches Optimierungsproblem unterliegt in der Realität zumeist Nebenbedingungen, die die Werte der Parameter beschränken.

2.1.2. Zeit- und ressourcenbeschränkte *Scheduling*-Probleme

Ein *Scheduling*-Problem besteht zunächst aus einer Menge von Operationen $V := \{0, 1, .., n+1\}$, wobei Operation 0 und Operation $n+1$ Dummy-Operationen ohne Dauer sind, die den Beginn bzw. das Ende des Projektes beschreiben. Die einzelnen Operationen haben eine Ausführungsdauer, die durch p_i beschrieben wird. Die durch die Planung zu bestimmende Startzeit der Operationen i wird mit S_i bezeichnet.

Zwischen den Operationen kann es zeitliche Restriktionen geben, beispielsweise einen minimalen zeitlichen Abstand zwischen Operation i und der darauf folgenden Operation j, der durch d_{ij}^{min} beschrieben wird. Es sind auch andere zeitliche Restriktionen möglich, wie beispielsweise eine maximale Pause zwischen zwei Aktivitäten. Diese können durch den Parameter d_{ij}^{max} festgelegt werden. Eine zeitliche Restriktion kann als geordnetes Paar $\langle i, j \rangle \in E$ aufgefasst werden, wobei E die Menge aller Beziehungen zwischen zwei Operationen umfasst.

Definition 2: Scheduling-Problem

Ein Ablaufplan ist als eine Sequenz von Startzeiten $S = (S_0, S_1, ..., S_{n+1})$ definiert, die jeder Operation i, auch Aktivitäten genannt, eine Startzeit S_i zuweist. So kann das Problem, einen Ablaufplan zu finden, der bezüglich einer Zielfunktion $f(S) \to \mathbb{R}$ optimal ist, als *Lineares Programm* formuliert werden:

$$\min f(S) \tag{2.1}$$

unter den Nebenbedingungen

$$\text{(Zeitliche Beziehungen)} \quad S_j - S_i \geq \delta_{ij} \quad (\langle i, j \rangle \in E) \tag{2.2}$$

$$\text{(Nichtnegativität)} \quad S_i \geq 0 \quad (i \in V) \tag{2.3}$$

$$\text{(Startzeitpunkt)} \quad S_0 = 0 \quad (S_0 \in V) \tag{2.4}$$

Innerhalb eines *Scheduling*-Problems sucht man einen zulässigen Plan, der durch Startzeiten S_i charakterisiert ist. Ein Plan heißt dabei zulässig, wenn alle Vorgängerbeziehungen eingehalten werden und, falls der Bedarf von Ressourcen berücksichtigt wird, zu jedem Zeitpunkt genügend Ressourcen zur Ausführung der eingeplanten Operationen zur Verfügung stehen. Im Rahmen der Optimierung sucht man einen Plan, der bezüglich einer

bestimmten Zielfunktion, beispielsweise der Minimierung der Gesamtdauer, optimal ist.

Die Restriktionen geben vor, dass die erste Operation zum Zeitpunkt 0 stattfindet ($S_0 = 0$) und alle Operationen nach Projektbeginn eingeplant sein müssen ($S_i \geq 0$). Des Weiteren müssen zeitliche Restriktionen δ_{ij} zwischen zwei Aktivitäten i und j eingehalten werden, falls diese vorgeschrieben sind.

Die zeitlichen Beziehungen zwischen den einzelnen Operationen (siehe 2.2) lassen sich mittels eines *Activity-On-Node*-Diagramms darstellen.

Definition 3: Activity-On-Node-Diagramm

Ein *Activity-On-Node*-Diagramm ist ein gerichteter gewichteter Graph $G := (V, E)$. Die einzelnen Aktivitäten beschreiben dabei die Knotenmenge V des Graphen, die Menge der Kanten E ergibt sich durch ggf. festgesetzte zeitliche Restriktionen. Ist ein minimaler zeitlicher Abstand zwischen den Aktivitäten i und j gegeben, so wird eine Kante $\langle i, j \rangle \in E$ mit dem Kantengewicht $\delta_{ij} = d_{ij}^{min}$ eingefügt. Ist eine maximale Verzögerung d_{ij}^{max} gegeben, so wird eine Rückwärtskante (j, i) dem Graphen hinzugefügt, deren Kantengewicht $\delta_{ij} = -d_{ij}^{max}$ entspricht.

Über die Kantengewichte lassen sich verschiedene Folgen von Operationen modellieren. Die Normalfolge, auch Ende-Anfang-Beziehung genannt, beschreibt dabei, dass eine Operation B erst dann begonnen werden kann, wenn eine Operation A beendet ist. Somit wäre das Kantengewicht zwischen den beiden Operationen so zu wählen, dass es der Dauer der Operation A entspricht. Ähnlich lassen sich Anfangsfolgen (Anfang-Anfang-Beziehungen) oder Endfolgen (Ende-Ende-Beziehungen) abbilden.

Viele Anwendungen erforden neben Zeit-Beschränkungen zusätzliche Ressourcen-Beschränkungen. Diese Ressourcen können sowohl erneuerbar als auch nicht erneuerbar sein. Beschreibt $\mathcal{R}$ die Menge von erneuerbaren Ressourcen in einem Projekt, so wird definiert, dass $R_\kappa \in \mathbb{N}$ die Kapazität der Ressource $\kappa \in \mathcal{R}$ ist und $r_{i\kappa} \in \mathbb{Z}_{\geq 0}$ der Ressourcenbedarf der Aktivität i an der Ressource κ.

Der Bedarf an Ressource κ zum Zeitpunkt t lässt sich mittels des Active Sets definieren.

Definition 4: Active Set

Bei einem gegebenen Ablaufplan S bilden die Aktivitäten, die zum Zeitpunkt t ausgeführt werden, das Active Set:

$$\mathcal{A}(S,t) = \{i \in V | S_i \leq t < S_i + p\}(t \geq 0) \tag{2.5}$$

Der eigentliche Bedarf an der Ressource κ zum Zeitpunkt t ergibt sich schließlich zu:

$$r_\kappa(S,t) = \sum_{i \in \mathcal{A}(S,t)} r_{i\kappa}(\kappa \in \mathcal{R}, t \geq 0) \tag{2.6}$$

Definition 5: Ressourcenbeschränktes Scheduling-Problem

Ein ressourcenbeschränktes *Scheduling*-Problem erweitert das zeitbeschränkte *Scheduling*-Problem um eine Restriktion, die besagt, dass für alle Ressourcen zu jedem Zeitpunkt t, wobei t zwischen 0 und $\bar{d}$, der Maximaldauer des Projektes, liegt, die Kapazität nicht überschritten werden darf.

Es ergibt sich das Optimierungsproblem

$$\min f(S) \tag{2.7}$$

unter den Nebenbedingungen

(Zeitliche Beziehungen)	$S_j - S_i$	$\geq \delta_{ij}$	$(\langle i,j \rangle \in E)$	(2.8)
(Nichtnegativität)	S_i	≥ 0	$(i \in V)$	(2.9)
(Startzeitpunkt)	S_0	$= 0$	$(S_0 \in V)$	(2.10)
(Kapazitätsbeschränkung)	$r_\kappa(S,t)$	$\leq R_\kappa$	$(\kappa \in \mathcal{R}, 0 \leq t \leq \bar{d})$	(2.11)

Im englischen Sprachgebrauch spricht man hierbei oft von einem *resource constrained project scheduling problem*, kurz *RCPSP*.

2.1.3. Multi-Mode-Ressource-Constrained-Project-Scheduling-Probleme

Als Erweiterung zum ressourcenbeschränkten *Scheduling*-Problem lässt sich ein *Multi-Mode-Ressource-Constrained-Project-Scheduling-Problem* (MMRCPSP) formulieren, bei dem jede Operation $i \in V$ in verschiedenen Moden M_i ausgeführt werden kann. Diese Moden können unterschiedliche Ressourcenanforderungen und unterschiedliche Dauern haben. Ein MMRCPSP ist dekomponierbar in ein Zuweisungsproblem (englisch: *mode assignment problem*) und in ein einfaches *ressourcenbeschränktes Scheduling-Problem*.

Definition 6: Mode-Assignment-Problem

$$\text{finde eine Zuordnung } \vec{x} = (x_{im_i}) \tag{2.12}$$

unter der Nebenbedingung

$$\text{(Exlusivitätsbedingung)} \quad \sum_{m_i \in M_i} x_{im_i} = 1 \qquad (i \in V) \tag{2.13}$$

$$\text{(Auswahlbedingung)} \quad x_{im} \in \{0,1\} \qquad (i \in V, m_i \in M_i) \tag{2.14}$$

Beschränkt man sich wie zuvor nur auf erneuerbare Ressourcen, ergibt sich der Bedarf an die Ressource k zum Zeitpunkt t zu:

$$r_\kappa(S,t,\vec{x}) = \sum_{i \in \mathcal{A}(S,t,x)} r_{i\kappa}(\vec{x}) \qquad (\kappa \in \mathcal{R}, t \geq 0), \tag{2.15}$$

wobei der Ressourcenbedarf $r_{i\kappa}$ der Operation i an der Ressource κ eine Funktion der gewählten Moden $\vec{x}$ der einzelnen Operationen ist. Es ergibt sich:

$$r_{i\kappa}(\vec{x}) = \sum_{m_i \in M_i} r_{i\kappa m_i} \cdot x_{im_i} \tag{2.16}$$

Das *Multi-Mode-Resource-Constrained-Scheduling*-Problem lässt sich wie folgt definieren:

Definition 7: Multi-Mode-Resource-Constrained-Scheduling-Problem (MMRCSP)

Ein *Multi-Mode-Resource-Constrained-Scheduling*-Problem ist ein ressourcenbeschränktes *Scheduling*-Problem, bei dem für jede Operation verschiedene Moden zur Verfügung stehen. Es wird ein Ablaufplan gesucht, der neben der Startzeit für jede Operation einen Modus festlegt und bezüglich der Zielfunktion $f(S) \to \mathbb{R}$ optimal ist.

$$\min f(S) \tag{2.17}$$

unter den Nebenbedingungen

$$\begin{aligned}
&\text{(Zeitliche Bedingungen)} && S_j - S_i && \geq \delta_{ij}(\vec{x}) && (\langle i,j\rangle \in E) && (2.18)\\
&\text{(Nichtnegativität)} && S_i && \geq 0 && (i \in V) && (2.19)\\
&\text{(Startzeitpunkt)} && S_0 && = 0 && && (2.20)\\
&\text{(Exlusivitätsbedingung)} && \sum_{m_i \in M_i} x_{im_i} && = 1 && (i \in V) && (2.21)\\
&\text{(Auswahlbedingung)} && x_{im_i} && \in \{0,1\} && (i \in V, m_i \in M_i) && (2.22)\\
&\text{(Kapazitätsbeschränkung)} && r_\kappa(S,t,x) && \leq R_\kappa && (\kappa \in \mathcal{R}, 0 \leq t \leq \bar{d}(x)) && (2.23)
\end{aligned}$$

2.1.4. Erweiterungen

Ein ressourcenbeschränktes *Scheduling*-Problem lässt sich noch um weitere Gesichtspunkte erweitern. Im Folgenden werden Ausführungszeitfenster für die einzelnen Operationen sowie sequenzabhängige Rüstzeiten eingeführt.

Ausführungszeitfenster von Operationen

Die einzelnen Operationen eines ressourcenbeschränkten *Scheduling*-Problems können weiteren zeitlichen Beschränkungen unterliegen. Für jede Operation O_i lässt sich beispielsweise eine früheste Operationenstartzeit ES_i (engl.: *earliest start*) und ein spätestes Operationenende LE_i (engl.: *latest end*) festlegen. Weiterhin ergeben sich unter Berücksichtigung der Transportdauer das früheste Operationenende EE_i (engl.: *earliest end*) sowie der späteste Operationenstart LS_i (engl.: *latest start*). Da die Operationen durch das *Activity-On-Node*-Diagramm in Beziehung miteinander stehen, müssen die

Ausführungszeitfenster über eine Vorwärts- und eine Rückwärtsterminierung angepasst werden.

Innerhalb der mathematischen Formulierung des Optimierungsproblems lässt sich der früheste Start als Randbedingung aufnehmen:

$$S_i \geq ES_i. \tag{2.24}$$

Ebenso lassen sich die anderen Grenzen des Ausführungszeitfensters einer Operation entweder als Restriktion oder im Optimierungsziel anführen.

Definition 8: Erweitertes *Activity-On-Node*-Diagramm

Wurde ein Ablaufplan S bestimmt, so lässt sich ein erweitertes *Activity-On-Node*-Diagramm betrachten, dass um die Relation der disjunkten Kanten $\mathcal{E}$ erweitert wird. Wurde eine Operation O_i vor einer Operation O_j auf einer Ressource κ ausgeführt, so kann diese Beziehung als geordnetes Paar $\langle i, j \rangle \in \mathcal{E}_\kappa$ aufgefasst werden. Die gesamte Relation ergibt sich als Vereinigung über die Ressourcen $\mathcal{E} = \bigcup_{i=1}^{n} \mathcal{E}_\kappa$.

Abbildung 2.1 zeigt beispielhaft den Aufbau eines erweiterten *Activity-On-Node*-Diagramms.

Sequenzabhängige Rüstzeiten

In einigen Anwendungsfällen treten sequenzabhängige Rüstzeiten auf. Wird auf einer Ressource nach einer Operation O_i eine Operation O_j eingeplant, so kann dies eine Rüstzeit s_{ij} erfordern.

Diese Rüstzeit lässt sich als Kantengewicht $\Delta_{ij}(x)$ im erweiterten *Activity-On-Node*-Diagramm berücksichtigt. Es ergibt sich somit:

$$\Delta_{ij}(\vec{x}) = S_i + p_i + s_{ij} \qquad (\langle i, j \rangle \in \mathcal{E}). \tag{2.25}$$

2.1.5. Shop-Scheduling-Probleme

Bei *Shop-Scheduling*-Problemen werden Operationen in Teiloperationen unterteilt. Man spricht von Jobs $(J_1, ..., J_n)$, die jeweils aus einer Folge von n_i Operationen $O_{i1}, ..., O_{in_i}$ bestehen. Des Weiteren können ausführende, erneuerbare Ressourcen als Menge M von Maschinen $M_1, ...M_m$ aufgefasst werden. Jede Operation O_{ij} kann auf einer Untermenge $M_{ij} \subseteq M$ ausgeführt werden und benötigt dafür eine Dauer von p_{ij}. Man spricht nun

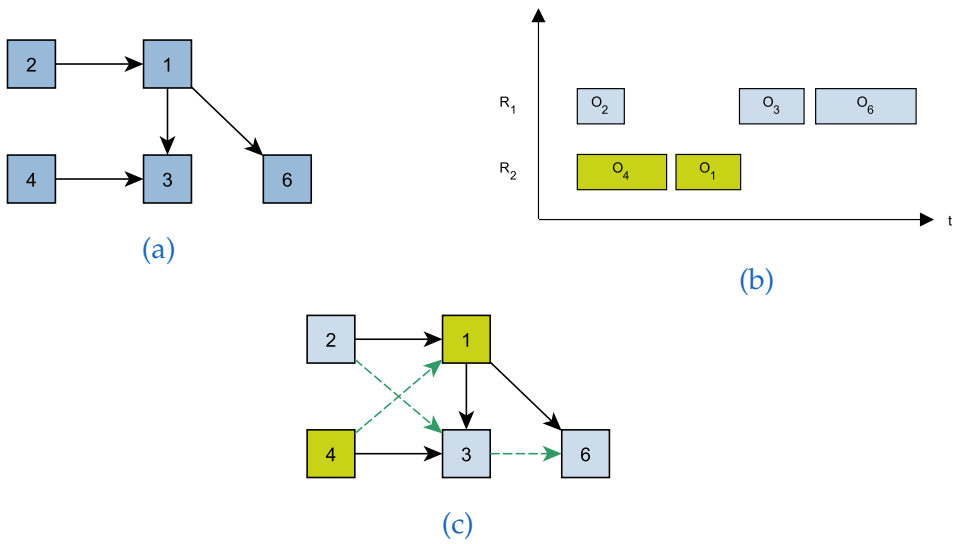


Abbildung 2.1.: Erweitertes *Activity-On-Node*-Diagramm. Zunächst werden technologische Vorschriften im Rahmen eines *Activity-On-Node*-Diagramm vorgegeben (2.1a). Danach folgt die Ablaufplanung, deren Ergebnis ein Ablaufplan ist (2.1b). Nach Erstellen des Ablaufplans lässt sich ein erweitertes *Activity-On-Node*-Diagramm erzeugen, welchem Kanten im Vergleich zum Ausgangsgraphen hinzugefügt wurden (2.1c). Diese Kanten geben die Reihenfolge auf den Ressourcen wieder.

von einem *Shop-Scheduling*-Problem, das sich mit Hilfe eines ressourcenbeschränkten *Scheduling*-Problems beschreiben lässt.

Definition 9: Shop-Scheduling-Problem

Ein *Shop-Scheduling*-Problem besteht aus einer Menge von n Jobs $J = \{J_1, ..., J_n\}$, die aus einer oder mehreren Operation(en) bestehen. Die Operation(en) eines Jobs werden in der Menge $O_i = \{O_{i1}, ..., O_{in_i}\}$ zusammengefasst. Die erneuerbaren Ressourcen werden als Menge von Maschinen $M\{M_1, ...M_m\}$ zusammengefasst. Den Operationen der Jobs werden wie in einem gewöhnlichen *Schedule*-Problem Startzeiten zugeordnet, die entsprechend einer Zielfunktion optimiert werden.

Flexible-Job-Shop

Ein Job J_i kann neben einer einzigen vorgegebenen Folge von Teiloperationen auch eine Menge von möglichen Operationsfolgen zur Auswahl haben. Analog zum MMRCSP gibt es für jeden Job J_i eine Menge von Folgen von Operationen $M_{J_i} = \left\{m^0_{J_i}, ..., m^k_{J_i}, ..., m^m_{J_i}\right\}$. Eine Folge $m^k_{J_i} = (O_{ik1}, ..., O_{ikp})$ beschreibt dabei eine mögliche Reihenfolge von Teilaufgaben, die zur Fertigstellung des Jobs ausgeführt werden müssen. Eine solche

Operationenfolge m_j^k wird im Folgenden Route genannt. Abbildung 2.2 zeigt den strukturellen Aufbau.

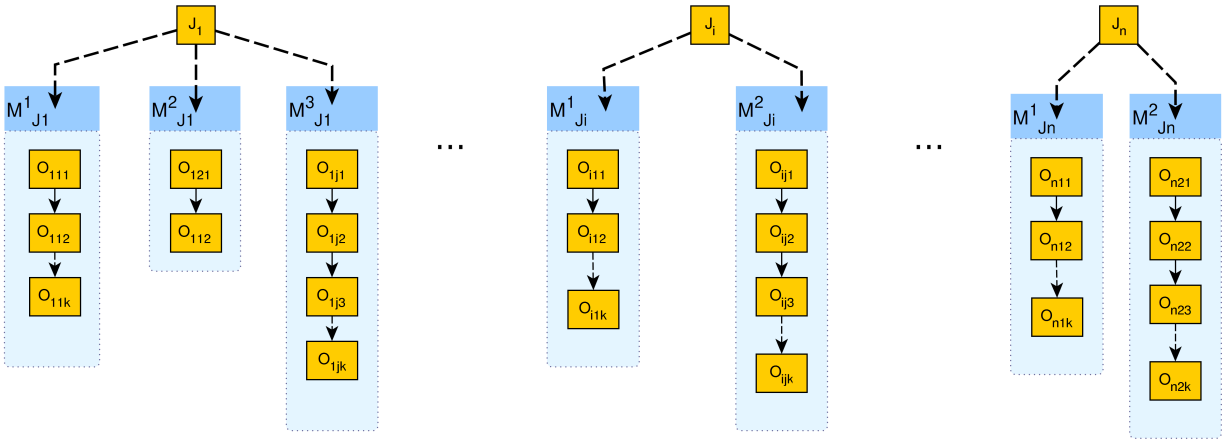


Abbildung 2.2.: Struktur der Jobs im Rahmen eines Flexible-Job-Shops. Die einzelnen Jobs können unterschiedliche Routen besitzen. Diese wiederum bestehen aus unterschiedlichen Operationen.

Definition 10: Prozessplanung

Bei der Prozessplanung wird für jeden Job J_i aus der zugehörigen Menge M_{J_i} eine Route ausgewählt und somit festgelegt, durch welche Folge von Operationen der Job abgearbeitet wird. Durch die Auswahl einer Route pro Job erhält man eine Menge V aller einzuplanenden Operationen.

Wie beim Multi-Mode-RCPS (siehe 2.1.3) lässt sich eine Mode-Assignment-Problem definieren, in dem die Variable $x_{\alpha\beta}$ angibt, ob die Route β des Jobs α ausgewählt wurde. Diese Zuordnung wird zur Definition der einzuhaltenden zeitlichen Restriktionen verwendet. Gibt es eine zeitliche Restriktion zwischen zwei Aktivitäten, so muss diese eingehalten werden, falls beide Aktivitäten ausgewählt wurden. Folglich existiert ein δ_{ij}, wenn $x_{\alpha\beta} = 1 \wedge x_{\varrho\sigma} = 1$ sowie $(\langle i, j \rangle \in E)$ gilt.

2.1.6. Lösung von Ablaufplanproblemen

Zur Optimierung von Ablaufplanproblemen werden zumeist Heuristiken eingesetzt. Diese Heuristiken werden als *Schedule-Generation-Scheme* (SGS) bezeichnet. Ein SGS erzeugt einen zulässigen Plan durch die iterative Erweiterung eines bestehenden Teilplans. Dabei wird grundsätzlich zunächst zwischen seriellen und parallelen SGS unterschieden. Ein serieller SGS arbeitet operationsorientiert und plant in jeder Iteration die betrachtete Operation ein. Ein paralleler SGS betrachtet in jeder Iteration einen neuen Entscheidungspunkt und die zu diesem Punkt zur Verfügung stehende Teilmenge von Operationen, die einplanbar sind.

Als Eingabeparameter für einen SGS gibt es grundsätzlich die Möglichkeiten, eine permutationscodierte Aktivitätsliste oder Prioritätsregeln zu verwenden. Eine permutationscodierte Aktivitätsliste gibt direkt vor, in welcher Reihenfolge versucht wird die Operationen einzuplanen. Prioritätsregeln gegeben die Einplanreihenfolge implizit vor. Sie werden zumeist in Kombination mit einem parallelen, zeitorientierten SGS verwendet. Eine detaillierte Beschreibung der SGS erfolgt in den Abschnitten 3.3.4 und 3.4.2.

Der SGS wird zur Lösung des Optimierungsproblems zumeist in eine Metaheuristik wie beispielsweise in einen Genetischen Algorithmus (Mattfeld (1996), Ozdamar (1999), Lin u. Chong (2015)), Simmulated Annealing (Osman u. Potts (1989), Van Laarhoven u. a. (1992)) oder der Tabu-Search (Dell'Amico u. Trubian (1993), Nowicki u. Smutnicki (1996)) eingebettet.

Im Folgenden wird näher auf den Aufbau von evolutionären Algorithmen eingegangen, da diese im Rahmen des Multiskalen-Algorithmus verwendet werden. Sie bieten durch die Erweiterung zu Koevolutionären Algorithmen die Möglichkeit der kombinierten Optimierung der Prozess- und Ablaufplanung.

Klassifikation und genereller Aufbau von evolutionären Algorithmen

Der Begriff Evolutionäre Algorithmen umfasst alle Algorithmen, die die Evolution in der Art imitieren, dass sie aus einer Population von Individuen diejenigen bestimmen, die sich einer zuvor definierten Umgebung mit limitierten Ressourcen am besten angepasst haben. Die Individuen codieren dabei eine Lösung des zugrundeliegenden Optimierungsproblems. Als Maß für die Anpassung werden die Individuen mit einer Güte bewertet, ihrer sogenannten Fitness. Die Population durchlebt mehrere Generationen, wobei anhand der Fitness ausgewählt wird, aus welchen Individuen ein Individuum der nächsten Generation erzeugt wird. Das Produzieren neuer Nachkommen geschieht mittels einer Rekombination und bzw. oder einer Mutation. Die Rekombination erzeugt aus zwei Individuen der Elterngeneration ein bzw. mehrere Individuen, wohingegen die Mutation aus einem Individuum der Elterngeneration wiederum ein Individuum

erzeugt. Den neu erzeugten Individuen wird ein Fitnesswert zugewiesen, anhand dessen sie mit den anderen Individuen verglichen werden. Nachfolgend wird mittels einer Selektion bestimmt, welche Individuen in die nächste Generation übernommen werden. Der Ablauf des Verfahrens wird in Abbildung 2.3 schematisch dargestellt.

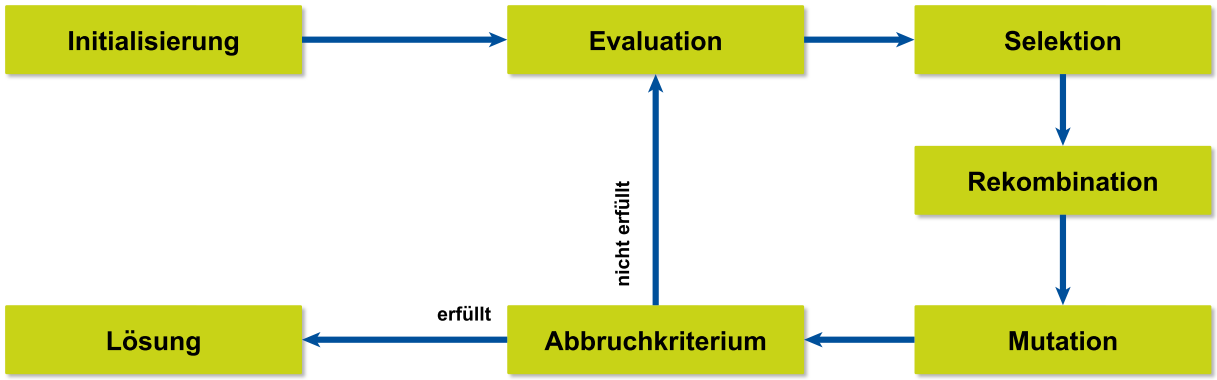


Abbildung 2.3.: Allgemeiner Ablauf eines Evolutionären Algorithmus

Die einzelnen Varianten der evolutionären Algorithmen wurden in den 1970er Jahren entwickelt. Vent (1973) und Schwefel (1975) entwickelten *Evolutionsstrategien (engl.: evolution strategy)* zur Optimierung in technischen Fragestellungen. Vor allem bei kontinuierlichen Optimierungsproblemen finden *Evolutionsstrategien* Anwendung. Hierbei wird das Problem direkt repräsentiert und größtenteils werden Mutationsoperatoren eingesetzt, um den Lösungsraum zu untersuchen. Neben der Mutation ist die Adaption der wichtigste Mechanismus des Verfahrens, der die Parameter der Mutation anpasst.

Das wohl am weitesten verbreitete Verfahren sind die von Holland (1975) entwickelten *Genetischen Algorithmen (engl.: Genetic Algorithm)*, die von De Jong (1975) erstmals auf Optimierungsprobleme angewandt wurden. In ihrer ursprünglichen Form wurde eine binäre Codierung des Problems verwendet, mittlerweile wurden unterschiedlichste problemspezifische Codierungen entwickelt. Weitere Verfahren wie die *Evolutionäre Programmierung* oder die *Genetische Programmierung (engl.: Genetic Programming)* wurden in der darauf folgenden Zeit veröffentlicht (eine genauere Übersicht ist in Gerdes u. a. (2013) zu finden). Abbildung 2.4 zeigt auf, wie die einzelnen Verfahren historisch klassifiziert wurden.

Lange Zeit wurde versucht, die Vor- und Nachteile der einzelnen Verfahren gegenüberzustellen. Der Oberbegriff *Evolutionärer Algorithmus (engl.: Evolutionary Algorithm)* wurde erst später eingeführt. Es sind eine Fülle von Varianten und Kombinationen der einzelnen Verfahren entstanden, wodurch sich die historische Unterteilung nicht mehr aufrecht erhalten lässt. Im allgemeinen Sprachgebrauch werden die Begriffe *Evolutionärer Algorithmus* und *Genetischer Algorithmus* daher heutzutage häufig synonym verwendet.

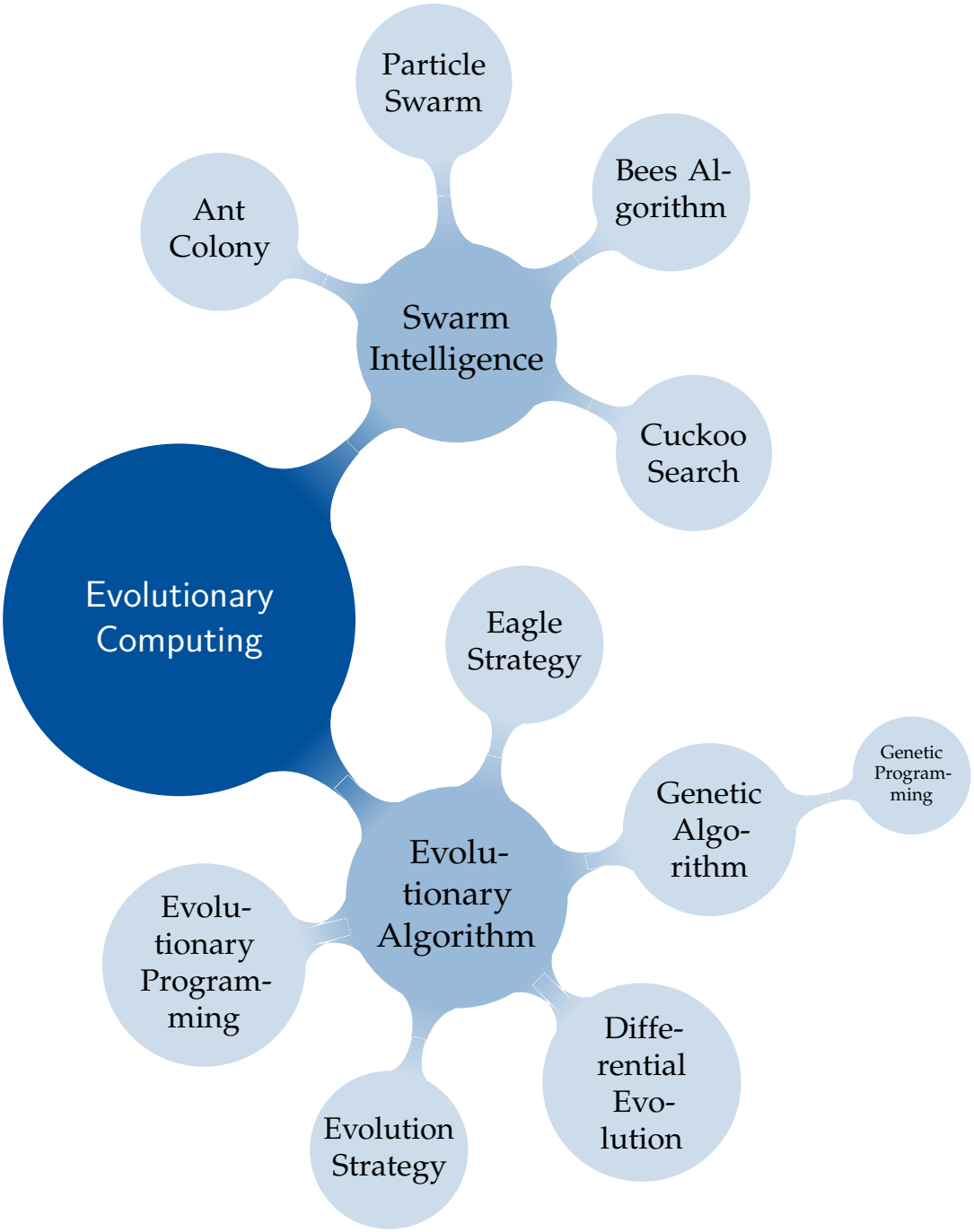


Abbildung 2.4.: Historische Klassifikation von Algorithmen des Evolutionary Computing

Komponenten von evolutionären Algorithmen

Die Analogie zur biologischen Evolution führt dazu, dass auch viele Begrifflichkeiten übernommen wurden. Tabelle 2.1 stellt die wichtigsten Begriffe und ihr Pendant aus der Biologie dar. Auch die Struktur eines Genetischen Algorithmus wird in Analogie zur Biologie aufgebaut.

Die Elemente des Suchraums, Individuen genannt, werden zunächst als Chromosom codiert. Danach wird eine Startpopulation generiert, wobei dies zumeist zufällig geschieht. In vielen Problemstellungen, wie beispielsweise auch bei der Optimierung der Ablaufplanung, kann es vorkommen, dass bei der Generierung der Startpopulation diverse Restriktionen berücksichtigt werden müssen, um valide Individuen mit passenden Chromosomen zu erzeugen.

Evaluation Die Fitness beschreibt den Grad der Anpassung an die Umwelt. Die biologische Evolution hat die Fähigkeit, sich an veränderte Umstände anzupassen. Diese Anpassung wird als *genetischer Drift* bezeichnet und führt dazu, dass Täler in der Fitness-Landschaft (siehe Definition 11) übersprungen werden können.

Die Zielfunktion des Optimierungsproblems entspricht der Fitnessfunktion, die zur Evaluation der Individuen benötigt wird. Unter Umständen muss diese angepasst werden, beispielsweise wenn ein Minimum gesucht werden soll. Weiterhin müssen bei der Fitnessfunktion etwaige ungültige Lösungen berücksichtigt werden und beispielsweise mit einem *Penalty*-Faktor belegt werden. Die Operatoren zur Rekombination, Mutation und Selektion müssen daraufhin der Problemstellung und der Codierung entsprechend ausgewählt werden.

Rekombination Bei der Rekombination zweier bestehender Chromosome wird ein Operator benötigt, der die Gene beider Chromosome neu anordnet. Die Eigenschaften der Eltern können auf unterschiedliche Art und Weise kombiniert werden, wobei oft in drei verschiedenen Arbeitsweisen unterschieden wird (sieh beispielsweise Weicker (2007)). *Kombinierende Operatoren* setzen die Gensequenzen beider Elternindividuen neu zusammen und können im Idealfall die Vorteile beider in einem neuen Individuen vereinen. Diese Art der Rekombination dient meistens zur Erforschung des Suchraums. Weiterhin gibt es die Klasse der *interpolierenden Operatoren*, die die Eigenschaften beider Elternindividuen vermischen, so dass das ein neues Individuum mit Eigenschaften entsteht, welches zwischen den Eigenschaften der Elternindividuen liegt. Diese Art der Rekombination braucht einen recht stark ausgeprägten Mutationsoperator, damit die Population nicht zu schnell konvergiert und der Lösungsraum ausreichend erkundet wird. Die dritte Klasse von Operatoren ist die Klasse der *extrapolierenden Operatoren*. Bei dieser Klasse von Operatoren werden Informationen aus mehreren Individuen abgeleitet und Annahmen zur Verteilung der Individuen im Lösungsraum getroffen,

Name	Biologische Evolution	Evolutionäre Algorithmen
Chromosom	Makromolekülkomplexe, die Gene enthalten und in ihrer Gesamtheit die Eigenschaften des Lebewesens festlegen	String oder Zahlenkette, die den Lösungskandidaten beschreibt
Gen	einzelnes Teilstück eines Chromosoms, das eine (Teil)-Eigenschaft festlegt	Variable
Genort	Ort eines Gens im Chromosom	Postion der Variable
Allel	mögliche Zustandsform eines Gens, das sich an einem bestimmten Genort befindet	Wert eines Zeichens
Genotyp	Das Erscheinungsbild des Chromosoms	codierte Lösung des Problems
Phänotyp	Das resultierende Erscheinungsbild des Lebewesens	Decodierte Form eines Individuums
Population	Menge von Lebewesen	Menge aller potentiellen Lösungskandidaten
Fitness	Überlebens- bzw. Vermehrungsfähigkeit eines Lebewesens. Grad der Anpassung an die Umwelt.	Güte einer Lösung
Generation	Population zu einem Zeitpunkt	Iterationsschritt der Population

Tabelle 2.1.: Gegenüberstellung der Begriffe aus der Biologie und der Optimierung.

um einen Bereich des Lösungsraums zu erreichen, in dem Individuen eine höhere Güte versprechen. Im Weiteren werden einige Rekombinationsoperatoren der einzelnen Klassen vorgestellt.

K-Point-Crossover-Operator Die Rekombinationsoperatoren *One-Point-Crossover* und *Two-Point-Crossover* sind die einfachsten und weitverbreitesten Methoden der Rekombination. Beim *One-Point-Crossover*-Operator wird eine Stelle im Chromosom zufällig bestimmt. Alle Allele vor dieser Stelle werden im ersten der neuen Kindindividuen vom ersten Elternindividuum bezogen, alle ab dieser Stelle bis zum Ende des Chromosoms vom zweiten Elternindividuum. Ein zweites Individuum lässt sich analog erzeugen, in dem die jeweils anderen Teile des Chromosoms benutzt werden. Beim *Two-Point-*

Crossover-Operator werden dahingegen zwei Stellen zufällig ausgewählt und die Allele zwischen diesen Stellen ausgetauscht. Dieses Prinzip lässt sich beliebig erweitern zum *K-Point-Crossover*-Operator.

Uniform-Crossover-Operator Beim *Uniform-Crossover*-Operator wird im Gegensatz zu *K-Point-Crossover*-Operatoren für jedes Allel festgelegt, aus welchem Elternindividuum es bezogen wird. Dies geschieht zumeist über eine Bitmaske, die angibt, aus welchem Elternindividuum welches Allel übernommen wird.

Rekombinationsoperatoren für permutationscodierte Chromosomen Della Croce u. a. (1995) beschreiben ausführlich die Tauglichkeit von verschiedenen Operatoren für spezielle Anwendungen. Bei permutationscodierten Chromosomen, wie sie bei der Ablaufplanung oder bei Reihenfolgeproblemen im Allgemeinen auftreten, muss auf problemspezifische Eigenschaften eingegangen werden. Häufig unterliegt die Reihenfolge innerhalb eines Chromosoms bestimmten Nebenbedingungen, wie beispielsweise fest vorgeschriebenen Vorgängerbeziehungen zwischen Operationen. Diese Beziehungen müssen nach dem Anwenden des Rekombinationsoperators weiterhin gewährleistet werden oder nachträglich repariert werden.

Goldberg u. Lingle (1985) entwickelten den *Partially-Matched-Crossover*-Operator (PMX), der ähnlich einem *Two-Point-Crossover*-Operator die Chromosomen der Elternindividuen auftrennt. Kommen Elemente an zwei verschiedenen Genorten vor, so wird es durch das korrespondierende Element des anderen Elternindividuums ersetzt. Ähnlich zu diesem Operator entwickelte Davis (1985) den *Order-Based-Crossover*-Operator (OX), der zyklisch arbeitet und nach der Schnittposition die restlichen Elemente, die noch nicht verwendet wurden, dem Nachkommen anfügt. Dies führt dazu, dass die Elemente in den Nachkommen nicht so verstreut auftauchen wie beim PMX-Operator. Deswegen wird er häufig beim *Travelling-Salesman*- Problem eingesetzt. Die relative Position der Elemente untereinander wird eher berücksichtigt als ihre absolute Position innerhalb des Chromosoms.

Der *Uniform-Order-Based-Crossover*-Operator von Syswerda (1991) ist wiederum an den *Uniform Crossover*-Operator angelehnt und verwendet ebenso eine Bit-Maske. So werden die Elemente vom ersten Individuum gewählt, bei denen in der Maske eine 1 hinterlegt ist. Die Reihenfolge der übrigen Elemente wird aus dem anderen Elternteil übernommen und die Elemente auf die freien Genorte verteilt.

Innerhalb von Scheduling-Problemen müssen allerdings, wie bereits erwähnt, gewisse Reihenfolge-Nebenbedingungen berücksichtigt werden, weswegen sich der *OX*-Operator oft nicht eignet. Falkenauer u. Bouffouix (1991) entwickelten daraufhin eine weitere Variante des OX, den sogenannten *Linear-Ordered-Crossover*-Operator (LOX). Ein weiterer Vertreter der Rekombinationsoperatoren für Reihenfolgeprobleme ist der *Cycle-Crossover*-Operator, der stark verwandt mit dem *Uniform-Crossover*-Operator ist.

Es gibt sowohl bei den Standard-Rekombinationsoperatoren als auch bei denen für Reihenfolgeproblemen angepassten noch viele weitere Operatoren, die für bestimmte Codierungen und Problemstellungen entwickelt wurden. Eine vollständige Aufzählung ist daher kaum möglich. Eine gute Übersicht über eine etwas größere Anzahl bieten beispielsweise Gerdes u. a. (2013).

Mutation Die Mutation wird benötigt, um neues Gengut in die Population mit einzubringen. Der Mutationsoperator dient folglich zur Erkundung des Lösungsraums und bewirkt zumeist eine zufällige Änderung eines oder mehrerer Gene eines Chromosoms. Genetische Algorithmen haben den großen Nachteil, dass sie oft vorzeitig konvergieren. Im Extremfall führt dies dazu, dass alle Individuen einer Population den gleichen Genotyp aufweisen. Viele Autoren, die einen Überblick über Genetische Algorithmen geben (beispielsweise Weicker (2007), Gerdes u. a. (2013)), führen die Ansicht von Spears (1995) an, der die Meinung vertritt, dass Mutation in kleinen Populationen eine sehr wichtige Rolle spielt, um den Lösungsraum abzudecken. In großen Populationen, in denen eine größere Vielfalt vorherrscht, kann die Mutation allerdings gute Lösungen zerstören. Häufig wird, wenn die Evaluation eines Individuums nicht zu viel Rechenzeit in Anspruch nimmt, daher das unmutierte und das mutierte Individuum bewertet.

Für unterschiedliche Codierungen eignen sich unterschiedliche Arten von Mutation. Bei binären Codierungen wird oft lediglich zufällig ein Wert geändert (*Bit-Flip*), oder zwei Gene getauscht (*Bit-Swap*). Bei permutationsbasierten Codierungen für Reihenfolgeprobleme werden zumeist Teile des Chromosoms verschoben oder gespiegelt.

Selektion Das Auswahlverfahren, mit dessen Hilfe bestimmt wird, welche Eltern zur Reproduktion ausgewählt werden, nennt sich Selektion. Zur Auswahl wird zumeist die Fitness der Lösungskandidaten herangezogen. Einer der wichtigsten Punkte bei der Selektion der Elternindividuen ist dabei der sogenannte Selektionsdruck, der angibt wie stark gute Individuen bei der Auswahl bevorzugt werden. Der Selektionsdruck steuert somit, wie schnell der Genetische Algorithmus konvergiert. Herrscht ein großer Selektionsdruck, werden gute Lösungen stark bevorzugt. Ist der Druck zu hoch, vermehren sich diese guten Individuen so stark, dass die Vielfalt der Population zu gering wird und der Algorithmus schnell konvergiert, zumeist gegen ein lokales Optimum. Bei einem zu niedrigen Selektionsdruck werden gute Individuen kaum bevorzugt, was dazu führt, dass zum Teil zu viele schlechte Individuen in der Population verbleiben und der Algorithmus nicht oder nur langsam konvergiert. Das Verfahren beschreibt in diesem Fall fast eine Zufallssuche.

Am weitesten verbreitet sind die Methoden der *fitnessproportionalen* Selektion, der *Turnierselektion* und der *Rangselektion*. Als einfachste Art der Selektion können auch die n besten Individuen einer Population ausgewählt werden. Dieses Verfahren führt

häufig zu einer sehr schnellen Konvergenz, zumeist zu einer Lösung in einem lokalen Optimum.

Bei der *fitnessproportionalen* Selektion (auch *Roulette-Wheel*-Selektion genannt) wird die Selektionswahrscheinlichkeit $p_s(I_i)$ eines Individuums I_i direkt proportional zu seiner Fitness bestimmt und ergibt sich zu:

$$p_s(I_i) = \frac{f(I_i)}{\sum_{j=1}^{n} f(I_j)}. \tag{2.26}$$

Der Selektionsdruck ist bei diesem Verfahren relativ gering, wenn die Fitness der Individuen der Population recht dicht beieinander liegt. Die Konvergenz ist somit langsam, die Vielfalt allerdings hoch.

Bei der *Rangselektion* werden die Individuen entsprechend ihrer Fitness absteigend in einer Liste sortiert. Die Position innerhalb dieser Liste gibt den Rang eines Individuums an. Die Selektionswahrscheinlichkeit steht hierbei in Relation zum Rang des Individuums, und wird zumeist über einen Parameter $1 \leq E_{max} \leq 2$ geregelt, der den Erwartungswert angibt, dass die beste Lösung ausgewählt wird. Die Selektionswahrscheinlichkeit der übrigen ergibt sich daraufhin mit der Gesamtanzahl an Individuen n und dem Rang $r(I_i)$ des Individuums I_i zu:

$$p_s(I_i) = \frac{2 - E_{max}}{n} + \frac{2 \cdot r(I_i) \cdot (E_{max} - 1)}{n \cdot (n-1)} \tag{2.27}$$

Die Selektionswahrscheinlichkeit eines Individuums ist somit nicht direkt von der Fitness abhängig, sondern vom Rang des Individuums. Der Selektionsdruck lässt sich über den Parameter E_{max} steuern.

Bei der *Turnierselektion* werden 2 bis k Individuen zufällig ausgewählt und deren Fitness-Werte verglichen. Das Individuum mit der höchsten Fitness gewinnt das Turnier und wird zur Erzeugung von Nachkommen verwendet. Der Selektionsdruck lässt sich über die Anzahl an ausgewählten Individuen k beeinflussen. Je mehr Individuen gewählt werden, desto höher ist der Selektionsdruck. Bei geringen Werten kann es sein, dass die besten Individuen nicht selektiert werden.

Prinzipien evolutionärer Algorithmen

In diesem Abschnitt erfolgt eine mathematische Sicht auf die Genetischen Algorithmen. Hierbei wird zunächst die Fitness-Landschaft definiert, die als Darstellungsweise des Suchraums verstanden werden kann. Nachfolgend wird die Vorgehensweise eines Genetischen Algorithmus aus Sicht der Stochastik beurteilt. Die Ausführungen dienen einem besseren Verständnis der Arbeitsweise von evolutionäre Algorithmen und bilden die Grundlage zum Design von problemspezifischen Operatoren.

Fitness-Landschaft Das Konzept der Fitness-Landschaft wurde vom theoretischen Biologen Wright (1932) eingeführt, um in der biologischen Evolution die Dynamik der Optimierung von Lebewesen darzustellen; genauer gesagt zur Darstellung der Relation zwischen dem Genotyp der einzelnen Individuen und deren Reproduktionsrate. Bei dieser Darstellungsweise nehmen hoch angepasste Arten Bereiche nahe den Gipfeln der Landschaft ein. Dieses Konzept wurde auf Optimierungsprobleme und evolutionäre Algorithmen übertragen. Jeder Lösung, repräsentiert durch ein Individuum, wird eine Fitness zugewiesen. Die Individuen werden so in einem abstrakten Raum angeordnet, dass veranschaulicht werden kann, wie die Fitness über die Lösungen des Optimierungsproblems verteilt ist und wie Algorithmen durch diese Fitness-Landschaft navigieren.

Definition 11: Fitness-Landschaft

Die Fitness-Landschaft ergibt sich zunächst als Hyperfläche, die durch die Gene und ihre möglichen Ausprägungen aufgespannt wird. Die verschiedenen Individuen mit ihren möglichen Ausprägungen werden als Punkte der Landschaft repräsentiert. Ihre Höhe wird durch die jeweilige Fitnessfunktion bzw. Reproduktionsrate bestimmt. Formal besteht eine Fitness-Landschaft $L(S, f, d)$ aus:

- einer Menge von Konfigurationen der Gene (Lösungen) S
- einer Fitness Funktion $f : S \to \mathcal{R}$, die jeder Lösung einen Fitness-Wert zuordnet
- und einer Metrik $d : S \times S \to \mathcal{R}$.

Eine Beschreibung der Nachbarschaft zweier Lösungen erfolgt dabei durch eine durch die Metrik, die den Abstand zwischen zwei Lösungen definiert, induzierte Topologie. Häufig wird als Metrik eine Funktion verwendet, die den Abstand zwischen Lösung x zu Lösung y durch die Anzahl an Schritten, die der Mutationsoperator benötigt, um von Lösung x zu Lösung y zu gelangen, festlegt. Dadurch ergeben sich unterschiedliche Landschaften bei der Verwendung unterschiedlicher Operatoren. Für die Metrik gilt weiterhin $d_{min} \leq d(s_1, s_2) \leq d_{max} \quad \forall s_1, s_2 \in S, s_1 \neq s_2$. Zwei Lösungen s_1 und s_2 sind somit benachbart, wenn $d(s_1, s_2) = d_{min}$ gilt.

Unter diesen Voraussetzungen lässt sich die Fitness-Landschaft auch als ungerichteter gewichteter Graph $G = (V, R)$ verstehen. Die Knotenmenge entspricht dabei der Menge an Lösungen S. Die Kanten des Graphen werden durch die Relation $R \subset S \times S | d(s, s') = d_{min}$ repräsentiert, wobei die Kanten somit eine Nachbarschaftsbeziehung beschreiben.

Evolutionäre Algorithmen aus der Sicht der Stochastik Betrachtet man die Menge der Lösungen S, so können Übergangswahrscheinlichkeiten T_{xy} angegeben werden, die beschreiben, mit welcher Wahrscheinlichkeit, ausgehend von der Lösung $x \in S$, die Lösung $y \in S$ berechnet wird. Mit dieser Erkenntnis lässt sich ein Evolutionärer Algorithmus als eine Markov-Kette interpretieren. In Rudolph (1996) wird dieses Vorgehen ausführlich beschrieben. Zur Einordnung der evolutionären Algorithmen folgen einige Definitionen, die auf den Grundbegriffen der Stochastik (siehe Abschnitt 2.2.1) aufbauen.

Definition 12: Stochastischer Prozess

Ein *stochastischer Prozess* ist die mathematische Beschreibung einer Folge von Zufallsexperimenten, die zeitlich geordnet sind. Die Folge lässt sich mittels Zufallsvariablen X_t beschreiben, wobei $t \in T$ gilt, und T den *Parameterraum* beschreibt, der die möglichen Zeitpunkte im System enthält. Ein stochastischer Prozess X kann als Abbildung $X : \Omega \times T \to \mathcal{R}$ aufgefasst werden.

Definition 13: Stochastische Kette

Bei einer *stochastischen Kette* handelt es sich um einen stochastischen Prozess, bei dem der Parameterraum diskret ist, d.h. t und somit auch X_t können nur abzählbar viele *Zustände* $X_0, X_1, ...X_n$ annehmen. Die Zustände stammen dabei aus der endlichen *Zustandsmenge* $S = s_1, .., s_m$,.

Beispielsweise ist das n-malige, zeitlich nacheinander ausgeführte Würfeln eine stochastische Kette.

Zur Beschreibung des Wahrscheinlichkeitsraums sowie von Maßräumen werden σ-Algebren verwendet, die im Folgenden definiert werden. Neben der nachfolgenden Definition von Markov-Ketten werden sie ebenfalls bei der Motivation der Plausibilitätstheorie in Abschnitt 3.3.9 verwendet.

Definition 14: Sigma-Algebra

Eine σ-Algebra $\mathcal{A}$ mit $A \subseteq \mathcal{P}(\Omega)$ ist eine mengentheoretische Struktur, die mit Hilfe der Potenzmenge $\mathcal{P}$ ein Mengensystem auf einer festen Grundmenge Ω bezeichnet.

Dabei muss gelten, dass

- $\Omega \in \mathcal{A}$
  die Grundmenge in der Struktur enthalten ist.
- $A \in \mathcal{A} \Rightarrow A^c \in \mathcal{A}$
  wenn $\mathcal{A}$ eine Teilmenge A von Ω enthält, auch deren Komplement enthalten sein muss.
- $A_1, A_2, \cdots \in \mathcal{A} \Rightarrow \bigcup_{n\in\mathbb{N}} A_n \in \mathcal{A}$
  wenn für jede natürliche Zahl n die Menge $\mathcal{A}_n$ in $\mathcal{A}$ enthalten ist, auch die abzählbarer Vereinigung aller $\mathcal{A}_n$ in $\mathcal{A}$ liegt.

Eine *Markov-Kette* ist ein Spezialfall eines stochastischen Prozesses, der häufig auftritt. Im Gegensatz zu *Markov-Prozessen* ist die Zustandsmenge bei Markov-Ketten diskret, wohingegen dieser bei Markov-Prozessen stetig ist. Zusätzlich lassen sich sowohl Markov-Ketten als auch -Prozesse anhand des Parameterraums in stetige und diskrete Markov-Ketten bzw. -Prozesse unterscheiden. Hier soll, im Hinblick auf die spätere Anwendung, lediglich auf diskrete Markov-Ketten näher eingegangen werden.

Definition 15: Markov-Kette

Eine Markov-Kette ist eine stochastische Kette, die nur von den n vorangehenden Zuständen abhängt. Es ergibt sich:

$$P(X_{t+1} \in A \mid X_t, X_{t-1}, \ldots, X_0) = P(X_{t+1} \in A \mid X_t, \ldots, X_{t-n+1}). \tag{2.28}$$

mit endlicher Zustandsmenge $S = s_1, ...s_m$ und der Ordnung n, die angibt, von wie vielen vorangehenden Zuständen der Zustand zum Zeitpunkt $t+1$ abhängig ist. Das Ereignis A ist dabei ein Element der σ-Algebra $\mathcal{A}$.

Bei Markov-Ketten erster Ordnung hängt der nächste Zustand nur vom aktuellen Zustand ab:

$$P(X_{t+1} \in A \mid X_t, X_{t-1}, \ldots, X_0) = P(X_{t+1} \in A \mid X_t). \tag{2.29}$$

Die *Übergangswahrscheinlichkeit* gibt an, mit welcher Wahrscheinlichkeit das System von einem Zustand s_i zum darauf folgenden Zustand s_j wechselt. Formal definieren lässt

sich die Übergangswahrscheinlichkeit als

$$p_{ij}(t) := P(X_{t+1} = s_j \mid X_t = s_i), \quad i,j = 1,\ldots,m. \tag{2.30}$$

Die Übergangswahrscheinlichkeiten für beliebige i und j, für die $i,j = 1,\ldots,m$ gilt, lassen sich in Übergangsmatrizen $\mathbf{P}(\mathbf{t})$ zusammenfassen. Im Allgemeinen sind diese von der Zeit abhängig und ergeben sich zu:

$$\mathbf{P}(\mathbf{t}) = \begin{pmatrix} p_{11}(t) & \ldots & p_{1m}(t) \\ \vdots & \ddots & \vdots \\ p_{m1}(t) & \ldots & p_{mm}(t) \end{pmatrix}. \tag{2.31}$$

Von einer *homogenen Markov-Kette* spricht man, wenn die Übergangswahrscheinlichkeit zu jedem Zeitpunkt gleich ist.

Im Rahmen eines Genetischen Algorithmus werden in jeder Generation zwei Elternindividuen benutzt, um durch Mutation und Selektion ein neues Individuum zu erzeugen. Daher handelt es sich bei der Optimierung mittels dieses Metaverfahrens aus mathematischer Sicht um einen Markov-Prozess. Der Zustand zur Zeit t (aktuelle Generation) hängt nur vom Zustand zur Zeit $t-1$ (der Generation der Elternindividuen) ab. Eine Übergangsmatrix für den Prozess lässt sich aus den Operatoren des Genetischen Algorithmus herleiten. Grundsätzlich lässt sich festhalten, dass es sich bei Genetischen Algorithmen um ein stochastisches Verfahren handelt. Sie sind verwandt mit den Monte-Carlo-Methoden. Im Grunde genommen erweitern sie die Zufallssuchmethode der Monte-Carlo-Methoden um Verfahren, die der Evolution angelehnt sind, um eine gerichtete Zufallssuche zu erhalten, die eine bessere Konvergenz verspricht.

2.1.7. Ursachen von Unsicherheit

Die Ablaufplanung unterliegt je nach Problemstellung verschiedenen Unsicherheitsaspekten. Dabei werden Unsicherheiten zumeist in zwei Kategorien unterschieden. *Aleatorische* Unsicherheiten leiten sich vom inhärenten Zufall eines betrachteten Systems ab. *Epistermische* Unsicherheit folgt auf einen Mangel an exakten Informationen. Die Informationslage lässt sich teilweise nur sehr aufwändig verbessern, was somit nicht immer sinnvoll ist. Zimmermann (2000) führt fünf Ursachen auf, die einen Großteil der Möglichkeiten von Unsicherheiten abdecken:

- Informationsmangel: Die wohl häufigste Ursache ist das nicht Vorhandensein jeglicher Information über ein Ereignis.
- Informationskomplexität: Ist ein Entscheidungsträger aufgrund eines sehr komplexen Systems mit einer Informationsflut konfrontiert, kann eine Filterung der Daten zu Unsicherheiten führen.
- Informationskonflikt: Für einen Sachverhalt können widersprüchliche Informationen zur Verfügung stehen, beispielsweise aufgrund verschiedener Expertenaussagen.
- Informationsmehrdeutigkeit: Gleiche linguistische Informationen können unterschiedlichen numerischen Angaben entsprechen.
- Subjektivität: Vorliegende Informationen sind unsicher, wenn sie auf subjektiven Einschätzungen basieren.

Weiterhin treten beim Abstraktionsprozess der Modellierung Unsicherheiten auf. Beim Abbilden der Realität auf ein Modell werden Vereinfachungen getroffen, die dazu führen, dass scharfe Eingabeparameter zu unscharfen mit Unsicherheit belasteten Ergebnissen führt. Dies entspricht im Kern der gleichen Problematik wie bei der Informationskomplexität.

In vielen Bereichen des täglichen Lebens und der Wissenschaft sind wir von Unschärfen umgeben. Gerade bei der Entscheidungsfindung und anderen Optimierungsproblemen können Unsicherheiten in den Parametern großen Einfluss haben. Um die Unsicherheiten und ihre Auswirkungen zu verdeutlichen, wird zunächst ein Problem der linearen Optimierung betrachtet.

Allgemein lässt sich ein lineares Optimierungsproblem, dass dadurch gekennzeichnet ist, dass die Zielfunktion linear ist und die Nebenbedingungen durch ein System linearer Gleichungen und Ungleichungen darstellbar sind, als ein Minimierungsproblem in kanonischer Form formulieren:

$$\min f(x) = c^T x \tag{2.32}$$

unter den Nebenbedingungen

$$Ax \leq b \tag{2.33}$$
$$x \geq 0 \tag{2.34}$$

In der Realität unterliegen die Parameter A, b und c Unsicherheiten. Das bedeutet, dass weder die Nebenbedingungen noch der Einfluss der einzelnen Parameter am Wert der Zielfunktion deterministisch beschreibbar sind. Somit ist auch der Wert der Zielfunktion nicht frei von Unsicherheiten.

Bei den hier betrachteten Ablaufproblemen können somit sämtliche Werte, wie beispielsweise die Ausführungszeitfenster der einzelnen Operationen, der Ressourcenbedarf einer Operation genauso wie der bestimmte Wert der Zielfunktion Unsicherheiten unterliegen.

2.2. Mathematische Methoden zur Beschreibung von Unsicherheiten

Im Folgenden werden kurz die Grundlagen der Stochastik und der *Fuzzy-Set*-Theorie als Mittel zur mathematischen Beschreibung von Unsicherheiten vorgestellt. Nachfolgend wird ein Vergleich zwischen den Methoden vorgenommen.

2.2.1. Unsicherheitsbeschreibung mit Mitteln der Stochastik

Eine Möglichkeit, Unsicherheiten mathematisch zu beschreiben, ist die Verwendung von Methoden aus der Stochastik. Im Gegensatz zu der Verwendung von deterministischen Größen werden unsichere Größen als Zufallsvariable aufgefasst. Die Verteilung der Zufallsvariable lässt sich als mathematische Funktion beschreiben. Auf Basis dieser Verteilungen lassen sich weiterführende Berechnungen durchführen.

Grundbegriffe der Stochastik

Ein *Zufallsexperiment* ist ein Vorgang, der genau beschreibbar und auch wiederholbar ist. Der Ausgang des Vorgangs ist jedoch zufällig. Führt man ein Zufallsexperiment aus, beispielsweise das Würfeln mit einem Würfel, so umfasst der *Wahrscheinlichkeitsraum* $(\Omega, \mathcal{F}, P)$ die *Ergebnismenge* Ω sowie die *Ereignisalgebra* $\mathcal{F}$ und das *Wahrscheinlichkeitsmaß* P, welches auf $\mathcal{F}$ definiert ist.

Als *Elementarereignisse* bezeichnet man alle möglichen Ereignisse, die bei einem Zufallsexperiment eintreten können. Im obigen Beispiel sind alle Zahlen von 1 bis 6 Elementarereignisse. Die *Ergebnismenge* Ω enthält nun alle diese möglichen *Elementarereignisse*, die auftreten können, im Beispiel $\Omega := \{1, 2, 3, 4, 5, 6\}$.

Die *Ereignismenge* $\mathcal{F}$ ist als σ-Algebra auf Ω definiert. Es werden also nur Teilmengen der möglichen Ergebnisse betrachtet, die durch bestimmte Ereignisse beschrieben sind. Häufig wird hierbei die *Potenzmenge* des Ergebnisraumes herangezogen, die alle Teilmengen enthält. Möchte man beispielsweise das Ereignis betrachten, dass die gewürfelte Zahl gerade ist, so ist dies das Ereignis $A = \{2, 4, 6\}$.

Da Ereignisse Teilemengen des *Ergebnisraumes* $\mathcal{F}$ sind, lassen sich Beziehungen zwischen den Ereignissen mit den Mitteln der Mengenalgebra beschreiben. Betrachtet man das Ereignis $A := 2, 4, 6$ „Es wird eine gerade Zahl gewürfelt“ und $B := \{1, 3\}$ „Die Augenzahl ist 1 oder 3“, so lässt sich feststellen, dass diese Ereignisse nicht gleichzeitig stattfinden können, es gilt also $A \cap B = \emptyset$. In diesem Falle spricht man davon, dass A und B *disjunkte Ereignisse* sind.

Das *Wahrscheinlichkeitsmaß* P ist eine Funktion, die jedem Ereignis $A \in \mathcal{F}$ eine Zahl zwischen 0 und 1 zuweist. Dabei handelt es sich um die Wahrscheinlichkeit von A. Bei einem Würfel ist die Wahrscheinlichkeit für alle möglichen Elementarereignisse gleich, sie liegt bei $1/6$. Es lässt sich festhalten, dass ein Wahrscheinlichkeitsmaß normiert ist. Diese Eigenschaft besagt, dass die Ergebnismenge Ω eine Wahrscheinlichkeit von 1 hat. Weiterhin gilt die Eigenschaft der σ-Additivität des Wahrscheinlichkeitsmaßes P. Diese besagt, dass sich die Wahrscheinlichkeit für ein aus disjunkten Ereignissen zusammengesetzte Ereignis als Summe der einzelnen Wahrscheinlichkeiten der disjunkten Ereignisse ergibt. Definiert man das Ereignis E „ungerade Zahl oder gerade Zahl“, so ergibt sich die Wahrscheinlichkeit dieses Ereignisses zu $P(E) = P(A \cup B) = P(A) + P(B)$.

Als *Zufallsgröße* bezeichnet man die Variable, die innerhalb des Versuches vom Zufall abhängig ist. Nimmt man das Würfelexperiment zur Hand, so kann man beispielsweise festlegen, dass der Spieler eine Gewinnsumme von 1 Euro ausgezahlt bekommt, wenn er eine 6 würfelt. Die Auszahlungssumme X sei dann wie folgt definiert:

$$X(\omega) = \begin{cases} 1, \text{wenn } \omega = 6 \\ 0, \text{sonst,} \end{cases} \tag{2.35}$$

wobei $\omega \in \Omega$ die Realisierung der Zufallsvariable beschreibt.

Wahrscheinlichkeitsinterpretationen in der Stochastik

Über die Jahre haben sich verschiedene Interpretationen des Begriffes Wahrscheinlichkeit durchgesetzt, die in Ocker (2010) näher beschrieben sind.

Die *klassische Wahrscheinlichkeitsinterpretation* nach Laplace legt besonderen Wert auf die kombinatorische Natur der zugrundeliegenden Experimente. Bei einem Laplace-Experiment müssen alle Elementarereignisse mit der gleichen Wahrscheinlichkeit eintreten, der Wahrscheinlichkeitsraum ist abwählbar und besitzt endlich viele Elementarereignisse. Die Ermittlung der Wahrscheinlichkeit ergibt sich aus den kombinatorischen Möglichkeiten, die als Ergebnis eintreten können. So werden die günstigen Ereignisse durch die Gesamtanzahl der Elementarereignisse geteilt, um die Wahrscheinlichkeit zu bestimmen. Wahrscheinlichkeiten werden nach der klassischen Methode bestimmt, falls das *Indifferenzprinzip*, auch Prinzip vom unzureichenden Grund genannt, gilt. Dies besagt, dass, wenn kein Grund erkennbar ist, der ein Elementarereignis mehr oder weniger wahrscheinlich erscheinen lässt, alle Elementarereignisse gleichverteilt sind.

Der Begriff der *frequentistischen Wahrscheinlichkeit*, den vor allem Richard von Mises und Egon Pearson geprägt haben, setzt zunächst voraus, dass ein Zufallsexperiment mehrfach unter identischen Bedingungen durchgeführt werden kann. Die Bestimmung

der Wahrscheinlichkeit basiert auf der relativen Häufigkeit. Hierzu wird das Zufallsexperiment mehrfach durchgeführt. Die *frequentistische Wahrscheinlichkeit* eines Ereignisses A ergibt sich als Quotient zwischen der Anzahl der Zufallsvorgänge, bei denen das Ereignis A auftritt, und der Gesamtanzahl an Zufallsvorgängen. Werden nur wenige Zufallsexperimente durchgeführt, ist die Wahrscheinlichkeit, dass die relative Häufigkeit von der tatsächlichen Wahrscheinlichkeit abweicht, groß. Das *Empirische Gesetz* nach Bernoulli beschreibt allerdings, dass die relative Häufigkeit eines Ereignisses sich bei einer großen Anzahl an Zufallsexperimenten an einen Grenzwert annähert, der die Wahrscheinlichkeit des Ereignisses darstellt. Da das Zufallsexperiment beliebig oft unter identischen Bedingungen ausgeführt werden muss, kann die *frequentistische Wahrscheinlichkeit* nicht für Ergebnisse abgewandt werden, die lediglich einmalig auftreten. Insgesamt wird oft angeführt, dass die Annäherung an einen bestimmten Grenzwert ein rein mathematisches Phänomen ohne reale Bedeutung sein kann (vgl. Bourier (2002)).

Die *subjektive Wahrscheinlichkeitsermittlung* basiert darauf, dass sachkundige Personen die Eintrittswahrscheinlichkeiten eines Ereignisses rational beurteilen und bestimmen können. Der Experte bildet sich unter Zuhilfenahme seines Fachwissens eine subjektive Meinung über die Wahrscheinlichkeiten, die in Zahlen ausgedrückt werden müssen. Dieses Vorgehen birgt die Gefahr, dass die angenommenen Wahrscheinlichkeiten willkürlich festgelegt worden sind. Durch die Überführung der Wahrscheinlichkeiten in den soliden Rahmen der Stochastik können Ergebnisse einer Berechnung mit diesen Werten eine Aufwertung beim späteren Entscheidungsträger erlangen, die nicht gerechtfertigt ist.

2.2.2. Unsicherheitsbeschreibung mit Mitteln der Fuzzy-Set-Theorie

Mit seiner Arbeit entwickelte Zadeh (1965) die theoretische Basis zur Modellierung von Unschärfe. Er beschrieb eine Erweiterung der klassischen zweiwertigen Logik, von der ausgehend er unscharfe Mengen definierte. Der folgende Abschnitt beschreibt die Grundlagen der *Fuzzy-Set*-Theorie.

Motivation und mathematische Definition

In der klassischen Mengentheorie nach Cantor gehört ein Element entweder zu einer Menge oder nicht. Diese Zugehörigkeit kann man mittels einer binären charakteristischen Funktion modellieren. Betrachtet man die Grundmenge X, so gibt die charakteristische Funktion $\chi : X \to \{0, 1\}$ an, ob ein Element zu einer Menge gehört ($\chi(x) = 1$) oder nicht ($\chi(x) = 0$).

Diese Definition der scharfen Menge ist ausreichend für sehr viele Anwendungen. Betrachtet man das Beispiel der Menge aller Rechtecke A $\{x \in X | x \text{ ist ein Rechteck}\}$, so

lassen sich die geometrischen Figuren in Abbildung 2.5a eindeutig der Menge zuordnen. Im Beispiel sind a.), c.) und d.) Rechtecke.

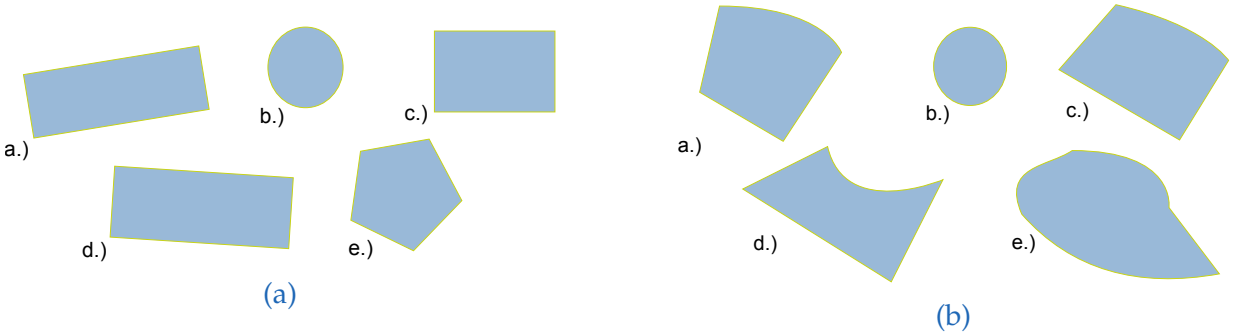


Abbildung 2.5.: Motivation der Einführung unscharfer Mengen. Geometrische Objekte, deren Zugehörigkeit zur Menge der Rechtecke festgestellt werden soll.

Betrachtet man hingegen Abbildung 2.5b, so ist zwar klar, dass die Figuren a.) und c.) keine Rechtecke sind, sie ähneln allerdings einem Rechteck deutlicher mehr als die Figuren d.) und e.). Das Kreiselement b.) würden wohl die wenigsten Menschen der Menge aller Rechtecke zuordnen.

Hier setzt die *Fuzzy-Set*-Theorie an. Erweitert man die Bildmenge der charakteristischen Funktion um das gesamte Intervall $[0, 1]$, sprich man lässt auch Werte zwischen 0 und 1 zu, so kann man angeben, zu welchem Grad ein Element in der Menge enthalten ist. Da die Funktion den Grad der Zugehörigkeit zu einer Menge angibt, wird sie Zugehörigkeitsfunktion genannt. Über diese Funktion lässt sich, wie bereits zuvor motiviert, eine unscharfe Menge $\tilde{A}$ definieren:

$$\tilde{A} = \{(x, \mu_{\tilde{F}}(x)) | x \in X\} \text{ mit } \mu_{\tilde{A}} : X \to 0, 1 \tag{2.36}$$

So könnte man beispielsweise festlegen, dass die Elemente a.) und e.) in Abbildung 2.5b einen Zugehörigkeitswert von $\mu_{\tilde{A}}(a) = 0.8$ beziehungsweise $\mu_{\tilde{A}}(e) = 0.1$ haben.

Erweiterungsprinzip nach Zadeh

Möchte man zwei Aussagen miteinander verbinden, so bietet die scharfe Logik hierfür mehrere logische Verknüpfungen. Unter Zuhilfenahme der charakteristischen Funktion $\chi(x)$ schaut man sich nun zunächst die Verknüpfungen „und“ und „oder“ an. Möchte man wissen, ob der Ausdruck „a und b“ wahr ist, so lässt sich dies mit der charakteristischen Funktion überprüfen:

$$\chi(a \wedge b) = \begin{cases} 1, & \text{wenn a wahr und b wahr sind} \\ 0, & \text{sonst} \end{cases} \tag{2.37}$$

$$\chi(a \vee b) = \begin{cases} 1, & \text{wenn a oder b oder beide wahr sind} \\ 0, & \text{sonst} \end{cases}. \tag{2.38}$$

Übertragen auf den unscharfen Fall ergibt sich:

$$a \wedge b := \min(\mu(a), \mu(b)) \tag{2.39}$$
$$a \vee b := \max(\mu(a), \mu(b). \tag{2.40}$$

Betrachtet man den Fall einer unendlichen Menge von Aussagen a_i, so lässt sich dies verallgemeinern zu:

$$\forall i : a_i = \inf a_i \tag{2.41}$$
$$\exists i : a_i = \sup a_i \tag{2.42}$$

Um das Erweiterungsprinzip nach Zadeh zu motivieren, betrachtet man zunächst, was passiert, wenn man eine scharfe Menge durch eine gegebene Funktion $f : X \to Y$ in den Bildraum Y abbildet (siehe Abbildung 2.6).

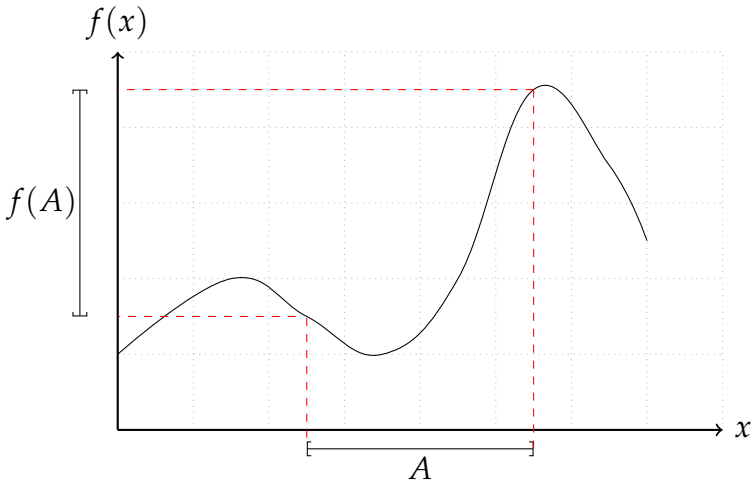


Abbildung 2.6.: Anwendung einer Funktion auf eine scharfe Menge A. Die Menge wird dadurch auf die Bildmenge der Funktion abgebildet.

Betrachtet man dasselbe Problem auf unscharfen Mengen (siehe Abbildung 2.7), so gilt nach dem Erweiterungsprinzip von Zadeh, dass, wenn mehrere Elemente aus dem Urbild auf das gleiche Element z der Bildmenge abgebildet werden, der Zugehörigkeitswert von z zur unscharfen Menge $f(\tilde{A})$ dem maximalen Zugehörigkeitswert unter den Elementen des Urbildes entspricht.

Dies führt zur Ergebnis-Fuzzy-Menge $f(\tilde{A})$ mit folgender Zugehörigkeitsfunktion:

$$\mu_{f(\tilde{A})}(y) = \begin{cases} \sup\limits_{y=f(x)} \mu_a(x), & \text{falls } \exists y = f(x) \\ 0, & \text{sonst} \end{cases} \tag{2.43}$$

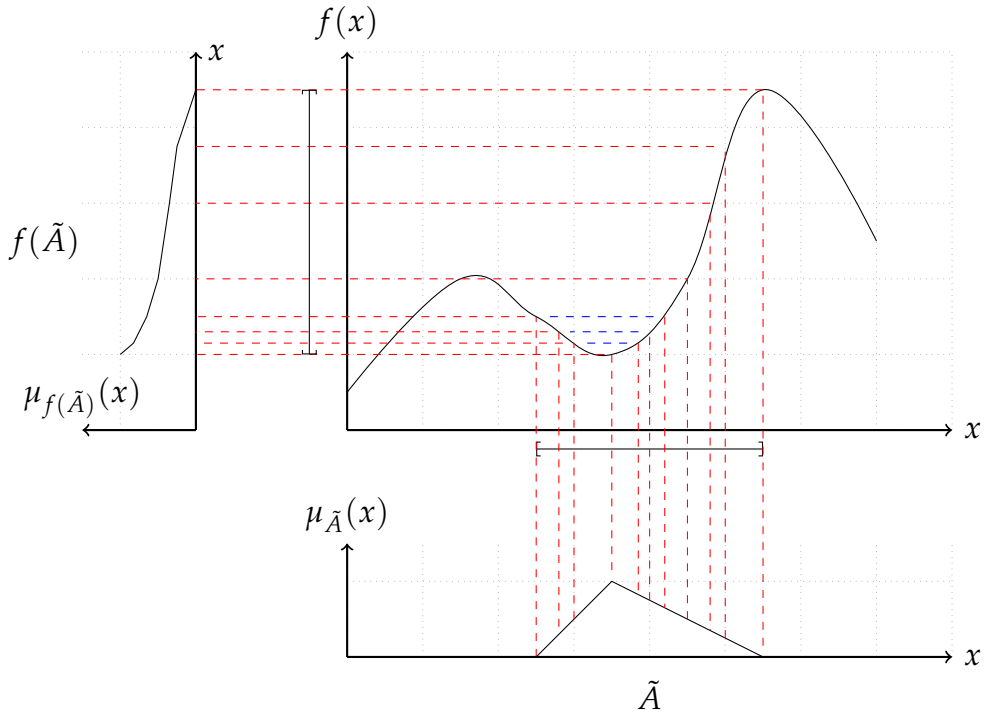


Abbildung 2.7.: Erweiterungsprinzip nach Zadeh. Anwendung einer Funktion auf eine unscharfe Menge $\tilde{A}$. Die Menge wird dadurch auf die Bildmenge der Funktion abgebildet. Die blauen Linien deuten an, dass an diesen Stellen das Supremum aus den Zugehörigkeitswerten gewählt werden muss.

Erweiterungsprinzip für zweistellige Operationen Auch für zweistellige Operationen, wie die Addition oder Subtraktion, kann das Erweiterungsprinzip angewandt werden. Betrachtet man zunächst wiederum den Fall von scharfen Mengen und die zweistellige

Funktion $f : X \times Y \to Z$, der Einfachheit halber hier die Addition $x + y = z$ zweier Mengen, so ergibt sich:

$$A + B = \{y | (\exists a)(\exists b) y = a + b \wedge a \in A \wedge b \in B\}. \tag{2.44}$$

Überführt man dies auf zwei Fuzzy-Mengen $\tilde{A}$ und $\tilde{B}$, die auf X bzw. Y definiert sind, so lässt sich die Ergebnis-Fuzzy-Menge über die charakteristischen Funktionen angeben. Mit dem Übergang auf die Zugehörigkeitsfunktionen $\mu_{\tilde{A}}$ und $\mu_{\tilde{B}}$ lässt sich eine Vorschrift für die Addition zweier Fuzzy-Mengen herleiten:

$$\mu_{\tilde{A}+\tilde{B}}(z) = \sup_{x,y} \{\chi(y = x + y) \wedge \chi(x) \wedge \chi(b)\} \tag{2.45}$$

$$= \sup_{x,y:z=x+y} \{\min(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(y))\}. \tag{2.46}$$

Eigenschaften von Fuzzy-Mengen

Definition 16: Alpha-Niveau-Menge

Als Alpha-Niveau-Menge, auch α-Cut genannt, bezeichnet man die Teilmenge G_α der Grundmenge G, deren Zugehörigkeit $\mu_{\tilde{A}}$ größer oder gleich $\alpha \in [0, 1]$ ist. Es handelt sich dabei um eine scharfe Menge, für deren einzelnen Elemente $x \in G$ folglich gelten muss:

$$x \in G_\alpha \quad | \quad x : \mu_{\tilde{A}}(x) \geq \alpha \tag{2.47}$$

Definition 17: Träger einer Fuzzy-Menge

Die Stützmenge, im englischen „support“ einer Fuzzy-Menge $\tilde{A}$ ist definiert durch $\mu_{\tilde{A}} \neq 0$. Sie gibt folglich die Teilmenge der Grundmenge G an, die zu einem Grad > 0 der Fuzzy-Menge angehört. Mathematisch lässt sich dies ebenso definieren als:

$$supp(\tilde{A}) := \{x | \mu_{\tilde{A}} > 0, x \in X\} \subseteq X \tag{2.48}$$

Fuzzy-Zahlen und deren Arithmetik

Ausgehend von der Möglichkeit, unscharfe Mengen zu beschreiben, bieten sich Möglichkeiten bei der Modellierung von Unsicherheiten. Häufig werden mit Vagheit behaftete Aussagen getroffen, etwa bei Aussagen und Meinungen von Experten. Diese Aussagen müssen formal in einem mathematischen Rahmen beschrieben werden können. Dazu können Fuzzy-Zahlen als Spezialfall einer Fuzzy-Menge verwendet werden. So lässt sich beispielsweise eine Aussage der Art „die Operation dauert ungefähr 4 Minuten“ so formulieren, dass sie für spätere Berechnungen weiterverwendet werden kann.

Definition 18: Fuzzy-Zahl

Eine Fuzzy-Zahl $\tilde{A}$ ist eine normalisierte, konvexe Fuzzy-Menge über dem Raum der reellen Zahlen. Das bedeutet, es muss gelten:

- $\mu_{\tilde{A}}(x) = 1$ für genau ein $x \in \mathbb{R}$
- $\mu_{\tilde{A}}$ muss stückweise stetig sein.

Dreieck-Fuzzy-Zahlen Eine Dreick-Fuzzy-Zahl lässt sich durch das Tupel $\tilde{A} = (m, \alpha, \beta)$ oder über $\tilde{A} =< c_1, m, c_2 >$ mit $c_1 = m - \alpha$ und $c_2 = m + \beta$ beschrieben. Für die Addition und Subtraktion zweier Dreieck-Fuzzy-Zahlen ergeben sich folgende Regeln:

$$\tilde{A} + \tilde{B} = (m^{\tilde{A}} + m^{\tilde{B}}, \alpha^{\tilde{A}} + \alpha^{\tilde{B}}, \beta^{\tilde{A}} + \beta^{\tilde{B}}) \tag{2.49}$$

$$\tilde{A} - \tilde{B} = (m^{\tilde{A}} - m^{\tilde{B}}, \alpha^{\tilde{A}} + \alpha^{\tilde{B}}, \beta^{\tilde{A}} + \beta^{\tilde{B}}) \tag{2.50}$$

Bei der Addition und der Subtraktion addiert sich die Unschärfe, die sich durch den α- und den β-Bereich der beiden Operatoren ergibt, auf. Für die Multiplikation und die Division gibt es verschiedene Näherungsformeln. Auf diese soll hier allerdings nicht weiter eingegangen werden, da diese Operatoren im Rahmen dieser Arbeit nicht verwendet werden. Abbildung 2.8 zeigt eine Darstellung einer Dreieck-Fuzzy-Zahl.

Vier-Punkt-Fuzzy-Intervalle Neben Fuzzy-Zahlen lassen sich auch Fuzzy-Intervalle definieren. Ein Vier-Punkt-Fuzzy-Intervall lässt sich durch $\tilde{A} =< c_1, m_1, m_2, c_2 >$ beschreiben. In der Literatur werden Fuzzy-Intervalle auch häufig als Trapez-Fuzzy-Zahlen bezeichnet (siehe Bansal (2011)). Formal gesehen beschreiben Intervalle keine Fuzzy-Zahlen, da für mehr als ein $x \in \mathbb{R}$ der Zugehörigkeitswert bei 1.0 liegt. Für nicht negative Vier-Punkt-Fuzzy-Intervalle lassen sich die Grundrechenoperationen Addition und Subtraktion wie folgt definieren:

$$\tilde{A} + \tilde{B} =< c_1^{\tilde{A}} + c_1^{\tilde{B}}, m_1^{\tilde{A}} + m_1^{\tilde{B}}, m_2^{\tilde{A}} + m_2^{\tilde{B}}, c_2^{\tilde{A}} + c_2^{\tilde{B}} > \tag{2.51}$$

$$\tilde{A} - \tilde{B} =< c_1^{\tilde{A}} - c_2^{\tilde{B}}, m_1^{\tilde{A}} - m_2^{\tilde{B}}, m_2^{\tilde{A}} - m_1^{\tilde{B}}, c_2^{\tilde{A}} - c_1^{\tilde{B}} > \tag{2.52}$$

Für die Multiplikation und die Division gibt es wiederum Näherungsformeln. Abbildung 2.9 zeigt eine Darstellung eines Vier-Punkt-Fuzzy-Intervalls.

Sechs-Punkt-Fuzzy-Intervalle Bei einem Sechs-Punkt-Fuzzy-Intervall wird zusätzlich zu den markanten Punkten bei einem Vier-Punkt-Fuzzy-Intervall noch ein weiterer α-Cut beim Niveau λ ausgezeichnet. Es ergibt sich das Tupel $\tilde{A} =< c_1, c_{1\lambda}, m_1, m_2, c_{2\lambda}, c_2 >$. Abbildung 2.10 zeigt eine exemplarische Darstellung.

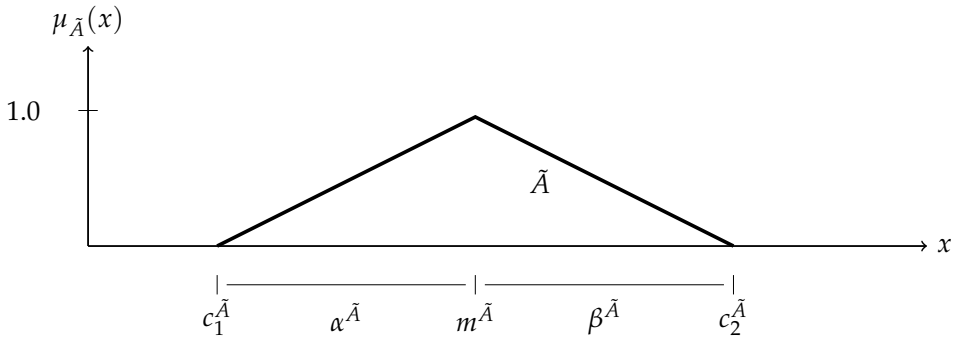


Abbildung 2.8.: Darstellung einer Dreieck-Fuzzy-Zahl

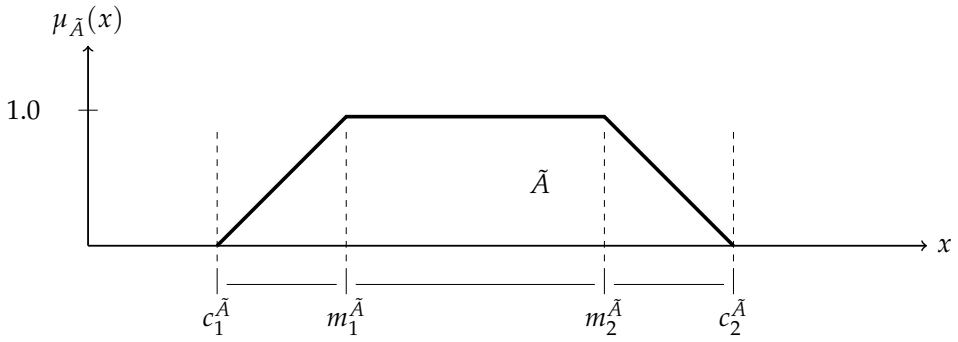


Abbildung 2.9.: Darstellung eines Vier-Punkt-Fuzzy-Intervalls

Verallgemeinerung mittels diskreter Fuzzy-Zahlen Die zuvor beschriebenen Fälle lassen sich durch eine Verallgemeinerung beschreiben. So ist es beim Einsatz von

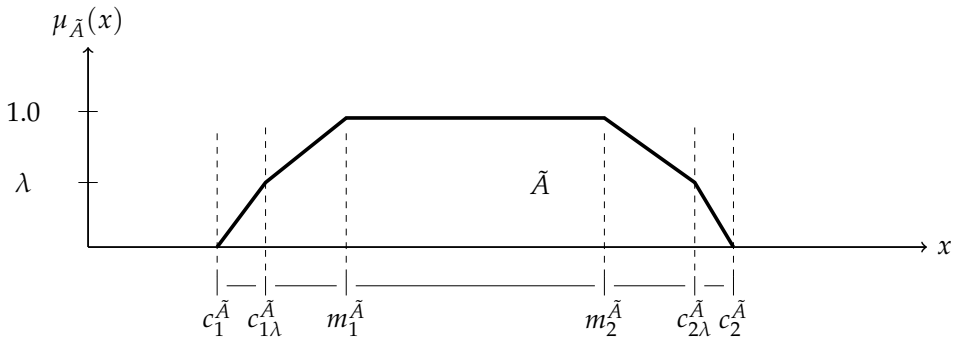


Abbildung 2.10.: Darstellung eines Sechs-Punkt-Fuzzy-Intervalls

diskreten Fuzzy-Zahlen möglich, beliebig viele Niveaus anzugeben. Hierzu werden für jedes Niveau die Alpha-Cuts explizit angegeben. Die Rechenregeln für Addition und Subtraktion ergeben sich nun aus der Anwendung der Intervall-Arithmetik auf die einzelnen Alpha-Cuts.

2.2.3. Vergleich zwischen Stochastik und Fuzzy-Set-Theorie

Sowohl die Methoden der Stochastik als auch die der *Fuzzy-Set*-Theorie lassen sich nutzen, um Unsicherheiten zu modellieren. Im Folgenden werden an einigen exemplarischen Eigenschaften Unterschiede der beiden Methoden hervorgehoben, um später daraus Schlüsse für die Verwendung der Methoden zu ziehen.

Mathematische Struktur und Semantik

Untersucht man die mathematische Struktur beider Methoden, so betrachtet man bei der Stochastik zunächst die *Kolmogorov'schen Axiome*.

Ein Wahrscheinlichkeitsmaß P, das jedem Ereignis A der Ergebnismenge Ω eine Wahrscheinlichkeit zuweist, muss die *Kolmogorov'schen Axiome* erfüllen. Hierzu gilt für Teilmengen die Ereignisse $A_1, A_2 \in X$

- $P(A_1) \geq 0$
- $P(\Omega) = 1$
- $P(\emptyset) = 0$
- $A_1 \cap A_2 = \emptyset \Rightarrow P(A_1 \cup A_2) = P(A_1) + P(A_2)$.

Diese Axiome bedeuten, dass jedes Ereignis eine Wahrscheinlichkeit größer gleich 0 haben muss. Das sichere Ereignis Ω hat die Wahrscheinlichkeit 1. Die Wahrscheinlichkeit,

dass kein Ereignis eintritt, liegt bei 0. Das letzte Axiom besagt, dass die Wahrscheinlichkeit der Vereinigung zweier oder mehrerer Ereignisse, die paarweise disjunkt sind, sich einfach als Summe der Einzelwahrscheinlichkeiten ergibt.

Im späteren Verlauf der Arbeit wird der Betrieb auf einem Containerterminal genauer betrachtet. Als Motivationsbeispiel dient daher hier die Modellierung von Zugverspätungen. Auf der scharfen endlichen Menge X aller Zugankünfte werden mehrere Teilmengen definiert, die wie folgt lauten:

- A_1 Menge aller Züge, die pünktlich eintreffen.
- A_2 Menge aller Züge, die fast pünktlich eintreffen.
- A_3 Menge aller Züge, die verspätet eintreffen.
- A_4 Menge aller Züge, die deutlich verspätet eintreffen.

Zunächst soll nicht weiter festgelegt sein, was „punktlich“, „fast pünktlich“, „verspätet“ und „deutlich verspätet“ bedeutet. Auf der Menge X lässt sich ein Wahrscheinlichkeitsmaß P definieren, das jeder Teilmenge eine Zahl $P(A) \in [0,1]$ zuordnet, die man als Wahrscheinlichkeit interpretieren kann.

Das Wahrscheinlichkeitsmaß gibt an, mit welcher Wahrscheinlichkeit eine Variable $x \in X$, sprich ein Zug in einer der *scharfen* Teilmengen liegt. Liegt die Wahrscheinlichkeit für einen Zug, der deutlich verspätet ist, bei $P(A_4) = 0.05$, so impliziert diese Aussage, dass der Zug entweder pünktlich, fast pünktlich, verspätet oder deutlich verspätet sein kann und die Wahrscheinlichkeit, dass er deutlich verspätet ist, bei 5 % liegt. Die Tatsache, dass der Zug nur einer Teilmenge zugehörig ist und dies mit einer bestimmten Wahrscheinlichkeit, wird auch „Gesetz der ausgeschlossenen Mitte“ genannt.

Innerhalb der Fuzzy-Theorie lassen sich auf der scharfen, endlichen Menge X unscharfe Mengen $\tilde{A}_i$ definieren, die durch eine Zugehörigkeitsfunktion $\mu_x : X \to [0,1]$ beschrieben werden, die den Grad der Zugehörigkeit für ein x der Menge X zu einer unscharfen Menge $\tilde{A}_i$ festlegt. Auf das obige Beispiel bezogen, bedeutet ein Zugehörigkeitswert von $\mu_x(\tilde{A}_4) = 0.8$, dass der Zug zu einem Grad von 0.8 zur unscharfen Menge der deutlich verspäteten Züge gehört. Ebenso kann er zu einem anderen Zugehörigkeitsgrad auch zu einer der anderen Teilmengen gehören.

Die scharfe Abgrenzung wird ebenso deutlich, wenn man festlegen möchte, was „punktlich“, „fast pünktlich“, „verspätet“ und „deutlich verspätet“ bedeutet. In der Theorie der Stochastik müssen alle Teilmengen scharf definiert werden, beispielsweise wie folgt:

- Zugverspätung liegt im Intervall von $[0,5]$ min $\to x \in A_1$
- Zugverspätung liegt im Intervall von $[5,10]$ min $\to x \in A_2$
- Zugverspätung liegt im Intervall von $[10,30]$ min $\to x \in A_3$
- Zugverspätung liegt im Intervall von $[30,\infty]$ min $\to x \in A_4$

Nun stellt sich die Frage, warum ein Unterschied bei der Zuordnung zu den Teilmengen zwischen einem Zug, der eine Verspätung von $4:59$ Minuten hat und einem anderen,

der eine Verspätung von 5 : 01 Minuten hat, gemacht werden soll. Genau hier setzt die *Fuzzy-Set*-Theorie an, die es ermöglicht, dass ein Zug zu einem gewissen Grad zu beiden Teilmengen gehören kann.

Unterschiede der mathematischen Operationen

Ein weiterer relevanter Unterschied beider Theorien liegt in der Arbeitsweise der mathematischen Operationen begründet. Die Summe aus zwei Zufallsvariablen wird über ein Faltungsintegral bestimmt. Die Summe zweier unscharfer Zahlen wird hingegen mit dem Erweiterungsprinzip bestimmt. Abbildung 2.11 zeigt das Ergebnis einer Addition zweier gleichverteilter Zufallsvariablen. Die resultierende Wahrscheinlichkeitsdichte ist nicht gleichverteilt, sie nimmt erst zu und fällt daraufhin wieder ab. Werte im mittleren Bereich sind somit häufiger als Randwerte. Dies lässt sich beispielsweise beim Würfeln mit zwei Würfeln beobachten. In Summe eine Augenzahl von zwei oder zwölf zu würfeln ist unwahrscheinlicher als eine Augenzahl von sechs.

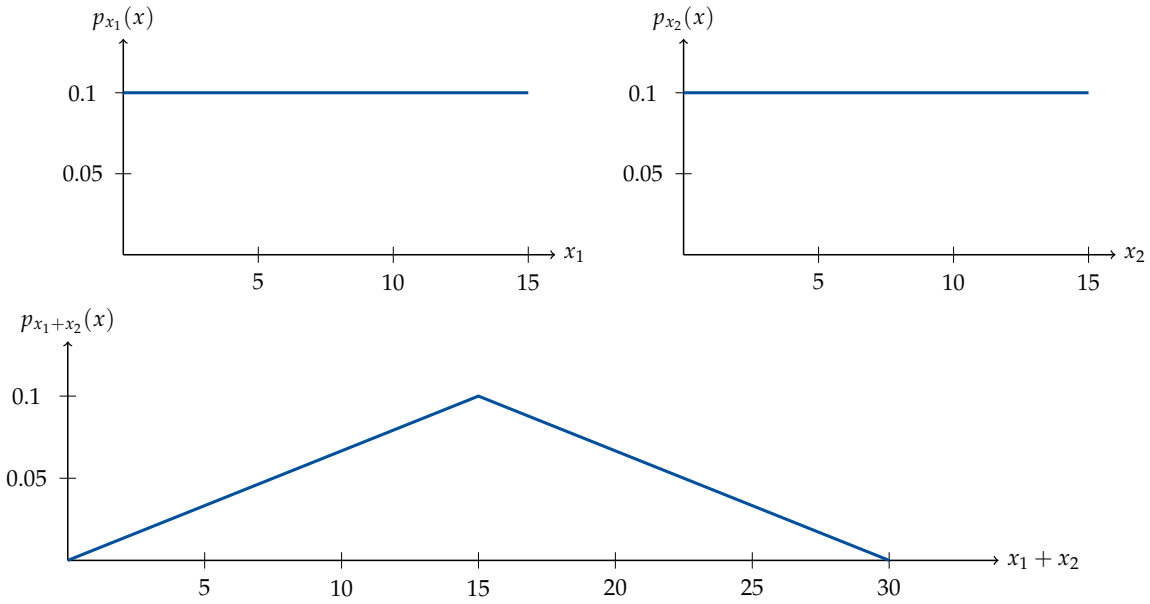


Abbildung 2.11.: Addition zweier Zufallsvariablen. Das Beispiel ist angelehnt an Schulz (2002)

Betrachtet man die Addition zweier Fuzzy-Intervalle mit konstanter Zugehörigkeitsfunktion (siehe Abbildung 2.12), ergibt sich wiederum ein Fuzzy-Intervall mit konstanter Zugehörigkeitsfunktion. Während man bei einer stochastischen Modellierung bei einer Gleichverteilung davon spricht, dass alle Werte gleich wahrscheinlich sind, spricht man bei einer Modellierung mittels der *Fuzzy-Set*-Theorie vielmehr davon, dass alle Werte die gleiche Möglichkeit beziehungsweise Glaubwürdigkeit haben. Informationen darüber, wie häufig einzelne Ereignisse eintreten, werden bei der *Fuzzy-Set*-Theorie nicht berücksichtigt. Zusammenfassend lässt sich festhalten, dass eine Wahrscheinlichkeitsverteilung

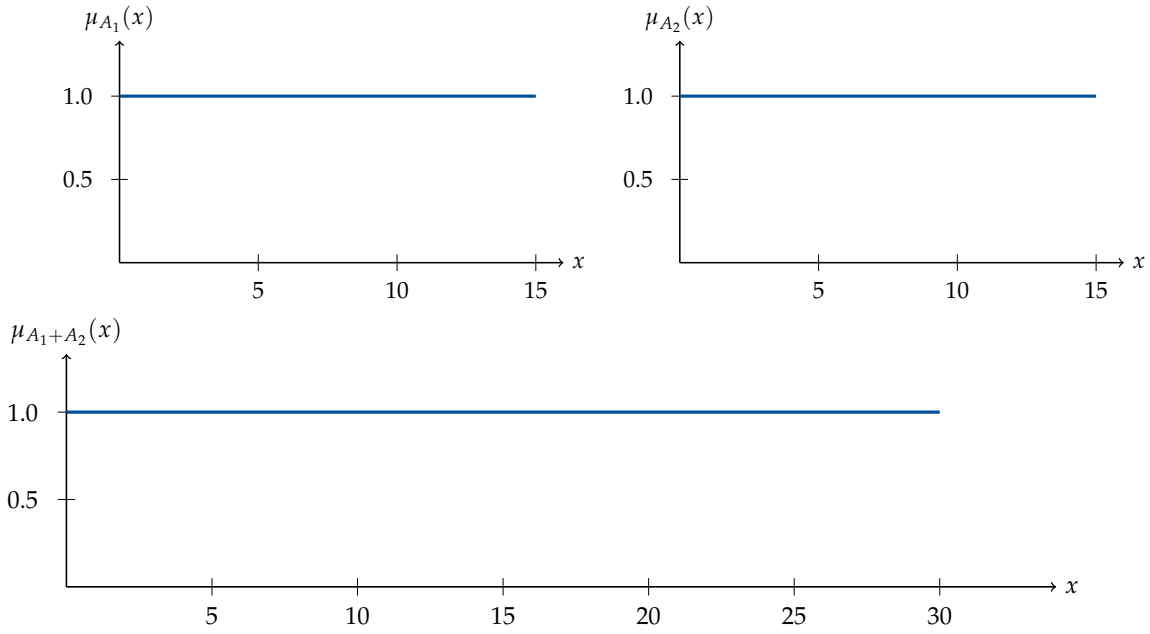


Abbildung 2.12.: Addition zweier unscharfer Zahlen. Das Beispiel ist angelehnt an Schulz (2002)

konkretere Informationen über die betrachtete Größe beschreibt als eine Modellierung über eine Zugehörigkeitsfunktion.

Kombination von stochastischen Methoden und der Fuzzy-Set-Theorie

Bei realen Problemen treten zumeist sowohl aleatorische als auch epistemische Arten von Unsicherheiten gleichzeitig auf (siehe Abschnitt 2.1.7). Um beide Unsicherheiten kombiniert zu behandeln wurden in der Literatur diverse Konzepte vorgestellt. Negoita u. a. (1978) formulierten im *Consistency Principle* die formalen Zusammenhänge zwischen Möglichkeit und Wahrscheinlichkeit. Die Grundaussage des Prinzips besteht darin, dass Ereignisse, die möglich sind, nicht unbedingt wahrscheinlich sein müssen. Umgekehrt müssen Ereignisse, die wahrscheinlich sind, auch möglich sein und Ereignisse, die unwahrscheinlich sind, können allerdings durchaus möglich sein.

Darauf aufbauend lassen sich Verfahren ableiten, die es ermöglichen, aus stochastischen Informationen einzelner Parameter Zugehörigkeitsfunktionen abzuleiten. Weiterhin gibt es Verfahren wie Fuzzy-Zufallszahlen, die beide Theorien vereinen (siehe Guang-yuan u. Zhong (1993), Liu u. Liu (2002), Möller u. Beer (2004)). Somit können aleatorische und epistemische Unsicherheiten gemeinsam behandelt werden.

2.3. Behandlung von Unsicherheiten bei der Ablaufplanung

Die Ablaufplanung unterliegt in der Ausführung den in Abschnitt 2.1.7 beschriebenen Unsicherheiten. In der Literatur werden verschiedene Methoden vorgestellt, um effizient und adäquat mit Unsicherheiten während der Erstellung von Ablaufplänen umzugehen. In diesem Abschnitt erfolgt zunächst eine Definition der grundlegenden Begrifflichkeiten, um darauf aufbauend den Stand der momentanen Forschung darzustellen.

Pläne lassen sich bezüglich ihrer Eigenschaften bei tatsächlicher Ausführung unterscheiden. Um diese Kategorisierung vornehmen zu können, werden zunächst einige Definitionen benötigt. Diese Definitionen sind an Bidot (2005) angelehnt, der einen guten Überblick gibt.

Definition 19: Stabilität

Ein Plan ist umso stabiler, je weniger Änderungen während der Ausführung an selbigen vorgenommen werden müssen. Bei einem stabilen Plan müssen folglich einmal vorgenommene Entscheidungen über die Reihenfolge und Ausführungszeit einer Operation während der Ausführung des Plans nicht oder nur zu einem sehr geringen Grad angepasst werden. Reaktive Pläne (vgl. 2.3.1), die während der Ausführung auf externe Einflüsse und Randbedingungen reagieren, sind naturgemäß nicht stabil.

Definition 20: Flexibilität

Die Flexibilität eines Plans bezeichnet die Möglichkeit, Operationen innerhalb des Plans tauschen zu können oder anderen Ressourcen zuzuordnen. Bei einem flexiblen Plan sind noch nicht alle Entscheidungen getroffen worden. Einige Entscheidungen werden erst während der Ausführung getroffen. Ein flexibler Plan ermöglicht folglich mehr Reaktionsmöglichkeiten als ein unflexibler Plan.

Definition 21: Robustheit

Ein robuster Plan ist ein Plan, dessen Qualität nicht oder nur in geringem Maße abnimmt, wenn schlecht voraussagbare Ereignisse während der Ausführung des Plans auftreten.

Anschaulich könnte man meinen, dass jeder flexible Plan gleichzeitig robust sein muss, dies ist allerdings nicht immer der Fall. Nimmt man beispielsweise zwei Pläne s_1 und s_2, wobei s_1 bezüglich der Startzeiten der einzelnen Operationen flexibler als s_2 ist, s_1 allerdings zwei kritische Pfade hat und s_2 lediglich einen, ist es möglich, dass der Plan s_1 eine größere Anfälligkeit gegenüber Störungen besitzt als s_2. Robustheit muss problemspezifisch definiert werden in Bezug zu anderen Kriterien und den auftretenden Störungen. Robustheit kann unter anderem durch Flexibilität und bzw. oder Stabilität gewährleistet werden.

Definition 22: Prädiktiver Ablaufplan (engl.:predictive schedule)

Bei einem prädiktiver Ablaufplan sind alle Entscheidungen vor Projektstart getroffen wurden. Häufig wird auch der Begriff *baseline scheduling* für dieses Vorgehen verwendet.

Definition 23: Reaktiver Ablaufplan (engl. reactive schedule)

Bei einem reaktiven Ablaufplan werden möglichst viele Entscheidungen zum spät möglichsten Zeitpunkt getroffen.

Definition 24: Offline-Ablaufplanung

Offline-Ablaufplanung führt zu einem statischen vorausschauenden Ablaufplan. Die Erzeugung eines Planes erfolgt vor Beginn der Ausführung und dient beratend bei der Entscheidungsfindung.

Definition 25: Online-Ablaufplanung

Online-Ablaufplanung erfolgt gleichzeitig mit der physikalischen Ausführung der zu planenden Prozesse. Die Ablaufplanung erfolgt inkrementell bis zu einem festgelegten Horizont unter Berücksichtigung der aktuellen Ressourcenauslastung. Ein solcher Ablaufplan kann nicht nur beratend vor Beginn der Ausführung des Projekts dienen, sondern auch im Verlauf des Projekts beratend unterstützen, indem die Planung auf Ereignisse während der Ausführung reagiert.

Sowohl beim Offline- als auch bei Online-Scheduling-Verfahren werden Entscheidungen getroffen, bei denen in einem gewissen Rahmen Vorhersagen benötigt werden, welche Situation während der Ausführung der Operationen eintritt. Diese Vorhersagen unterliegen gewissen Unsicherheiten. Mehrere Autoren bieten einen ausführlichen

Überblick über die verbreiteten Methoden zur Behandlung dieser Unsicherheiten bei der Ablaufplanung (siehe z.B. Li u. Ierapetritou (2008) oder Herroelen u. Leus (2005)). Grundsätzlich lässt sich in Methoden der *reaktiven Ablaufplanung*, der *stochastischen Ablaufplanung*, der *robusten Ablaufplanung* und Methoden der *Fuzzy-Set*-Theorie bei der Ablaufplanung unterscheiden.

2.3.1. Reaktive Ablaufplanung

Bei der reaktiven Ablaufplanung wird eine Start-Ablaufplanung unter deterministischen Annahmen vorgenommen. Dieser Start-Ablaufplan wird aktualisiert, wenn ein unerwartetes Ereignis auftreten sollte, das ein Eingreifen erfordert. Der Start-Ablaufplan hat die Eigenschaften eines *predictive schedule*s. Durch die Aktualisierung des Ablaufplans bekommt der Ablaufplan einen reaktiven Charakter.

Es gibt viele verschiedene Verfahren, um auf unerwartet eintretende Ereignisse zu reagieren. Zunächst sind Verfahren zu nennen, die lediglich auf kleinen Zeitfenstern arbeiten oder kleine Änderungen am ursprünglichen *predicitve schedule* vornehmen (siehe z.B. Sadeh u. a. (1993)). Andere Ansätze führen eine komplette Neuberechnung durch.

Zur Generierung von reaktiven Ablaufplänen bieten sich stabile und/oder flexible Ausgangspläne an. Um die Flexibilität zu erhöhen, setzten viele Ablaufplanungsverfahren auf sogenannte *Least-Commitment*-Strategien (vgl. Sacerdoti (1975), Draper u. a. (1994) und Keng u. Y. Y. Yun (1989)).

Definition 26: Least-Commitment-Strategien

Least-Commitment-Strategien repräsentieren einen Ablaufplan in einer möglichst flexiblen Form, die es erlaubt, konkrete Entscheidungen erst zum Ausführungszeitpunkt zu treffen. Lediglich die essentielle Ordnung der Operationen ist vorgeschrieben.

Eine oft genutzte Möglichkeit der Beschreibung von losen Operationsfolgen ist die Verwendung von sogenannten Prioritätsregeln, die im englischen Sprachgebrauch *dispatching rules* genannten werden. Eine Beschreibung eines Ablaufplans durch eine Folge von Prioritätsregeln benötigt eine Auswertung der Regeln und eine Übersetzung der Beschreibung in eine Folge von Operationen spätestens zum Ausführungszeitpunkt. Gibt eine Prioritätsregel z.B. die Sortierung nach der Prozessdauer an, so lässt sich dieses online zum Entscheidungszeitpunkt auswerten. Ein konkreter Ablaufplan kann somit inkrementell zur physischen Ausführung erstellt werden.

Im Rahmen eines *Rolling-Horizon*-Ansatzes (siehe Li u. Ierapetritou (2010), Ovacik u. Uzsoy (1995)), werden die Anpassungen an einen Ablaufplan unter Berücksichtigung

möglicher zukünftiger Verletzungen von Nebenbedingungen innerhalb eines Zeitfensters vom aktuellen Betrachtungspunkt ausgehend getroffen.

Méndez u. a. (2006) untersuchten verschiedene Möglichkeiten, einen bestehenden Ablaufplan anzupassen, wie beispielsweise das Anpassen der Startzeiten von Operationen, eine lokale Umsortierung von Operationen, sowie ein Wechsel in der Ressourcenzuordnung.

2.3.2. Stochastische Ablaufplanung

Eine der am weitesten verbreiteten Verfahren zur Handhabung von Unsicherheiten in der Ablaufplanung ist deren Modellierung mittels stochastischer Variablen. Im englischen Sprachgebrauch wird hierbei oft der Begriff *stochastic programming* verwendet. Einen allgemeinen Überblick liefern Birge u. Louveaux (2011). Stochastische Methoden versuchen einen Ablaufplan dahingehend zu optimieren, dass er unter den durchschnittlich auftretenden Störeinflüssen, wie etwa Verspätungen, optimal ist.

Die zu erwartende Realisierung der Optimierungsgrößen wird unter Berücksichtigung der entsprechenden Variablen optimiert. So lassen sich zum Beispiel die Dauern von Operationen als Zufallsvektor beschreiben.

Stochastische Verfahren zur Ablaufplanung lassen sich in zwei- und mehrstufige Verfahren unterscheiden. Die Kernidee eines zweistufigen Verfahrens ist es, dass Entscheidungen auf vorhandenen Daten zum Zeitpunkt der Entscheidung getroffen werden und nicht von zukünftigen Beobachtungen abhängen. Hierzu wird zunächst das Optimierungsproblem, beispielsweise

$$\min f(x) = c^T x \tag{2.53}$$

unter den Nebenbedingungen

$$Ax \leq b \tag{2.54}$$
$$x \geq 0 \tag{2.55}$$

um eine Zufallsvariable ω erweitert.

Man unterteilt die Zielfunktionen und Nebenbedingungen in zwei Teile mit verschiedenen Parametersätzen. Der erste Parametersatz auf der ersten Stufe c, A, B, x beschreibt die Parameter, die vor der Realisierung der unsicheren Daten bekannt sind. Der Vektor x beschreibt die Ausgangsentscheidung, die zu Beginn des Planungszeitraums unter Unsicherheit bestimmt werden muss.

Auf der zweiten Stufe ist eine Realisierung der unsicheren Daten, sprich der Zufallsvariable ω, bekannt. Es lässt sich ein Parametersatz q, T, W, h, y einführen, der die zufallsbehafteten Kosten q und die stochastischen Nebenbedingungen mittels T, W und H beschreibt. Der Vektor y ist sowohl durch die Ausgangsentscheidung x, als auch durch ω beeinflusst und wird als Rekurs-Vektor bezeichnet.

Das Problem lässt sich formulieren als:

$$min f(x, \omega) = c^T x + q(\omega)^T y(\omega) \tag{2.56}$$

unter den Nebenbedingungen

$$Ax = b \tag{2.57}$$
$$T(\omega)x + Wy(\omega)) = h(\omega) \tag{2.58}$$
$$x, y(\omega) \geq 0 \tag{2.59}$$

In der ersten Stufe werden die Kosten der ersten Stufe $c^T x$ sowie die zu erwartenden Kosten der zweiten Stufe optimiert. Die zweite Stufe kann als einfaches Optimierungsproblem aufgefasst werden, welches das vermeintlich optimale Verhalten beschreibt, wenn die Realisierung bekannt ist. Dieses Optimierungsproblem kann als Minimierung der Kosten $q(\omega)^T y$ betrachtet werden, die durch das Ausgleichen von Abweichungen zur Ausgangsentscheidung entstehen.

Ziel ist es, eine Einplanstrategie zu finden, die für fast alle möglichen Realisierungen der Zufallsvariable den zu erwartenden Wert einer Funktion über die Entscheidungen maximiert. Stork (2001) stellt hierfür eine Auswahl von Einplanstrategien vor.

Die stochastische Ablaufplanung wird in der Literatur von vielen Autoren und auf diverse Problemstellungen angewandt. Bonfill u. a. (2004) benutzen dieses Verfahren beispielsweise, um Risiken bei der Kurzzeitplanung von Anlagen mit mehreren verschiedenen Produktchargen unter unsicherem Bedarf zu optimieren. Till u. a. (2007) verwenden das Verfahren bei der Ablaufplanung in der chemischen Industrie. Zur Lösung des Problems setzen sie in der ersten Stufe einen evolutionären Algorithmus ein, wobei auf der zweiten Stufe ein gemischt-ganzzahliges Programm aufgestellt und gelöst wird.

Auf gleicher Basis lassen sich auch mehrstufige Verfahren formulieren. Balasubramanian u. Grossmann (2004) entwickelten ein mehrstufiges Modell, wobei bestimmte Entscheidungen unabhängig von der Realisierung der unsicheren Parameter getroffen werden und andere anhand der Realisierung der Zufallsvariable.

2.3.3. Robuste Ablaufplanung

Ziel von robuster Ablaufplanung ist es, die Auswirkungen von Störungen auf die Bewertung eines Ablaufplans zu minimieren und somit einen robusten Ablaufplan zu erhalten. Um dies zu gewährleisten, wird versucht, dass sich der reale Ablaufplan möglichst wenig vom ursprünglichen Ablaufplan unterscheidet.

Einen guten Überblick über die verschiedenen Definitionen des Begriffs Robustheit und der daran angelehnten robusten Ablaufplanung stellen Herroelen u. Leus (2005) zusammen. Davenport u. a. (2014) bezeichnen einen robusten Ablaufplan als „Ablaufplan, der es ermöglicht, einen gewissen Grad von unerwarteten Ereignissen zu absorbieren, ohne dass eine Neuberechnung nötigt ist“. Leon u. a. (1994) definieren einen robusten Ablaufplan als „ein *a-priori offline schedule*, der eine hohe Performance [bezüglich einer gegebenen Zielfunktion] beim Vorhandensein von Störungen gewährleistet“. Kouvelis u. a. (2000) fassen robuste Ablaufplanung als „die Bestimmung eines Ablaufplans, dessen Performance relativ unempfindlich gegenüber den potentiellen Realisierungen der Operationsdauern der einzelnen Jobs ist“ auf.

Mulvey u. a. (1995) entwickelten das Konzept der Robusten Optimierung. Dabei unterscheiden sie zwischen Lösungsrobustheit und Modellrobustheit. So definieren sie verschiedenen Szenarien, die beschreiben, welche Realisierungen verschiedene unsichere Parameter annehmen können. Eine Lösung wird als lösungsrobust bezeichnet, falls sie für alle möglichen Szenarien, die durch Störungen eintreten können, nahe am Optimum bleibt. Modellrobustheit beschreibt die Tatsache, dass die Lösung für die meisten Szenarien gültig bleibt. Das eigentliche Verfahren besteht darin, dass das Originalproblem dahingehend umformuliert wird, dass eine gegenüber Unsicherheiten robuste Lösung gefunden wird.

Stochastische Ablaufplanung optimiert nur das erste Moment der Verteilung der unsicheren Parameter und vernachlässigt höhere Momente sowie die Präferenz des Entscheidungsträgers ein Risiko einzugehen (siehe Bidot (2005)). Robuste Optimierung hingegen kann auch höhere Momente, wie etwa die Varianz, minimieren. Weiterhin benötigt man lediglich eine deterministische Menge von Szenarien, die unter Unsicherheit eintreten können, anstelle einer zumeist schwer zu erzeugenden Verteilungsfunktion.

Robuste Optimierung wurde in vielen Anwendungen eingesetzt, beispielsweise beim *Supply-Chain*-Planungsproblem der Fertigung, Montage und des Vertriebs (Escudero u. a. (1999)), bei dem sowohl der Produktbedarf, als auch die Lieferkosten und die Lieferzeit Unsicherheiten unterliegen. Bertsimas u. a. (2013) verwendeten die Robuste Optimierung für das sogenannte *Unit-Commitment*-Entscheidungsproblem, bei dem in einem Stromversorgungsnetz die Kosten für das Hoch- und Herunterfahren von Kraftwerken sowie die Kosten für das Befördern der Strommengen minimiert werden sollen. Dabei gehen sie von einem prognostizierten Systemverbrauch unter Berücksichtigung

verschiedener physikalischer und zeitlicher Einschränkungen für die Erzeugung von benötigten Ressourcen aus.

2.3.4. Sensitivitätsanalyse

Die Sensitivitätsanalyse ist ein Ansatz, um auf Basis von gewonnen Informationen auf Unsicherheiten zu reagieren. Es wird die Empfindlichkeit einer Lösung analysiert, wenn man Änderungen an den Eingangsdaten des Problems vornimmt. Eines der grundsätzlichen Probleme der Sensitivitätsanalyse ist es, dass die Unsicherheiten nicht in die Problemstellung aufgenommen werden sondern separate nach der Optimierung des Problems behandelt werden.

2.3.5. Fuzzy-Ablaufplanung

Auf Basis der *Fuzzy-Set*-Theorie nach Zadeh (1965) wurden Algorithmen zur Behandlung von Unsicherheiten bei der Ablaufplanung entwickelt. Grundsätzlich lassen sich zwei verschiedene Schwerpunkte bei der Modellierung von Unsicherheiten mittels der *Fuzzy-Set*-Theorie in der Literatur feststellen. Dazu zählt zunächst die Ablaufplanung unter flexiblen Nebenbedingungen und andererseits die Ablaufplanung unter unvollständigen oder unpräzisen Informationen.

Methoden der Stochastik können nur konsequent angewandt werden, wenn die Eingangsgrößen zufälliger Natur sind und genügend Informationen zur Modellierung der Wahrscheinlichkeitsdichtefunktion vorhanden sind. Die Ablaufplanung unter unvollständiger Informationslage mittels nicht-probabilistischer Methoden, wie der Intervallanalyse, der Möglichkeitstheorie sowie der Fuzzy-Set-Theorie war daher in den letzten Jahrzehnten Gegenstand der Forschung.

Fortemps (1997) entwickelte grundsätzliche Ideen zum Einsatz der *Fuzzy-Set*-Theorie bei der Ablaufplanung und optimiert einen Ablaufplan mittels *Simulated Annealing* in Hinblick auf dessen unscharfe Projektdauer.

Dubois u. Prade (1999) und Dubois u. a. (2003) stellten eine Kombination aus Berücksichtigung von flexiblen Nebenbedingungen und unvollständigen, unpräzisen Informationen vor, die zu einem robusten und fehlertoleranten Ablaufplan führen soll, der alle Nebenbedingungen zu einem gewissen Umfang mit einem hinreichenden Grad der Vertrauenswürdigkeit erfüllt. Eines der Optimierungsziele bei einem Ablaufplanproblem mit Präferenzen ist die Maximierung des gesamtheitlichen Zufriedenheitsgrades. Beispielsweise kann mittels der *Fuzzy-Set*-Theorie das späteste Operationsende einer Operation unscharf formuliert werden, um zu beschreiben, dass eine Fertigstellung zu einem gewissen Zeitpunkt wünschenswert ist, ein späterer Abschluss der Operation allerdings noch annehmbar ist. Teilweise sind die gewünschten Nebenbedingungen

inkompatibel, da der Fall eintreten kann, dass kein Ablaufplan alle gegebenen Nebenbedingungen komplett erfüllen kann. In diesem Fall wird eine Relaxation der Nebenbedingung angestrebt.

Hapke u. a. (1999) entwickelten einen parallelen SGS, der unsichere Verfügbarkeitszeitpunkte und Dauern von Operationen berücksichtigt. Allerdings wird dabei die Auslastung innerhalb der Ressourcen deterministisch angenommen. Masmoudi u. Haït (2013) stellten darauf aufbauend eine auf die Possibilitätstheorie gestützte Modellierung der Auslastung vor und erhalten unscharfe Auslastungsprofile, die sie im Rahmen eines Genetischen Algorithmus glätten.

Die Netzplantechnik wird insbesondere in der Terminplanung von Bauprojekten verwendet und setzt auf den gängigen Konzepten der Graphentheorie auf. Sie wird verwendet, um kritische Aktivitäten, die ein gesamtes Projekt verzögern können, zu identifizieren und eine Abschätzung über das frühste und das späteste Ende eines Projektes zu erlangen. Im Rahmen der *Programm Evaluation and Review Technique* (PERT) werden für jeden Vorgang drei mögliche Zeiten angenommen: eine optimistische, eine wahrscheinliche und eine pessimistische Dauer. Unter Annahme einer Beta-Verteilung wird eine Zufallsvariable definiert. Im weiteren Verlauf wird mit dem Erwartungswert und der Varianz der Dauer jeder einzelnen Operation das Ende des Projektes bestimmt. Freundt (2004) diskutierte die Eignung dieses Verfahrens und verweist darauf, dass das berechnete wahrscheinliche Projektende um bis zu 30 % von der Realität abweicht. Darum erweitert er die Netzplantechnik dahingehend, dass er für die Modellierung der Dauern und der Verfügbarkeitszeiten Fuzzy-Zahlen verwendet. Dabei geht er auf die Problematik ein, dass die Unschärfe, bedingt durch die Vergrößerung des Trägers einer Fuzzy-Zahl durch arithmetische Operatoren, im Laufe der Berechnung steigt. Steigt die Unschärfe zu sehr, so lässt sich schwierig eine Aussage aus den Ergebnissen ableiten. Daher zerlegt er den Ablaufplan an Meilensteinen in Teil-Ablaufpläne.

3. Multiskalen-Algorithmus für reaktive Prozess- und Ablaufplanung

Die Bestimmung eines optimalen Prozess- sowie Ablaufplans für reale Problemstellungen birgt durch seine Anzahl an Kombinationsmöglichkeiten Schwierigkeiten. So müssen, um einen detaillierten Ablaufplan zu bestimmen, alle Operationen unter Berücksichtigung der Nebenbedingungen eingeplant werden. Die Überprüfung dieser Nebenbedingungen kann in realen Anwendungen unter Umständen recht komplex sein, etwa die Vermeidung von Kollisionen zwischen einzelnen beweglichen Ressourcen. Durch die große Anzahl an Kombinationsmöglichkeiten, die durch die Reihenfolge der Operationen und durch die Wahl einer Route pro Job gegeben ist, ergibt sich ein sehr großer Lösungsraum, der untersucht werden muss.

Weiterhin unterliegen reale Probleme in der Praxis vielen äußeren Einflüssen und somit Veränderungen an den Restriktionen des Optimierungsproblems (siehe 2.1.7). Aufgrund der Unsicherheiten in den Restriktionen ist eine detaillierte Optimierung der Prozess- und Ablaufplanung über den gesamten Projektraum nicht zu empfehlen. Die zumeist rechenintensiv berechnete Lösung wird durch kleinere Änderungen an den Restriktionen unzulässig und kann nicht mehr genutzt werden. Einer der Schlüsselaspekte bei realen Projekten ist folglich, dass ein Algorithmus sich dem Systemoptimum aus ganzheitlicher Sicht annähert und dabei die auftretenden Unsicherheiten berücksichtigt.

Um dieser Tatsache Rechnung zu tragen, wird ein Multiskalen-Algorithmus in Anlehnung an Stefansson u. a. (2006) entwickelt, der verschiedene Detailstufen des Problems betrachtet. Im Folgenden wird lediglich von zwei Ebenen ausgegangen, die bei Bedarf allerdings um weitere Zwischenebenen ergänzt werden können.

Auf der makroskopischen Ebene wird eine Optimierung des Prozessplans sowie des Ablaufplans unter Vernachlässigung von exakten Beschreibungen der einzelnen Operationen durchgeführt. Hierfür wird ein Koevolutionärer Algorithmus verwendet (siehe Cai u. Peng (2002) und Smith (2007)). Die Optimierung auf der mikroskopischen Ebene erfolgt iterativ lediglich auf einem kleinen Zeitfenster. Auf dieser Ebene wird nur die Operationenreihenfolge optimiert, der Prozessplan wird nicht verändert. Zwischen den Ebenen wird ein Austausch der Informationen berücksichtigt, der sicher stellt, dass der makroskopische Prozess- und Ablaufplan immer der aktuellen Ausprägung der Nebenbedingungen entspricht. Die makroskopische Ebene beschreibt auf der einen Seite eine Kontrollebene, die zur Abschätzung der Machbarkeit und der zu erwartenden

Ressourcennutzung dient. Auf der anderen Seite hat sie die Funktion einer Voroptimierung, um den Lösungsraum zu verkleinern. Der entwickelte Multiskalen-Algorithmus beschreibt ein Konzept zum Einsatz im Online-Scheduling und kann dementsprechend während der physikalischen Ausführung der Operationen parallel arbeiten.

Maravelias u. Sung (2009) stellen verschiedene Ansätze für die kombinierte Betrachtung der Prozess- und der Ablaufplanung vor. Hierbei geht es um die Bereitstellung von Ressourcen im Rahmen des *Supply-Chain-Management*. Auf einer übergeordneten Ebene wird über die Produktion von Rohmaterial unter Berücksichtigung von Lagerkosten entschieden, wohingegen auf niedriger Ebene die Verteilung des Endproduktes im Vordergrund steht. Gerade bei der Fertigung von pharmazeutischen Erzeugnissen wurden hier viele Veröffentlichungen über eine Betrachtung auf unterschiedlichen Detaillierungsebenen vorgenommen. Diese Veröffentlichungen beschäftigen sich mit der strategischen Planung bei der Produktion und Lagerung.

Hier soll allerdings auf einer übergeordneten Ebene eine Prozess- und Ablaufplanung vorgenommen werden, bei der nur unter großem Aufwand die zur Verfügung stehenden Ressourcen erhöht und angepasst werden können. Vielmehr geht es darum, das kombinierte Prozess- und Ablaufplanproblem an sich, unter den zur Verfügung stehenden Ressourcen, auf einem höheren Abstraktionslevel überschlägig zu lösen, um auf detaillierter Ebene mit geringeren Hardwareanforderungen einen konkreten Ablaufplan zu erzeugen. Aspekte, die im Rahmen des *Supply-Chain-Management* gewonnen worden sind, werden hierbei angepasst und auf die geänderte Problematik angewendet.

Als Anwendungsbeispiel folgt im Kapitel 4 die Betrachtung der Optimierung des Betriebs auf einem Containerterminal. Gerade bei hochmodernen Containerterminals erfolgt die Steuerung der einzelnen mobilen Ressourcen halbautomatisch bis vollautomatisch, so dass eine Integration des Multiskalen-Algorithmus in die Betriebsleittechnik vor Ort eine direkte Nutzung der optimierten Lösung ermöglicht.

3.1. Motivation

Als Ausgangspunkt wird ein *Flexible-Job-Shop*-Problem (siehe 2.1.5) betrachtet, bei dem es für die einzelnen Jobs Ausführungszeitfenster gibt, in denen sie zur Ausführung zur Verfügung stehen. Im Rahmen der Prozessplanung muss bestimmt werden, auf welche Art und Weise die Fertigstellung jedes einzelnen Jobs geschehen soll. In der Praxis wird häufig durch Experten eine Grobplanung vorgenommen und die Auswirkungen einer Routenwahl abgeschätzt. In einem nächsten Schritt folgt daraufhin die Bestimmung einer konkreten Operationensequenz pro Ressource, sprich die Erzeugung eines konkreten Ablaufplans unter Berücksichtigung aller Nebenbedingungen.

Der hier entwickelte Multiskalen-Algorithmus greift dieses Konzept auf und bietet somit eine Möglichkeit, das *Flexible-Job-Shop*-Problem auf einer groben Detaillierungsebenen

holistisch zu optimieren und einen Prozessplan sowie einen groben Ablaufplan zu erhalten, der im Rahmen eines *Online-Scheduling*-Verfahrens auf einer detaillierteren Ebene konkretisiert wird.

Das Ziel der makroskopischer Ebene ist es dabei, unter Vernachlässigung von exakten Nebenbedingungen und Informationen einen optimalen Ablaufplan zu generieren, der einen Prozessplan und eine grobe Operationenreihenfolge umfasst. Der generierte Prozessplan und die grobe Operationenreihenfolge dienen zur Abschätzung über den gesamten Projektzeitraum. So ist es möglich, Aussagen über die Machbarkeit zu treffen und auftretende Engstellen und Zeitfenster mit besonderer Ressourcenauslastung zu identifizieren, um bereits im Vorhinein darauf reagieren zu können. Zur Optimierung auf der makroskopischen Ebene wird der in Abschnitt 3.3 vorgestellte Koevolutionäre Algorithmus verwendet.

Um die Prozess- und Ablaufplanung als begleitendes System zur Entscheidungsfindung im praktischen Betrieb nutzen zu können, ist eine Betrachtung der Begebenheiten auf möglichst hohem Detailniveau nötig. Um die Anforderungen an die eingesetzte Computerhardware zu verringern, wird ein *Rolling-Horizon*-Algorithmus eingesetzt (vgl. Li u. Ierapetritou (2010)). Im Rahmen eines solchen Algorithmus wird eine detaillierte Betrachtungsweise des Ressourcenbedarfs und ähnlichen Restriktionen lediglich innerhalb eines relativ kleinen Zeitfensters, ausgehend vom momentanen Zeitpunkt, vorgenommen. Iterativ werden die einzelnen Zeitfenster abgearbeitet, wobei bereits ausgeführte Operationen unverändert bleiben, Entscheidungen für zukünftige Operationen hingegen iterativ angepasst werden (siehe Abbildung 3.1).

Die auf makroskopischer Ebene bestimmte Lösung dient als Initiallösung für eine Optimierung des aktuell betrachteten Zeitfensters. Es wird eine detaillierte Ablaufplanung mit detaillierteren Ressourcenanfragen und Kapazitätsüberprüfungen der Operationen durchgeführt, die im aktuellen Zeitfenster des makroskopischen Ablaufplans liegen.

Anpassungen auf mikroskopischer Ebene führen zu Änderungen des Prozess- und Ablaufplans auf makroskopischer Ebene und haben somit Einfluss auf die nachfolgenden Zeitfenster. Parallel zur lokalen Optimierung innerhalb des aktuellen Zeitfensters wird der Gesamtplan über die gesamte Projektdauer, ausgehend vom aktuellen Zeitpunkt, in regelmäßigen Abständen neu evaluiert und optimiert. Folglich ist eine Änderung des Prozessplans und dementsprechend der Operationensequenz für den noch nicht eingeplanten Teil einer Route möglich.

Die Codierung im Rahmen des koevolutionären Algorithmus für die kombinierte Prozess- und Ablaufplanung auf makroskopischer Ebene wird so gewählt, dass sie einer *Least-Commitment*-Strategie folgt. Lediglich eine grobe Operationsreihenfolge soll bestimmt werden, um globale Aussagen treffen zu können. Daher wurde auf eine Codierung mittels Prioritätsregeln (siehe 2.3.1) zurückgegriffen. Diese Codierung dient sowohl zur Prozessplanung als auch zur Ablaufplanung.

Auf mikroskopischer Ebene wird eine exakte Operationenreihenfolge für einen kleinen Zeitraum mit einer kleinen Menge von Operationen bestimmt. Die Menge der einzuplanenden Operationen ergibt sich aus der in diesem Zeitraum auf makroskopischer Ebene bestimmten Menge an eingeplanten Operationen.

Aufgrund der Modellierung des Problems auf mehreren Ebenen und der Verwendung der zuletzt bestimmten optimierten Lösungen ist es möglich, schnell auf Änderungen der Restriktionen zu reagieren ohne die zuvor gewonnenen guten Lösungen zu verwerfen.

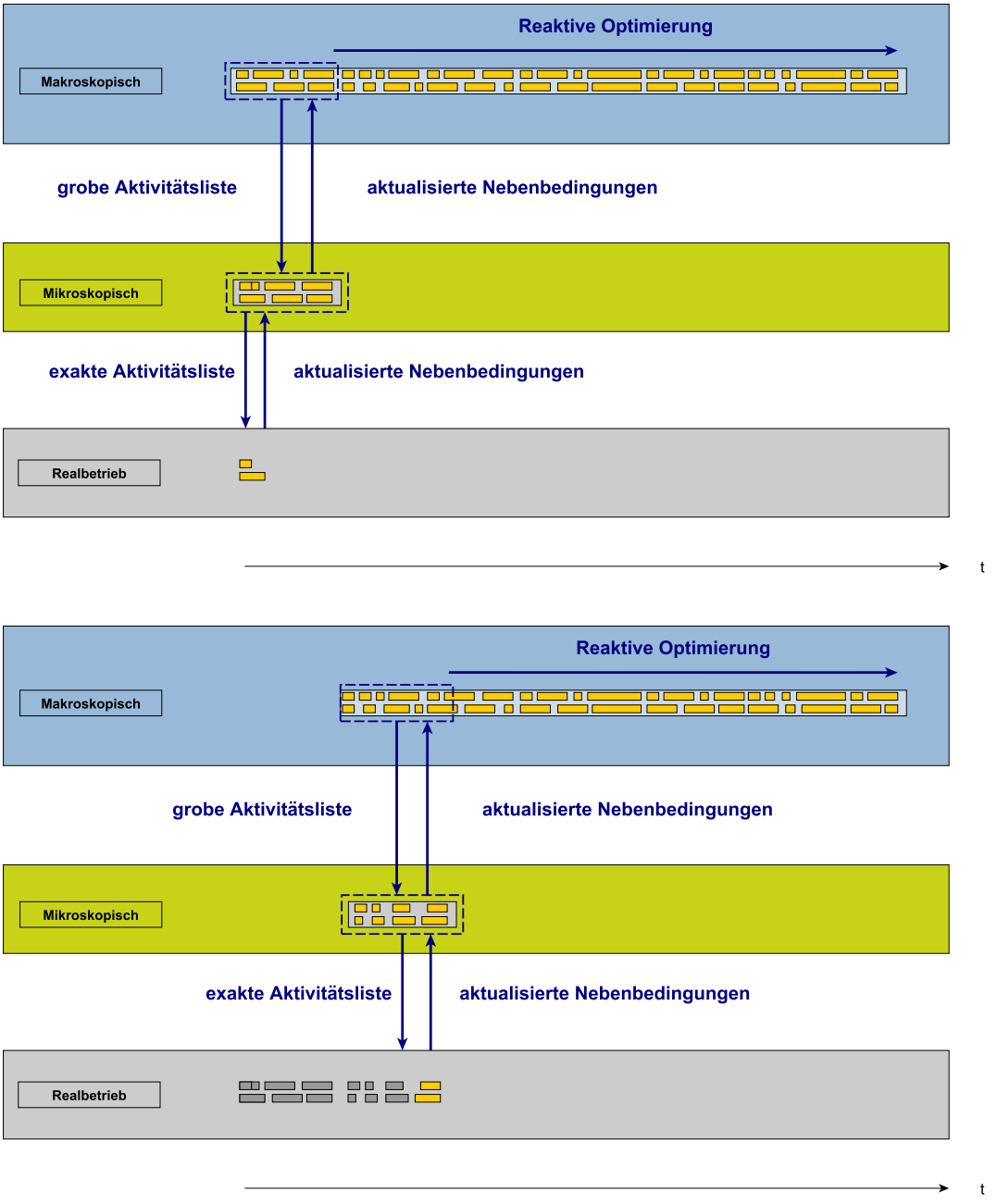


Abbildung 3.1.: Darstellung der Wechselwirkung zwischen makroskopischem Algorithmus und lokalem mikroskopischen Einplanverfahren.

3.2. Ablauf des Multiskalen-Algorithmus

Neben der eigentlichen Problemstellung, die die zeitlichen Restriktionen und technologischen Vorschriften umfasst, wird pro Skala für jede Ressource eine Einplanregel benötigt, die den Detaillierungsgrad der Überprüfung beschreibt. Weiterhin kann es unter Umständen möglich sein, dass die bereits bestimmten Routen nicht ausreichen, da zu große Änderungen an den Nebenbedingungen aufgetreten sind, so dass vorhandene Routen nicht mehr gültig sind. Daher kann optional ein Verfahren berücksichtigt werden, das neue Routen bestimmt.

Vor Beginn des Projekts wird auf makroskopischer Ebene eine Prozess- und eine Ablaufplanung vorgenommen. Diese kann zunächst als Machbarkeitsaussage verwendet werden. Gegebenenfalls muss der Entscheidungsträger vor Ort Maßnahmen vornehmen, um die Ressourcenkapazitäten zu erhöhen oder es muss die Anzahl an anzufertigenden Jobs verringert werden. Weiterhin dient diese Planung zur Auswahl der Operationen, die auf mikroskopischer Ebene unter Berücksichtigung eines höheren Detailgrades eingeplant werden sollen. Zur Generierung des Prozess- und des Ablaufplans wird der Koevolutionäre Algorithmus verwendet.

Für ein kleines Zeitfenster, das beispielsweise die ersten 30 Minuten des Betriebs umfasst, werden diejenigen Operationen aus dem makroskopischen Ablaufplan ausgewählt, die innerhalb des Zeitfensters liegen. Dies beinhaltet auch Operationen, die lediglich innerhalb dieses Zeitfensters starten. Diese Aktivitätsliste wird nun auf mikroskopischer Ebene weiterverarbeitet. Mittels eines genetischen Algorithmus werden diese Operationen optimal eingeplant. Die Prozessplanung wird dabei vernachlässigt.

Der auf dieser Ebene erzeugte Ablaufplan wird nun an den auszuführenden Ressourcen bestmöglich umgesetzt. Im Rahmen des bereits erwähnten Containerterminal-Problems wäre dies beispielsweise die Übermittlung einer Liste von Kranaufträgen an die jeweiligen Kranführer. Die Kranführer arbeiten diese der Reihe nach ab und bestätigen jeweils den Abschluss einer Operation. Auf mikroskopischer Ebene lässt sich der durch diese Bestätigung gewonnene Informationsstand über den Zustand auf dem Terminal auf die Nebenbedingungen übertragen und die Optimierung begleitend zur Ausführung weiterführen.

Der Informationsstand bezüglich der Nebenbedingungen ändert sich somit mit fortschreitender Zeit. Alleine die exakte Einplanung einer Ressource hat Auswirkungen auf die Zeitfenster der noch nicht eingeplanten Operationen. Unter Berücksichtigung dieser geänderten Informationen wird im nächsten Iterationsschritt auf makroskopischer Ebene ausgehend von der Lösung des vorherigen Iterationsschrittes optimiert. Hier werden die bereits ausgeführten Operationen als nicht veränderliche zeitliche Restriktionen gesetzt und eventuelle Änderungen an den Verfügbarkeitszeiten von Ressourcen oder an den Ausführungsfenstern von Operationen berücksichtigt. Ausgehend vom frühestmöglichen Zeitpunkt, auf dem auf einer Ressource eine Operation eingeplant werden

kann, und unter Berücksichtigung des momentanen Zeitpunkts im Realbetrieb wird die makroskopische Optimierung weitergeführt.

Der gesamte Ablauf wird so lange wiederholt, bis das Ende des Projekts erreicht ist. Abbildung 3.3 sowie Algorithmus 3.1 erläutern den Ablauf detailliert und zeigen das Zusammenspiel der einzelnen Komponenten.

Bei der Zerlegung des Gesamtproblems in kleine Zeitfenster kann es passieren, dass durch die Optimierung auf mikroskopischer Ebene Ressourcen die ihnen zugeordneten Operationen beendet haben, bevor das Ende des Zeitfensters erreicht ist. Um Leerlaufzeiten zu vermeiden, wird daher der minimale Endzeitpunkt c_{min} unter den letzten Operationen der Ressourcen bestimmt. Das nachfolgende Zeitfenster wird um den Bereich von c_{min} bis zum Ende des aktuellen Zeitfensters erweitert. Abbildung 3.2 zeigt das Vorgehen schematisch. Dadurch ist es möglich, Operationen, die auf makroskopischer Ebene im nachfolgenden Zeitfenster eingeplant sind, frühestmöglich, bereits im ursprünglich vorangegangen Zeitfenster, einzuplanen.

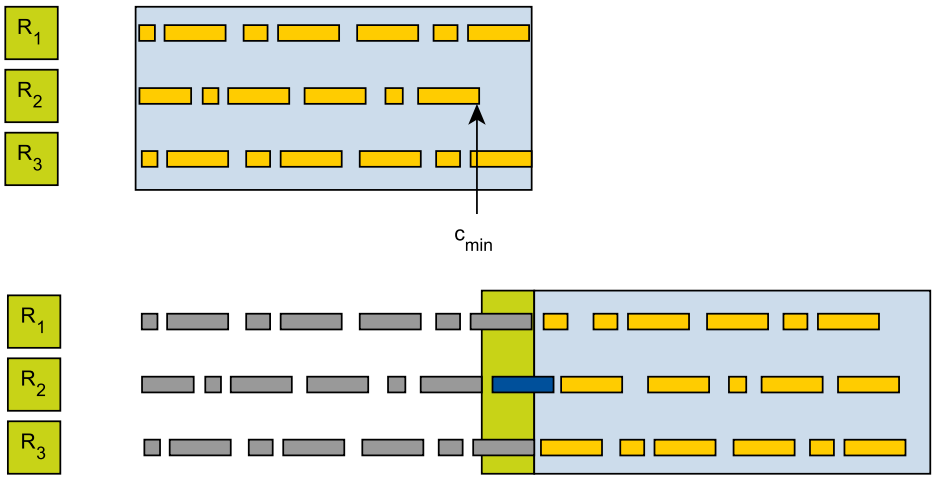


Abbildung 3.2.: Bestimmung des Startzeitpunktes des aktuellen Einplanzeitraums.

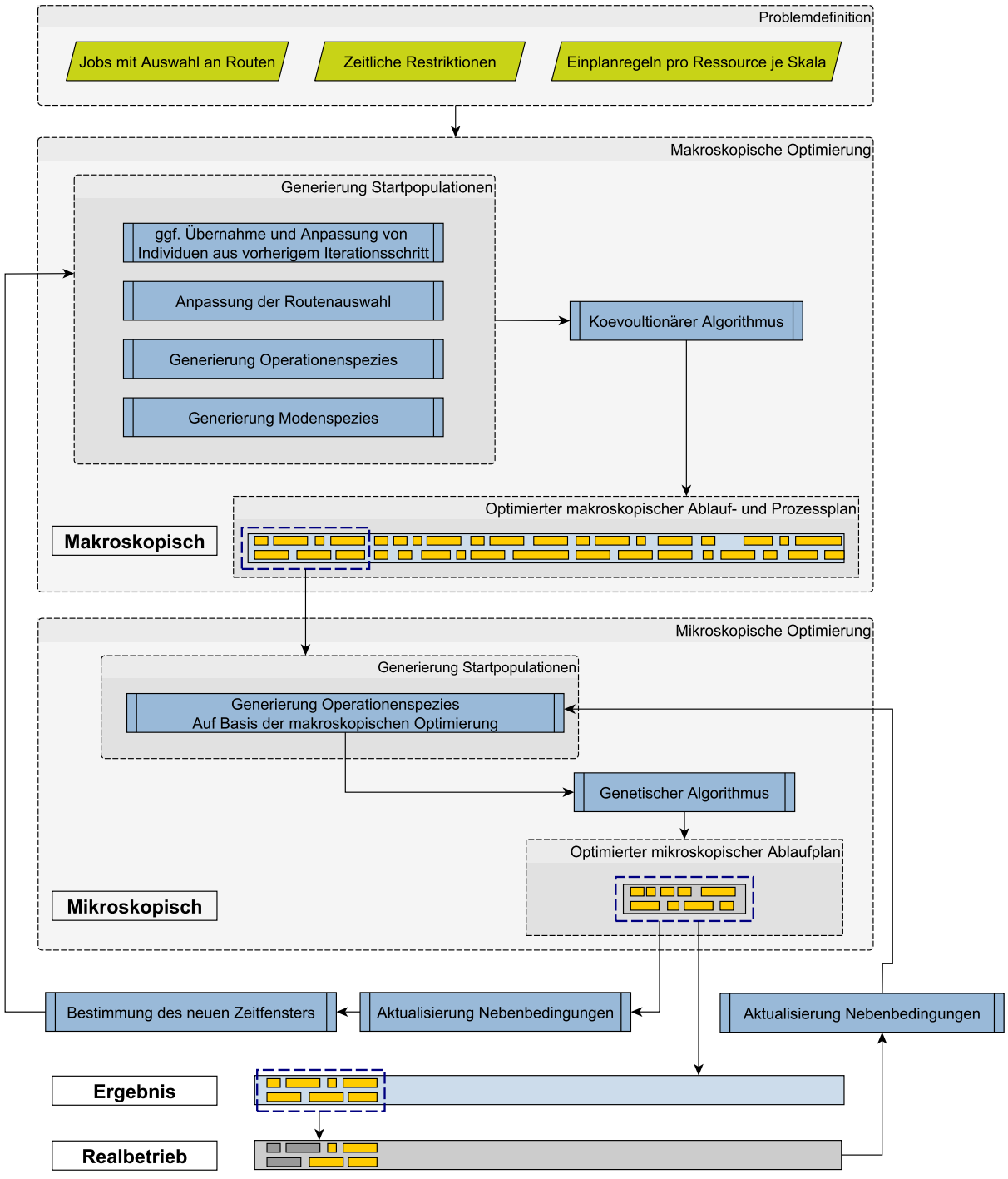


Abbildung 3.3.: Ablauf des Multiskalen-Algorithmus

Algorithmus 3.1 Ablauf des Multiskalen-Algorithmus mit zwei Ebenen

Eingabe: P ▷ *Flexible-Job-Shop*-Problem
Eingabe: SR_{micro}, SR_{macro} ▷ Einplanregeln
Eingabe: d ▷ Dauer des Zeitfensters auf Mikro-Ebene
Eingabe: $OT = [0, \bar{d}]$ ▷ gesamter Optimierungszeitraum
  Initialisiere Schedule S
  Initialisiere Problem P_{macro} mit den Einplanregeln SR_{macro} über OT
  Initialisiere momentan betrachtetes Zeitfenster $CT = [CT_1 = 0, CT_2 = d]$
  while $CT_2 < \bar{d}$ **do**
    Optimiere P_{macro} und bestimme die beste Lösung Fit_{macro}
    Wähle die Operationen O_{CT} aus Fit_{macro}, deren Einplanzeitpunkt innerhalb von CT liegt sowie die Operationen, die bereits ausgewählt wurden, allerdings noch nicht eingeplant werden konnten.
    Erzeuge Mikro-Problem P_{micro} mit O_{CT}, SR_{micro} sowie den Restriktionen und Ausführungszeitfenstern von P_{macro}
    Bestimme Startpopulation aus den Individuen der letzten Population der Optimierung von P_{macro}
    Optimiere P_{micro} und bestimme die beste Lösung Fit_{micro}
    Füge die neu hinzugekommenen Operationen Fit_{micro} dem Schedule S hinzu
    Bestimme Start des neuen Zeitfensters aus dem Minimum c_{min} der Endzeiten der Ressourcen von Fit_{micro}
    Setze $CT = [c_{min}, CT_2 + d]$
    Erzeuge neues Problem P_{macro} auf Makro-Ebene durch Überführung der bisher eingeplanten Operationen aus S
  end while
  return S

3.3. Optimierung auf makroskopischer Ebene

Das Ziel der Optimierung auf makroskopischer Ebene ist es, unter Verwendung von vereinfachten Modellen zur Bestimmung von Operationsdauern und zur Überprüfung der Einplanbarkeit, einen Prozess- sowie einen Ablaufplan zu generieren, der eine Routenwahl und eine grobe Operationenreihenfolge umfasst. Diese Planung dient zur Steuerung der Optimierung auf mikroskopischer Ebene und zur ständig aktuellen Abschätzung der Machbarkeit und des zu erwartenden Ressourcenbedarfs des gesamten Projekts.

3.3.1. Koevolutionärer Algorithmus

Die Bestimmung des Prozess- und des Ablaufplans erfolgt im Rahmen eines Koevolutionären Algorithmus. Ein Koevolutionärer Algorithmus ist angelehnt an die in der biologischen Evolution auftretende Wechselwirkung zwischen zwei stark interagierenden Arten. Da das Problem der Prozess- und Ablaufplanung sich grundsätzlich in zwei Teilprobleme auftrennen lässt, die sich allerdings stark beeinflussen, bietet sich ein solcher Algorithmus zur Behandlung des Problems an.

Die Güte einer Prozessplanung kann nur nach einer Ablaufplanung bestimmt werden und eine Ablaufplanung lässt sich erst nach erfolgter Prozessplanung durchführen. Für jedes Teilproblem wird eine eigene Teilpopulation angelegt. Die genetischen Operatoren zur Bestimmung der nachfolgenden Population werden nur auf Individuen innerhalb der jeweiligen Population angewandt. Interaktion zwischen den Populationen herrscht im Allgemeinen lediglich bei der Bestimmung der Fitness vor.

Im konkreten Fall wird ein Superindividuum eingeführt, welches aus zwei Subindividuen, dem Prozessplan-Individuum und dem Ablaufplan-Individuum, besteht. Für beide Populationen wird ein eigenständiger genetischer Algorithmus verwendet, wobei für die Bestimmung der Fitness eines Subindividuums ein Repräsentant aus der jeweiligen anderen Population benötigt wird. Dies bedeutet, dass ein Ablaufplan nicht ohne ein Prozessplan evaluiert werden kann sowie analog, dass ein Prozessplan nur in Verbindung mit einem Ablaufplan evaluiert werden kann. Die Auswahl des Repräsentanten kann durch verschiedenen Verfahren geschehen. Weiterhin ist eine Parallelisierung und somit asynchrone Berechnung der einzelnen Generationen möglich, genauso wie eine synchrone, hintereinander ausgeführte Berechnung der nächsten Generation der einzelnen Subpopulationen. Abbildung 3.4 zeigt den Ablauf des Koevolutionären Algorithmus.

Bei der Verwendung einer Codierungen, die die Reihenfolge der Operationen angibt, tritt der Nachteil auf, dass generierte Lösungen häufig ungültig sind und durch Reparaturmechanismen angepasst werden müssen. Gerade bei der Betrachtung im Rahmen des *Flexible-Job-Shop*-Problems mit der Berücksichtigung von unterschiedlichen Routen

können sich die ungültigen Lösungen mehren. Bei der Reparatur tritt wiederum die Frage auf, auf welche Art und Weise ungültige Operationenreihenfolgen oder zeitlich nicht realisierbare Prozessplankombinationen aufgelöst werden sollen. Wiederum ändern sich die einzuplanenden Operationen je nach Prozessplan. Eine explizite Codierung hätte hier den Nachteil, dass sie alle möglichen Operationen enthalten müsste. Die Operationen, die in der Prozessplanung nicht ausgewählt wurden, müssten in einem zweiten Schritt herausgefiltert werden. Dies führt wiederum zu einer schlechten Konvergenz des Verfahrens.

Um diese Problematiken zu vermeiden, wird hier sowohl für den Prozessplan als auch für die Operationenreihenfolge eine implizite Codierung gewählt, die zusammen mit einem geeigneten Decodierungsverfahren gültige Lösungen erzeugt.

3.3.2. Spezies der Prozessplanung

Bei der Prozessplanung gibt es die Tatsache zu berücksichtigen, dass nicht für jeden Job die optimale Operationensequenz gewählt werden kann, um ein Optimum bezüglich globaler Ziele zu erreichen. Für einige Jobs muss eine andere als die beste Route gewählt werden, etwa um kapazitive Restriktionen zu erfüllen. Weiterhin sind nicht alle Routenkombinationen möglich. Gründe dafür, dass eine Route nicht einplanbar ist, sind beispielsweise Verletzungen der Ausführungszeitfenster oder problemspezifische Restriktionen. Daher wird ein Verfahren entwickelt, welches gültige Kombinationen von Routen für die Jobs bestimmt.

Job-On-Node-Diagramm

Als Grundlage für die iterative Bestimmung einer gültigen Routenkombination wird ähnlich dem *Activity-On-Node*-Diagramm auf Operationenebene (siehe Definition 3), das Vorgängerbeziehungen zwischen Operationen darstellt, ein *Job-On-Node*-Diagramm auf Job-Ebene erstellt.

Eine Kante $< J_2, J_1 >$ im *Job-On-Node*-Diagramm beschreibt eine technologische Vorschrift zwischen der ersten Operation einer jeden Route des Jobs J_2 und der letzten Operation einer jeden Route des Jobs J_1. Konkret heißt dies, wenn eine Kante $< J_2, J_1 >$ existiert, so muss die erste Operation der jeweils in der Prozessplanung bestimmten Route des Jobs J_2 vor der letzten Operation der ausgewählten Route des Jobs J_1 stattfinden.

Die Beziehungen zwischen den Jobs haben Auswirkungen auf das *Activity-On-Node*-Diagramm (vgl. Definition 3). In der Darstellung als *Activity-On-Node*-Diagramm werden Vorgängerbeziehungen zwischen zwei Operationen O_i und O_j sowie deren zeitlichen Restriktionen als Kanten zwischen den entsprechenden Knoten mit den Kantengewichten δ_{ij} der Kantenmenge E bzw. der Relation des Graphen hinzugefügt.

Im Rahmen der Prozessplanung wird ein Graph G_{ALL} generiert, dessen Knotenmenge alle Operationen aus den möglichen Routen der einzelnen Jobs enthält (siehe Abbildung 3.5). Die Operationen einer Route werden mittels Kanten verbunden, um die Vorgängerbeziehungen abzubilden. Aus der Kantenmenge des *Job-On-Node*-Diagramms lassen sich nun die Relationen zwischen den Operationen der unterschiedlichen Jobs herleiten. Wie bereits beschrieben, bedeutet eine Kante $< J_2, J_1 >$ zwischen zwei Jobs, dass die erste Operation des Jobs J_2 vor der letzten Operation des Jobs L_1 stattfinden muss. Dies gilt folglich für alle möglichen Routen und deren erster bzw. letzten Operation.

Das Beispiel in Abbildung 3.5 zeigt die Beziehungen zwischen den Operationen, wobei die erste Route des Jobs J_2 und die dritte Route des Jobs J_1 gewählt wurden. Bevor die letzte Operation von J_1, die Operation O_{134}, ausgeführt werden kann, muss sowohl die

vorangegangene Operation der gewählten Route, die Operation O_{133}, fertig gestellt sein, als auch die Operation O_{211} des Jobs J_2.

Dieser Graph dient als Ausgangspunkt für die Bestimmung der einzelnen *Activity--On-Node*-Diagramme, die im Rahmen der Prozessoptimierung erstellt werden müssen. Je nach gewählten Routen für die einzelnen Jobs ergibt sich eine Teilmenge $\bar{O}$ der Operationenmenge O. Aus dem Graphen G_{ALL} lässt sich über diese Teilmenge ein Subgraph generieren, der für die weitere Auswertung zur Verfügung steht. Dabei ist das Kantengewicht δ_{ij} eine Funktion von $x(\alpha\beta)$. Das heißt, dass die Kanten lediglich im Graphen enthalten sind, wenn sowohl Operation O_i als auch Operation O_j zu Routings gehören, die ausgewählt wurden.

Das *Job-On-Node*-Diagramm lässt sich in Zusammenhangskomponenten (ZK) unterteilen, die jeweils eine Teilmenge aller Jobs umfassen, die in Beziehung zueinander stehen (siehe Abbildung 3.6). Diese Unterteilung ermöglicht eine Lösung des Gesamtproblems in Teilschritten. Die Zerlegung in Zusammenhangskomponenten kann beispielsweise effizient nach dem Algorithmus von T. (1972) erfolgen. Die bestimmten Zusammenhangskomponenten sind übertragbar auf die Operationen innerhalb des *Activity-On-Node*-Diagramms.

Iterative Prozessplanung

Pro Job steht eine Menge von Routen zur Verfügung, die nach ihrer individuellen Güte, im Bezug auf den spezifischen Job, sortiert sind. Innerhalb eines iterativen Verfahrens wird getestet, ob die momentan betrachtete Route eines Jobs einplanbar ist. Dies ist gleichbedeutend damit, dass alle zeitlichen Restriktionen eingehalten werden können. Sollte dies nicht der Fall sein, so wird versucht, die nächste Route einzuplanen.

Dabei ist die Reihenfolge, in der für die Jobs eine passende Route gesucht wird, entscheidend. Wird ein Job erst später ausgewählt und eingeplant, so kann es sein, dass durch bereits festgelegte Routen anderer Jobs eine schlechtere Route für diesen Job gewählt werden muss. Diese Reihenfolge kann entweder explizit angegeben werden oder aber über Prioritätsregeln bestimmt werden. Der Algorithmus 3.2 beschreibt das grundsätzliche Vorgehen. Die Sortierung der Jobs gemäß der Prioritätsregel oder einer vorgegebenen Sequenz innerhalb einer Zusammenhangskomponente für die Einplanung hat durch die Anwendung des Algorithmus einen impliziten Einfluss auf die Routenwahl.

Prioritätsregelbasierte Codierung

Als Eingabe für das in Algorithmus 3.2 vorgestellte Verfahren zur Prozessplanung kann entweder eine feste Reihenfolge vorgegeben werden, in der die einzelnen Jobs

Algorithmus 3.2 Bestimmung einer gültigen Prozessplanung

Eingabe: $G := (J, E)$ ▷ *Job-On-Node*-Diagramm
Eingabe: $R\{S_1, ..., S_\lambda, ..S_n\}$ ▷ Pro ZK eine Sequenz von Jobs
initialisiere $AON := (\{\}, \{\})$ ▷ *Activity-On-Node*-Diagramm
for $\lambda = 1 \rightarrow N$ **do**
Betrachte die Zusammenhangskomponente ZK_λ
for J_i in S_λ **do**
for $\beta = 1 \rightarrow k$ **do**
Versuche Route m^i_β des Jobs J_i einzuplanen
if Einplanen der Route m^i_β möglich **then**
Füge Operationen und Vorgängerbeziehungen der Route (die innerhalb der Route, sowie die zu Operationen anderer Jobs) in AON ein
BREAK
end if
end for
end for
end for
return AON

eingeplant werden oder die Reihenfolge kann auf Basis von Prioritätsregeln bestimmt werden. Letzteres ist im Sinne der *Least-Commitment*-Strategien (vgl. Sacerdoti (1975), Draper u. a. (1994) und Keng u. Y. Y. Yun (1989)), die bei der Ablaufplanung eingesetzt werden und den Plan in einer möglichst flexiblen Form repräsentieren, die es erlaubt, konkrete Entscheidungen erst zum Ausführungszeitpunkt zu treffen. Die ursprünglich für die Ablaufplanung vorhergesehene Codierung wurde hier zur Prozessplanung eingesetzt.

Im Rahmen der prioritätsregelbasierten Codierung wird für ein Individuum aus der Population der Prozessplanoptimierung pro Zusammenhangskomponente eine Prioritätsregel festgelegt (siehe Abbildung 3.7). Zur Formulierung der Prioritätsregeln lassen sich beispielsweise die Ausführungszeitfenster $[S_i, E_i]$ der Jobs mit dem frühsten Start S_i und dem spätesten Ende E_i benutzen. Weiterhin erscheint eine Betrachtung der Mindest-Transportzeit t_i^{min} der besten Route eines Jobs als sinnvoll. Tabelle 3.1 listet einige mögliche Prioritätsregeln auf.

Zusammenhangskomponenten im *Job-On-Node*-Diagramm können unter Umständen viele Jobs beinhalten. Weiterhin können die Jobs einer Zusammenhangskomponente zeitlich verteilt über die Projektdauer sein. Das bedeutet, dass die Ausführungszeitfenster der Jobs eher nacheinander angeordnet sind. Die Entscheidung, einen Job innerhalb einer Zusammenhangskomponente zu bevorzugen, hat daher schwer erfassbare Auswirkungen. Die Interaktion zwischen zeitgleichen Operationen ist zumeist stärker einzuschätzen. Daher ist die Güte der Wahl einer Prioritätsregel für eine Zusammen-

Bezeichnung	Abkürzung	Kriterium
Min Availability	MA	$\min(E_i - S_i)$
Min Due Date	MDD	$\min(E_i)$
Max Min Processing Time	MMPT	$\max(t_i^{min})$

Tabelle 3.1.: Prioritätsregeln für die Prozessplanung

hangskomponente abhängig von der Wahl der Prioritätsregeln für Zusammenhangskomponenten, die Operationen besitzen, die zeitgleich zu Operationen der betrachteten Zusammenhangskomponente ausgeführt werden. So werden etwaige Zeitpunkte mit Ressourcenknappheit indirekt aufgelöst.

Permutationsbasierte Codierung

Im Gegensatz zu der zuvor vorgestellten Bestimmung der Einplanreihenfolge der Jobs pro Zusammenhangskomponente lässt sich diese Reihenfolge auch über eine permutationsbasierte Codierung bestimmen. Für jede Zusammenhangskomponente wird ein Teilchromosom erzeugt, dass eine Permutation der Jobs der Komponente beschreibt (siehe Abbildung 3.8).

3.3.3. Spezies der Ablaufplanung

Die Bestimmung der Operationenreihenfolge auf makroskopischer Ebene geschieht ebenfalls implizit. Zum Decodieren eines Individuums und der Generierung eines Ablaufplans wird im Allgemeinen ein *Schedule Generation Scheme* (SGS) verwendet. In diesem Fall wird nach der Bestimmung der Routen durch das vorgestellte Verfahren für ein Individuum der Prozessplanspezies ein zeitorientierter paralleler SGS genutzt, bei dem in jeder Iteration ein neuer Entscheidungspunkt betrachtet wird (siehe Abschnitt 3.3.4). In jeder Iteration wird eine Teilmenge von Aktivitäten eingeplant. Hierzu wird die Teilmenge von Operationen bestimmt, die die Operationen umfasst, die zum aktuellen Entscheidungszeitpunkt einplanbar sind.

Codierung

Bei der Auswahl der nächsten einzuplanenden Operation aus der Menge der möglichen Operationen kann nach verschiedenen Kriterien vorgegangen werden. Tabelle 3.2 bietet

einen Überblick über die zur Verfügung stehenden Prioritätsregeln, z.B. über den frühsten Start ES_j einer Operation oder dessen Dauer p_j.

Anstatt die Operationenreihenfolge über eine Aktivitätsliste vorzugeben, die die Reihenfolge der einzuplanenden Operationen exakt vorgibt, codiert ein Chromosom der Population II zur Ablaufplanung für die Entscheidungspunkte eine Prioritätsregel, die auf die Teilmenge der ausführbaren Operationen innerhalb des Iterationsschrittes des SGS angewendet wird. Die Operationen werden der zugehörigen Prioritätsregel entsprechend sortiert und der Reihenfolge nach abgearbeitet. Abbildung 3.9 zeigt den prinzipiellen Aufbau des Chromosoms. Eine solche Codierung hat unter anderem den Vorteil, dass keine ungültigen Lösungen entstehen können, da die Reihenfolge im Rahmen des SGS festgelegt wird und nicht direkt codiert ist. Eine Codierung mittels Aktivitätsliste muss teilweise mit recht hohem Aufwand repariert werden, um technologische Vorschriften zu berücksichtigen.

Der betrachtete Optimierungszeitraum wird diskretisiert. Pro Zeitintervall wird eine Prioritätsregel festgelegt. Die Komplexität des Lösungsraums lässt sich anhand eines kleinen Beispiels verdeutlichen. Betrachtet man ein Projektzeitraum von acht Stunden und eine Diskretisierung von 2 Minuten, ergeben sich 240 Zeitintervalle. Bei 14 zur Verfügung stehenden Prioritätsregeln ergeben sich $14^{240} \approx 1,1769 * 10^{275}$ verschiedene Lösungskandidaten. Zu beachten ist, dass die Wahl der Diskretisierung in Abhängigkeit der Operationsdauer getroffen werden sollte.

3.3.4. Evaluation und Erzeugung von Ablaufplänen

Für die Evaluierung der Individuen muss ein Selektionsmechanismus festgelegt werden, der aus der jeweils anderen Spezies ein Individuum auswählt, mit dessen Hilfe die Fitness am Gesamtproblem bestimmt werden kann. Es wird ein Superindividuum erzeugt, das aus einem Individuum für die Prozessplanung und einem Individuum für die Ablaufplanung besteht. Die Fitness des Super-Individuums wird dem zu untersuchenden Individuum zugewiesen.

Zum Erzeugen eines Ablaufplans wird nun das Super-Individuum mittels eines SGS ausgewertet. Für konkrete Aufgabenstellungen muss das Schema unter Umständen angepasst werden, beispielsweise um Rüstzeiten zu berücksichtigen. Im Rahmen des SGS wird für die aktuell betrachtete Operation je nach Detaillevel überprüft, ob die Kapazität aller beteiligten Ressourcen für die Dauer der Operation nicht überschritten wird. Werden alle Nebenbedienungen eingehalten, so wird die Operation eingeplant und die Auslastung der Ressourcen aktualisiert. Sollte dies nicht der Fall sein, so wird versucht, die nächste Operation der sortierten Menge einzuplanen.

Grundsätzlich wird dabei zwischen seriellen und parallelen SGS unterschieden. Um bei einem seriellen SGS Operation einzuplanen, wird zunächst versucht, die Operation

Bezeichnung	Abkürzung	Kriterium
Min Earliest Start Time	EST	$\min(ES_i)$
Min Late Start Time	LST	$\min(LS_i)$
Min Earliest End Time	EET	$\min(EE_i)$
Min Latest End Time	LET	$\min(LE_i)$
Min Slack	MINSLK	$\min(LS_i - ES_i - p_i)$
Max Slack	MAXSLK	$\max(LS_i - ES_i - p_i)$
Shortest Processing Time	SPT	$\min(p_i)$
Longest Processing Time	LPT	$\max(p_i)$
Least Immediate Successors	LIS	$\min(\|Suc_i\|)$
Most Immediate Successors	MIS	$\max(\|Suc_i\|)$
Most Total Successors	MTS	$\max(\|\bar{Suc}_i\|)$
Greatest Rank Positional Weight	GRPW	$\max(p_i + \sum_{j \in Suc_i} p_j)$
Least Rank Positional Weight	LRPW	$\min(p_i + \sum_{j \in Suc_i} p_j)$
Least Vertex Class	LVC	$\min(rank(i))$
Min Setup Time	MINSETUP	$\min s_{ij}$

Tabelle 3.2.: Prioritätsregeln für die Ablaufplanung, angelehnt an Masmoudi u. Haït (2013) und Ozdamar (1999)

in ggf. vorhandene Lücken im Ablaufplan zu füllen. Sollte dies nicht möglich sein, wird die Operation an das Ende des Ablaufplans angehängt. Im Gegensatz zu dieser operationenorientierten Vorgehensweise handelt es sich beim parallelen SGS um ein zeitorientiertes Verfahren zum Erzeugen eines Ablaufplans. Der Ablaufplan wird zeitlich fortlaufend generiert und die zum aktuell betrachten Zeitpunkt einplanbaren Operationen untersucht. Dementsprechend lassen sich auch nur am Ende eines Ablaufplans neue Operationen einfügen.

Bei der hier gewählten Codierung mittels Prioritätslisten bietet sich ein paralleler SGS an. Die Codierung beschreibt für ein Zeitintervall von wenigen Minuten die aktuell zu verwendende Prioritätsregel.

Bei einem seriellen SGS ist eine Zuordnung eines Genortes zu einem Zeitintervall nicht möglich. Man könnte die Prioritätsregeln auf die Anzahl an einzuplanenden Operationen beziehen und beispielsweise eine Prioritätsregel für die nächsten fünf einzuplanenden Operationen festlegen. Dies führt im Rahmen der kombinierten Be-

trachtung von Prozess- und Ablaufplanung allerdings zu Problemen, da die Anzahl an einzuplanenden Operationen vor der Auswertung eines Individuums nicht bekannt ist. Erst nachdem das Chromosom für die Prozessplanung evaluiert wurde, ist es möglich eine Aussage über die Anzahl an Operationen zu treffen.

Bei einem parallelen SGS hängt die Anzahl an Entscheidungspunkten von den gewählten Routen ab. Somit lässt sich die Anzahl der Genorte nicht auf die Anzahl an Entscheidungspunkten beziehen. Eine Genlänge, die der maximal möglichen Anzahl an Entscheidungspunkten entsprechen würde, würde dazu führen, dass nicht relevante Teile mit in der Codierung enthalten sind, die unter Umständen das Konvergenzverhalten des genetischen Algorithmus beeinflussen. Algorithmus 3.3 beschreibt das allgemeine Vorgehen des SGS, der auf zuvor beschriebene Weise die Prioritätsregeln je Zeitintervall berücksichtigt.

Algorithmus 3.3 Paralleler SGS auf Basis von Prioritätsregeln

Eingabe: R:= $\{R_1, ...R_\lambda, ..., R_m\}$ ▷ Prioritätsregel-Liste
Eingabe: $\mathbb{O} := \{O_1, O_2, ..., O_i, ..., O_n\}$ ▷ Menge der einzuplanenden Operationen
Eingabe: $O_i \mapsto \{(ES_i, LS_i, EE_i, LE_i)\}$ ▷ Ausführungsfenster der Operationen
$t = \min_{i \in V}(ES_i)$
Bestimme die Menge E_t von Operationen O_i, die keine Vorgängeroperationen haben und zum aktuellen Zeitpunkt t eingeplant werden können ($ES_i \leq t$).
Bestimme die Menge A_t von Operationen O_i, die keine Vorgängeroperationen haben, aber noch nicht eingeplant werden können ($ES_i > t$).
while nicht alle Operationen eingeplant sind **do**
  while $E_t \neq \emptyset$ **do**
    Bestimme aktuelle Prioritätsregel R_t anhand t
    Wähle Operation $O_j \in E_t$ anhand der Prioritätsregel R_t
    if eine Ressource k mit $rjk > R_k(\tau)$ für ein $\tau \in [t, t + p_j[$ existiert **then**
      CONTINUE
    else
      Plane die Operation j in den Ablaufplan S ein.
    end if
  end while
  Bestimme neues t als Minimum der Endzeiten der bereits eingeplanten Operationen und dem frühsten ES_i der Operationen aus A_t.
  Aktualisiere E_t und A_t
end while
return S

3.3.5. Diskussion der Prioritätsregeln

Im Folgenden werden die vorgestellten Prioritätsregeln genauer analysiert und ihre Eignung für die Optimierung diskutiert. Einige der Prioritätsregeln führen zu ähnlichen Ergebnissen, andere erscheinen als nicht sonderlich sinnvoll für die konkrete Aufgabenstellung der Optimierung im Rahmen eines *Flexible-Job-Shop*-Problems mit Ausführungszeitfenstern. Weiterhin ist zu überlegen, auf welchen Informationsstand bei der Berechnung der Prioritäten zurückgegriffen werden soll.

Start- und Endzeitenbezogene Regeln

Ein Großteil der in der Literatur beschrieben Regeln beziehen sich auf die Ausführungszeitfenster der Operationen. Dafür muss ein Problem vorliegen, wo diese von Wichtigkeit sind. Die frühsten und spätesten Start- und Endzeiten einer Operation lassen sich mit einfachen Mitteln der Netzplantechnik bestimmen.

Min Earliest Start Time & Min Earliest End Time Eine Sortierung nach den frühsten Start- oder Endzeiten ist lediglich sinnvoll, wenn keine zeitkritischen Operationen vorliegen. Zumeist treten in diesem Fall allerdings Optimierungsziele, wie beispielsweise die Minimierung des Energiebedarfs, etc. in den Vordergrund. Daher werden diese Regeln hier nicht verwendet.

Min Latest End Time & Min Late Start Time Operationen, deren spätestes Ende beziehungsweise spätester Start früher liegt, werden bevorzugt. Dies verringert das Risiko, dass diese Operationen nicht rechtzeitig abgefertigt werden können. Die beiden Regel *LST* und *LET* sind dabei sehr ähnlich, da bei der Startzeit lediglich die Operationsdauer mit berücksichtigt wird. Da gerade bei Operationen von langer Dauer diese Regel allerdings einige Vorteile birgt, wird die *LET* im Weiteren nicht berücksichtigt.

Min Slack & Max Slack Das Ziel der *Min-Slack*-Regel ist es, kritische Operationen zu bevorzugen. Eine Operation ist dabei kritisch, wenn es wenig Pufferzeiten gibt, diese Operation auszuführen. Das bedeutet, dass das Zeitfenster, in dem die Operation ausgeführt werden kann, unter Berücksichtigung der Operationsdauer recht klein ist. Die *Max-Slack*-Regel bevorzugt unkritische Operationen und wird hier nicht weiter verwendet.

Operationsdauerbezogene Regeln

Die Dauer einer Operation ist eine der wichtigsten Größen, die bei der Einplanung berücksichtigt werden muss. Die Operationsdauer in Verbindung mit dem Bedarf an Ressourcen hat den größten Einfluss auf die Ressourcenprofile.

Shortest Processing Time Operationen mit geringerer Dauer werden bevorzugt. Diese Prioritätsregel bietet sich an, wenn kleine Lücken im Ablaufplan mit kurzen Operationen gefüllt werden sollen.

Longest Processing Time Operationen mit längerer Dauer werden bevorzugt. Diese Operationen binden die ausführenden Ressourcen recht lange. Diese Regel bietet sich an, wenn große Leerzeiten im Ablaufplan zur Verfügung stehen.

Graphbezogene Regeln

Die Zusammenhänge zwischen Operationen lassen sich anschaulich in einem *Activity-On-Node*-Diagramm darstellen. Technologische Vorschriften lassen sich so mit Hilfe der Graphentheorie genauer analysieren.

Most Immediate Successors & Least Immediate Successors Priorisiert die einzelnen Operationen nach der Anzahl der direkt nachfolgenden Operationen. Eine Operation mit vielen direkten Nachfolgern bildet einen Flaschenhals, da die nachfolgenden Operationen erst ausgeführt werden können, wenn die entsprechende Operation ausgeführt wurde. Eine Sortierung nach der minimalen Anzahl (*Least-Immediate-Successors*) erscheint nicht sinnvoll und wird nicht weiter verfolgt.

Most Total Successors Die Anzahl an Nachfahren berücksichtigt den gesamten nachfolgenden Abschnitt innerhalb des Graphen ab dem betrachteten Knoten. Neben der Identifikation von Engstellen im Graph werden auch Operationen, die eine lange Kette weiterer Operationen nach sich ziehen, bevorzugt.

Greatest Rank Positional Weight & Least Rank Positional Weight Die beiden Regeln sind stark verwandt mit den Regeln für die direkte Nachfolger. Allerdings werden die Operationsdauern berücksichtigt. Die *Greatest-Rank-Positional-Weight*-Regel wird größtenteils von der *Most-Immediate-Successors*-Regel abgedeckt und somit nicht verwendet. Die *Least-Rank-Positional-Weight*-Regel wird aus den gleichen Gründen wie bei der *Least-Immediate-Successors*-Regel ebenfalls vernachlässigt.

Least Vertex Class Bevorzugt Operationen, die einen geringeren Rang in der Topologie des Graphen haben. Ermöglicht es, neue Teilgraphen zu eröffnen und bietet dadurch im späteren Verlauf mehr Auswahlmöglichkeiten.

Sequenzabhängige Regeln

Anwendung in der Realität haben häufig Eigenschaften, die erschwerend bei der Planung hinzukommen. Eine nicht zu vernachlässigende Größe dabei ist die Rüstzeit, die die Dauer für die Bereitstellung einer Ressource beschreibt. In vielen Anwendungen ist die Rüstzeit nicht pauschal anzunehmen, sondern abhängig von der Reihenfolge der Operationen.

Min Setup Time Bevorzugt Operationen, die eine geringe Rüstdauer benötigen. Auf makroskopischer Ebene bedeutet dies, dass, wenn diese Regel ausgewählt wurde, keine zeitkritischen Operationen vorliegen und daher auf Energieeffizienz optimiert werden kann.

3.3.6. Komponenten des Koevolutionären Algorithmus

Bei Genetischen Algorithmen muss im Allgemeinen die Balance zwischen dem Erkunden des unter Umständen recht großen Lösungsraums und der Untersuchung der Nachbarschaft von bereits bestimmten, guten Lösungen, gefunden werden. Dazu lässt sich zunächst die Rekombination genauer betrachten.

Wenn die Fitness-Landschaft des Problems (siehe Definition 11) eine Struktur besitzt und nicht zerklüftet ist, so kann mittels *respektvollen* Rekombinationsoperatoren im Allgemeinen eine sehr gute Konvergenz gewährleistet werden (siehe Radcliffe (1991)).

Dieser bewirkt, dass ein Nachkommen z zweier Individuen x und y in der Fitness-Landschaft zwischen den beiden Eltern-Individuen liegt. Dies bedeutet, dass der Abstand zwischen den beiden Eltern-Individuen gleich oder größer ist, als der Abstand zwischen den Nachkommen und den Eltern.

Liegt der Nachkommen weiterhin auf dem kürzesten Weg zwischen den Eltern, so gilt die Gleichung $d(x,z) + d(z,y) = d(x,y)$ und man spricht von einem *sortierenden* Rekombinationsoperator.

Liegt der Ort des Nachkommens in der Fitness-Landschaft in einem Bereich, der nicht zwischen den beiden Eltern liegt, so spricht man von einem irreleitenden Rekombinationsoperator. Abbildung 3.10 zeigt das Verhalten aller drei Typen geometrisch dargestellt.

Die Verwendung eines *K-Point-Crossover*-Operators oder eines *Uniform-Crossover*-Operators bedeutet bei der Anwendung auf binärcodierte Chromosomen, dass

1. Allele, die in beiden Eltern-Individuen vorkommen, in den Nachkommen erhalten bleiben,
2. Nachkommen nur Allele von einem der beiden Eltern-Individuen enthalten können.

Punkt 1. beschreibt dabei die Eigenschaft respektvoll, Punkt 2. die Eigenschaft sortierend. Bei permutationsbasierter Codierung ist die Einhaltung des 2. Punkts nicht möglich, da Elemente auf andere Genorte verschoben werden müssen. Andernfalls können Werte an zwei verschiedenen Genorten vorkommen. Eine permutationscodierter Lösungsraum ermöglicht somit lediglich den Einsatz von respektvollen, aber nicht von sortierenden Rekombinationsoperatoren.

Operatoren für die prioritätsregelbasierte Codierung

Für die Rekombination und Mutation sowohl der Prozess- als auch der Ablaufplanung werden aus oben genannten Gründen die Standardoperatoren *Uniform*-Crossover und

Swap-Mutation gewählt. Sowohl bei der Auswahl der Eltern als auch für die Evaluierung am Gesamtproblem wird ein fitnessproportionaler Selektionsalgorithmus gewählt.

Akzeptanzmechanismen

Genetische Algorithmen konvergieren oft vorzeitig und verbleiben in lokalen Optima. Neben Anpassungen an den Mutationsoperatoren gibt es die Möglichkeit, bei der Generierung der nächsten Generation einer Population Akzeptanzmechanismen einzusetzen, die nicht alle neu erzeugten Individuen in die nächste Generation übernehmen. Im Sinne eines Elitismus lässt sich beispielsweise grundsätzlich eine gewisse Anzahl der besten Individuen unverändert in die nächste Generation übernehmen.

Hier wird ein Schwellwertauswahlverfahren verwendet, das noch um eine Möglichkeit des Zurücksetzens der Population erweitert wurde. Hierbei wird die Fitness des schlechtesten Individuums aus der vorangegangenen Generation als Schwellenwert verwendet. Alle Individuen aus der neuen Generation, die eine bessere Fitness haben, ersetzen nach und nach die schlechtesten Individuen aus der vorangegangenen Generation. Sollte die maximale Fitness der Population länger als eine festzulegende Anzahl an Generationen stagnieren, so werden lediglich wenige der besten Individuen aus der aktuellen Generation beibehalten und alle restlichen Individuen durch zufällig neu erzeugte Individuen ersetzt.

Diversitätsmaß

Um Aussagen über die Konvergenz und die genetische Vielfalt eines Genetischen Algorithmus zu treffen, empfiehlt es sich, eine Diversitätsanalyse durchzuführen. Bei der hier vorliegenden Codierung ist die Verwendung eines *paarweisen* Hamming-Distanzmaß (siehe Horn (1997)) nach einigen Anpassungen möglich. In seiner generalisierten Form ist es definiert als:

$$D(x_k, x_j) = \sum_{j=1}^{j=P-1} \sum_{k=j+1}^{k=P} d(x_k, x_j) \tag{3.1}$$

mit der Anzahl an Individuen P und einem Distanzmaß $d(x_k, x_j)$.

Im hier vorliegenden Fall ist eine Verwendung des Hamming-Distanzmaß nicht möglich, da es sich nicht um binäre Werte handelt. Daher wird ein anderes Distanzmaß gewählt, das die Anzahl an Allelen zählt, die unterschiedlich sind.

Selektionsverfahren zur Generierung des Superindividuums

Zur Evaluierung wird aus beiden Populationen jeweils ein Individuum benötigt, da eine Bewertung der Fitness nur durch beide Informationen, die Prozess- und die Ablaufplanung, möglich ist. Potter u. De Jong (1994) beschreiben den Gedanken, dass die Fitness eines Subindividuums über die Abschätzung bestimmt wird, wie gut es mit den anderen Subpopulationen kooperiert. Als einfachste Annahme wählen sie zunächst das jeweils beste Individuum der anderen Populationen aus. Sie weisen allerdings darauf hin, dass die Wahl des Selektionsverfahrens zur Bestimmung der Fitness viel Potential hat und näher untersucht werden sollte. Aufbauende Arbeiten wie die von Cai u. Peng (2002) verwenden zumeist jedoch auch einen Selektionsmechanismus, der das jeweils beste Individuum der anderen Subpopulationen auswählt. Ladjici u. a. (2014) verwenden zur Fitnessbestimmung eine Anzahl von zufällig bestimmten Individuen der jeweils anderen Teilpopulationen, bilden den Mittelwert aus den bestimmten Fitnesswerten und weisen diesen Mittelwert dem zu evaluierenden Individuum als Fitnesswert zu.

Für die erste Generation wird ein Selektionsmechanismus gewählt, der zufällig ein Individuum aus den anderen Spezies wählt.

3.3.7. Wiederverwendbarkeit der Individuen

Die gewählten Codierungen der Individuen für die Prozess- und Ablaufplanung haben den Vorteil, dass sie im Rahmen des Multiskalen-Algorithmus wiederverwendet werden können. Die Prozessplanung codiert Informationen für die einzelnen Zusammenhangskomponenten, die sich im Laufe der Optimierung nicht ändern. Für die Ablaufplanung gibt die Codierung Prioritätsregeln für aufeinanderfolgende Zeitfenster an. Wurden bereits Operationen aus Zeitfenstern eingeplant, sei es in der Simulation oder im realen Betrieb, so lassen sich die Individuen, die auf makroskopischer Ebene bestimmt wurden, weiter verwenden. Der Optimierungszeitraum verringert sich lediglich, da der bereits eingeplante Zeitraum wegfällt. Die Chromosomen der Individuen aus der vorangegangenen Optimierung werden folglich im vorderen Bereich so verkürzt, dass die bereits eingeplanten Zeiträume wegfallen. So können Lösungen, die in einem früheren Iterationsschritt des Multiskalen-Algorithmus zu guten Ergebnissen geführt haben, zu einem späteren Iterationsschritt als Initiallösung dienen.

Sollten sich die Nebenbedingungen nicht zu stark geändert haben, so sollten diese Lösungskandidaten im Rahmen des Genetischen Algorithmus verhältnismäßig schnell erneut zu guten Ergebnissen und somit zu einer hohen Fitness führen. Weiterhin dienen die Teilchromosome als eine Startlösung für die mikroskopische Ebene. Die Lösungen der Prozessplanung sind ohne Anpassungen zu übernehmen, da sich die Zusammenhangskomponenten nicht ändern.

3.3.8. Modellierung der Zielfunktion unter Berücksichtigung der Unsicherheit

Häufig werden an Problemstellungen mehrere Anforderungen gestellt, die bestmöglich erfüllt werden müssen. Man spricht von einer mehrkriteriellen Optimierung. Die unterschiedlichen Optimierungsgrößen können beispielsweise im Rahmen einer Pareto-Optimierung oder einer Lexikographischen Ordnung (siehe Definition 39) berücksichtigt werden. Bei einer Pareto-Optimierung können bekannte Varianten des genetischen Algorithmus, die Probleme mit mehreren Zielgrößen berücksichtigen, wie beispielsweise der NSGA-II (siehe Deb u. a. (2000)), eingesetzt werden.

Definition 27: Mehrkriterielle Optimierung

Die mehrkriterielle Optimierung bezeichnet das Lösen eines Optimierungsproblems unter Berücksichtigung mehrerer Zielfunktionen. Die resultierende Zielfunktion eines solchen Optimierungsproblems lässt sich als Vektor auffassen. Sie wird beschrieben durch:

$$f(\vec{x}) = [f_1(x), f_2(x), ..., f_n(x)]^T \tag{3.2}$$

Auf makroskopischer Ebene des Multiskalen-Algorithmus muss die Robustheit der Lösung als weitere Zielgröße der mehrkriteriellen Optimierung aufgenommen werden.

Im Rahmen der makroskopischen Optimierung werden Kenngrößen nicht exakt berücksichtigt. Beispielsweise kann modellbedingt ein vereinfacher Weg einer sich bewegenden Ressource angenommen werden. Dadurch kann es passieren, dass die mikroskopische Ausprägung der Operation eine längere Operationsdauer oder einen höheren Ressourcenbedarf aufweist als die in der makroskopischen Optimierung bestimmten Werte.

Weiterhin unterliegen reale Problemstellung grundsätzlich problemspezifischen Unsicherheiten. Im Folgenden werden zwei alternative Optimierungsgrößen vorgestellt, die bei der Generierung von robusten Ablaufplänen eingesetzt werden können, die die zuvor genannten Faktoren berücksichtigen.

Resource-Leveling

Ein häufig verwendeter Ansatz verfolgt das Ziel, eine gewisse Ausgeglichenheit der Ressourcennutzung zu erreichen. Man spricht von einer Bedarfsglättung (engl.: *resource leveling*) (siehe beispielsweise Leu u. Yang (1999) und Zhao u. Liu (2006)). Hierbei wird der Projektzeitraum in Zeitintervalle diskretisiert. Das Optimierungsziel verfolgt eine

gleichmäßige Auslastung über diese Zeitintervalle. Je nach Problemstellung kann es problematisch sein, diese Bedarfsglättung vorzunehmen, gerade bei der Berücksichtigung von Ausführungszeitfenstern für einzelne Operationen.

Definition 28: Resource-Leveling-Index

Mathematisch formulieren lässt sich die Bedarfsglättung als Minimierung des *Resource-Leveling*-Index RI, der wie folgt definiert ist:

$$RI := \sum_{\kappa=1}^{K} RI_\kappa \tag{3.3}$$

Für alle Ressourcen $1, ..., \kappa, ..., K$ wird über alle Zeitintervalle $1, ...t, ..., n_T$ die mittlere absolute Differenz zwischen der tatsächlichen und der gleichmäßigen Ressourcenauslastung RI_κ bestimmt.

Dazu werden alle Operationen $1, ..., i, ..., n_O$ betrachtet. Die mittlere absolute Differenz ergibt sich somit zu

$$RI_\kappa = \sum_{t=1}^{n_T} \sum_{i=1}^{n_O} \left(\theta_{i\kappa t} - \bar{\theta}_\kappa\right)^2, \tag{3.4}$$

wobei $\theta_{i\kappa t}$ den anteiligen Ressourcenbedarf der Operation i an der Ressource κ im Zeitintervall t beschreibt. Dieser anteilige Ressourcenbedarf ergibt sich unter der Berücksichtigung der Intervall-Länge T, der Ressourcenkapazität R_κ und der Dauer p_{it}, die angibt wie lange die Operation i im Intervall t ausgeführt wird, zu

$$\theta_{i\kappa t} = \frac{r_{i\kappa} \cdot p_{it}}{T \cdot R_\kappa}. \tag{3.5}$$

Die mittlere Ressourcenauslastung $\bar{\theta}_\kappa$ ist durch

$$\bar{\theta}_\kappa = \frac{1}{n_T} \sum_{i=1}^{N} \theta_{i\kappa t} \tag{3.6}$$

gegeben. Abbildung 3.12 stellt die benötigten Größen nochmals dar.

Maximierung der Pufferzeiten

Die Problemstellungen, die im Rahmen dieser Arbeit behandelt werden, zeichnen sich unter anderem durch das Vorhandensein von Ausführungszeitfenstern von Operationen aus (siehe Abschnitt 2.1.4). Diese Ausführungszeitfenster sind bei der Maximierung der Robustheit ein wichtiger Faktor, den es zu berücksichtigen gilt.

Al-Fawzan u. Haouari (2005) betrachten Probleme der Netzplantechnik und definieren die freie Pufferzeit s_i einer Operation O_i als Größe des Zeitfensters, in dem eine Operation verschoben werden kann, ohne die Startzeit der nachfolgenden Operationen nach hinten zu verschieben. Zur Erzeugung von robusten Ablaufplänen schlagen sie vor, die Summe der freien Pufferzeiten aller Operationen zu maximieren. Aufbauend auf dieser Arbeit schlagen Kobylański u. Kuchta (2007) vor, die minimale Pufferzeit zu maximieren, um einen robusten Ablaufplan zu erhalten.

Diese Ansätze werden nun auf die Bewertung von Ablaufplänen, die im Rahmen eines SGS zur Lösung von *Flexible-Job-Shop*-Problemen mit Ausführungszeitfenstern generiert werden, übertragen. Die Pufferzeit beschreibt in diesem Zusammenhang die Zeitdauer zwischen dem spätesten Ende einer Operation und den durch die Heuristik bestimmten Startzeitpunkt S_i dieser Operation. Die hier betrachtete Pufferzeit P_i einer Operation O_i mit ihrem spätesten Ende LE_i und ihrer Dauer p_i lässt sich folglich als

$$P_i = LE_i - (S_i + p_i) \tag{3.7}$$

definieren.

Betrachtet man die Maximierung der Summe der Pufferzeiten nach Al-Fawzan u. Haouari (2005), führt ein solches Vorgehen dazu, dass eine groß Pufferzeiten einer Operation viele kleine Pufferzeiten anderer Operationen kompensiert. Besitzt von einer Menge an Operationen eine Operation eine besonders große Pufferzeit, die anderen Operationen allerdings nur recht kleine, so versagt das Kriterium der Maximierung der Summe. Der Ablaufplan hat dadurch nicht unbedingt die gewünschte Robustheit. Die Betrachtung der minimalen Pufferzeit nach Kobylański u. Kuchta (2007) umgeht dieses Problem, da das Maß unempfindlicher gegen Ausreißer ist.

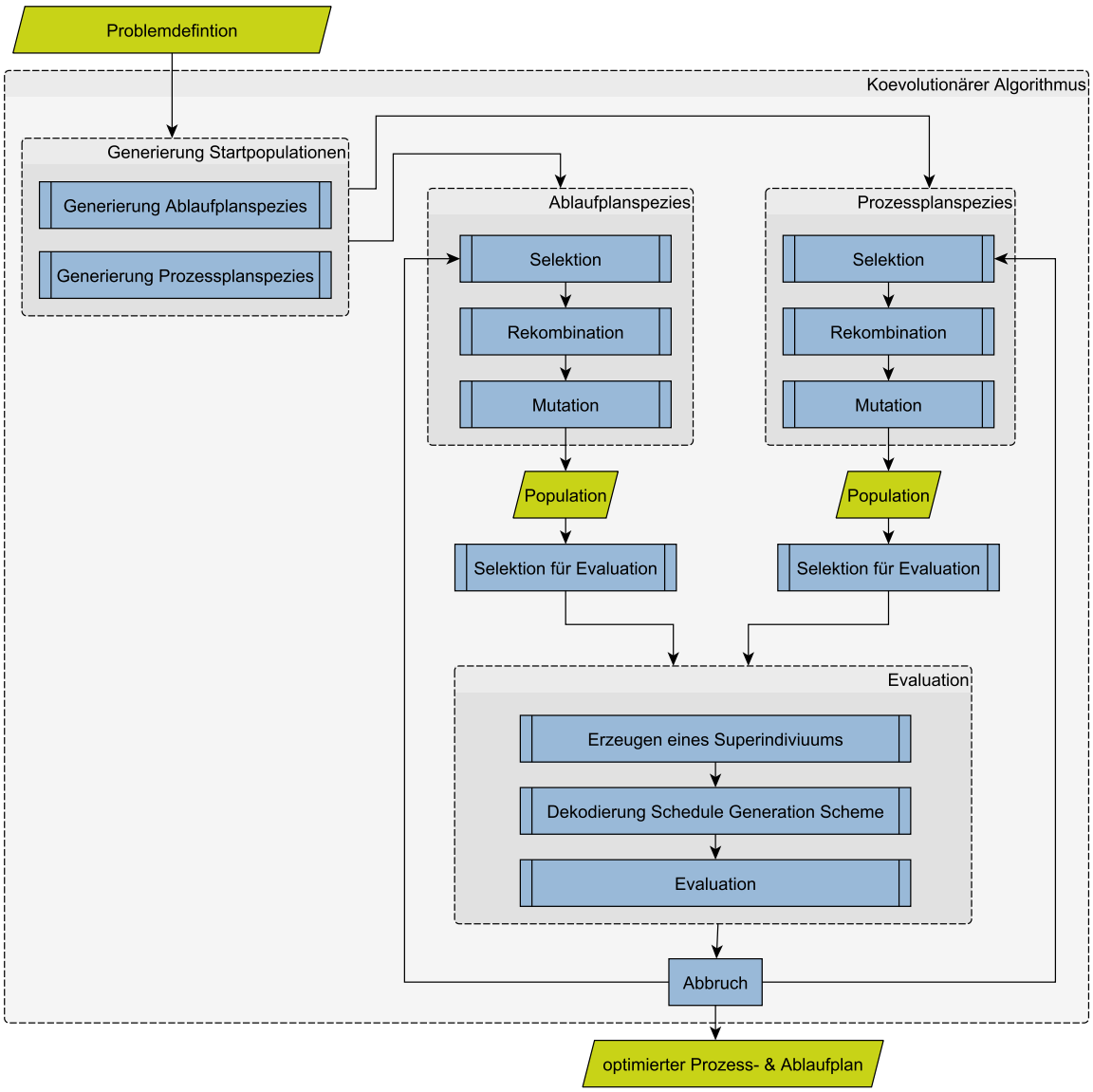


Abbildung 3.4.: Ablauf des Koevolutionären Algorithmus auf makroskopischer Ebene

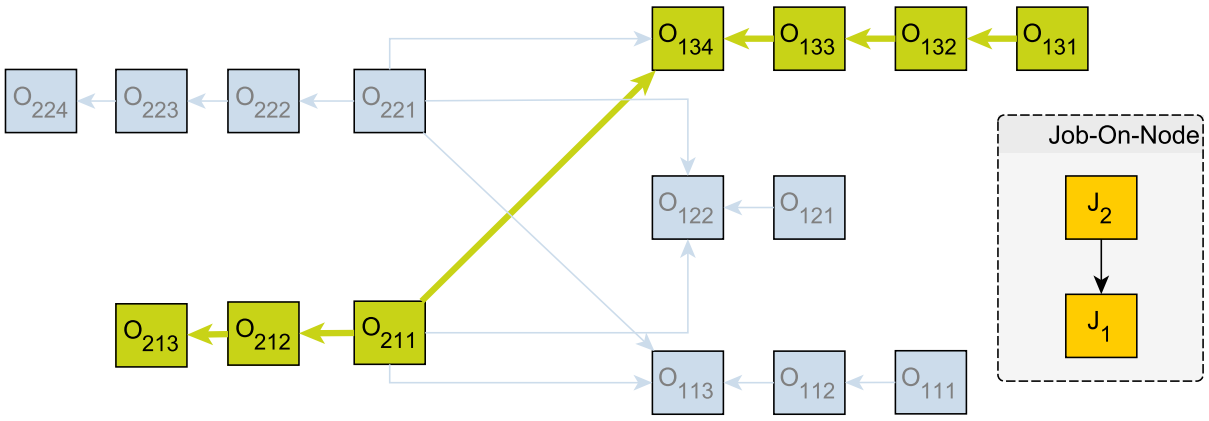


Abbildung 3.5.: *Activity-On-Node*-Diagramm für zwei Jobs J_1 und J_2, wobei die Routings $m^3_{J_1}$ und $m^1_{J_2}$ gewählt wurden.

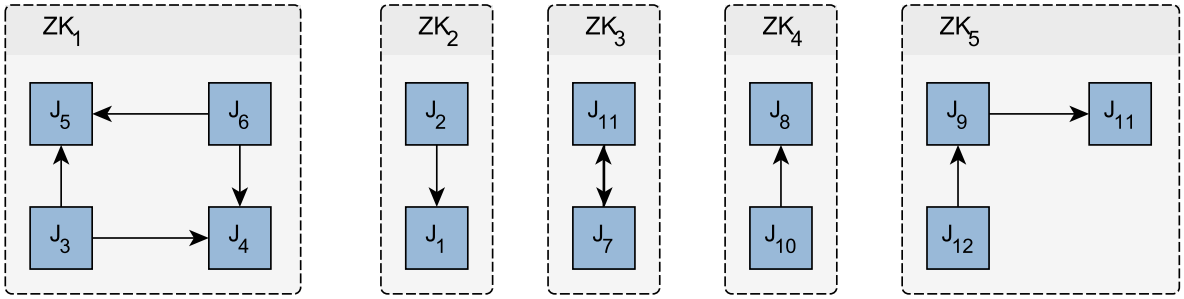


Abbildung 3.6.: Unterteilung des *Job-On-Node*-Diagramms in Zusammenhangskomponenten.

ZK_1	ZK_2	ZK_3	ZK_4	ZK_5	ZK_6	ZK_7	ZK_8
MA	MA	MMPT	MA	MDD	MA	MMPT	MDD

Abbildung 3.7.: Aufbau eines Chromosomes zur prioritätsregelbasierten Codierung der Prozessplanung

ZK_1	ZK_2	ZK_3	ZK_4	ZK_5	ZK_6	ZK_7	ZK_8
J_4	J_{12}	J_{98}	J_{56}	J_{77}	J_{43}	J_{14}	J_{22}
J_{72}	J_{11}	J_{31}	J_{48}			J_2	J_{28}
J_1		J_{79}	J_{51}				J_{13}
J_{82}							J_5

Abbildung 3.8.: Aufbau eines Chromosomes zur permutationsbasierten Codierung der Prozessplanung

$0-2\,\text{min}$	$2-4\,\text{min}$	$4-6\,\text{min}$	...	...	$476-478\,\text{min}$	$478-480\,\text{min}$
EST	LST	LST	...	...	LET	SPT

Abbildung 3.9.: Aufbau eines Chromosomes zur prioritätsregelbasierten Codierung der Ablaufplanung

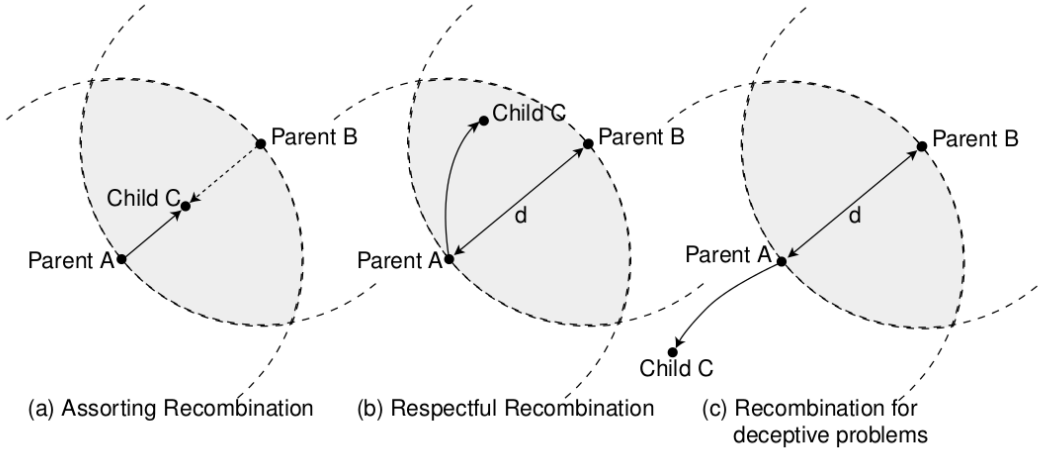


Abbildung 3.10.: Geometrische Visualisierung des Verhaltens von verschiedenen Rekombinationsoperatoren (aus Merz (2000))

$0-2\,\text{min}$	$2-4\,\text{min}$	$4-6\,\text{min}$	$6-8\,\text{min}$	...	$476-478\,\text{min}$	$478-480\,\text{min}$
EST	LST	LST	LVC	...	LET	SPT

Abbildung 3.11.: Wiederverwendung eines Chromosoms eines Individuums der Ablaufplanung. Das Zeitfenster zwischen $0-6\,\text{min}$ wurde bereits auf mikroskopischer Ebene eingeplant. Das Individuum kann einfach verkürzt werden, in dem die bereits eingeplanten Genorte nicht mehr berücksichtigt werden. Ein neues, verkürztes Chromosom wird generiert, das als Startlösung für den nächsten Schritt des Multiskalen-Algorithmus dient.

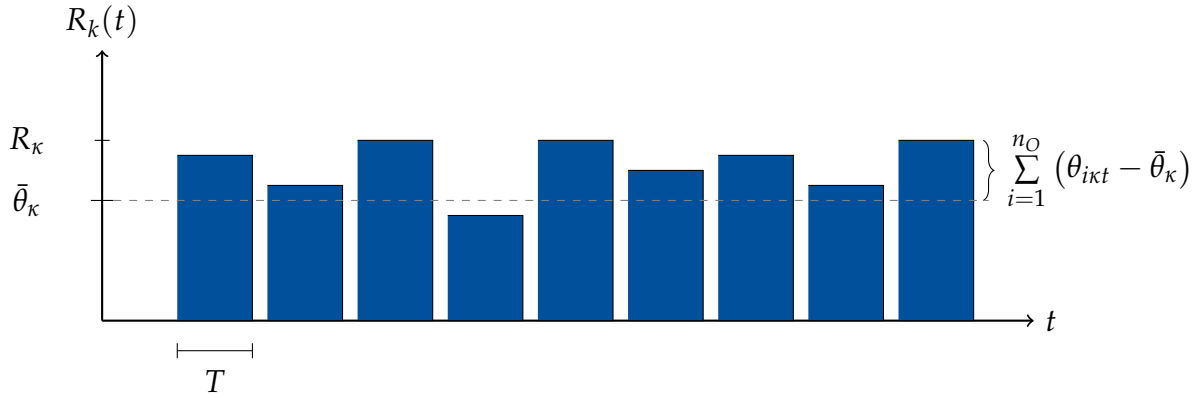


Abbildung 3.12.: Beschreibung des Resource-Leveling-Indexes. Der Index beschreibt die Summe der quadratischen Abweichungen vom Mittelwert.

3.3.9. Integration von Techniken der Fuzzy-Set-Theorie

Im Folgenden werden Ansätze diskutiert, die die Robustheit eines Ablaufplanes mit Hilfe der *Fuzzy-Set*-Theorie optimieren. Hierbei wird die stochastische Auslastungsformulierung aus Bode u. a. (2012) weiterentwickelt und auf die Possibilitätstheorie und die Arithmetik der Fuzzy-Zahlen übertragen und mit der Modellierung von unscharfen Auslastungen nach Masmoudi u. Haït (2013) kombiniert.

Die Ressourcennutzung einer Operation lässt sich über ihren Bedarf und ihre Start- sowie Endzeit bzw. Dauer bestimmen. Die unscharfe Dauer einer Operation impliziert eine ebenfalls unscharfe Auslastung der einzelnen an der Operation beteiligten Ressourcen.

Zunächst werden die Grundlagen der unscharfen Ablaufplanung unter Berücksichtigung eines unscharfen Auslastungsprofils betrachtet. Nachfolgend wird eine Analyse der Eigenschaften des vorgestellten Verfahrens vorgenommen.

Plausibilitätstheoretische Grundlagen

Die Possibilitätstheorie bietet die Möglichkeit, die *Fuzzy-Set*-Theorie in eine axiomatische Struktur einzubetten, die derjenigen der Wahrscheinlichkeitstheorie ähnelt. In Dubois u. Prade (1999) wird dies ausführlich diskutiert.

Eine *Möglichkeitsverteilungsfunktion* $\pi_x(\xi)$ beschreibt den Grad der Möglichkeit in einem Intervall $[0,1]$, dass eine Variable $x \in \mathbb{R}$ einen bestimmten Wert ξ annimmt. Die beiden Grenzfälle $\pi_x(\xi) = 0$ und $\pi_x(\xi) = 1$ beschreiben die Tatsache, dass $x = \xi$ unmöglich, bzw. $x = \xi$ voll und ganz möglich ist. Hierbei sei erwähnt, dass sich eine Möglichkeitsverteilungsfunktion mit einer Zugehörigkeitsfunktion einer Fuzzy-Menge vergleichen lässt. Es handelt sich folglich um eine Abbildung:

$$\pi_x(\xi) := X \rightarrow [0,1], \tag{3.8}$$

wobei X die Menge aller möglichen Werte von x darstellt. Weiterhin lassen sich zwei Maße definieren, die Möglichkeit und die Notwendigkeit.

Definition 29: Möglichkeit

Betrachtet man die nichtleere Menge $A \subset \mathbb{R}$ und die Variable x mit der Möglichkeitsverteilungsfunktion $\Pi_x(\xi)$, so lässt sich ein Maß definieren, dass die Möglichkeit angibt, dass $x \in A$ gilt:

$$\Pi_x(A) = \sup_{\xi \in A} \pi_x(\xi) \tag{3.9}$$

Zusammenfassend bedeutet dies, dass sich mit Hilfe der Möglichkeitsverteilungsfunktion $\pi_x(\xi)$ angeben lässt, mit welcher Möglichkeit die Variable x einen Wert ξ annimmt, der Teil der Menge A ist. Die Möglichkeit, dass $x \in A$ gilt, ergibt sich folglich aus dem Supremum aller möglichen Werte für $\xi \in A$. Die Möglichkeit lässt sich auch als Maß für die Plausibilität betrachten.

Definition 30: Notwendigkeit

Die Notwendigkeit $N_x(A)$, dass $x \in A$ ist, ergibt sich zu

$$N_x(A) = 1 - \Pi_x(\bar{A}), \tag{3.10}$$

wobei $\bar{A}$ das Komplement der Menge A angibt. Die Notwendigkeit definiert folglich ein Maß für die Sicherheit, dass $x \in A$.

Weiterhin lassen sich zwei Axiome für die Vereinigung und den Durchschnitt definieren. So gilt

$$\Pi(A \cup B) = \max(\Pi(A), \Pi(B)) \tag{3.11}$$

und

$$N(A \cap B) = \min(N(A), N(B)). \tag{3.12}$$

Die Möglichkeit, dass $x \in (A \cup B)$ gilt, ergibt sich aus dem Maximum der Möglichkeiten, dass $x \in A$ und $x \in B$ gilt.

Über die einfache Zuordnung des Possibilitätsmaßes zur Möglichkeitsverteilungsfunktion

$$\Pi_x(\xi) = \mu_x(\xi) \tag{3.13}$$

werden die *Fuzzy-Set*-Theorie und die Plausibilitätstheorie miteinander in Beziehung gesetzt. Dieser Zusammenhang ist ähnlich dem bei der Definition eines Wahrscheinlichkeitsmaßes über Wahrscheinlichkeitsdichtefunktion in der Stochastik.

Definition 31: Fuzzy-Grenzwerte

Es lassen sich bezüglich der Möglichkeit und der Notwendigkeit verschiedene Grenzwerte definieren. Hierzu betrachtet man ein Fuzzy-Intervall $\tilde{A}$ für das $m_2 = +\infty$ und $c_2 = +\infty$ bzw. $m_1 = -\infty$ und $c_1 = -\infty$ gilt sowie eine reale Zahl $\tau \in \tilde{A}$. Die Grenzwertbetrachtung für eine reelle Zahl t ergibt sich zu:

$$\Pi(t \leq \tau) = \mu_{(-\infty,\tilde{A}]}(t) = \sup_{u \geq r} \mu_{\tilde{A}}(u) \tag{3.14}$$

$$N(t \leq \tau) = \mu_{(-\infty,\tilde{A}[}(t) = \inf_{u < r} 1 - \mu_{\tilde{A}}(u) \tag{3.15}$$

$$\Pi(t \geq \tau) = \mu_{[\tilde{A},+\infty)}(t) = \sup_{u \leq r} \mu_{\tilde{A}}(u) \tag{3.16}$$

$$N(t \geq \tau) = \mu_{]\tilde{A},+\infty)}(t) = \inf_{u > r} 1 - \mu_{\tilde{A}}(u) \tag{3.17}$$

Abbildung 3.13 stellt die Grenzwerte nochmals graphisch dar.

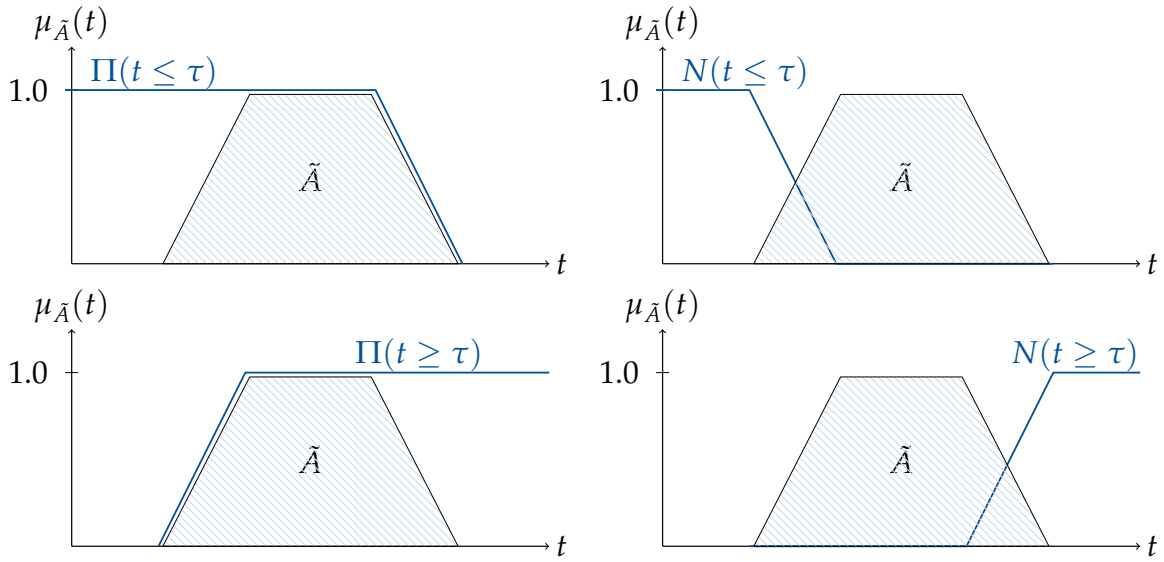


Abbildung 3.13.: Grenzwerte der Möglichkeit und der Notwendigkeit von Fuzzy-Zahlen.

Definition 32: Möglichkeit und Notwendigkeit des Ereignisses $\tau \leq t \leq \sigma$

Sind zwei Fuzzy-Mengen $\tilde{A}$ und $\tilde{B}$ gegeben, lassen sich für die Möglichkeit und die Notwendigkeit des Ereignisses $\tau \leq t \leq \sigma$, wobei $\tau \in \tilde{A}$ und $\sigma \in \tilde{B}$ gilt, folgende Ausdrücke herleiten:

$$\Pi(\tau \leq t \leq \sigma) = \mu_{[\tilde{A};\tilde{B}]}(t) = \mu_{[\tilde{A},+\infty)\cap(-\infty,\tilde{B}]} \tag{3.18}$$

$$N(\tau \leq t \leq \sigma) = \mu_{]\tilde{A};\tilde{B}[}(t) = \mu_{]\tilde{A},+\infty)\cap(-\infty,\tilde{B}[} \tag{3.19}$$

Abbildung 3.14 stellt den Verlauf dieser beiden Größen exemplarisch dar.

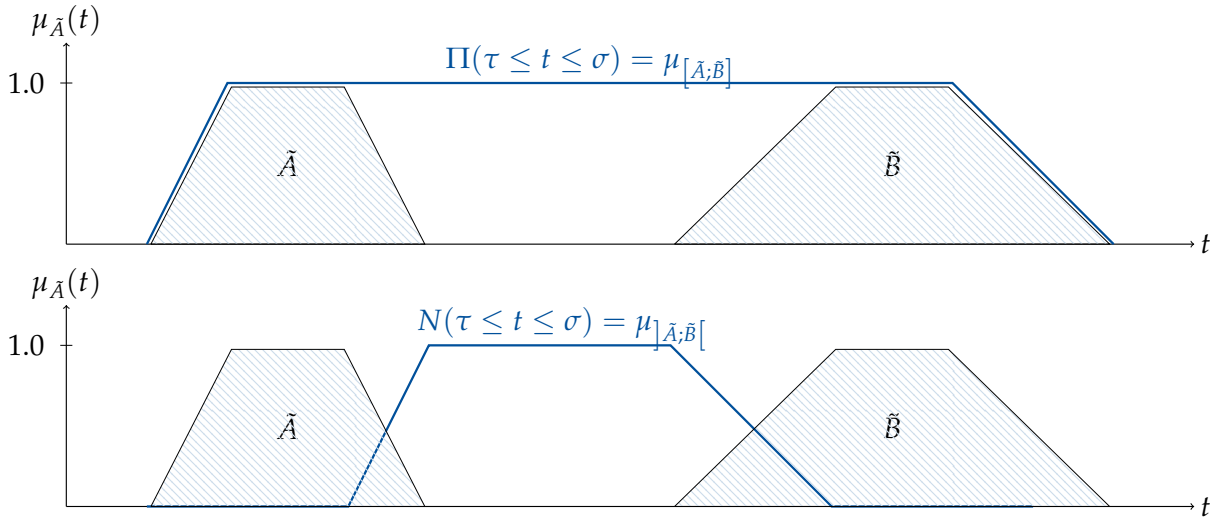
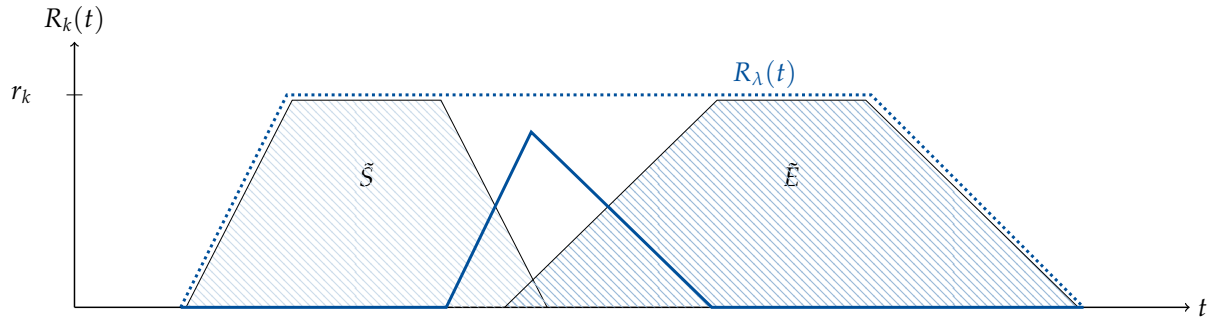


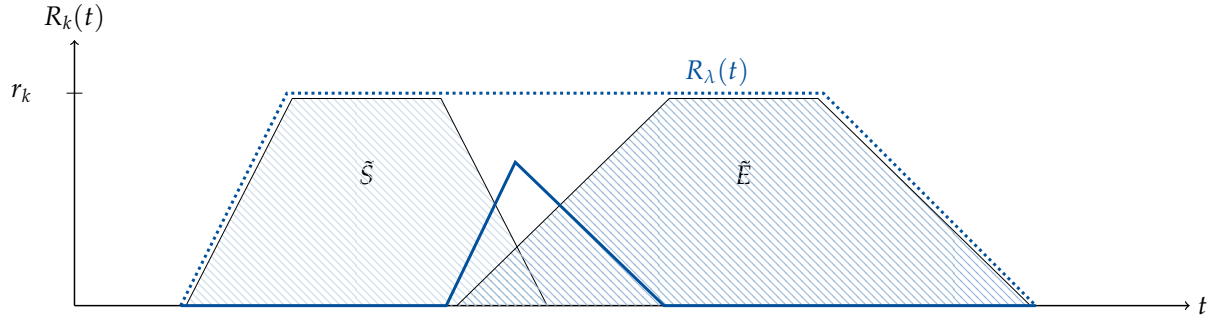
Abbildung 3.14.: Darstellung der Möglichkeit $\Pi(\tau \leq t \leq \sigma)$ und der Notwendigkeit $N(\tau \leq t \leq \sigma)$

Die Dauer einer Operation und deren Unschärfe haben einen großen Einfluss auf den Verlauf der Notwendigkeit und der Möglichkeit. Abbildung 3.15 zeigt den Einfluss der Dauer auf die Überlappung der Start- und der Endzeit einer Operation. Bei einer Vergrößerung der Überlappung verringert sich die Notwendigkeit, dass die Operation ausgeführt wird.

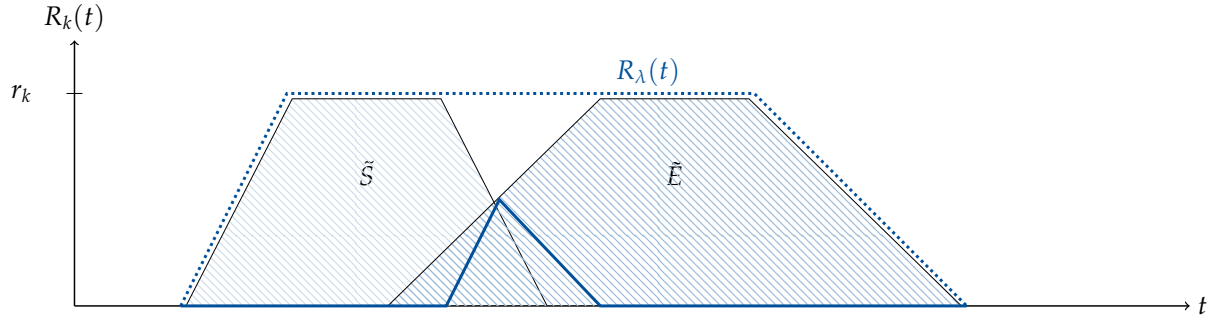
Zwischen der Möglichkeit Π und der Notwendigkeit N lässt sich eine affine Kombination definieren. Diese gibt die sogenannte Glaubwürdigkeit an.



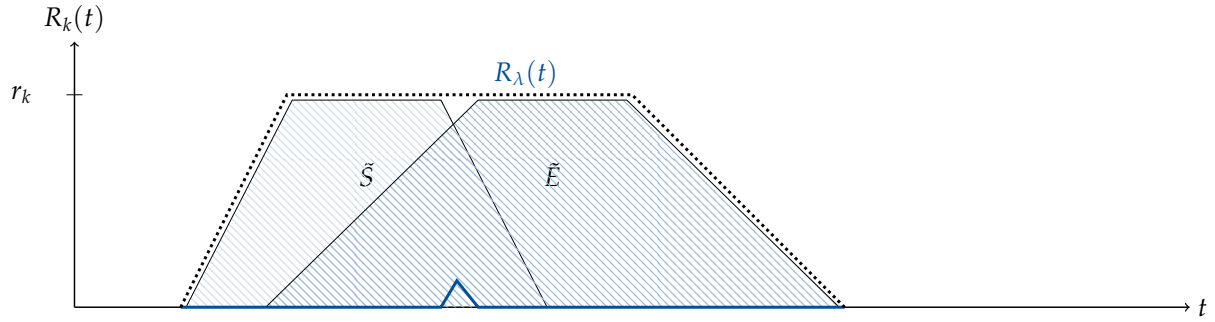
(a)



(b)



(c)



(d)

Abbildung 3.15.: Verlauf der Notwendigkeit in Abhängigkeit der Operationsdauer. Unterschiedliche Konfigurationen, bei denen die Dauer immer weiter abnimmt von a.) nach d.). Je geringer die Dauern, desto mehr überlappen sich Start und Ende und desto weniger groß ist der Bereich, in dem für die Notwendigkeit $\neq 0$ gilt.

Definition 33: Glaubwürdigkeit des Ereignisses $\tau \leq t \leq \sigma$

Die Glaubwürdigkeit des Ereignisses $\tau \leq t \leq \sigma$ lässt sich als gewichtete Summe zwischen der Möglichkeit und der Notwendigkeit mit dem Gewicht λ definieren (siehe Liu (2006)). Die Glaubwürdigkeit ergibt sich somit zu:

$$Cr(\tau \leq t \leq \sigma) = \lambda \cdot \Pi(t) + (1 - \lambda) \cdot N(t) \tag{3.20}$$

mit $\lambda \in [0, 1]$. Ein Wert nahe $\lambda = 1$ beschreibt eine pessimistischere, ein Wert nahe $\lambda = 0$ eine optimistischere Gewichtung.

Die Wahl eines geeigneten λ liegt später beim Entscheidungsträger. Im Rahmen der Ablaufplanung wird die *Fuzzy-Set*-Theorie im Folgenden auf Operationen und deren Startzeit, Dauer und Endzeit angewandt.

Unscharfe Ablaufplanung

Im Folgenden wird ausgehend von plausibilitätstheoretischen Grundlagen ein Verfahren zur Ablaufplanung unter Unsicherheitsaspekten auf Basis der Arbeit von Masmoudi u. Haït (2013) vorgestellt und erweitert. Hierzu wird zunächst eine unscharfe Auslastung definiert, die wiederum im Rahmen einen SGS Verwendung findet, der bezüglich der unscharfen Parameter angepasst wurde.

Fuzzy-Auslastung Besitzt eine Operation einen unscharfen Startzeitpunkt $\tilde{S}$ und eine unscharfe Dauer $\tilde{D}$, so ergibt sich ein ebenfalls unscharfes Ende $\tilde{E} = \tilde{S} + \tilde{D}$. Nun lassen sich zu jedem beliebigen α-Cut Auslastungen für die einzelnen benötigten Ressourcen bestimmen. Masmoudi u. Haït (2013) führten eine neue Technik zum Umgang ein, bei der sowohl der Ablaufplan als auch die zugehörigen Auslastungsprofile im Verlauf der Ablaufplanung weiterhin unscharf betrachtet werden. Diese Betrachtungsweise, die Masmoudi u. Haït (2013) anhand von Vier-Punkt-Fuzzy-Intervallen beschreiben, wird im Folgenden diskutiert und um einige Aspekte erweitert. So wird eine allgemeinere Formulierung auf Basis der oben eingeführten affinen Kombination gewählt und eine einheitliche Wahl des λ-Niveaus durch eine Skalierung eingeführt.

Definition 34: Fuzzy-Präsenz von Operationen

Die Fuzzy-Präsenz einer Operation entspricht der Glaubwürdigkeit $Cr(\tau \leq t \leq \sigma)$ mit $\tau \in \tilde{S}$ und $\sigma \in \tilde{E}$. Masmoudi u. Haït (2013) interpretieren dies als Wahrscheinlichkeit $P(t)$, dass eine Operation zum Zeitpunkt t stattfindet.

Die Aussage über die Fuzzy-Präsenz einer Operation muss nun auf die Ressourcenauslastung übertragen werden. Liegen die Ressourcenanforderung einer Operation O mit der Dauer D an die Ressource k bei einem scharfen Wert von r_k, so ergibt sich die scharfe Auslastung zu $A = r_k \cdot D$.

Aus einer unscharfen Dauer $\tilde{D} = (m_{\tilde{D}}, \alpha_{\tilde{D}}, \beta_{\tilde{D}})_{LR}$ folgt, dass sich die resultierende Auslastung als Produkt aus Ressourcenbedarf und einem Wert, der zwischen den Grenzen des Trägers der unsicheren Dauer liegt, ergibt. Man spricht bei den Grenzfällen von der optimistischen Auslastung $W_{min} = r_k \cdot D_{min}$ mit $D_{min} = m_{\tilde{D}} - \alpha_{\tilde{D}}$ und der pessimistischen Auslastung $W_{max} = r_k \cdot D_{max}$ mit $D_{max} = m_{\tilde{D}} + \beta_{\tilde{D}}$.

Definition 35: Fuzzy-Ressourcenbedarf

Der Fuzzy-Ressourcenbedarf einer Operation O an einer Ressource k ergibt sich durch die Skalierung der Präsenzverteilung $P_\lambda(t)$ der Operation mit dem benötigten scharfem Ressourcenbedarf r_k. Es gilt

$$R_{k,\lambda}(t) = P_\lambda(t) \cdot r_k \tag{3.21}$$

mit $\lambda \in [0,1]$, welches entsprechend vom Entscheidungsträger zu wählen ist.

Zur weiteren Betrachtung wird das Integral des Ressourcenbedarfs über die unscharfe Operationsdauer benötigt.

Definition 36: Gesamt-Fuzzy-Ressourcenbedarf einer Operation

Der Gesamt-Fuzzy-Ressourcenbedarf $W_{\lambda,k}$ einer Operation O an einer Ressource k ergibt sich aus dem Integral des Fuzzy-Ressourcenbedarfs über die unscharfe Operationsdauer $\tilde{D}$ der Operation O. Folglich gilt:

$$W_{k,\lambda} = \int_{-\infty}^{+\infty} P_\lambda(t) \cdot r_k dt = \int_{-\infty}^{+\infty} R_{k,\lambda}(t) dt, \tag{3.22}$$

wobei ein entsprechendes λ zu wählen ist.

Abbildung 3.16 zeigt für einen exemplarischen Ressourcenverlauf das Integral, das den Gesamt-Fuzzy-Ressourcenbedarf beschreibt.

Bei der Wahl des Parameters λ muss beachtet werden, dass bei den Extrema $\lambda = 0$ und $\lambda = 1$ das berechnete Integral W_k für die Auslastung der Ressource k die minimale Auslastung $W_k^{min} = r_k \cdot D_{min}$ und die maximale Auslastung $W_k^{max} = r_k \cdot D_{max}$ unter- bzw. überschreiten kann.

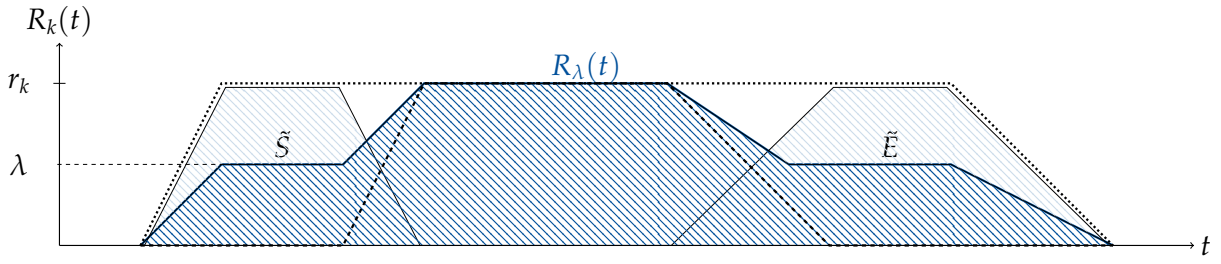


Abbildung 3.16.: Ressourcenprofil einer Ressource k, die durch eine Operation O ausgelastet ist. Das Integral $W_{k,\lambda}$ ist farblich hervorgehoben.

Daher wird ein λ_{min} und ein λ_{max} bestimmt, welche den Anforderungen bezüglich der maximalen bzw. minimalen Auslastung genügen:

$$\lambda_{min} = \frac{W^{min} - W_N}{W_\Pi - W_N} \tag{3.23}$$

$$\lambda_{max} = \frac{W^{max} - W_N}{W_\Pi - W_N} \tag{3.24}$$

Hierbei ist zu beachten, dass W_Π und W_N analog zum Gesamt-Fuzzy-Ressourcenbedarf einer Operation über die Möglichkeit und die Notwendigkeit definiert sind:

$$W_\Pi = \int_{-\infty}^{+\infty} \Pi(t) \cdot r_k \quad dt \tag{3.25}$$

$$W_N = \int_{-\infty}^{+\infty} N(t) \cdot r_k \quad dt \tag{3.26}$$

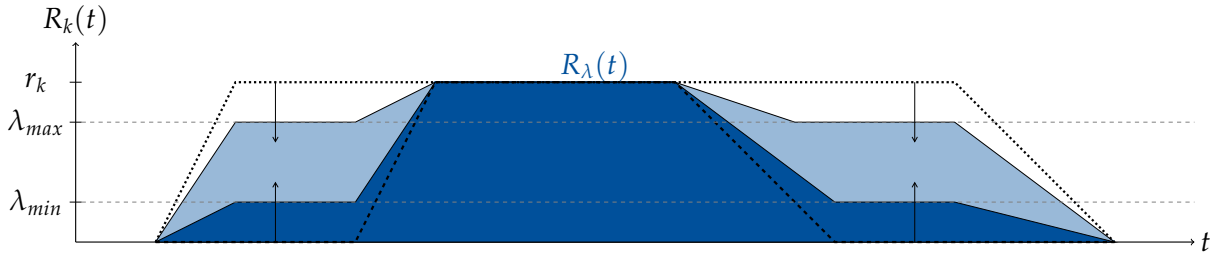


Abbildung 3.17.: Eingeschränktes Ressourcenprofil, das die Grenzfälle des Gesamt-Fuzzy-Ressourcenbedarfs einer Operation berücksichtigt.

Abbildung 3.17 zeigt exemplarisch die Auswirkungen der Einschränkung des Ressourcenprofils. Zur Vereinfachung lässt sich eine neue affine Kombination zwischen der Notwendigkeit und der Möglichkeit definieren, die auf das Intervall $[\lambda_{min}, \lambda_{max}]$ normiert ist.

Wahl eines Pessimismus-Levels Im Folgenden wird ein Pessimismus-Level eingeführt, das den Vorteil besitzt, dass die vom Entscheidungsträger gewählte Glaubwürdigkeit unabhängig vom Start- und Endzeitpunkt einer Operation gewählt werden kann. Das λ-Niveau muss immer an die entsprechenden minimalen und maximalen Werte angepasst werden. Eine Formulierung über den Pessimismus-Level ermöglicht eine Festlegung auf eine Glaubwürdigkeit für die betrachtete Operation, ohne den zum Einplanzeitpunkt zur Verfügung stehenden Spielraum kennen zu müssen.

Dazu werden zunächst die beiden extremen Verläufe

$$R_{k,min} = \lambda_{min} \cdot \Pi(t) + (1 - \lambda_{min}) \cdot N(t) \tag{3.27}$$

und

$$R_{k,max} = \lambda_{max} \cdot \Pi(t) + (1 - \lambda_{max}) \cdot N(t) \tag{3.28}$$

betrachtet.

Definition 37: Pessimismus-Level

Das Pessimismus-Level ρ drückt aus, wie pessimistisch bzw. optimistisch bei der Festlegung der zu erwartenden Auslastung vorgegangen wird. Der Fuzzy-Ressourcenbedarf einer Operation O an einer Ressource k liegt innerhalb des Intervalls $[R_{k,min}, R_{k,max}]$. Die zu erwartende Auslastung liegt je nach gewählten Pessimismus-Level ρ bei

$$R_{k,\rho}(t) = \rho \cdot R_{k,max} + (1 - \rho) \cdot R_{k,min} \tag{3.29}$$

mit $\rho \in [0, 1]$, welches wiederum entsprechend vom Entscheidungsträger zu wählen ist.

Die Wahl des Parameters λ bzw. des einheitlichen Parameters ρ ist eine der Kernaufgaben, die der Entscheidungsträger zu fällen hat. Die Wahl beeinflusst den Auslastungsgrad zum Teil erheblich. Um dies zu verdeutlichen wird ein unscharfer Startzeitpunkt $\tilde{S} =< 6, 10, 15, 19 >$ und eine unscharfe Dauer $\tilde{D} =< 25, 30, 40, 45 >$ und somit das unscharfe Ende $\tilde{E} =< 31, 40, 55, 64 >$ für eine Operation festgelegt (siehe Abbildung 3.18).

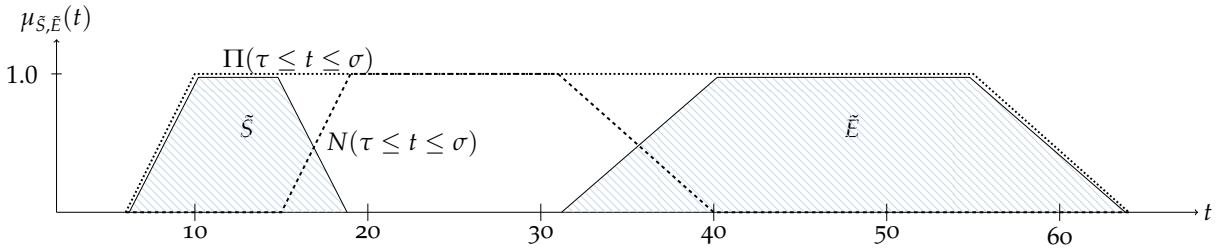


Abbildung 3.18.: Beispielhafte Operation und deren Start, Ende sowie Möglichkeit und Notwendigkeit

Nun wird zunächst der Parameter ρ zwischen 0.0 und 1.0 variiert und die Auslastung farblich über die Zeit dargestellt (siehe Abbildung 3.19). Der Bereich, an dem die Notwendigkeit bei 1.0 liegt, lässt sich im gesamten Diagramm schnell wiedererkennen.

Der Parameter ρ steht direkt in Relation zur unscharfen Dauer $\tilde{D}$ einer Operation. Ein $\rho = 0$ beschreibt den optimistischsten Fall, dass die Operation die geringste Dauer $c_{1\tilde{D}}$ in Anspruch nimmt, analog beschreibt $\rho = 1$ den pessimistischsten Fall, dass die Operation die Zeitdauer $c_{2\tilde{D}}$ benötigt. Der Parameter verläuft so über den gesamten Träger der Fuzzy-Zahl beziehungsweise des Fuzzy-Intervalls der Operationsdauer. Daher lässt sich der Parameter ρ direkt auf die Zugehörigkeitsfunktion $\mu_{\tilde{D}}$ beziehen. So kann man die Auslastung in Abhängigkeit des Parameters ρ mit dem Funktionswert der Zugehörigkeitsfunktion gewichten. Die sich ergebene Funktion (siehe Abbildung 3.20) kann man als zu erwartende Auslastung in Abhängigkeit vom Pessimismus-Level auffassen.

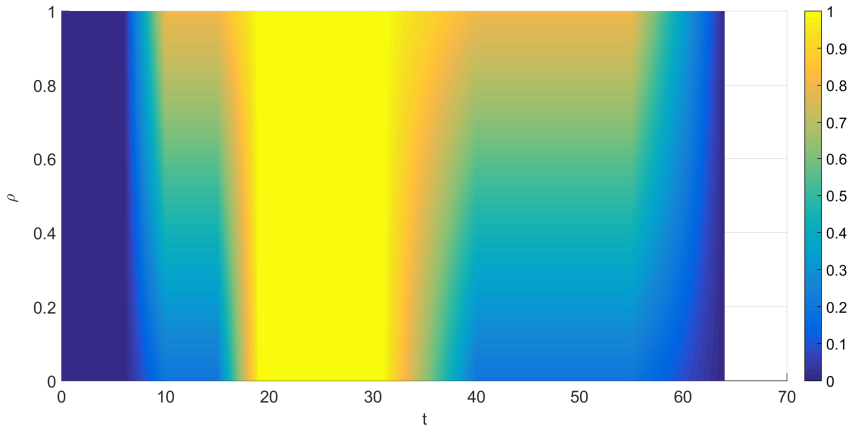


Abbildung 3.19.: Einfluss des Pessimismus-Levels - Beispielhaftes Ressourcenprofil $R_\kappa(t,\rho)$. Farbliche Darstellung von verschiedenen Pessimismus-Levels (vgl. Abbildung 3.18)

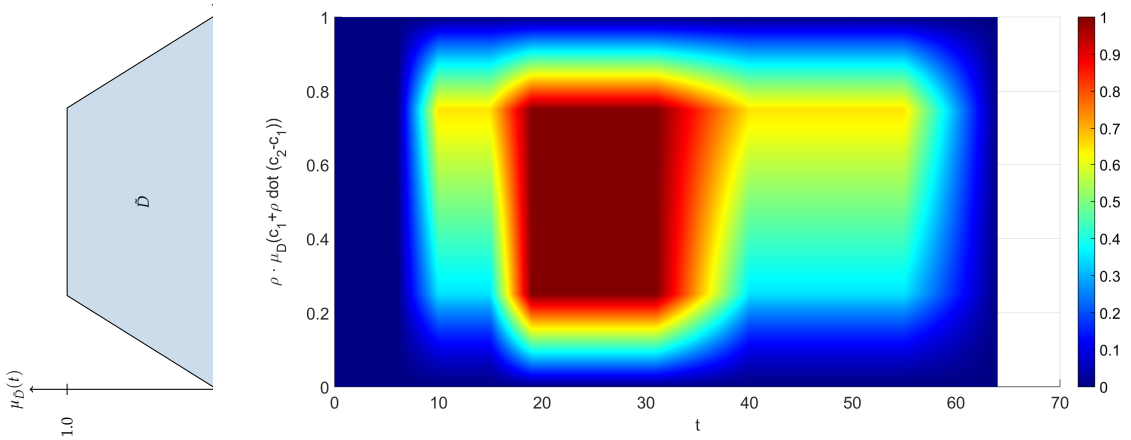


Abbildung 3.20.: Einfluss des Pessimismus-Levels - Ressourcenprofil $R_\kappa(t,\rho)$, skaliert mit der entsprechenden Zugehörigkeit $\mu_{\tilde{D}}(c_1 + \rho \cdot (c_2 - c_1))$, wobei die Dauer als Fuzzy-Intervall $\tilde{D} = (m_{\tilde{D}1}, m_{\tilde{D}2}, \alpha_{\tilde{D}}, \beta_{\tilde{D}})_{LR}$ mit den Grenzen $c_1 = m_{\tilde{D}1} - \alpha_{\tilde{D}}$ und $c_2 = m_{\tilde{D}2} + \beta_{\tilde{D}}$ definiert ist.

Ablaufplanerzeugung Die unscharfe Auslastung, die durch eine Operation mit einem unscharfen Startzeitpunkt und einer unscharfen Dauer impliziert wird, muss nun auf alle Operationen im Rahmen eines ressourcenbeschränkten Ablaufplanproblems angewandt werden. Hierzu wird ein angepasster *SGS* benötigt, der die Unschärfe in den Zeitangaben sowie der Ressourcenauslastung berücksichtigt.

Als einfachste Annahme kann zunächst davon ausgegangen werden, dass eine Operation immer zum frühestmöglichen Zeitpunkt unter Berücksichtigung aller technologischen und kapazitiven Restriktionen eingeplant werden soll. Die Operationen folgen einer Ende-Start-Beziehung, folglich wird im Rahmen der Generierung des Ablaufplans versucht, den unscharfen Start der nachfolgenden Operation auf das unscharfe Operationsende der vorausgegangenen Operation zu setzen.

Die resultierende Gesamtauslastungsverteilung überschreitet möglicherweise die Kapazität einer oder mehrerer Ressourcen. Um dies zu vermeiden, können entweder lokale Verfahren angewandt werden, die Veränderungen am Ressourcenprofil des Fuzzy-Ressorcenbedarfs vornehmen oder globale Verfahren. Zu den globalen Verfahren zählt beispielsweise das Tauschen von Operationen unter Berücksichtigung der technologischen Vorschriften, wie es im Rahmen eines Genetischen Algorithmus durchgeführt wird.

Der Algorithmus 3.4 enthält die Anpassungen, die an einem SGS auf Basis einer Aktivitätsliste vorgenommen werden müssen, damit die Unschärfe in den Zeitangaben sowie der Ressourcenauslastung anhand des vorgestellten Verfahrens berücksichtigt werden kann.

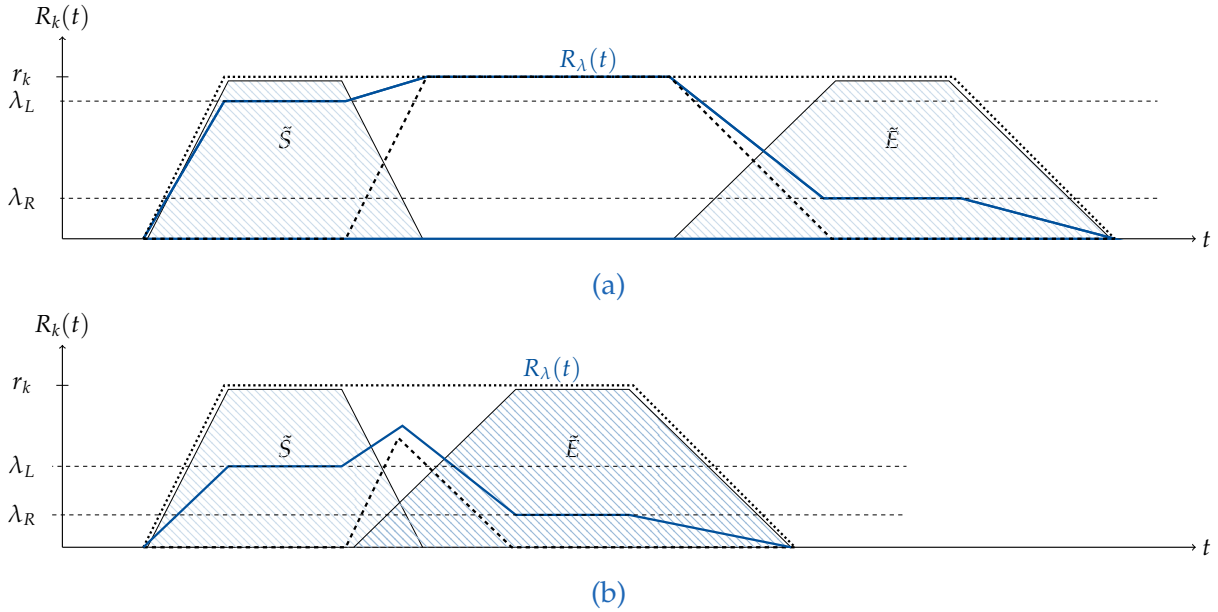


Abbildung 3.21.: Asymmetrische Ressourcenprofile

Zu den lokalen Anpassungen zählt die Aufteilung des Fuzzy-Ressourcenbedarfs $R_{k,\rho_i}(t)$ in einen linken und einen rechten Bereich. Ist eine Einplanung des entstehenden symmetrischen Ressourcenprofils $R_{k,\rho_i}(t)$ der Operation nicht möglich, so wird versucht, einen asymmetrischen Verlauf einzuplanen (siehe Abbildung 3.21). Hierbei wird ausgehend von einem Startwert λ_L^{Start} iterativ ein λ_L und ein entsprechendes λ_R gesucht, welches die kapazitiven Grenzen nicht überschreitet. Dabei ist zu berücksichtigen, dass der resultierende Gesamt-Fuzzy-Ressourcenbedarf $W_{k,\lambda_L,\lambda_R}$ dem durch das Pessimismus-Level vorgegebenen integralen Wert W_{k,ρ_i} entspricht.

Wurde ein passendes asymmetrisches Profil gefunden, so wird die entsprechende Operation eingeplant. Sollte dies nicht der Fall sein, so wird die Operation nicht eingeplant und zum nachfolgenden Entscheidungspunkt erneut untersucht.

Zur Bestimmung des nächsten Einplanzeitpunktes unterscheiden Masmoudi u. Haït (2013) zwischen zwei Fällen. Zunächst führen sie dazu einen Maximum-Operator für Vier-Punkt-Fuzzy-Intervalle ein:

$$max(\tilde{A},\tilde{B}) =< \max(c_1^{\tilde{A}}, c_1^{\tilde{B}})\, \max(m_1^{\tilde{A}}, m_1^{\tilde{B}}), \max(m_2^{\tilde{A}}, m_2^{\tilde{B}}), \max(c_2^{\tilde{A}} + c_2^{\tilde{B}}) > \tag{3.30}$$

Sind alle Operationen der Menge $E_{\tilde{t}}$, die alle Operationen enthält, die zum aktuellen Zeitpunkt eingeplant werden können, tatsächlich eingeplant worden, wird im Sinne der schwachen Ordnungsrelation (siehe A.1) zunächst die kleinste Fuzzy-Zahl $\tilde{l}_t$ unter den Startzeitpunkten der Operationen in $A_{\tilde{t}}$ und den Endzeiten der Operationen im *Active-Set* $\mathcal{A}(\tilde{t})$ zum Zeitpunkt $\tilde{t}$ bestimmt. Zur Bestimmung des neuen Einplanzeitpunktes wird der Maximum-Operator auf den aktuellen Zeitpunkt t sowie $\tilde{l}_t$ angewandt. Sind nicht alle Operationen eingeplant worden, so wird ausgehend vom linken Ende des Supports der aktuellen Zeit $c_1^{\tilde{t}}$ inkrementell eine Verschiebung $c_1^{\tilde{t}} + \Delta t$ vorgenommen und somit eine scharfe Zahl gebildet. Mittels des oben eingeführten Maximum-Operators wird der neue Zeitpunkt $\tilde{t}_{neu} = \max(\tilde{t}, c_1^{\tilde{t}} + \Delta t)$ bestimmt. Dadurch verringert sich die Unschärfe in der momentanen Startzeit, da der vordere Bereich abgeschnitten wird. Das ist durchaus vertretbar, da diese Verschärfung der Startzeit aus kapazitiven Gründen geschieht. Abbildung 3.22 verdeutlicht die Arbeitsweise des Algorithmus.

Dieses Vorgehen lässt sich auf Job-Shop-Probleme, bei denen wenig Interaktion zwischen den einzelnen Maschinen herrscht, einfach überführen, indem für jede Maschine eine einzelne Startzeit mitgeführt wird.

Algorithmus 3.4 Paralleler SGS auf Basis einer Aktivitätsliste für unscharfe Ablaufplanprobleme

Eingabe: $O := (O_{23}, O_5, ..., O_{12}, ..., O_4)$ ▷ Aktivitätsliste
Eingabe: $O_i \mapsto \left\{ (\widetilde{ES_i}, \widetilde{LS_i}, \widetilde{EE_i}, \widetilde{LE_i}) \right\}$ ▷ Ausführungsfenster der Operationen
Eingabe: $O_i \mapsto \{\rho_i\}$ ▷ Pessimismus-Level der Operationen
$\tilde{t} = \min_{i \in V}(\widetilde{ES_i})$
Bestimme die Menge $E_{\tilde{t}}$ von Operationen O_i, die keine Vorgängeroperationen haben und zum aktuellen Zeitpunkt t eingeplant werden können.
Bestimme die Menge $A_{\tilde{t}}$ von Operationen O_i, die keine Vorgängeroperationen haben, aber noch nicht eingeplant werden können aufgrund ihres frühsten Startzeitpunktes.
while nicht alle Operationen eingeplant sind **do**
  CurrentTimeStepLoop:
  while $E_t \neq \emptyset$ **do**
    Wähle Operation $O_j \in E_{\tilde{t}}$, die an minimaler Position in O steht.
    Bestimme das symmetrische Ressourcenprofil $R_{k,\rho}(t)$
    if eine Ressource k mit $R_{k,\rho}(\tau) > R_k(\tau)$ für ein $\tau \in [t, t + p_j[$ existiert **then**
      Bestimme Gesamt-Fuzzy-Ressourcenbedarf W_{k,ρ_i}
      Initialisiere $\lambda_L = \lambda_L^{Start}$
      while $\lambda_L > 0$ **do**
        Bestimme λ_R, so dass $W_{k,\lambda_L,\lambda_R} = W_{k,\rho_i}$ gilt.
        if kein passendes λ_R bestimmbar ist **then**
          $\lambda_L = \lambda_L + \delta\lambda$
        else
          Plane die Operation j ein.
          CONTINUE CurrentTimeStepLoop
        end if
      end while
      CONTINUE CurrentTimeStepLoop
    else
      Plane die Operation j ein.
    end if
  end while
  bestimme neues t
  if alle Operationen aus $E_{\tilde{t}}$ eingeplant **then**
    $\tilde{t} = \max(\tilde{t}, \tilde{l}_t)$
  else
    $\tilde{t} = \max(\tilde{t}, c_1^{\tilde{t}} + \Delta t)$
  end if
  aktualisiere $E_{\tilde{t}}$ und $A_{\tilde{t}}$
end while
return Schedule S

$\tilde{l}_t$ kleinste Wert im Sinne der schwachen Ordnungsrelation aus den frühsten Startzeiten ES_i der Operationen $O_i \in A_t$ und den Endzeiten der Operationen in $\mathcal{A}(\tilde{t})$

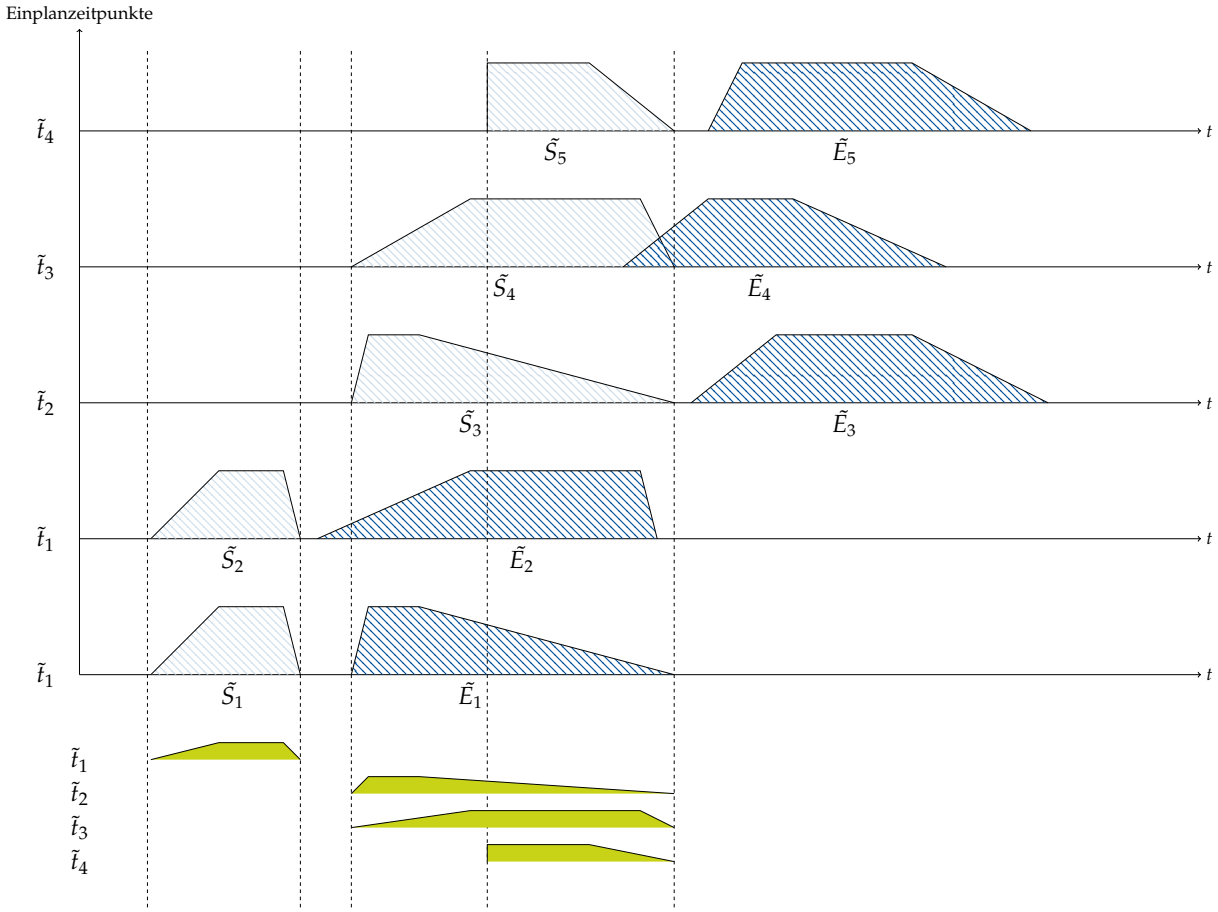


Abbildung 3.22.: Exemplarische Einplanung von unscharfen Operationen im Rahmen eines Single-Machine-Problems.

Maximierung der Robustheit

Die in 2.1.4 beschriebenen Zeitfenster für die Ausführung von Operationen unterliegen in vielen Anwendungen Unsicherheiten. So ist zumeist der frühestmögliche Start einer Operation nicht sicher, ebenso wie ihre Dauer.

Häufig ist die Einhaltung von Fälligkeitsterminen eine der Nebenbedingungen eines Ablaufplanproblems. Zunächst liegt der Gedanke nahe, diese als harte Nebenbedingung in der Problemstellung zu berücksichtigen. Gerade bei Probleminstanzen, bei denen es allerdings nicht möglich ist, alle diese Nebenbedingungen einzuhalten, wird die Einhaltung der Fälligkeitstermine mit in die Zielfunktion aufgenommen, anstatt diese als Nebenbedingungen zu behandeln (vgl. Dubois u. Prade (1999), Sakawa u. Kubota (2000)).

Maximierung des Agreement Index Der Fall, dass eine Operation nicht rechtzeitig abgeschlossen werden konnte, ist bei einer Berücksichtigung von unscharfer Startzeit und unscharfer Operationsdauer detaillierter zu betrachten. So lässt sich der Grad der Einhaltung der spätesten Endzeit $\tilde{LE}$ für eine Operation bestimmen (siehe Abbildung 3.23). Der eingeplante unscharfe Endzeitpunkt $\tilde{E}$ einer Operation kann dabei die spätestmögliche Endzeit $\tilde{LE}$ teilweise oder ganz überschreiten. Die Modellierung des spätesten Endzeitpunktes als Fuzzy-Zahl drückt folglich eine Präferenz aus, zu welchem Zeitpunkt die Operation fertiggestellt sein sollte.

Dies wird durch den nach links offenen Verlauf deutlich, der angibt, dass jede Fertigstellung in diesem Bereich komplett den Präferenzen entspricht, dass die Operation fristgerecht beendet wurde. Der abnehmende Verlauf des spätesten Endes auf der rechten Seite bis zum Wert 0 beschreibt die abnehmende Zufriedenheit bei einer späteren Beendigung der Operation. Der rechte Verlauf lässt sich bei Bedarf als vertikale Flanke modellieren, um die einzuhaltende späteste Endzeit als scharfe Zahl zu interpretieren.

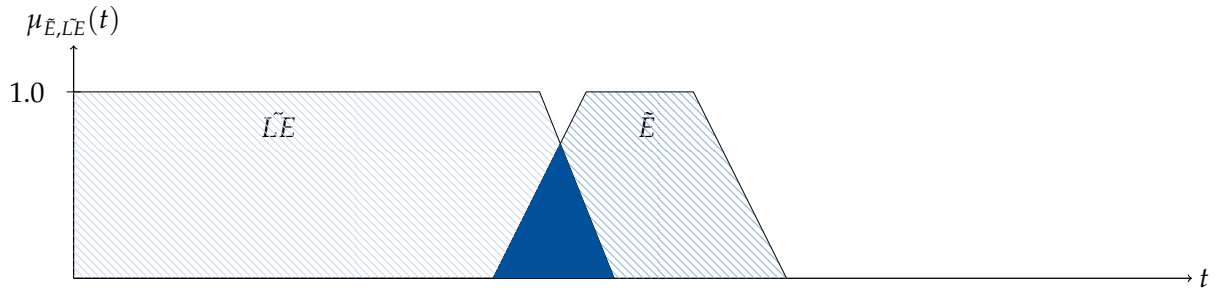


Abbildung 3.23.: Unscharfe Überschreitung des spätesten Endzeitpunktes.

Definition 38: Agreement Index

Sakawa u. Kubota (2000) führen einen *agreement index* (AI) ein, der beschreibt, wie gut die Einhaltung des spätesten Endes umgesetzt werden kann. Dieser ergibt sich aus dem Grad der Erfüllung der unscharf modellierten Präferenzen, der sich definieren lässt als:

$$AI := \frac{area(\tilde{E}) \cap area(\tilde{LE})}{area(\tilde{E})} \tag{3.31}$$

Eine mögliche Optimierung kann, neben anderen problemspezifischen Optimierungsgrößen wie beispielsweise der Minimierung der Gesamtdauer, die Maximierung des minimalen *agreement index* aller Operationen oder die Maximierung des mittleren *agreement index* umfassen. Dieses Vorgehen ist stark verwand mit der Maximierung der Pufferzeiten aus Abschnitt 3.3.8. Das Operationenende kann als Fuzzy-Zahl formuliert werden, deren Verlauf die Pufferzeiten berücksichtigt.

Unscharfe Kapazitätsgrenze Eine Erweiterung des vorgestellten Verfahrens umfasst neben der Berücksichtigung des *agreement index* eine unscharfe Modellierung der Kapazitätsgrenze der Ressourcen. Anstatt diese scharf zu setzen, wird eine ähnliche Modellierung wie beim Fälligkeitsdatum der einzelnen Operationen vorgenommen (siehe Abbildung 3.24). So lässt sich der Grad der Verletzung der Kapazitätsgrenze bestimmen und in Summe minimieren. Eine Verletzung über den Träger der unscharfen Kapazitätsgrenze ist dabei nicht erlaubt.

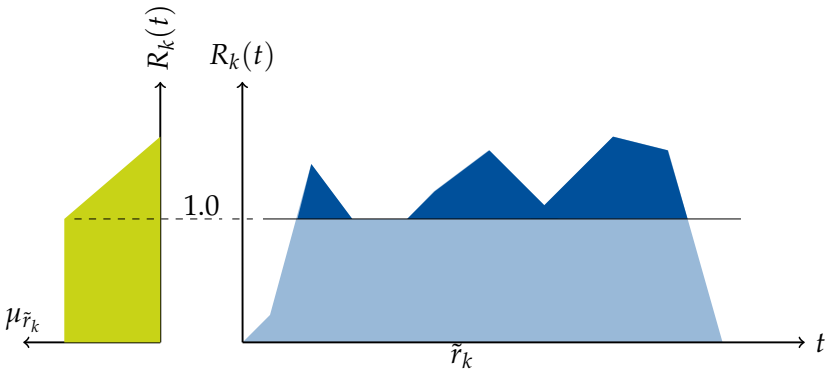


Abbildung 3.24.: Unscharfe Auslastungsgrenze, die eine Präferenz angibt.

Bestimmung eines Pessimismus-Levels für jede Operation Eine der entscheidenden Schwächen der vorgestellten Modellierung liegt darin, dass der Entscheidungsträger für jede Operation ein Pessimismus-Level wählen muss. Zwar erstreckt sich der Verlauf über den gesamten Bereich, in dem die Operation möglich ist, die Wahl des Levels beeinflusst dennoch stark die Ressourcenprofile und somit die Operationenreihenfolge. Ein anderer Ansatz wäre die Bestimmung des Pessimismus-Levels, das eintreten muss, damit die Operation an dem betrachteten Zeitpunkt einplanbar ist.

Diskussion der Einsatzmöglichkeiten

Masmoudi u. Haït (2013) verwenden das ursprüngliche Verfahren für ähnliche Problemstellungen wie die hier vorgestellten. Allerdings sind die Ressourcenbeschränkungen bei den vorgestellten Problemen zumeist kein stark limitierender Faktor, größtenteils werden die Kapazitäten nicht vollends ausgenutzt. Wichtig ist ihnen die Bestimmung, wie viele Ressourceneinheiten zu einem gewissen Zeitpunkt benötigt werden.

Betrachtet man eine Problemstellung, bei der die Ressourcenbeschränkung einen größeren Einfluss hat, treten andere Phänomene auf, die es zu berücksichtigen gilt. Hierzu sollen beispielhaft die Operationen aus Tabelle 3.3 der Reihe nach eingeplant werden.

Durch die sich vergrößernde Unschärfe, begründet durch das Aufaddieren der Unschärfe der Dauer und der Startzeit, überschneiden sich die Start- und Endzeiten der Operationen 3 und 4. Dies führt zu einer Abnahme der Notwendigkeit. Die Operationen 1 und 2 besitzen noch Plateaus, bei denen die Notwendigkeit bei 1.0 liegt. Dies ist implizit in Abbildung 3.25 durch den Verlauf der Ressourcenauslastung zu erkennen. Bei Operation 3 sinkt die Notwendigkeit, es gibt allerdings noch ein Peak, Operation 4 hingegen hat durch die große Überschneidung von Start- und Endzeit keinen Bereich mehr, bei dem die Notwendigkeit $\neq 0$ ist.

Operation	d	γ	Ressourcenbedarf
1	10	1	1
2	14	2	1
3	14	2	1
4	8	2	1

Tabelle 3.3.: Definitionen der Operationen des Beispiels zur ungleichmäßigen Auslastung

Die Tatsache, dass Operation 2, 3, und 4 sich überschneiden, führt dazu, dass der Verlauf der Auslastung deutlich unter 1.0 liegt (siehe Abbildung 3.26). Aufgrund der Form der einzuplanenden Operationen ist es im Folgenden schwer zu erreichen, eine weitere Operation, die ebenfalls einen Ressourcenbedarf von 1.0 besitzt, so einzuplanen, dass die Restkapazität effektiv genutzt werden kann. Bei der Fragestellung nach einer Abschätzung des Ressourcenbedarfs ist dieses Phänomen von untergeordnetem Interesse. Bei einem stark ressourcenbeschränkten Problem und der Suche nach einem robusten Ablaufplan führt dieses Verhalten allerdings dazu, dass die zur Verfügung stehenden Ressourcenkapazitäten nicht optimal genutzt werden, was gegen den Einsatz der vorgestellten Modellierung der Auslastungsprofile für solche Probleme spricht. Im Anhang A sind detaillierte Analyse weiterer unscharfen Auslastungsprofile zu finden.

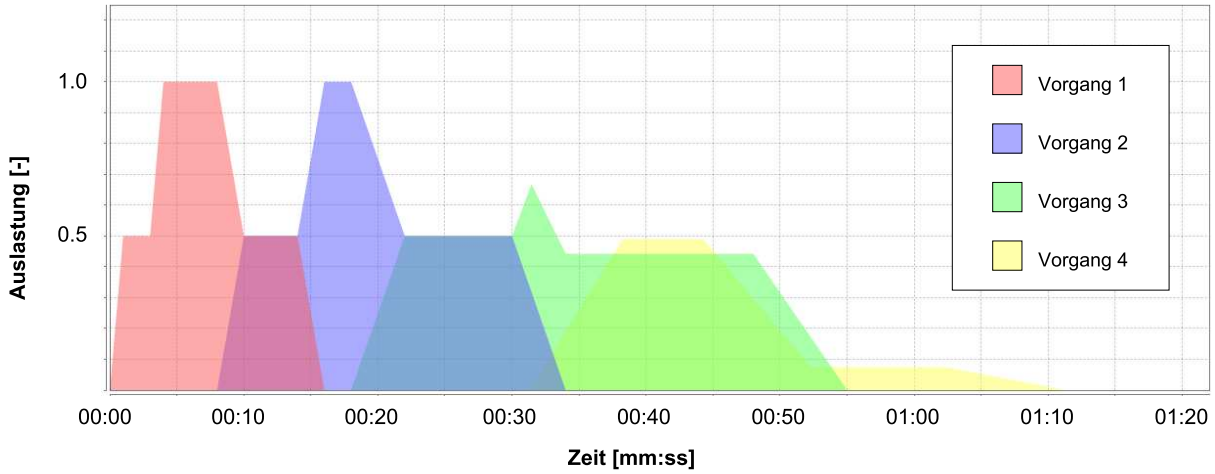


Abbildung 3.25.: Beispiel - Fuzzy-Auslastung der einzelnen Operationen

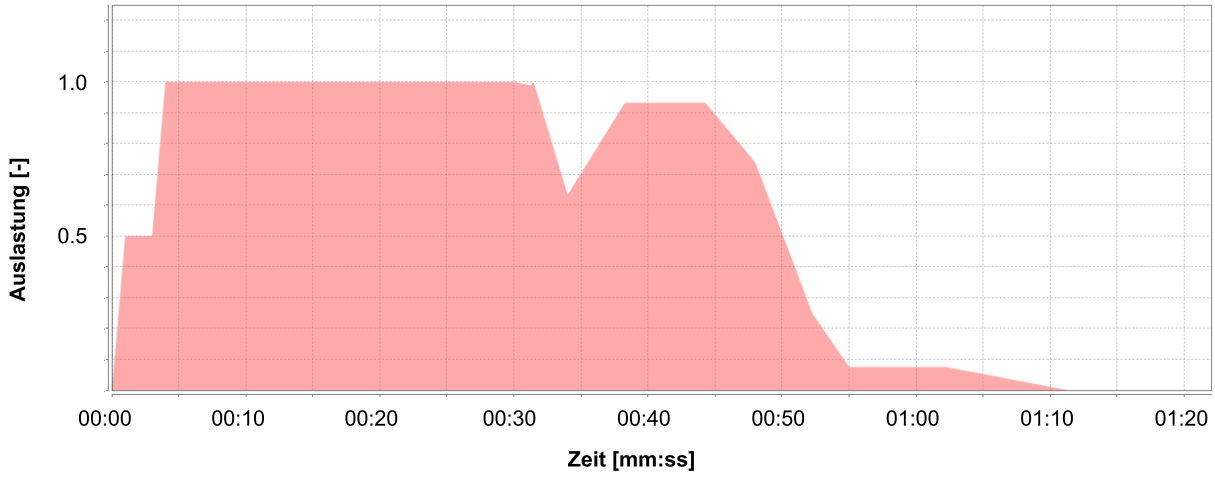


Abbildung 3.26.: Beispiel - Fuzzy-Gesamtauslastung

3.4. Optimierung auf mikroskopischer Ebene

Auf mikroskopischer Ebene werden keine Änderungen an der Routenwahl vorgenommen. Lediglich die Operationenreihenfolge wird für das betrachtete kleinere Zeitfenster optimiert. Daher wird ein Genetischer Algorithmus eingesetzt, die Interaktion im Rahmen eines Koevolutionären Algorithmus kann vernachlässigt werden. Der Vorteil der auf makroskopischer Ebene verwendeten Codierung mittels Prioritätsregeln liegt darin, dass der Ablaufplan eine gewisse Flexibilität bietet, da die Operationen nicht im Detail bekannt sein müssen und die Reihenfolge nicht direkt vorgegeben ist. Dadurch wird der Lösungsraum dahingehend eingeengt, dass nur Operationenreihenfolgen möglich sind, die aus der Auswertung einer der gewählten Prioritätsregeln hervorgehen.

3.4.1. Codierung

Auf mikroskopischer Ebene wird nur ein kleines Zeitfenster und eine geringe Anzahl an Operationen betrachtet. Weiterhin sind die zu erwartenden Unsicherheiten geringer. Daher wird eine Codierung gewählt, die auf Kosten der Flexibilität das Problem exakter erfasst und einen größeren Lösungsraum ermöglicht und somit eine größere Vielfalt an Lösungskandidaten. Diese Codierung orientiert sich an dem bei Rixen u. Kopfer (1994) vorgestellten Verfahren, das auf einer topologischen Sortierung des *Activity-On-Node*-Diagramms basiert. Es werden nur gültige Lösungen erzeugt, die die technologischen Vorschriften einhalten. Dadurch ist keine Reparatur oder Bestrafung ungültiger Lösungen nötig. Durch die Bestimmung der Routen in der Prozessplanung und der Ablaufplanung auf makroskopischer Ebene ergibt sich eine Menge von Operationen, die eingeplant werden müssen. Wiederum werden nur die Operationen betrachtet, die innerhalb des Zeitfensters eingeplant werden sollen. Auf Basis dieser Teilmenge von Operationen kann ein Untergraph des *Activity-On-Node*-Diagramms erzeugt werden. Das ursprüngliche *Activity-On-Node*-Diagramm wird zunächst topologisch sortiert. Dadurch ergibt sich eine Einteilung in Knotenklassen. Diese Knotenklassen dienen als Unterchromosomen. Pro Knotenklasse K ergibt sich eine Liste von Operationen, die die Sub-Prioritäten P_K der einzelnen Operationen beschreibt. Für die Decodierung des Chromosoms wird angenommen, dass jeder Operation eine Priorität P_{Ges} als Summe aus seiner Sub-Priorität P_K innerhalb seiner Knotenklasse sowie des Maximalwertes der Prioritäten seiner Vorgänger P_{pred}^{max} zugewiesen wird.

Besitzen zwei Operationen die gleiche Priorität P_{Ges}, so wird unterschieden, ob die Operationen aus derselben Knotenklasse stammen oder unterschiedlichen Knotenklassen angehören. Haben zwei Operationen, die aus der gleichen Knotenklasse stammen, die gleiche Priorität, so wird die Operation höher gewichtet, die innerhalb der Knotenklasse den höheren Wert hat. Stammen beide Operationen aus unterschiedlichen

Knotenklassen, so wird die Operation bevorzugt, die aus der niedrigeren Knotenklasse stammt.

Ein Chromosom enthält folglich für jeden Job J_α die Operationen aus der ausgewählten Route $O_{\alpha\beta}$. Diese Operationenmenge wird in knotenklassenabhängige Unterlisten aufgespaltet. Betrachtet man folgendes Beispiel mit drei Jobs J_1, J_2, J_3, mit den auf makroskopischer Ebene ausgewählten Routen

- $P_{12} := \{O_2, O_3, O_{10}\}$
- $P_{21} := \{O_8, O_9\}$
- $P_{33} := \{O_{11}, O_{12}\}$.

und nimmt an, dass keinerlei Beziehungen zwischen den Jobs vorgegeben sind und sich die Operationen nach einem topologischen Sortieren in den Knotenklassen K_1, K_2 und K_3 unterteilen lassen, so ergibt sich eine beispielhafte Codierung zu:

O_{11}	O_2	O_8	O_3	O_{12}	O_9	O_{10}

Die farbliche Unterscheidung dient der Unterteilung in den Knotenklassen. Die berechneten Prioritäten ergeben sich zu:

Operation	O_{11}	O_2	O_8	O_3	O_{12}	O_9	O_{10}
P_K	1	2	3	1	2	3	1
P_{pred}^{max}	-	-	-	2	1	3	3
P_{Ges}	1	2	3	3	3	6	4

Eine Aktivitätsliste als Input für das Schedule Generation Scheme ergibt sich nach oben beschriebenen Schema zu:

O_{11}	O_2	O_8	O_3	O_{12}	O_{10}	O_9

Als Operatoren wird für die Rekombination ein *Two-Point-Crossover* für jede einzelne Knotenklasse verwendet. Als Mutation dient ein *Swap*-Operator, der zufällig in einer der Knotenklassen zwei Operationen tauscht.

3.4.2. Ablaufplanerzeugung

Zur Erzeugung eines gültigen Ablaufplans wird ein paralleler SGS (siehe Algorithmus 3.5) verwendet, wie er häufig in der Literatur eingesetzt wird (beispielsweise Brucker u. Neyer (1998), Ayache (2005))

Algorithmus 3.5 Paralleler SGS auf Basis einer Aktivitätsliste

Eingabe: $O := (O_{23}, O_5, ..., O_{12}, ..., O_4)$ ▷ Aktivitätsliste
Eingabe: $O_i \mapsto \{(ES_i, LS_i, EE_i, LE_i)\}$ ▷ Ausführungsfenster der Operationen
$t = \min_{i \in V}(ES_i)$
Bestimme die Menge E_t von Operationen O_i, die keine Vorgängeroperationen haben und zum aktuellen Zeitpunkt t eingeplant werden können ($ES_i \leq t$).
Bestimme die Menge A_t von Operationen O_i, die keine Vorgängeroperationen haben, aber noch nicht eingeplant werden können ($ES_i > t$).
while nicht alle Operationen eingeplant sind **do**
  while $E_t \neq \emptyset$ **do**
    Wähle Operation $O_j \in E_t$, die an minimaler Position in O steht.
    if eine Ressource k mit $rjk > R_k(\tau)$ für ein $\tau \in [t, t + p_j[$ existiert **then**
      CONTINUE
    else
      Plane die Operation j ein.
    end if
  end while
  Bestimme neues t als Minimum der Endzeiten der bereits eingeplanten Operationen und dem frühsten ES_i der Operationen aus A_t.
  Aktualisiere E_t und A_t
end while
return Schedule S

3.4.3. Zielfunktion

Wie auf makroskopischer Ebene (siehe Abschnitt 3.3.8), so liegt auch auf mikroskopischer Ebene ein mehrkriterielles Optimierungsproblem vor. Neben der Zielgrößen, die vom grundsätzlichen Problem vorgeschrieben sind, muss berücksichtigt werden, dass auf mikroskopischer Ebene möglichst viele Operationen, die auf makroskopischer Ebene innerhalb des betrachteten Zeitraums eingeplant sind, ebenfalls eingeplant werden.

4. Evaluierung des Algorithmus anhand einer komplexen Problemstellung

Der vorgestellte Multiskalen-Algorithmus wird anhand eines Anwendungsbeispiels getestet. Es werden die Möglichkeiten und Grenzen des vorgestellten Algorithmus aufgezeigt. Dabei wird zunächst auf eine Implementierung ohne Berücksichtigung der Unschärfe mittels der *Fuzzy-Set*-Theorie eingegangen. Nachfolgend wird deren Eignung für die Problemstellung diskutiert.

4.1. Betrieb auf einem Containerterminal

In Deutschland gibt es viele Wirtschaftszentren mit einem hohen Potential an Transportaufkommen. Der entstehende Verkehr ist verteilt zwischen Städten und Ballungszentren. Um mit diesem immer größer werdenden Verkehrsvolumen umgehen zu können, werden unter anderem für den Güterverkehr neue Konzepte entwickelt. Eines dieser Konzepte ist das *Hub & Spoke*-System in Verbindung mit *Gateway*-Verkehr Jeong u. a. (2007).

Aus der Sicht der Eisenbahnverkehrsunternehmen ist es unter Berücksichtigung der Kosten- und Zeiteffizienz wünschenswert, dass ein Güterzug komplett von einem einzelnen Kunden in Anspruch genommen wird und alle Ladeeinheiten auf diesem Zug dasselbe Ziel haben. Für den einzelnen Kunden ist es allerdings schwierig, das Transportaufkommen so zu bündeln, dass ein regelmäßig verkehrender Zug mit einem konstant hohen Volumen zwischen zwei Städten exklusiv für ihn eingesetzt werden kann.

Daher bieten Eisenbahnverkehrsunternehmen einzelne Waggonplätze anstatt eines gesamten Zuges an, um den Kunden eine größere Flexibilität zu ermöglichen. Dies führt allerdings zu einer großen Menge von verschiedenen Destinationen der Ladeeinheiten auf einem Zug. Weiterhin führt der steigende Anteil an Interaktion zwischen verschiedenen Transportsystemen, wie beispielsweise der Transport auf der Straße, auf der Schiene oder zu See, dazu, dass die Heterogenität der Destinationen der einzelnen Ladeeinheiten eines Zuges ebenfalls steigt. Einer der wichtigsten Aspekte des intermodalen Verkehrs von Gütern besteht darin, dass ein Containerschiff, das an einem Hafen anlegt, schnell entladen werden muss. Das stetig steigende Güteraufkommen führt allerdings

dazu, dass die maximalen Lagerkapazitäten an den Häfen zumeist schnell erreicht sind. Um Zeit und Lagerplatz am Seehafen zu sparen, werden Güterzüge mit Ladeeinheiten beladen, die unterschiedliche Destinationen haben. Dadurch entsteht der Bedarf, dass diese Ladeeinheiten später im Binnenland sortiert werden müssen. Der Lagerbedarf sowie der Sortierungsaufwand wird dadurch vom Seehafen ins Binnenland verlagert.

Das *Hub & Spoke*-System setzt an diesen Punkten an und verbindet Flexibilität und Kosteneffizienz. In intermodalen Containerterminals kommen Züge aus unterschiedlichen Wirtschaftszentren oder Seehäfen an, die Ladeeinheiten mit sich führen, die für unterschiedliche Destinationen bestimmt sind. Während ihres Aufenthalts in einem Containerterminal findet der Umschlag der Güter zwischen den Zügen statt. Die Kernidee besteht darin, in den großen Containerterminals, den *Hubs* des Systems, die Ladeeinheiten ihrer Destination entsprechend zu bündeln und Ausgangszüge zu generieren, auf denen alle Ladeeinheiten das gleiche Ziel haben. Die *Hubs* agieren als *Gatewayterminals* und schlagen die Ladeeinheiten zwischen zwei verschiedenen Zügen um. Die *Spokes* des Systems bilden die Verkehrsverbindungen zwischen den Containerterminals. An den Wirtschaftszentren existieren zumeist klassische Containerterminals, auf denen die Ladeeinheiten auf die Straße umgeschlagen werden.

Im Gegensatz zu klassischen Containerterminals werden bei einem *Hub*, wie dem *MegaHub Lehrte* (DB Netz AG (2014)), spezielle Anforderungen an den Betrieb vor Ort gestellt. Der Umschlag findet zwar größtenteils ebenfalls durch Portalkrane statt, die mehrere Gleise überspannen, allerdings erfordert der gestiegene Anteil an Umschlägen von Zug zu Zug, dass weitere Gesichtspunkte bei der Optimierung berücksichtigt werden müssen. Rotter (2004) entwickelte ein neues Betriebskonzept für intermodale Umschlagterminals im *Hub & Spoke*-System. Konventionelle Containerterminals sind dafür ausgelegt, Ladeeinheiten vom Zug aufzunehmen und auf Lastkraftwagen umzuschlagen sowie umgekehrt Ladeeinheiten von Lastkraftwagen auf Züge umzuschlagen. Dabei werden die Fahrer der Lastkraftwagen an einen Ort delegiert, der möglichst nah am Standort der Ladeeinheit auf dem Zug ist. So ist der Transportweg zwischen Start und Ziel der Ladeeinheit zumeist gering. Im *Gateway*-Betrieb kann die Entfernung vom Startort der Ladeeinheit zu ihrem Bestimmungsort auf einem anderen Zug unter Umständen sehr groß sein. Zwar wird durch eine gute Verladeplanung versucht, diese Entfernung zu minimieren, allerdings ist dies nicht immer möglich und kann auch hinfällig sein, wenn ein Zug in umgekehrter Wagenfolge in den Terminal einfährt als zunächst angenommen, da er zuvor umgeleitet wurde. Weiterhin sind *Hubs* zumeist mit einer größeren Anzahl an Kranen ausgestattet als konventionelle Containerterminals. Dies führt dazu, dass bei einem Transport einer Ladeeinheit quer über den gesamten Terminal entweder die Portalkrane, die dem transportierenden Kran den Weg versperren, wegfahren müssen, oder die Ladeeinheit über eine Stafette von Kran zu Kran durchgereicht werden muss. Daher werden auf größeren Containerterminals und vor allem auf Hubs im *Hub & Spoke* System die Portalkrane durch Längsförderanlagen unterstützt (siehe Abbildung 4.1), um die kostenintensiven Längsfahrten der Portalkrane zu minimieren. Da die Ladeeinheiten in der Regel nur eine geringe Zeit im Terminal

verbringen und somit temporär abgestellt werden, werden die Stellplätze der Ladeeinheiten auf dem Terminal nicht als Lager bezeichnet, sondern als Zwischenabstellung. Auf einigen Umschlagsterminals sind außerdem mobile Umschlaggeräte im Betrieb, die Ladeeinheiten in entferntere Zwischenabstellungen transportieren, die nicht direkt an einem der Umschlagmodule positioniert sind.

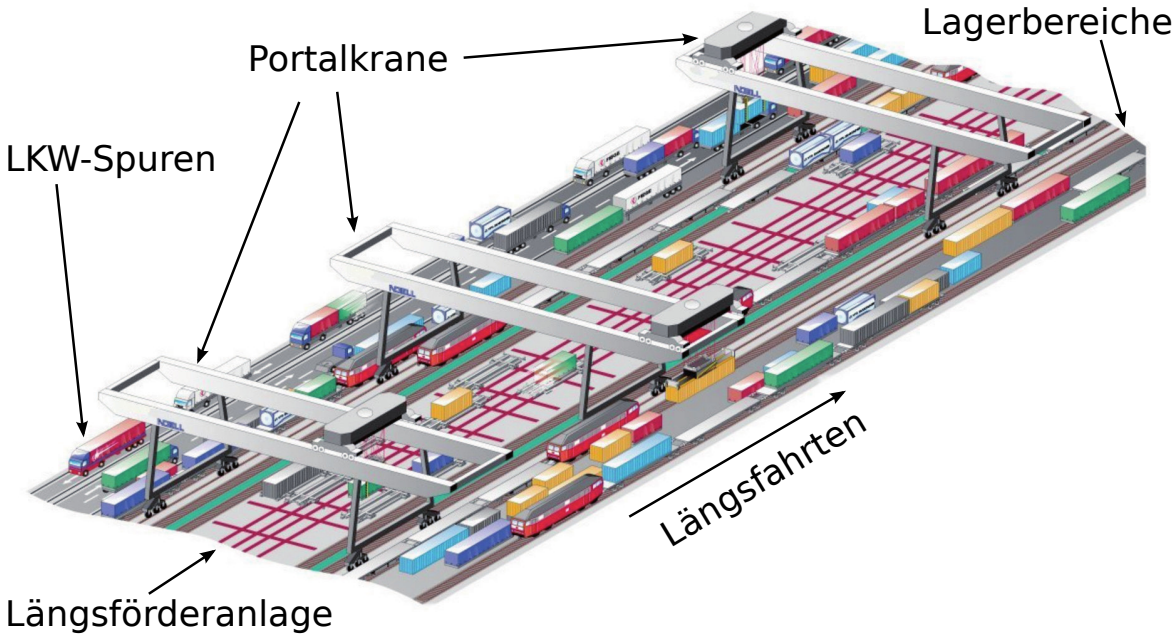


Abbildung 4.1.: Schematischer Aufbau des MegaHubs in Lehrte (angelehnt an Gg.-Noell-GmbH (1998))

Die Optimierung des Betriebsablaufes auf einem solchen Containerterminal lässt sich mathematisch als ein zeit- und ressourcenbeschränktes *Scheduling*-Problem (vgl. Abschnitt 2.1.2) beschreiben. Bei der Modellierung des *Scheduling*-Problems wird ein objektorientierter Ansatz gewählt. Das heißt, dass für die Bestandteile des Problems, die für die Optimierung wichtig sind, ein Objekt mit der entsprechenden Funktionalität modelliert wird. Grundsätzlich geht es bei dem betrachteten *Scheduling*-Problem um das Verladen von Ladeeinheiten. Man betrachtet eine *Ladeeinheit*, die einen Startort, beispielsweise einen Stellplatz eines Zuges oder eines Lagers hat, und analog einen Zielort besitzt. Weiterhin können im Rahmen der Optimierung eines reellen Problems weitere Eigenschaften wichtig sein, wie beispielsweise ein Attribut, welches angibt, ob es sich bei um einer Ladeeinheit um Gefahrgut handelt. Weiterhin kann das Greifwerkzeug, mittels dessen der Kran die Ladeeinheit heben kann, vermerkt werden. Zu den wichtigen Attributen gehören weiterhin die Ausdehnungen der Ladeeinheit in Länge, Breite und Höhe.

Eine Ladeeinheit induziert dadurch, dass sie von einem zum anderen Ort transportiert werden soll, einen Job. Dieser *Job* besteht aus einer Kette von einer oder mehreren Teiloperationen, die *Route* genannt werden. Die *Operationen* einer Route lassen sich

in Transportoperationen und Lageroperationen unterscheiden. *Transportoperationen* beschreiben den Transport einer Ladeeinheit von einem Zwischenabstellungsbereich des Terminals zu einem anderen Zwischenabstellungsbereich. Ausgeführt werden diese Transportoperationen von Umschlagsystemen, wie beispielsweise Kranen, oder von *automatic guided vehicles* (AGV). Bei den AGVs handelt es sich um gummibereifte oder schienengebundene Fahrzeuge, die eine oder mehrere Ladeeinheiten aufnehmen können und selbstständig über einen bestimmten Bereich des Terminals navigieren können, um die Ladeeinheit zu transportieren. Eine Transportoperation ist immer von zwei *Lageroperationen* umspannt. Vor Beginn des Transportvorganges ist eine Lageroperation für die Lagerung der Ladeeinheit an ihrem Startort eingeplant, nach Beendigung der Transportoperation folgt eine Lageroperation am Zielort der Transportoperation.

Der eigentliche Transport vom Startort der Ladeeinheit zu ihrem endgültigen Zielort besteht folglich aus einer Kette von Transportoperationen über mehrere Zwischenabstellungen und den damit verbundenen Lageroperationen. Die Operationen haben eine Dauer, auf die sich verschiedene Einflüsse auswirken. So ist die Dauer einer Transportoperation abhängig vom zurückgelegten Weg, der Geschwindigkeit der Transportsysteme und etwaige Verzögerungen durch das Vermeiden von Kollisionen. Eine Lageroperation ist abhängig von der vorangegangen sowie der nachfolgenden Transportoperation und etwaigen Reservierungen im Lager.

Die Transportsysteme und Zwischenabstellungen werden als *Ressourcen* aufgefasst, die die entsprechenden Operationen ausführen. Dabei haben die Ressourcen eine Kapazität, die beispielsweise die Anzahl an gleichzeitig ausführbaren Operationen beschränkt.

Das zu entwickelnde Modell des Terminals soll gleichermaßen für alle Optimierungs- und Simulationszwecke anwendbar sein. Modelle, die zur Simulation eingesetzt werden, lassen sich verschieden klassifizieren (vgl. Ludewig (2002)). Ein *deskriptives* (beschreibendes) Modell dient dem Studium von bestehenden Systemen. Ein *präskriptives* (vorschreibendes) Modell dient zu Planung von noch nicht vorhandenen realen Systemen. Ebenso lassen sich *transiente* Modelle bilden, die den Übergang auf Basis eines bestehenden Systems in ein angepasstes neues System beschreiben. Das im Folgenden entwickelte Modell kann sowohl während der Planungsphase eines Terminals eingesetzt werden als auch während dessen Betriebs. In der Planungsphase wird es präskriptiv eingesetzt, um ein noch nicht vorhandenen Containerterminal zu modellieren. Ebenso soll es bei Erweiterungen eines bestehenden Terminals als transientes Simulationsmodell eingesetzt werden, um die Auswirkungen der Änderungen an der Infrastruktur zu bestimmen. Weiterhin sollen bestehende Terminals deskriptiv abbildbar sein. Abbildung 4.2 zeigt eine Aufnahme des Betriebs auf einem Container-Umschlagterminal, dessen Komponenten im Folgenden kurz vorgestellt werden.

Abbildung 4.2.: Aufnahme des Betriebs auf dem Container-Umschlagterminal Hamburg-Billwerder (Eigene Aufnahme)

4.1.1. Ladeeinheiten

Im kombinierten Verkehr bezeichnet der Oberbegriff Ladeeinheit alle Arten von Behälter zur Aufnahme von Transportgut, der als Stückgut zwischen verschiedenen Orten umgeschlagen wird. Der Inhalt einer Ladeeinheit wird während des gesamten Transportvorgangs nicht verändert. Im Allgemeinen gibt es drei verschiedene Typen von Ladeeinheiten auf einem intermodalen Containerterminal: ISO-Container, Wechselbrücken und Sattelauflieger. Diese Ladeeinheiten werden mit unterschiedlichen Greifwerkzeugen vom Kran aufgenommen und transportiert. Sattelauflieger sind grundsätzlich nicht stapelbar, wohingegen ISO-Container, die kein Gefahrgut beinhalten, grundsätzlich stapelbar sind.

4.1.2. Transportprogramm

Der Betreiber eines Containerterminals bietet sogenannte Slots an, die den Umschlag einer Ladeeinheit in einem bestimmten Zeitfenster umfassen. Nach Anmeldung aller Ladeeinheiten und der entsprechenden Slots wird ein Transportprogramm erstellt, welches die Ankunfts- und Abfahrtszeiten der einzelnen Züge und die georderten Slots für die

Ladeeinheiten enthält. Dieses Transportprogramm muss im Folgenden im realen Betrieb bestmöglich abgearbeitet werden. Bei der Erstellung eines Transportprogramms ist die Verwendung des vorgestellten Multiskalen-Algorithmus ebenfalls von Nutzen, da durch die Planung auf makroskopischer Ebene eine Abschätzung der Kapazitätsauslastung und somit eine Optimierung des Transportprogramms und ggf. die Erweiterung des Angebots um zusätzliche Slots möglich ist.

4.1.3. Jobs und Operationen

Die Prozessplanung auf einem Containerterminal beschäftigt sich mit der Fragestellung, auf welche Art und Weise unter Verwendung welcher Transportsysteme eine Ladeeinheit von ihrem Start- zu ihrem Zielort befördert werden kann. Hierbei gibt es die Möglichkeit, die Ladeeinheiten mittels Direkttransport oder mittels mehrerer Teiltransporte mit Zwischenabstellung vom Start- zum Zielort zu befördern. Die Auswahl der Teiltransporte erfolgt unter Berücksichtigung einer zuvor definierten vektoriellen Zielfunktion, um das Gesamtsystem bezüglich bestimmter Zielgrößen zu optimieren. Die Bestimmung der einzelnen Folgen von möglichen Teiltransporten folgt in Abschnitt 4.1.5.

Job-On-Node-Diagramm

Zwischen den Transporten der Ladeeinheiten bestehen technologische Relationen. Zunächst betrachtet man das Problem, dass eine Ladeeinheit L_1, die vom Waggonplatz W_A eines Zuges auf den Waggonplatz W_B eines anderen Zuges umgeschlagen werden soll. Dieser Umschlag wird durch den Job J_1 beschrieben. Weiterhin existiert eine weitere Ladeeinheit L_2, die sich bei Einfahrt des Zuges bereits auf dem Waggonplatz W_B, dem Zielplatz von L_1, befindet. Im Rahmen des Jobs J_2 soll diese Ladeeinheit auf den Lagerplatz S_C abgestellt werden.

Da der Waggonplatz W_B nur eine Ladeeinheit aufnehmen kann, muss L_2 zunächst vom Platz aufgenommen werden, um den Platz für L_1 verfügbar zu machen. Die Ladeeinheit L_2 kann entweder direkt oder über eine oder mehrere Zwischenstationen zu ihrem Zielort S_C transportiert werden. Ist der Platz freigeräumt worden, kann die Ladeeinheit L_1 auf den Waggonplatz W_B abgestellt werden.

Diese Tatsache bedeutet, dass die erste Operation des Jobs J_2 vor der letzten Operation des Jobs L_1 stattfinden muss. Diese Relation lässt sich im Rahmen eines *Job-On-Node*-Diagramms (siehe Abschnitt 3.3.2) darstellen, bei dem die Jobs die Knotenmenge bilden und die Kanten die beschriebenen Relationen darstellen. Obiges Beispiel lässt sich in Abbildung 4.3 nachvollziehen.

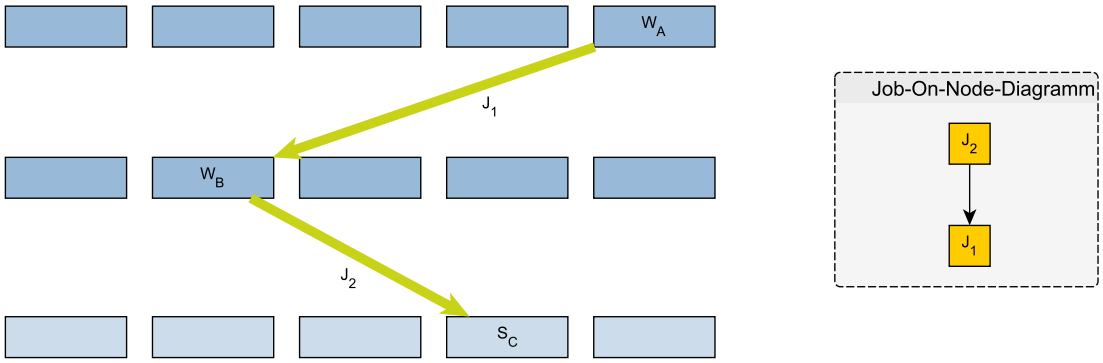


Abbildung 4.3.: Basis des *Job-On-Node*-Diagramms. Links: Transporte der Ladeeinheiten L_1 und L_2 im Grundriss eines Terminals dargestellt. Rechts: entstehendes *Job-On-Node*-Diagramm.

Dieses *Job-On-Node*-Diagramm lässt sich nun in starke Zusammenhangskomponenten unterteilen (siehe Abbildung 4.4).

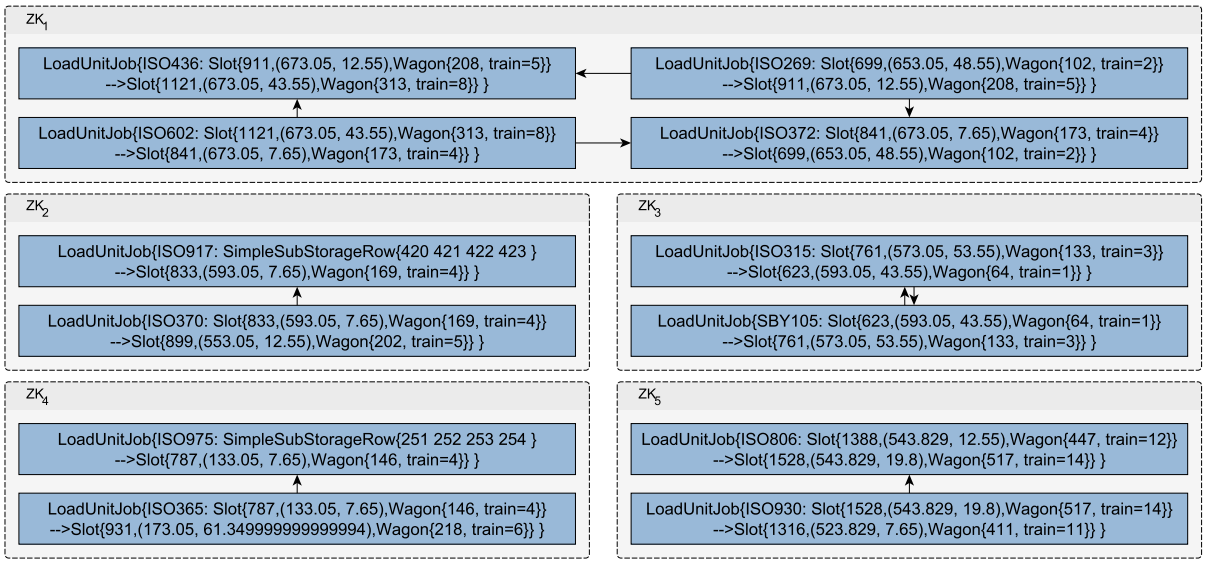


Abbildung 4.4.: Zerlegung eines *Job-On-Node*-Diagramms in Zusammenhangskomponenten.

Ausführungszeitfenster von Operationen

Die einzelnen Operationen des Containerterminal-Problems unterliegen zeitlichen Beschränkungen, die durch die Ankunfts- und Abfahrtszeiten der Züge vorgegeben werden. Die einzelnen Ladeeinheiten haben Ausführungszeitfenster (siehe 2.1.4), innerhalb derer sie umgeschlagen werden müssen. Die Ausführungszeitfenster für die

einzelnen Operationen einer Route werden mittels einer Vorwärts- und Rückwärtsrechnung bestimmt. Zur Bestimmung der Dauer der jeweiligen Operation werden die Einplanregeln der makroskopischen Ebene verwendet (siehe 4.1.6). So ergibt sich der frühste Start ES der ersten Operation einer Route aus der Zugankunft des Startzuges, das späteste Ende LE der letzten Operation einer Route aus der Abfahrt des Zielzuges. Für die einzelnen Operationen der Route ergeben sich die ES aus den EE der vorangegangen Operationen. Diese Größen müssen nachfolgend unter Berücksichtigung der technologischen Vorschriften zwischen den einzelnen Jobs auf Basis des *Activity-On-Node*-Diagramms angepasst werden. Innerhalb der mathematischen Formulierung des Optimierungsproblems lässt sich der früheste Start als Randbedingung aufnehmen:

$$S_i \geq ES_i. \tag{4.1}$$

Rüstfahrten

Da bei Transportoperationen von Umschlaggeräten wie Kranen oder AGVs die benötigten Ressourcen erst zum Startort der Transportoperation bewegt werden müssen, treten sequenzabhängige Rüstzeiten auf (siehe 2.1.4).

Der Kran oder das AGV befinden sich in der Regel zum Zeitpunkt des Transportbeginns nicht am Startort des Tranports. Die Rüstzeit ergibt sich dann durch die Fahrt zum Startort und ggf. zusätzliche Zeiten zum Umrüsten der Hebewerkzeuge, beispielsweise wenn nach einem Container eine Wechselbrücke von einem Kran transportiert werden soll.

Wird auf einer Ressource nach einer Operation O_i eine Operation O_j eingeplant, so kann dies eine Rüstzeit s_{ij} erfordern. Diese Rüstzeit wird beim Kantengewicht $\delta_{ij}(x)$ berücksichtigt. So ergibt sich:

$$\delta_{ij}(x) = S_i + p_i + s_{ij}. \tag{4.2}$$

4.1.4. Ressourcen

Eine Ressource bezeichnet ein Mittel, um eine Operation ausführen zu können. Dazu zählen sowohl die Lager-Ressourcen wie ein Waggonstellplatz oder ein Platz in der Zwischenabstellung als auch transportierende Ressourcen wie Portalkrane und die Längsförderanlage. Die Unterscheidung zwischen lagernden und transportierenden Ressourcen ist dabei nicht immer scharf zu gewährleisten, da beispielsweise auch die AGV der Längsförderanlage eine Ladeeinheit aufnehmen und diese unter Umständen eine gewisse Zeit lagern können, beispielsweise wenn keine Abstellmöglichkeit auf den Austauschbereichen gegeben ist. Abbildung 4.5 zeigt den Querschnitt durch einen typischen Containerterminal mit einer Längsförderanlage.

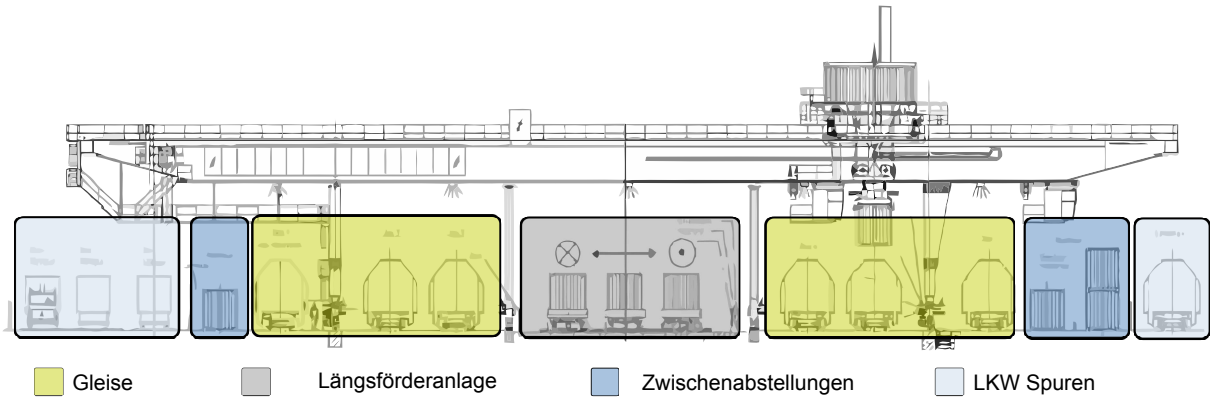


Abbildung 4.5.: Querschnitt durch einen Containerterminal mit Längsförderanlage. Nachbearbeitete Darstellung aus Projektunterlagen zur Verfügung gestellt durch DB Netz AG / Deutsche Umschlaggesellschaft Schiene-Straße mbH

Transportsysteme

Transportsysteme bewegen eine Ladeeinheit von einer Lager-Ressource zu einer anderen Lager-Ressource. Es gibt eine Vielzahl von Varianten von Transportsystemen, die auf Containerterminals eingesetzt werden. Ein Transportsystem besitzt eine Kapazität, die angibt, wie viele Operationen ein Transportsystem parallel ausführen kann. Dies entspricht folglich der Anzahl an Ladeeinheiten, die es zeitgleich aufnehmen kann.

Transportsysteme lassen sich unterschiedlich hierarchisch definieren. So kann entweder eine ganze Kranbahn, die aus mehreren Kranen besteht, als Transportsystem aufgefasst werden oder jeder Kran für sich als eigenes System, das in starker Wechselwirkung mit anderen Transportsystemen steht. Wichtige Kenngrößen, wie die Geschwindigkeit, in der ein Transport stattfinden kann, sowie der Bewegungsbereich der einzelnen Elemente und die auftretenden Kosten, die ein Transport verursacht, müssen festgelegt werden. Die Kosten setzen sich vor allem aus dem Energieverbrauch, Verschleisserscheinungen und Lohnkosten für die Bedienung der Transportsysteme zusammen. Nachfolgend werden die Systeme, die am häufigsten zum Umschlag eingesetzt werden, kurz erläutert.

Portalkrane Ein Portalkran ist ein ortsgebundener, jedoch beweglicher Kran. In Containerterminals werden Portalkrane zumeist im Verbund auf einer Kranbahn eingesetzt. Diese Kranbahn überspannt die Gleise und führt die Krane. Die einzelnen Portalkrane laufen dabei längs zu den Gleisen auf den zwei parallelen Schienen der Kranbahn. Die Bewegung in Querrichtung wird durch eine Laufkatze ermöglicht. An der Katze ist das Hubwerk befestigt, welches die Ladeeinheiten anhebt.

Die einzelnen Krane auf der Kranbahn können sich im Allgemeinen nicht überholen. Es gibt Konstruktionen, die durch eine unterschiedliche Höhe der einzelnen Krane und mehrere parallel verlaufende Schienen ein Überholen ermöglichen, dies ist allerdings nicht die Regel und wird im weiteren Verlauf nicht weiter verfolgt. Die Krane besitzen generelle Arbeitsbereiche, in denen sie sich bewegen können. Weiterhin ist zu einem konkreten Zeitpunkt zu beachten, dass benachbarte Krane diesen generellen Arbeitsbereich eingrenzen.

Längsförderanlage Eine Längsförderanlage dient zur Unterstützung der Portalkrane beim Umschlag von Gütern auf einem Containerterminal. Mehrere führerlos betriebene, automatisierte Containertransportfahrzeuge, sogenannte *AGV (automated guided vehicle)*, bilden im Zusammenspiel ein System, welches in Längsrichtung des Terminals einen Transport der Ladeeinheiten ermöglicht und somit Kranstaffetten sowie lange Kranfahrten vermeidet.

Bei dem Transport einer Ladeeinheit von einem Ende des Terminals zum anderen Ende gilt es zu berücksichtigen, dass sich die Krane im Allgemeinen nicht überholen können. Die Ladeeinheiten müssen folglich vom ersten Kran, in dessen Arbeitsbereich die Ladeeinheit steht, aufgenommen werden und in den Übergabebereich zum nachfolgenden Kran transportiert und dort abgestellt werden. Der nachfolgende Kran muss die Ladeeinheit aufnehmen und sie wiederum zum nächsten Übergabebereich transportieren. Diese Prozedur wiederholt sich so oft, bis die Ladeeinheit vom Kran im Zielgebiet aufgenommen werden kann. Da die Aufnahme und das Abstellen einer Ladeeinheit sehr zeitintensiv ist, hat ein solcher Transport einen hohen Ressourcenbedarf. Weiterhin ist die Längsbewegung eines Krans energieintensiv, da die gesamte Stahlkonstruktion bewegt werden muss. Ein Transport in Querrichtung bedarf nur einer Bewegung der Katze.

Eine Längsförderanlage unterstützt den Transport der Krane und führt zu einer Verminderung der Kranspiele pro Ladeeinheit. Der Kran, in dessen Arbeitsbereich die Ladeeinheit steht, kann diese aufnehmen und entweder direkt auf einen AGV oder auf extra vorgesehene Austauschbereiche abstellen. Das AGV transportiert die Ladeeinheit nun in den Arbeitsbereich des Krans, der die Ladeeinheit auf den Bestimmungsort umschlagen kann. Dabei kann ein AGV durch den Arbeitsbereich anderer Krane fahren, ohne diese zu beeinflussen. Ohne eine Längsförderanlage muss die Ladeeinheit durch einen Kran in die Übergabebereiche zum angrenzenden Kranbereichen abgestellt und vom nächsten Kran wieder aufgenommen werden.

Grundsätzlich ist eine Längsförderanlage flurgebunden und lässt sich in zwei Varianten unterscheiden. Schienengebundene Längsförderanlagen (siehe Abbildung 4.6) und schienenungebundene Längsförderanlage. Während schienengebundene Längsförderanlagen von Natur aus die Fahrwege der einzelnen AGVs vorgeben, wird die Navigation in einem schienenungebundenen System mittels Sensoren im Boden vorgenommen.

Abbildung 4.6.: Die im MegaHub-Lehrte angedachte Längsförderanlage besteht aus schienengebundenen AGVs, die mittels eines Linearmotors angetrieben werden. (Quelle: Rotter (2004))

In Seehäfen werden nicht schienengebundene AGVs seit längerer Zeit eingesetzt. Der erste Einsatz einer Längsförderanlage in einem Binnenterminal ist von der DB Netze für den MegaHub Lehrte geplant.

Lastkraftwagen Im *Hub & Spoke*-System ist ein Terminal dahingehend ausgelegt, dass größtenteils Zug-zu-Zug-Umschläge stattfinden. Trotzdem gibt ist weiterhin einen Anteil von Ladeeinheiten, die auf Lastkraftwagen umgeschlagen werden. Da dieser Anteil unter 5 % liegt, wird in der weiteren Modellierung vereinfacht angenommen, dass diese Ladeeinheiten einen Platz in der Zwischenabstellung zugewiesen bekommen, der nach einer mittleren Verweildauer von 20 Minuten wieder freigegeben wird.

Zwischenabstellungen

Bei der Abstellung der Ladeeinheiten auf einem Containerterminal muss zwischen zwei grundlegenden Arten unterschieden werden, der kurz- und der mittelfristigen Abstellung. Bleibt eine Ladeeinheit für eine längere Dauer im Terminal, so darf diese Ladeeinheit keinen zentralen Platz verbrauchen. Zur Einlagerung solcher Ladeeinheiten ist es unter Umständen sinnvoll, längere Transportwege zurückzulegen.

Das Hauptaugenmerk bei einer Optimierung, wie sie im Rahmen dieser Arbeit vorgenommen wird, liegt allerdings auf der temporären Lagerung, der Zwischenabstellung. Im operativen Betrieb ist es häufig nötig, eine Ladeeinheit zwischenzeitlich auf einer Lagerspur abzustellen. Lediglich Direktumschläge zwischen zwei Zügen können ohne Zwischenabstellung ausgeführt werden. Zu berücksichtigen sind hierbei die geänderten Prioritäten im Vergleich zu einer längeren Lagerung. Im Extremfall kann der verbrauchte Platz durch das Abstellen der Ladeeinheit fast komplett vernachlässigt werden, wenn dadurch ein schneller Umschlag der Ladeeinheit möglich ist. Im realen Betrieb wird versucht, ein Systemoptimum zu finden, bei dem die Präferenzen zwischen Lagerplatzausnutzung und Transportwegen allerdings zeitlich veränderlich sind.

Abstellungsflächen werden hierarchisch modelliert. Ein Lagerbereich besteht aus Abstellspuren, die aus einer Anzahl an aneinanderliegenden Stellplätzen bestehen. Es besteht die Möglichkeit, verschiedene Teilbereiche einer oder mehrerer Spuren zusammenzufassen. Dies ist vor allem bei der Bestimmung der zugehörigen Abstellflächen der *transfer area*s bei der Bestimmung der Operationensequenzen wichtig. Dies wird im folgendem Abschnitt genauer beschrieben.

4.1.5. Bestimmung von Operationensequenzen

Die Prozessplanung auf einem Containerterminal besteht darin, festzulegen, auf welche Art und Weise eine Ladeeinheit transportiert wird. Es wird festgelegt, welches Transportsystem die Ladeeinheit aufnimmt und welche Zwischenabstellungen benötigt werden. Zur Bestimmung von möglichen Operationssequenzen für die Prozessplanung wird auf graphenbasierte Wegfindungsalgorithmen zurückgegriffen.

Transportgraph

Zur Bestimmung der Routen wird zunächst ein Graph benötigt, der die möglichen Wege für einen Transport enthält. Hierzu wird die Problemstellung aus Abbildung 4.7 zur Motivation der Modellierung des Graphen verwendet. Eine Ladeeinheit soll aus einer Zwischenabstellung am linken Rand des schematisch dargestellten Terminals an einen Platz am rechten Rand transportiert werden.

Abbildung 4.7.: Schematischer Grundriss eines Terminals. Start- und Zielort sind exemplarisch angegeben.

Um den Graphen aufzubauen, werden zunächst die generellen Arbeitsbereiche der Transportsysteme betrachtet. Diese sind in Abbildung 4.8 dargestellt. Die Arbeitsbereiche der Krane überspannen alle Gleise sowie die Zwischenabstellung und überschneiden sich. Diese sich überschneidenden Bereiche dienen als Übergabebereiche zwischen den Kranen. Weiterhin ist der generelle Arbeitsbereich der Längsförderanlage zu erkennen. Dieser hat mit den Arbeitsbereichen aller Krane Überschneidungsbereiche.

Die Überschneidungsbereiche werden im Folgenden *transfer area*s genannt und lassen sich jeweils durch die Transportsysteme, die eine Ladeeinheit in diesem Bereich aufnehmen und abstellen können, und die Lager-Ressourcen, die als Übergabebereich dienen, beschreiben. Exemplarisch sei hier der Überschneidungsbereich zwischen Kran 1 und Kran 2 genannt. Die Lager-Ressourcen sind hierbei konkret die Lagerplätze der

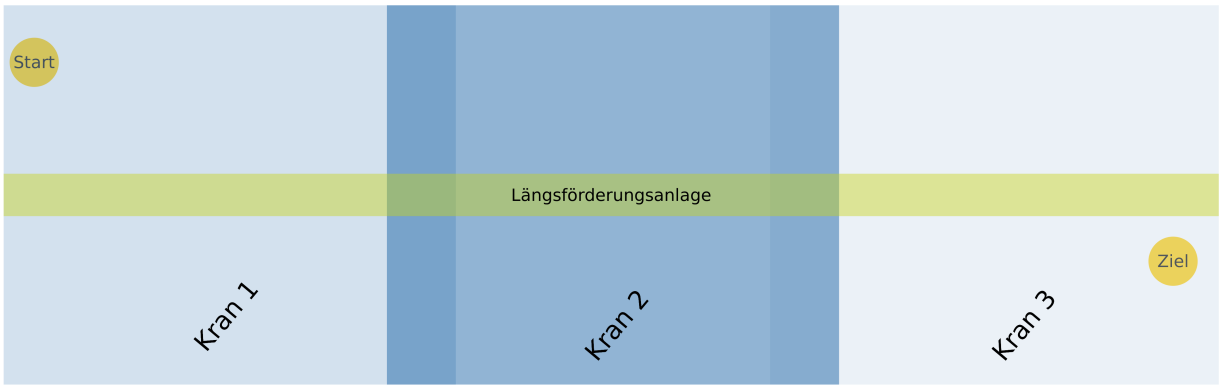


Abbildung 4.8.: Generelle Arbeitsbereiche der Transportsysteme

beiden Zwischenabstellungen oben und unten, die geometrisch innerhalb des Überschneidungsbereiches liegen. Abbildung 4.9 zeigt die entstandenen *transfer area*s und die jeweils beteiligten Transportsysteme. Hinzu kommen noch die *transfer areas*, die lediglich von einem Transportsystem bedient werden können. Diese dienen vor allem zur Abstellung von Ladeeinheiten, deren Start- und Endzug nicht gleichzeitig im Terminal stehen.

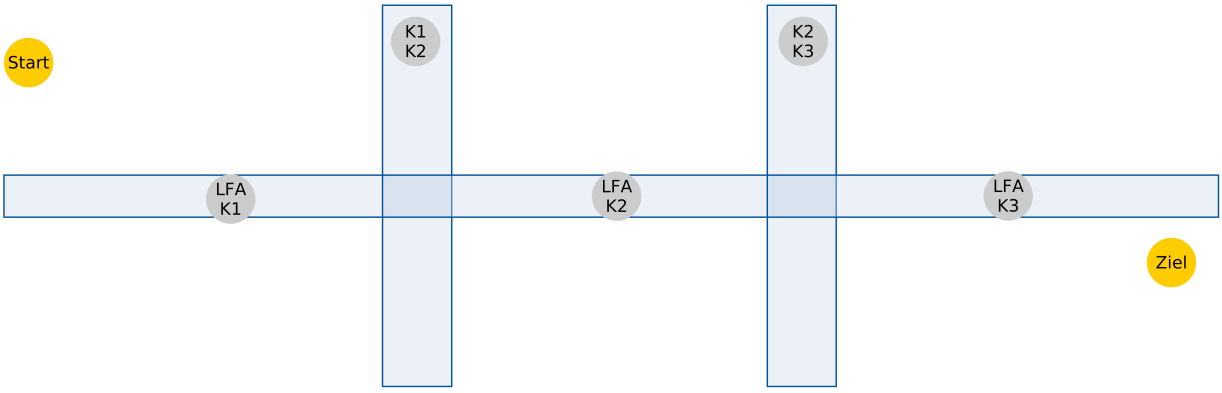


Abbildung 4.9.: Darstellung der geometrischen Flächen der *transfer area*s

Die *transfer area*s können als Knoten eines Graphen aufgefasst werden, die durch Kanten miteinander verbunden sind. Die Kanten beschreiben hierbei den eigentlichen Transport zwischen den *transfer area*s. Eine Kante zwischen zwei Knoten kann folglich nur existieren, wenn das Transportsystem in beiden *transfer area*s grundsätzlich agieren kann (siehe Abbildung 4.10).

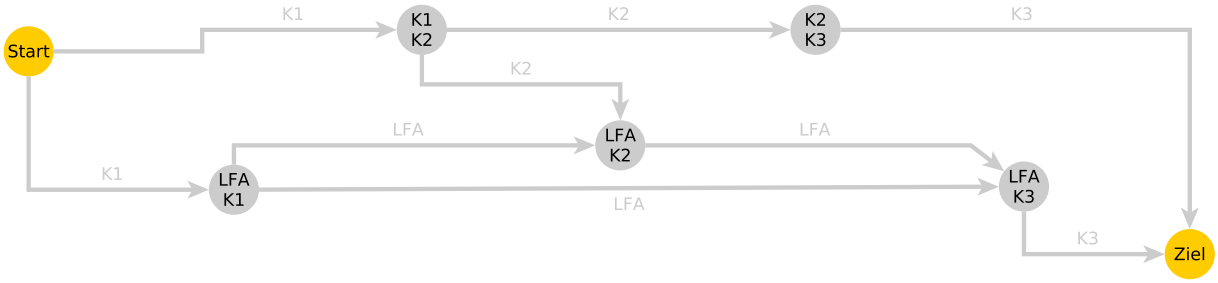


Abbildung 4.10.: Transportgraph mit den *transfer areas* als Knoten und den Transporten als Kanten

k-beste-Wegesuche

Auf dem Transportgraphen werden Algorithmen der Wegesuche angewandt, um die möglichen Routen für einen Transport einer Ladeeinheit zu bestimmen. Dazu müssen zunächst die Kanten des Transportgraphen bewertet werden. In diese Kosten-Bewertung geht zunächst die Transportdauer der Operation ein. Weiterhin können Parameter berücksichtigt werden, beispielsweise der Energiebedarf oder die Abnutzung der einzelnen Transportsysteme. Da der Transport mittels Portalkranen extrem energieintensiv ist, wird hier eine einfache Gewichtung vorgenommen, die Transporte über die Längsförderanlage bevorzugt. Dabei ergibt sich die Bewertung einer gesamten Route durch den Terminal durch eine Bewertung der einzelnen Kanten mit der Dauer des zugehörigen Transportes bei der Längsförderanlage und der doppelten Dauer bei einer Kranfahrt.

Um den Lösungsraum des Problems der Prozessplanung übersichtlich zu gestalten, werden die k besten Lösungen pro Job im Vorhinein der Optimierung bestimmt (siehe Abbildung 4.11). Dies bedeutet, dass neben der besten Lösung noch weitere Wege gesucht werden, die sich von der besten Lösung unterscheiden und die nächstbeste Bewertung haben. Die Anzahl an minimal zu bestimmenden Routen ist problemabhängig und muss auf die jeweilige Instanz angepasst werden. Teilweise kann es möglich sein, dass ein gegebenes Szenario für einen oder mehrere Jobs eine extrem hohe Anzahl an Routen benötigt. Das bedeutet, dass erst eine für den einzelnen Job recht schlechte Route aufgrund von Wechselwirkungen mit anderen Jobs der Zusammenhangskomponente eingeplant werden kann.

Bei der Bestimmung der k-besten-Wege kann auf diverse Algorithmen zurückgegriffen werden, beispielsweise den Algorithmen nach Yen (1971), Eppstein (1998) oder Althöfer (1999). Die Algorithmen werden zumeist bei der Navigation im Straßenverkehr angewandt. Im vorliegenden Problem ist eine in weiten Teilen mit der besten Route übereinstimmende Lösung akzeptabel, falls dadurch eine stark ausgelastete Ressource gemieden wird. Algorithmen wie das *Choice-Routing* nach Althöfer (1999) versuchen, die bestmöglichen Routen zu finden, die möglichst wenige Kanten mit den anderen besten Routen gemeinsam haben. Dies erscheint beim Straßenverkehr wünschenswert, bei

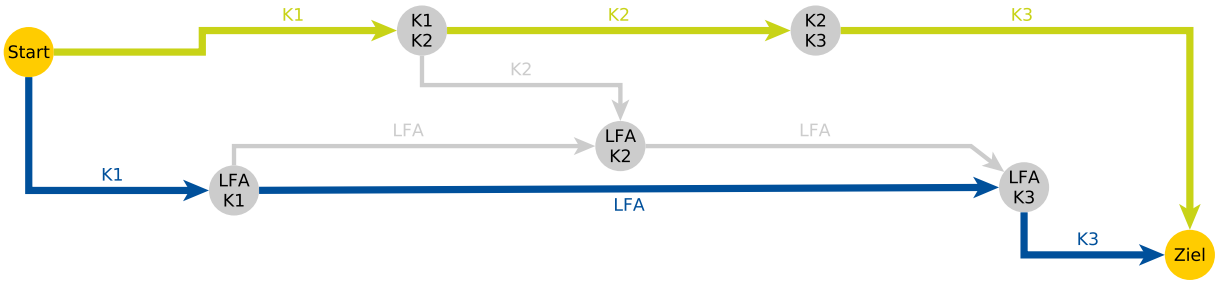


Abbildung 4.11.: Transportgraph mit zwei bestimmten Routen

der hier betrachteten Problemstellung allerdings nicht. Daher wird ein abgewandelter Yen-Algorithmus zur Routenbestimmung eingesetzt.

Aufsplitten von direkten Zyklen

Im *Job-on-Node*-Diagramm können Zyklen entstehen. Dies geschieht beispielsweise, wenn eine Ladeeinheit L_1, die auf dem Lagerplatz A steht, durch einen Auftrag A_1 zu einem Lagerplatz B transportiert werden soll und gleichzeitig eine Ladeeinheit L_2 auf diesem Lagerplatz B steht und innerhalb eines Auftrags A_2 zum Lagerplatz A transportiert werden soll.

Enthält eine der Zusammenhangskomponenten direkte Zyklen, so müssen diese aufgesplittet werden, da angenommen wird, dass es nicht möglich ist, dass beide Aufträge A_1 und A_2 durch Direkttransporte umgeschlagen werden können. In diesem Fall muss bei einem der beiden Aufträge der Direkttransport aus der Menge der möglichen Routen entfernt werden. Hier wird angenommen, dass der Job des Zyklus, der zuerst gestartet werden kann, nicht mehr als Direkttransport ausgeführt wird und der Zyklus an dieser Stelle aufgespaltet wird.

Bei einer direkten Codierung der Routenwahl pro Job wäre eine gesonderte Überprüfung dieser Konstellation nötig und es müsste entschieden werden, bei welchem Job der Direktumschlag ermöglicht wird. Durch das in Algorithmus 3.2 vorgestellte Verfahren und die Codierung mittels Prioritätsregeln wird dieses Problem umgangen.

4.1.6. Modellierung der Einplanregeln

Ein Modell beschreibt eine abstrakte Abbildung eines realen Systems. Im Rahmen der Modellbildung müssen die wesentlichen Gesichtspunkte und Einflussfaktoren identifiziert werden, die für die Problemstellung wichtig sind. Im vorliegenden Fall muss berücksichtigt werden, dass das Modell generisch gehalten wird, um eine möglichst große Vielfalt an unterschiedlichen konkreten Problemstellungen zu modellieren. Da das Modell im Rahmen eines Multiskalen-Algorithmus eingesetzt wird, muss es die Möglichkeit bieten, verschiedene Detaillierungsgrade einzelner Elemente des Systems abzubilden. So muss es beispielsweise möglich sein, eine strukturelle Modellierung der Transportprozesse einer Ressource, die eine möglichst detaillierte Beschreibung der einzelnen Teilprozesse beschreibt, durch eine abstrahierte Modellierung zu ersetzen.

Um ein beliebiges Problem möglichst effizient mittels eines Multiskalen-Algorithmus lösen zu können, müssen zunächst die Anzahl an verwendeten Skalen und die Skalen selbst definiert werden. Dies kann sich je nach Problemstellung unterscheiden. Die Anzahl der Skalen muss abgewägt werden, indem der Nutzen der einzelnen Ebenen klar definiert sein muss. Grundsätzlich bietet die Definition von Detaillierungsgraden allerdings einigen Spielraum.

Für das Anwendungsbeispiel hat es sich als sinnvoll erwiesen, lediglich zwei Detaillierungsgrade zu definieren, da der Informationsgewinn durch eine zusätzliche dritte Detaillierungsebene zu gering ist. Der Aufwand zur Berechnung eine guten Lösung in einer weiteren Ebene ist im Vergleich zum Nutzen bei der Abschätzung der Ressourcenauslastungen, zu hoch. Der vorgestellte Algorithmus lässt sich jedoch problemlos um weitere Detaillierungsgrade erweitern, sollte das zu betrachtende Problem dies nötig machen.

Für die Modellierung der Einplanregeln müssen einige grundsätzliche Modellannahmen getroffen werden. Da eine Ladeeinheit von einem Ort zum anderen zu transportieren werden muss, werden bewegliche, transportierende Ressourcen benötigt, die diesen Transport ausführen. Alle Transportsysteme sind somit bewegliche Ressourcen. Gibt es mehr als eine bewegliche Ressource, beispielsweise mehrere Portalkrane, die in einer Kranbahn angeordnet sind, so dass sie sich nicht überholen können, müssen Kollisionen berücksichtigt werden. Ressourcen, in diesem Beispiel die Krane, können sich gemeinsam benötigte Ressourcen, in diesem Fall den Platz auf der Kranbahn, teilen. Um Kollisionen zu vermeiden, müssen die Krane, die dem transportierendem Kran im Weg sind, entweder den benötigten Raum freigeben, was einer Verwendung von dynamischen Arbeitsbereichen entspricht, oder der Transport muss in mehrere Teiltransporte, eine sogenannte Kranstafette, aufgeteilt werden. In diesem Fall besitzen die Krane statische Aktionsbereiche, in denen sie Transporte ausführen können. Zur Ausführung einer Kranstafette gibt es zwischen diesen Aktionsbereichen allerdings Übergabebereiche, in denen die Ladeeinheiten übergeben werden können.

Makroskopische Modellierung

Auf makroskopischer Ebene wird für die einzelnen Komponenten eine Modellierung nach der *Black-Box*-Theorie angenommen. Diese dient dazu, komplexe Systeme möglichst einfach abzubilden, so dass lediglich das äußere Verhalten der betrachteten Komponenten abgebildet wird. Konkrete Vorgänge in der inneren Struktur der Komponenten werden allerdings vernachlässigt. Es wird lediglich eine *Input-Output*-Beziehung beschrieben.

Krane Bei der Modellierung der Einplanregeln für die Krane sind unterschiedliche Varianten möglich. Hier werden zwei Varianten vorgestellt, die theoretisch kombiniert im Rahmen eines dreistufigen Multiskalen-Algorithmus eingesetzt werden können. In diesem Fall würde die zweite Variante die mesoskopische Ebene beschreiben, da hierbei eine etwas detailliertere Einplanung vorgenommen wird.

Eine stark makroskopisch geprägte Beschreibung des Modells ist die Diskretisierung des gesamten Optimierungszeitraums in Intervalle. Diese Intervalle besitzen pro Kran eine Intervallkapazität, die sich als integrale Größe aus dem Produkt der Kapazität $k = 1$, die angibt, dass ein Kran nur eine Operation gleichzeitig ausführen kann, und der Intervalldauer ergibt. Für jede Operation wird ein Intervall bestimmt, in dem sie ausgeführt werden soll. Dabei wird überprüft, ob der Ressourcenbedarf als Produkt aus $r_k = 1$ und der Operationsdauer in Summe mit der bisherigen Auslastung im Intervall die Kapazitätsgrenze nicht überschreitet. Weiterhin ist es möglich, eine Operation auf zwei benachbarte Intervalle aufzuteilen. Ledigliche Interaktion zwischen den Kranen, wie beispielsweise ein nötiges Ausweichmanöver, wird vernachlässigt. Für einen angefragten Transport wird die Dauer unter Berücksichtigung der Geschwindigkeit und der Beschleunigung der Krane berechnet. Die Dauer der Rüstfahrten wird pauschal angenommen.

Eine etwas detailliertere Modellierung beschreibt die Auslastung der Krane als eine einfache Treppenfunktion. Im Gegensatz zur vorherigen Variante werden für alle Operationen exakte Startzeitpunkte und nicht nur die Zugehörigkeit zu einem Intervall bestimmt. Die Dauer der Rüstfahrten wird pauschal bestimmt, obwohl eine Operationenreihenfolge vorhanden ist. Da diese Operationenreihenfolge allerdings dem Einfluss diverser Modellvereinfachungen unterliegt, wie beispielsweise der Vernachlässigung der Interaktionen mit anderen Kranen innerhalb der Übergabebereiche, ist eine exakte Bestimmung der Rüstfahrten nicht sinnvoll.

Im weiteren Verlauf wird letztere Variante verwendet, da die recht grobe Beschreibung der zuvor gestellten Variante zu großen Abweichungen führt. Die Abweichungen zwischen makroskopischer und mikroskopischer Planung sind zu groß, so dass der Mehrnutzen der makroskopischen Ebene entfällt.

Rüstfahrten und Rüstdauern der Krane Die Auswertung von Realdaten eines typischen Betriebstages auf einem 700 m langen Containerterminal bezüglich der Krane und deren Rüstfahrten (siehe Anhang B.1.2), lässt erkennen, dass eine empirische Standardabweichung von 22 s bei einem Mittelwert von 40 s bei den Rüstfahrtendauern vorliegt. Dies führt zu einem Variationskoeffizienten, der größer als 0.5 ist. Das heißt, dass die dimensionslose Streuung relativ groß ist. Abbildung 4.12 zeigt ein Histogramm mit den relativen Häufigkeitsdichten.

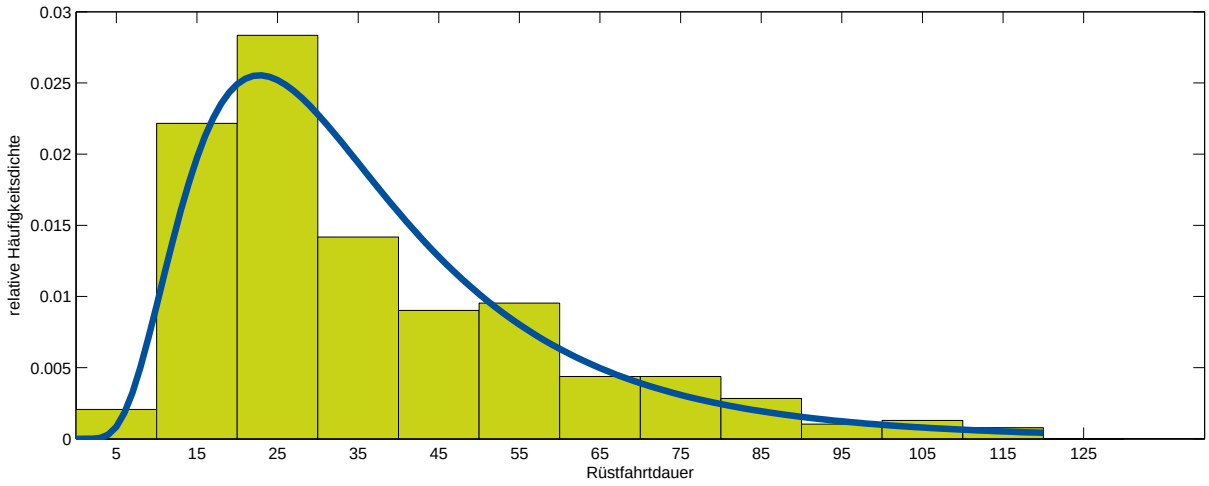


Abbildung 4.12.: Histogramm der Rüstfahrten auf einem Terminal mit fünf Kranen. Weiterhin ist die Dichtefunktion einer logarithmischen Normalverteilung mit den Parametern $\mu = 3.46134$ und $\sigma = 0.580295$ dargestellt.

Bei der Betrachtung des Histogramms fällt auf, dass es sich um eine unsymmetrische Verteilung handelt, die bis zum Maximum x_{max} steil ansteigt und Bereich $x > x_{max}$ abfällt. Dies ist charakteristisch für eine logarithmische Normalverteilung. Dadurch, dass der Disponent versucht, die Rüstfahrten zu minimieren, verschiebt sich die Verteilung nach links. Allerdings ist es zumeist nicht möglich, die Rüstfahrten auf einen Wert von 0 zu minimieren, weswegen für das Maximum $x_{max} > 0$ gilt. Weiterhin gibt es trotz Optimierung Ausreißer, die sich in der Verteilung rechts vom Maximum ablesen lassen.

Die Annahme einer logarithmischen Normalverteilung unter Zuhilfenahme der Maximum-Likelihood-Methode führt zu den Parametern $\mu = 3.461$ und $\sigma = 0.580$. Der Verlauf der Dichtefunktion ist ebenfalls in Abbildung 4.12 dargestellt. Im Rahmen eines *Kolmogorow-Smirnow*-Tests wird die Richtigkeit der Nullhypothese überprüft. Die Nullhypothese sagt in diesem Fall aus, dass die Zufallsvariable, die die Rüstfahrtdauer beschreibt, einer logarithmischen Normalverteilung folgt. Bei einem Signifikanzniveau von 5 % wird diese Hypothese nicht zurückgewiesen. Es liegt ein p-Wert von 0.8086

vor, der angibt, dass mit knapp 81 prozentiger Wahrscheinlichkeit ein solches Stichprobenergebnis auftreten kann, wenn eine logarithmische Normalverteilung vorliegt.

Es lässt sich festhalten, dass die Dauer der Rüstfahrten allem Anschein nach einer logarithmischen Normalverteilung folgt. Bei einer pauschalen Berücksichtigung der Rüstfahrten muss die große Standardabweichung berücksichtigt werden.

Längsförderanlage Ähnlich wie bei der Modellierung der Krane wurde eine *Black-Box*-Implementierung vorgenommen. Zur Berechnung der Operationsdauer wird mittels der Manhattan-Metrik eine Distanz zwischen Start- und Zielort ohne Kollisionsberücksichtigung bestimmt. Dabei wird nicht berücksichtigt, welches AGV der Längsförderanlage den Transport ausführt, sondern lediglich überprüft, ob im entsprechenden Zeitraum ein freies AGV zur Verfügung steht. Die Rüstzeiten werden auch hier pauschal aufgeschlagen.

Zwischenabstellung Die zur Verfügung stehenden Flächen für die Zwischenabstellung werden diskretisiert. Somit bildet beispielsweise eine Aneinanderreihung von mehreren Stellplätzen eine Lagerspur. Lagerspuren lassen sich wiederum zu Lagerbereichen zusammenfassen. Grundsätzlich lassen sich Subressourcen auf diesen Strukturen definieren, die nur einen gewissen Bereich umfassen. Die einzelnen Stellplätze besitzen dabei eine Ausdehnung in Längsrichtung von 3.10 m, so dass ein 20-Fuß-Container unter Berücksichtigung, dass der Kran einen gewissen Platz zum Aufnehmen der Ladeeinheit braucht, zwei Stellplätze benötigt.

In der makroskopischen Betrachtung werden bei der Lagerung von Ladeeinheiten auf Zügen exakte Positionen berücksichtigt, da diese vom Transportprogramm vorgegeben sind. Die Positionierung der Ladeeinheiten erfolgt somit unter Berücksichtigung der zur Verfügung stehenden, freien Stellplätze.

Für Zwischenabstellungen auf den Lagerspuren des Containerterminals wird kein exakter Abstellplatz und somit keine exakte Folge von zusammenhängenden Stellplätzen bestimmt, sondern lediglich die zu den *transfer area*s (siehe Abschnitt 4.1.5) zugehörigen Zwischenabstellungen als Ressource angefragt. Diese bestehen in der Regel aus mehreren Stellplätzen. Wiederum wird überprüft, ob grundsätzlich die zur Verfügung stehenden Stellplätze zur Aufnahme der Ladeeinheit ausreichen. Dabei wird nicht berücksichtigt, ob es sich bei den zur Verfügung stehenden Stellplätzen um benachbarte Plätze handelt. Als Standort der Ladeeinheit wird der Mittelpunkt der zugehörigen Zwischenabstellung verwendet. Dieser wird zur Berechnung der Transportdauer eingesetzt.

Auf einem Containerterminal werden Ladeeinheiten, die dies erlauben, aufeinander gestapelt, um die Kapazität der Zwischenabstellungen zu erhöhen. Der reale Stapelfaktor ergibt sich aus dem Bedarf an die Zwischenabstellung und der vorhandenen

Kapazität auf dem Boden. Zur Kapazitätsbestimmung auf makroskopischer Ebene muss ein Stapelfaktor angenommen werden, der den Anteil an stapelbaren Ladeeinheiten an der Gesamtanzahl der Ladeeinheiten berücksichtigt. Weiterhin führt ein zu hoher Stapelfaktor zu einer hohen Anzahl an Umstapelungen und somit zu einer erhöhten Kranauslastung. Je nach Probleminstanz und den Möglichkeiten auf dem Containerterminal bewegt sich ein realistischer Stapelfaktor im Intervall $[1.0; 3.0]$.

Bei der Bestimmung der Kapazität und der Auslastung muss berücksichtigt werden, dass sich *transfer area*s überschneiden. Abbildung 4.13 zeigt die Arbeitsbereiche zweier Krane. Aus diesen Arbeitsbereichen ergeben sich die *transfer area* ZA K1, in der der Kran 1 Ladeeinheiten abstellen kann, die später wieder durch den Kran 1 aus der Abstellung genommen werden können. Analog dazu ist dem Kran 2 eine Zwischenabstellung ZA K2 zum gleichen Zweck zugeordnet. Der Überschneidungsbereich dieser beiden *transfer area*s bildet eine gemeinsame *transfer area* AB K1/K2, die für die Übergabe von Ladeeinheiten zwischen Kranen verwendet wird.

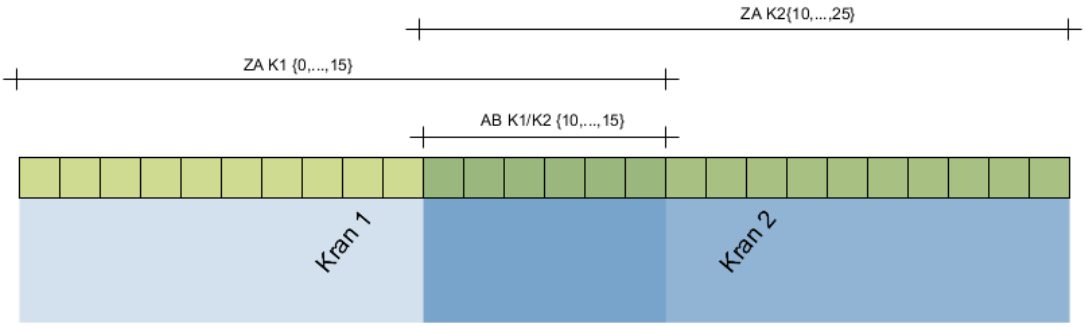


Abbildung 4.13.: Zwischenabstellungsbereiche und Arbeitsbereiche der Krane

In der Regel wird versucht, in den Übergabebereich auch nur Ladeeinheiten abzustellen, die von beiden Kranen bewegt werden müssen. Allerdings kann bei hoher Auslastung der Zwischenabstellungen der Fall eintreten, dass auch Ladeeinheiten, die nicht in den Arbeitsbereich eines anderen Krans verladen werden müssen, innerhalb des Übergabebereiches abgestellt werden müssen. Dies ist häufig der Fall, wenn viele Ladeeinheiten zwischen unterschiedlichen Zugbündeln umgeschlagen werden müssen. Ein Zugbündel umfasst dabei eine komplette Belegung aller Gleise durch Züge. Bei einem Umschlag zwischen zwei unterschiedlichen Zugbündeln kann es zu dem Fall kommen, dass kein Direktumschlag möglich ist. Dies tritt auf, wenn der Start- und der Zielzug nicht zeitgleich im Terminal sind. Folglich muss die Ladeeinheit zwischengelagert werden.

Es empfiehlt sich, die Bestimmung der Kapazität der einzelnen *transfer area*s auf makroskopischer Ebene abhängig vom Transportprogramm zu wählen. Dies lässt sich an einem Beispiel zeigen. Wird die Zwischenabstellung mittels nummerierter Lagerplätzen diskretisiert, so wird im Rahmen dieses Beispiels angenommen, dass ZA K1 die Plätze 0

bis 15 umfasst. Die *transfer area* ZA K2 soll die Lagerplätze 10 bis 25 umfassen. Die Plätze 10 bis 15 bilden den Übergabebereich (siehe Abbildung 4.13). Insgesamt stehen somit 26 Plätze zur Verfügung. ZA K1 und ZA K2 umfassen jeweils 16, der Übergabebereich 6 Plätze. Die Gesamtkapazität der Lagerspur beträgt bei einem Stapelfaktor von 1.0 somit 26 Einheiten. Die exklusiv für ZA K1 zur Verfügung stehende Gesamtkapazität beträgt 10 Einheiten, analog gilt gleiches für ZA K2. Die restliche Kapazität von 6 Einheiten kann nun unterschiedlich verteilt werden.

Müssen nach Transportprogramm viele Ladeeinheiten vom Arbeitsbereich von Kran 1 in den Arbeitsbereich von Kran 2 transportiert werden, so ist es sinnvoll, die gesamte restliche Kapazität der *transfer area* AB K1/K2 zuzuweisen. Dies entspricht dem oben beschriebenen Verfahren, in der Regel nur Ladeeinheiten in den Übergabebereich abzustellen, die über diesen transportiert werden müssen. Es kann auch der Fall eintreten, dass es sich anbietet, die restliche Kapazität aufzuteilen. Dies sollte bedacht werden, wenn es weniger Ladeeinheiten gibt, die zwischen den Kranbereichen übergeben werden müssen und mehr Ladeeinheiten, die von einem Zug auf einen Zug eines anderen Zugbündels umgeschlagen werden, wobei sich das Ziel im gleichen Kranbereich befindet. So kann man beispielsweise den Zwischenabstellungen ZA K1 und ZA K2 jeweils eine Gesamtkapazität von 12 Plätzen zuweisen, dem Übergabebereich eine Kapazität von 4 Plätzen. Bei der Berücksichtigung von Stapelungen wird die zuvor bestimmte Kapazität der einzelnen *transfer area*s dementsprechend mit dem Stapelfaktor multipliziert.

Unscharfe makroskopische Modellierung

Für eine unscharfe Betrachtung auf makroskopischer Ebene muss festgelegt werden, welche Bereiche mit Unsicherheit behaftet sind. Dies sind zunächst die Zugankunftszeiten, die durch Verspätungen eine Unsicherheit besitzen. Weiterhin ist auf makroskopischer Ebene die modellbedingte Unschärfe ein wesentlicher Aspekt. Durch die Modellierung der Zwischenabstellungen als *Black-Box*-System, bei dem der einzelnen Ladeeinheit kein konkreter Platz im Lager zugeordnet wird, unterliegen die Start- und Zielorte eines Transportes einer Unsicherheit. Diese geometrischen Parameter sind bei der Bestimmung der Transportdauer von entscheidender Bedeutung, da die Unschärfe im Ort direkt die Unschärfe in der Dauer impliziert. Weiterhin entsteht eine Unsicherheit durch die sequenzabhängigen Rüstzeiten, da die Operationenreihenfolge auf makroskopischer Ebene lediglich zur Steuerung des Multiskalen-Algorithmus dient. Die makroskopische Operationenreihenfolge wurde unter dem Gesichtspunkt der Unsicherheit in den Operationsdauern optimiert. Auf mikroskopischer Ebene werden diese Dauern allerdings der Situation entsprechend konkretisiert. Dadurch wird auch eine andere Operationenreihenfolge auf mikroskopischer Ebene durch die Optimierung bestimmt. Folglich ändern sich auch die Rüstfahrten, die somit auf makroskopischer Ebene auch Unsicherheiten unterliegen.

Die Zugankünfte können auf Basis von Expertenwissen und Beobachtungen als Fuzzy-Zahl oder Fuzzy-Intervall modelliert werden. Für die Berücksichtigung der Unsicherheiten bezüglich der Rüstfahrten lassen sich statistische Daten erheben (siehe Abbildung 4.12), die mittels verschiedener Methoden in Zugehörigkeitsfunktionen überführt werden können. Civanlar u. Trussell (1986) und Turksen (1991) stellen einige Methodiken vor, die beispielsweise bezüglich eines Konfidenzkriterium optimale Zugehörigkeitsfunktionen herleiten, die das *Consistency Principle* nach Negoita u. a. (1978) erfüllen.

Zur Beschreibung der Unsicherheit, die durch die Positionierung der Ladeeinheiten im makroskopischen Lager entsteht, muss die Lagerlogik berücksichtigt werden. Die einfachste Annahme ist, dass jede Lager-Position für eine Ladeeinheit die gleiche Wahrscheinlichkeit hat und somit gleich möglich ist. Wie zuvor beschrieben wird versucht, Lageroperationen von Ladeeinheiten, die nicht zwischen zwei Kranbereichen umgeschlagen werden müssen, möglichst nicht innerhalb des Übergabebereiches abzustellen. Trotzdem soll diese Möglichkeit gegeben sein, wenn die anderen Lagermöglichkeiten ausgeschöpft sind. Über die Modellierung der Fuzzy-Intervalle zur Beschreibung der Position einer Ladeeinheit lassen sich dementsprechende Präferenzen abbilden.

Zu der modellbedingten Unsicherheit muss noch der Faktor Mensch berücksichtigt werden. Aus sicherheitstechnischen Gründen wird der Umschlag auf einem Containerterminal im Binnenland halbautomatisch durchgeführt. Die Feinjustierung der Katze erfolgt immer noch durch einen Kranführer. Neben diesem Faktor spielt auch noch der Zustand der jeweiligen Ladeeinheit eine Rolle, da durch abgenutzte Aufnahmevorrichtungen Verzögerungen auftreten können. Diese müssen bei der Modellierung ebenso berücksichtigt werden.

Mikroskopische Modellierung

Innerhalb der mikroskopischen Betrachtung wird für jede Transportoperation eine exakte Beschreibung der Bewegung der ausführenden Transportressource unter Berücksichtigung der Beschleunigung durchgeführt. Auf Basis dieser Beschreibung wird mittels geometrischer Schnittberechnung eine exakte Kollisionsvermeidung durchgeführt. Bei der Abstellung von Ladeeinheiten wird für die entsprechende Ladeeinheit ein exakter Platz innerhalb des Abstellbereiches festgelegt.

Krane Die Kollisionsvermeidung der Krane wird auf mikroskopischer Ebene dadurch sichergestellt, dass jeder Kran einen polygonal brandeten Arbeitsbereich in Raum und Zeit besitzt. Dieser beschreibt den für den Kran befahrbaren Raum in Längsrichtung des Terminals über die Zeit. Bei der Anfrage einer Einplanung wird überprüft, ob die durch den Transport entstehende Trajektorie des Krans unter Berücksichtigung der Breite des Krans und eines Sicherheitsabstandes komplett innerhalb dieses Arbeitsbereiches liegt. Dabei muss neben dem Heben und Senken der Katze zur Aufnahme und zum Abstellen

der Ladeeinheit, sprich dem eigentlichen Transport, auch die Rüstfahrt zum Startort der Ladeeinheit berücksichtigt werden. Die Arbeitsbereiche der benachbarten Krane werden entsprechend der eingeplanten Operationen angepasst. Abbildung 4.14 stellt die Transporte in einem Weg-Zeit-Diagramm dar. Die Rüstfahrten wurden hier nicht visualisiert, wurden jedoch berücksichtigt.

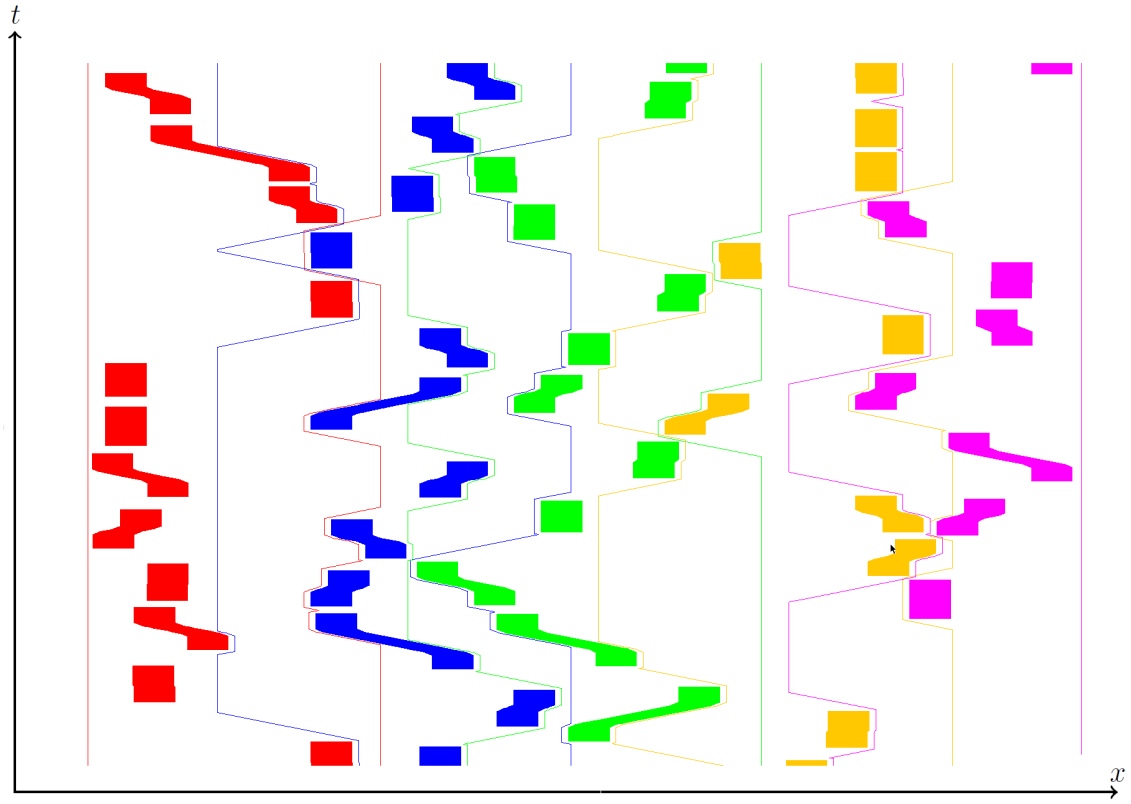


Abbildung 4.14.: Ausschnitt aus einem Weg-Zeit-Diagramm, das die Kranwege und die Arbeitsbereiche von fünf Kranen, die auf einer Kranbahn arbeiten, darstellt.

Die Form der Polygone im Weg-Zeit-Diagramm ergibt sich aus einem Kranspiel. Ein Kranspiel beschreibt die Rüstfahrt zum Transport, das Absenken der Lastkatze ohne eine Ladeeinheit, die Positionierung durch den Kranführer und das eigentliche Aufnehmen der Ladeeinheit. Daraufhin folgt das Anheben der Ladeeinheit und der eigentliche Transport, die Fahrt zum Zielort der Operation. Am Zielort wird die Ladeeinheit abgesenkt und über dem Bestimmungsort positioniert. Die Katze wird am Ende eines Lastspiels ohne Last hochgefahren. Abbildung 4.15 verdeutlicht den gesamten Vorgang nochmals. Die Teilbereiche des Aufnehmens der Ladeeinheit und des Absenkens lassen sich im Weg-Zeit-Diagramm als senkrechte Teilpolygone wiederfinden, der eigentliche Transport führt zur einer Veränderung in Längsrichtung.

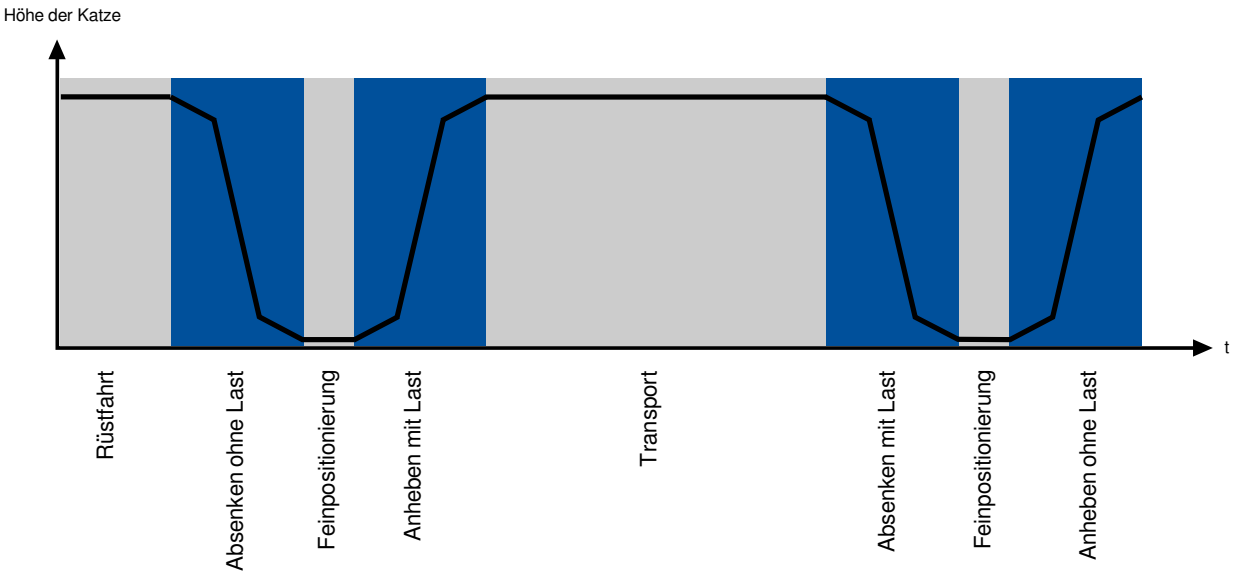


Abbildung 4.15.: Zeitlicher Ablauf eines Kranspiels

Längsförderanlage Auf mikroskopischer Ebene werden die Bewegungen jedes einzelnen AGVs bestimmt, die zur Durchführung eines Transportauftrages nötig sind. Im Rahmen der Modellierung wurde ein Wegegraph hinterlegt, der die möglichen Fahrwege der AGVs beschreibt. Auf diesem Wegegraph wurde ein angepasster A*-Algorithmus entwickelt, der die kürzesten Wege unter Berücksichtigung von Kollisionen bestimmt. Zur Kollisionsüberprüfung werden dreidimensionale Polytope erzeugt, die im zweidimensionalen Raum und der Zeit verschieblich sind. Wird kein Schnitt des einzuplanenden Polytops mit den bereits vorhanden Polytopen festgestellt, kann die angefragte Operation eingeplant werden.

Zwischenabstellung Im Gegensatz zur makroskopischen Modellierung der Zwischenabstellungen werden auf mikroskopischer Ebene für jede Lagerung exakte Positionen bestimmt. Der hierarchische Aufbau der Zwischenabstellungen erlaubt eine Anfrage auf Ebene der *transfer area*s. Im weiteren Verlauf werden iterativ Unterstrukturen der modellierten Lagerspuren erzeugt. Diese umfassen so viele Stellplätze, wie für die Lagerung der angefragten Ladeeinheit benötigt werden. Steht für den angefragten Zeitraum auf allen Stellplätzen genug Kapazität zur Verfügung, so wird eine mögliche Lageroperation erzeugt. Die Menge aller möglichen Lageroperationen wird daraufhin bewertet und sortiert und im Rahmen des Prozesses der Ablaufplanung zur Auswahl gestellt. Dabei werden zunächst Stapelungen vermieden, wenn möglich. Für Ladeeinheiten, die nicht in einen anderen Kranbereich transportiert werden müssen, werden Lageroperationen, die nicht im Übergabebereich zwischen zwei Kranen liegen, gegenüber Lageroperationen im Übergabebereich präferiert.

4.1.7. Modellierung der Zielfunktion

Die Zielfunktion beschreibt die Optimierungsziele und somit die Größen, die maximiert oder minimiert werden sollen. Im Rahmen des Containerterminalbeispiels gibt es hier mehrere Größen, die entweder gewichtet aufaddiert, hierarchisch berücksichtigt oder im Rahmen einer Pareto-Optimierung berücksichtigt werden können. Es handelt sich folglich um eine mehrkriterielle Optimierung (siehe Definition 27). Zunächst wird hier auf die wichtigen Größen eingegangen, nachfolgend wird eine Kombination der Optimierungsziele diskutiert.

Verspätete Operationen und Jobs

Es kann sein, dass ein Job nicht rechtzeitig zur Abfahrt des Zielzuges beendet werden kann, falls der Terminal zu stark ausgelastet ist. Zumeist wird in einem solchen Fall die Minimierung der Summe der Verspätungen in der Zielfunktion berücksichtigt. In dieser Anwendung wird diese Zielfunktion dahingehend abgeändert, dass die Anzahl der Ladeeinheiten, die zum Abfahrtszeitpunkt des Zielzuges nicht auf diesen abgestellt werden konnten, minimiert wird.

Hiermit ergeben sich noch weitere Punkte, die berücksichtigt werden müssen. So sind die Jobs untereinander nicht vollkommen unabhängig. Ist es nicht möglich, eine Ladeeinheit pünktlich vom Zug abzuladen, so sind auch die nachfolgenden Operationen eines anderen Jobs unter Umständen nicht ausführbar, wenn diese Operationen den Transport der Ladeeinheit zu dem Platz beschreiben, der durch die nicht pünktlich abgefertigte Ladeeinheit weiterhin blockiert ist. Innerhalb der Erzeugung des Ablaufplans muss sichergestellt werden, dass Nachfolgeoperationen einer verspäteten Operation, die einem anderen Job angehören, weiterhin ausführbar sind, so weit dies möglich ist. Das heißt, dass für den Job, der die verspätete Operation enthält, alle nachfolgenden Operationen ebenfalls als *nicht einplanbar* deklariert werden. Weiterhin ist es bei Jobs, die verspätet sind, möglich, die zugehörigen Ladeeinheiten in die Zwischenabstellung umzuleiten, um den Stellplatz auf dem Zug frei zu machen. So können nachfolgende Operationen andere Ladeeinheiten, die vom entsprechenden Platz auf dem Zug abhängig sind, ausgeführt werden. Ist der späteste Start der verspäteten Operation allerdings vom Zeitpunkt der Zugabfahrt abhängig, ist es nicht mehr möglich, die Ladeeinheit vom Zug abzuladen und ggf. eine andere Ladeeinheit auf dem entsprechenden Platz abzustellen.

Minimierung der verspäteten Jobs

Eine Operation O_i wird als verspätet bezeichnet, wenn das Operationende c_i der Operation nach ihrem *spätesten Ende* LE_i liegt. Im Rahmen des Containerterminal-Problems

werden bei verspäteten Operationen alle nachfolgenden Operationen des entsprechenden Jobs ebenfalls als verspätet markiert. Hängen andere Jobs durch Vorgängerbeziehungen von diesem Job ab, so wird, wenn möglich, die Ladeeinheit ins Zwischenlager umgeleitet, so dass der weitere Betrieb ungestört möglich ist. Die Minimierung der Anzahl an verspäteten Jobs, im folgenden als DNF (*Did not finish*) bezeichnet, ist das Hauptkriterium der Zielfunktion.

Minimierung der Kranlängsfahrten

Die Kranlängsfahrten sind sehr energieintensiv, weswegen diese in Summe minimiert werden. Dies hat zunächst Auswirkungen auf die Routenwahl, weil somit mehr Transporte auf die Längsförderanlage umgeleitet werden. Weiterhin ist eine geschickte Einplanung der Rüstfahrten des Krans eine Möglichkeit, die Kranlängsfahrten zu minimieren.

Minimierung der Rüstfahrten

Die Rüstfahrten sind nötig, allerdings lässt sich durch geschickte Anordnung der Operationen im Ablaufplan für eine einzelne Ressource die Dauer der Rüstfahrten minimieren.

Minimierung der Kranspiele pro Ladeeinheiten

Ungeschickte Stapelung im Lager führt dazu, dass es unter Umständen nötig ist, eine Ladeeinheit umzustellen, um an eine andere zu gelangen. Daher ist es sinnvoll, die mittlere Anzahl an Kranspielen pro Ladeeinheit zu minimieren. Der Kran ist das einzige Transportsystem, das eine Ladeeinheit anheben kann. Somit dient die Anzahl an Kranspielen als Indiz für die Anzahl an Umstapelungen.

Berücksichtigung der Auslastung

Auf makroskopischer Ebene erscheint eine gleichmäßige Verteilung der Auslastung über den gesamten Zeitbereich sinnvoll, um etwaige Änderungen am Ablaufplan abzufangen. Hierzu muss zunächst klar sein, was der Begriff Auslastung im Rahmen des Containerterminal-Problems aussagt.

Der Begriff Nutzung ist in der Technik definiert als die Verwendung einer Ressource zur Erfüllung einer Arbeitsaufgabe. Die Auslastung dient als Maß dafür, wie viel von der theoretisch zur Verfügung gestellten Arbeitsfähigkeit (Kapazität) einer Ressource tatsächlich für die Erfüllung seiner Arbeitsaufgabe verwendet wird (siehe REFA

Verband für Arbeitsstudien und Betriebsorganisation e. V. (1984)). Eine Maximierung der Robustheit muss die Auslastung der einzelnen Ressourcen berücksichtigen. Im Rahmen des Containerterminal-Problems wird zunächst zwischen Transportsystemen und Lagerressourcen unterschieden.

Der Begriff Auslastung lässt sich bei Transportsystemen exakt definieren. Ein Kran kann entweder einen Transport ausführen oder ruhig stehen. Die Auslastung ergibt sich somit aus der Einsatzzeit einer Ressource, bezogen auf die Länge eines Zeitfensters. Ebenso lässt sich die Auslastung für gemeinsam genutzte Ressourcen definieren. Betrachtet man beispielsweise den Raum auf der Kranbahn, so kann dieser entweder von einem Kran in Anspruch genommen werden oder frei sein. Auch die Kapazität lässt sich einfach definieren, so kann der Raum der Kranbahn nur von einem Kran gleichzeitig genutzt werden, da sonst beide Krane kollidieren würden.

Bei den Zwischenabstellungen kann ein Stellplatz entweder belegt oder frei sein, was dazu führt, dass sich die Auslastung zu einem Zeitpunkt aus den belegten Plätzen in Relation zu den verfügbaren Plätzen berechnen lässt. Allerdings lässt sich hier bemerken, dass auch noch Stapelungen möglich sind. So existieren weitere Stellplätze auf einer Ladeeinheit, die am Boden steht. Ebenso problematisch ist der Auslastungsbegriff in Relation zur Menge der zur Verfügung stehenden Ladeeinheiten zu sehen. Soll eine Menge von 40-Fuss-Container abgestellt werden, es stehen allerdings nur nicht benachbarte 20-Fuss-Stellplätze bereit, so muss umgestapelt werden, um die freie Kapazität zu nutzen.

Grundsätzlich muss der Begriff Auslastung immer unter dem Gesichtspunkt der aktuellen Verhältnisse auf dem Terminal gesehen werden. Dies zeigt sich sowohl bei der oben genannten Problematik der Ladeeinheiten-Größe als auch bei etwaigen Optimierungskriterien für die Transportsysteme.

Soll ein *Resource Leveling* vorgenommen werden (siehe Abschnitt 3.3.8), so muss berücksichtigt werden, dass sich durch das Transportprogramm und die Zugankünfte die Anzahl der zu einem Zeitpunkt umzuschlagen Ladeeinheiten ändert. Zu Beginn eines 8h Nachtbetriebs sind zumeist nur zwei bis drei Züge im Terminal, während nach 4h Betriebszeit alle Gleise belegt sein können. Weiterhin ist der Grad der Auslastung einer Zwischenabstellung von den möglichen Stapelungen abhängig, was wiederum Einfluss auf das *Resource Leveling* hat.

Zielfunktion

Sowohl auf makroskopischer als auch auf mikroskopischer Ebene wird eine mehrkriterielle Optimierung (siehe Definition 27) vorgenommen, die dem Prinzip der *Lexikographischen Ordnung* folgt (siehe Jee u. a. (2007)). Dabei handelt es sich um einen hierarchischen Ansatz, bei dem die unterschiedlichen Optimierungsgrößen eine unterschiedliche Priorität besitzen.

Definition 39: Lexikographische Ordnung

Eine lexikographische Ordnung ermöglicht es, für zusammengesetzte Objekte eine lineare Ordnung zu erhalten. Dabei wird auf der Menge M ein Vergleich zwischen allen $x, x^* \in M$ vorgenommen.

Formal kann eine Zielfunktion auf dieser Basis wie folgt definiert werden:

$$\min f_i(x) \tag{4.3}$$

unter den Nebenbedingungen

$$f_j(x) \leq f_j(x_j^*) \tag{4.4}$$

wobei $i = 1, 2, ...n$ und $j = 1, 2, ..., i-1,$ falls $i > 1$ gilt.

Auf makroskopischer Ebene wird eine lexikographische Zielfunktion verwendet, die zwei Einträge berücksichtigt, wobei der erste Eintrag die Anzahl an DNF, sprich der Ladeeinheiten, die verspätet abgearbeitet wurden, angibt und der zweite Eintrag die minimale Pufferzeit beschreibt. Die Anzahl an DNF soll minimiert werden, die minimale Pufferzeit maximiert, wobei die Minimierung der Anzahl an DNF eine höhere Priorität besitzt. Der Fitnessvektor ergibt sich folglich zu

$$f_{makro} = [f_1, f_2]^T \tag{4.5}$$

mit

$$f_1 = \text{Anzahl DNF}$$
$$f_2 = -\min_i P_i.$$

Eine Minimierung des *Resource-Leveling-Index* wäre eine alternative Optimierungsgröße zur minimalen Pufferzeit, bei hoch ausgelasteten Containerterminals erreicht man durch eine gleichmäßige Verteilung der Auslastung allerdings keine robustere Lösung. In den meisten Zeitfenstern liegt die Auslastung bei 100 %, eine gleichmäßige Verteilung liegt somit vor, allerdings nahe der Kapazitätsgrenze.

Auf mikroskopischer Ebene wird ein Fitnessvektor mit drei Einträgen betrachtet. Der erste Eintrag wird wiederum von den DNF gebildet. An zweiter Stelle steht die Anzahl NS an Operationen, die laut makroskopischer Planung vorgegeben wurden, allerdings nicht eingeplant werden konnten. Der letzte Eintrag beschreibt die späteste Endzeit C_{max} im Sinne einer Minimierung der Projektdauer (engl.: *minimize makespan*). Diese

Minimierung führt indirekt zu einer Minimierung der Rüstzeiten. Für den Fitnessvektor ergibt sich folglich:

$$f_{mikro} = [f_1, f_2, f_3]^T \tag{4.6}$$

mit

$$\begin{aligned} f_1 &= \text{Anzahl DNF} \\ f_2 &= \text{Anzahl NS} \\ f_3 &= C_{max}. \end{aligned}$$

Auf beiden Ebenen dient diese Fitnessbeschreibung zur Bestimmung des Rangs eines Individuums, welcher im Rahmen einer Rang-Selektion Anwendung findet.

4.1.8. Mixed-Integer-Programming-Formulierung

Das Problem der Optimierung des Betriebs auf einem Containerterminal lässt sich als *Mixed-Integer-Programming*-Problem mathematisch formulieren. Die Zielfunktion $f(S)$ entspricht dabei je nach Skala den Ausführungen in 4.1.7. Zusammenfassend lässt sich das Problem wie folgt mathematisch formulieren:

$$\min f(S)$$

unter den Nebenbedingungen

$$\text{(Exlusivitätsbedingung)} \quad \sum_{m_i \in M_i} x_{im_i} = 1 \qquad (i \in V) \tag{4.7}$$

$$\text{(Auswahlbedingung)} \quad x_{im_i} \in \{0,1\} \qquad (i \in V, m_i \in M_i) \tag{4.8}$$

$$\text{(Zeitliche Bedingungen)} \quad S_j - S_i \geq \delta_{ij}(\vec{x}) \qquad (\langle i,j \rangle \in E) \tag{4.9}$$

$$\text{(Frühester Start)} \quad S_i \geq ES_i \qquad (i \in V) \tag{4.10}$$

$$\text{(Kapazitätsbeschränkung)} \quad r_k(S,t,\vec{x}) \leq R_k \qquad (k \in \mathcal{R}, 0 \leq t \leq \bar{d}) \tag{4.11}$$

$$\text{(Rüstzeiten)} \quad S_j - S_i \geq \Delta_{ij}(\vec{x}) \qquad (\langle i,j \rangle \in \mathcal{E}) \tag{4.12}$$

Dabei beschreiben die beiden Gleichungen 4.7 und 4.8 die Nebenbedingung, dass für jeden Job genau eine Route gewählt werden muss. Gleichung 4.9 berücksichtigt die zeitlichen Restriktionen zwischen einzelnen Operationen. Da Zeitfenster für die einzelnen Jobs und somit auch für die Operationen berücksichtigt werden, gibt Gleichung 4.10 die Nebenbedingungen an, dass eine Operation erst nach ihrem Verfügbarkeitszeitpunkt ausgeführt werden darf. Weiterhin beschreibt Gleichung 4.11, dass für den gesamten Projektzeitraum von 0 bis $\bar{d}$ für jeden Zeitpunkt t der Ressourcenbedarf an einer Ressource k, die zur Menge aller Ressourcen $\mathcal{R}$ gehört, nicht höher sein darf als die Kapazität R_k. Gleichung 4.12 beschreibt die Berücksichtigung der Rüstfahrten $\Delta_{ij}(\vec{x}) = S_i + p_i + s_{ij}$, falls O_i auf der gleichen Maschine direkt vor O_j ausgeführt wird.

4.1.9. Erzeugung von Ablaufplänen

Bei der Erzeugung eines Ablaufplans beim Anwendungsbeispiel des Containerterminals müssen neben Transportoperationen auch noch die zugehörigen Lageroperationen sowie etwaige Rüstoperationen eingeplant werden.

Mehrere Autoren haben sich mit sequenzabhängigen Rüstzeiten beschäftigt, beispielsweise Ayache (2005), und Konzepte entwickelt, diese bei der Erzeugung des Ablaufplans mit aufzunehmen. Weiterhin muss für jede Transportoperation eine Lageroperation

am Zielort möglich sein. Algorithmus 4.1 beschreibt das allgemeine Vorgehen des entwickelten parallelen SGS auf makroskopischer Ebene. Auf mikroskopischer Ebene wird zur Auswahl der Operationen anstelle der Prioritätsregeln eine Aktivitätsliste verwendet.

Die Menge E_t enthält alle Operationen, die zum Zeitpunkt t einplanbar sind. Zu Beginn enthält diese die Operationen O_i^R, die keine Vorgängeroperationen im *Activity-On-Node*-Diagramm besitzen, für deren frühsten Start $ES_i \leq t$ gilt. Ist ein Iterationsschritt für ein t durchgelaufen, wird zunächst die Menge A_t aktualisiert, daraufhin ein neues t bestimmt und die Menge E_t aktualisiert. Die Menge A_t enthält alle Operationen O_i, deren gesamte Vorgängeroperationen eingeplant sind, die selbst noch nicht eingeplant sind und für die gilt $ES_i \leq t$. Die Bestimmung des neun Einplanzeitpunktes t erfolgt durch die Bildung des Minimums aller Endzeiten der im aktuellen Iterationsschritt eingeplanten Operationen und der frühsten Startzeiten der Operationen aus A_t. Innerhalb des Iterationsschrittes wird versucht, die eigentliche Transportoperation und die zugehörigen Rüst- und Lageroperationen einzuplanen.

Grundsätzlich folgt die Ablaufplanerzeugung diesem Verfahren. Gerade bei der Einplanung von Kran-Operationen auf mikroskopischer Ebene muss allerdings berücksichtigt werden, dass ein gewisser Einplanhorizont zur Verfügung gestellt werden muss. Ist zum Einplanzeitpunkt t eine Kran-Operation unter den einzuplanenden Operationen, so wird für diese eine Rüstfahrt bestimmt. Die Startzeit der eigentlichen Kran-Transportoperation liegt dementsprechend nach dem aktuellen Zeitpunkt. Bei der Einplanung der Rüstfahrten $s^k_{j,(q+1)}$ auf der Ressource k wird berücksichtigt, dass die Fahrt vom Zielort der letzten Transportoperation O^s_q zum Startort der aktuell betrachteten Transportoperation möglich ist und auf jeden Fall der Kran eine Position hat, an der er sich aufhalten kann. Dies kann dazu führen, dass eine Rüstfahrt ersetzt werden kann durch eine verkürzte Rüstfahrt $s^k_{q,j}$, eine Transportoperation O^s_j und eine Rüstfahrt $s^k_{j,(q+1)}$, die das Ende der vorherigen Rüstfahrt beschreibt und zur bereits eingeplanten Transportoperation O^s_{q+1} führt. Abbildung 4.16 beschreibt das generelle Vorgehen unter der Verallgemeinerung, dass mehrere Ressourcen an einem Transport beteiligt sind. Dies erweitert den SGS um eine serielle Eigenschaft für den Spezialfall der sequenzabhängigen Rüstfahrten.

Algorithmus 4.1 SGS für das Containerterminal-Problem

Eingabe: R :=$\{R_1, ...R_\lambda, ..., R_m\}$ ▷ Prioritätsregel-Liste
Eingabe: $O^R := \{O_1^R, O_2^R, ..., O_i^R, ..., O_n\}$ ▷ Routing-Transport-Operationen
Eingabe: $O_i \mapsto \{(ES_i, LS_i, EE_i, LE_i)\}$ ▷ Ausführungsfenster der Operationen

$t = \min_{i \in V}(ES_i)$
Initialisiere E_t
Initialisiere A_t
while Nicht alle Operationen eingeplant sind **do**
  while $E_t \neq \emptyset$ **do**
    Bestimme aktuelle Prioritätsregel R_t anhand t
    Wähle Routing-Transport-Operation $O_j^R \in E_t$ anhand der Prioritätsregel R_t
    Bestimme am Zielort von O_j^R einen Lagerort zum Zeitpunkt $t + p_j$
    Bestimme am Zielort eine Lageroperation S_i
    if keine Lageroperation S_i möglich **then**
      CONTINUE
    end if
    Erzeuge eine Operation O_j^s, die der aktuell betrachteten Skala genügt.
    for $k \in \mathcal{R}(O_j^s)$ **do**
      Bestimme zum Zeitpunkt t eine Rüstfahrt $s_{q,j}^k$ zur Transportoperation O_j^s.
      if $rjk > R_k(\tau)$ für ein $\tau \in \left[t, t + p_{s_{q,j}^k} + p_{O_j^s}^k + p_{s_{j,(q+1)}^k}\right[$ existiert **then**
        CONTINUE
      end if
    end for
    Verkürze vorangegangener Lageroperation der aktuell transportierten Ladeeinheit
    Plane die Operation O_j^s sowie die Rüstfahrt $s_{q,j}^k$ ein
    Plane die nachfolgende Lageroperation S_i bis zum Ende der Verfügbarkeit des Zielortes ein
  end while
  aktualisiere die Menge A_t
  bestimme neues t
  aktualisiere E_t.
end while
return Schedule S

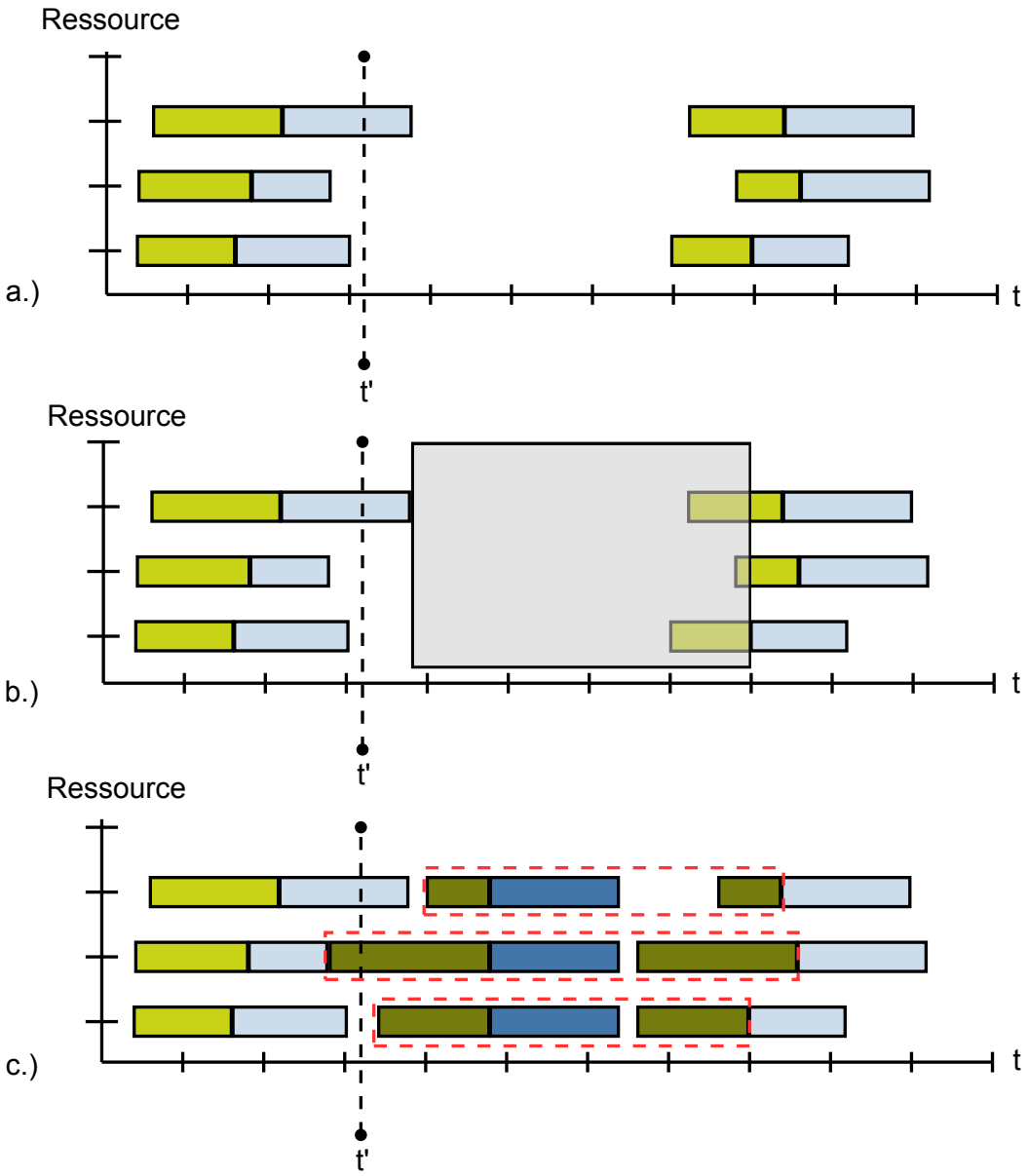


Abbildung 4.16.: Einplanen einer neuen Transportoperation unter der Berücksichtigung von Rüstoperationen. Grundsätzlich besteht die Möglichkeit, mehrere Ressourcen zu berücksichtigen. Transportoperationen sind blau gekennzeichnet, Rüstfahrten gelb. In c.) sind die neu hinzugekommenen Operationen dunkel hervorgehoben.

4.2. Ergebnisse des Anwendungsbeispiels

Zur Evaluierung des entwickelten Multiskalen-Algorithmus wurde ein sich im Bau befindlicher zukünftiger *Hub* im deutschen Eisenbahnnetz betrachtet. Bei dem betrachteten Containerterminal handelt es sich um einen Containerterminal mit einem Modul von 700 m Länge und sieben Gleisen. Der Umschlag wird durch fünf Portalkrane und eine Längsförderanlage mit gummibereiften AGVs bewerkstelligt. Die Portalkrane haben eine Geschwindigkeit von 1.67 $\frac{\mathrm{m}}{\mathrm{s}}$ in x-Richtung und 2.00 $\frac{\mathrm{m}}{\mathrm{s}}$ in y-Richtung. Weiterhin sind Beschleunigungswerte in x-Richtung sowie y-Richtung von 0.25 $\frac{\mathrm{m}}{\mathrm{s}^2}$ gegeben. In z-Richtung wird der Hub der Ladeeinheiten mit einer Greifwerkzeuggeschwindigkeit von 0.75 $\frac{\mathrm{m}}{\mathrm{s}}$ und einer Beschleunigung von ebenfalls 0.25 $\frac{\mathrm{m}}{\mathrm{s}^2}$ durchgeführt. Die Krane werden halbautomatisch gesteuert, lediglich die genaue Positionierung der Krankatze zum Aufnehmen und Abstellen der zu transportierenden Ladeeinheit wird durch einen Kranführer vorgenommen. Der zeitliche Aufwand für diese Positionierung wird sowohl beim Aufnehmen als auch beim Abstellen der Ladeeinheit auf 10 s gesetzt. Abbildung 4.17 zeigt den Grundriss des betrachteten Terminals.

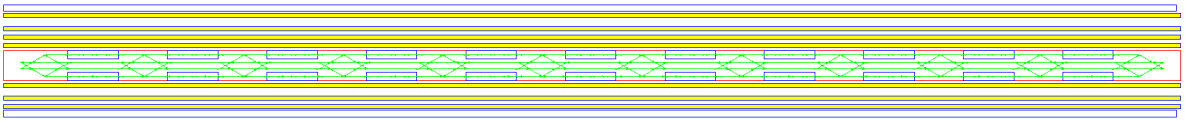


Abbildung 4.17.: Grundriss des betrachteten Terminals. Es gibt im Norden und im Süden (hier oben und unten) jeweils eine Lagerspur (blau). Es folgen insgesamt sechs Gleise (gelb). Im mittleren Bereich ist die Längsförderanlage mit den Übergabebereichen dargestellt. Weiterhin sind die möglichen Fahrwege der Agenten der Längsförderanlage zu erkennen (grün).

4.2.1. Probleminstanzen und deren Komplexität

Die Infrastruktur wurde durch verschiedene Transportprogramme über 8 Stunden mit 15 Zügen á 35 Waggons belastet. Die Transportprogramme sind so gewählt, dass sie für den Terminal überdimensioniert sind. Dabei ist die Kapazität der Krane der begrenzende Faktor. Die Zwischenabstellungen, die nicht Bestandteil der Übergabebereiche zwischen zwei Zügen sind, sind hingegen schwach ausgelastet. Dadurch liegt der reale Stapelfaktor nahe 1.0. Eine genauere Beschreibung der Transportprogramme ist im Anhang zu finden (siehe B). Das Transportprogramm TP1 enthält weniger Ladeeinheiten, die eine große Entfernung zwischen Start- und Zielort vorweisen, als die Transportprogramme TP2 und TP3.

Um eine Vorstellung von der Mächtigkeit des Lösungsraums zu erhalten, muss zunächst berücksichtigt werden, dass für 688 Ladeeinheiten, die nicht Bestandteil der Stammrelation sind, die folglich verladen werden müssen, jeweils vier verschiedene Routen

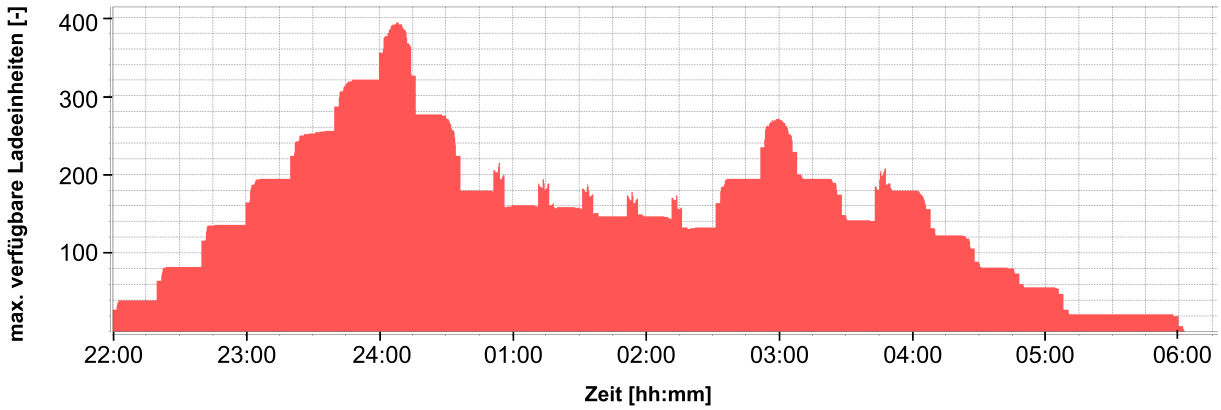


Abbildung 4.18.: Addition der Verfügbarkeitsfenster der einzelnen Operationen des TP 1

bestimmt werden, was alleine zu $688^4 \approx 22 \cdot 10^{11}$ verschiedenen Routenkombinationen führt. Betrachtet man eine einzige Routenkombination, so lässt sich die Möglichkeit der verschiedenen Operationenreihenfolgen durch eine Betrachtung der Verfügbarkeitsfenster der einzelnen Operationen (siehe Abbildung 4.18) ableiten. Die Abbildung sagt aus, dass zum Zeitpunkt 22:15 Uhr 40 Operationen zur Verfügung stehen, die ausgeführt werden können. In der Spitze sind aus kombinatorischer Sicht bis zu 390 Operationen ausführbar. Im Rahmen einer Simulation sinken diese Werte mit dem Fortschreiten der Zeit, da sich die Verfügbarkeitsfenster auf die tatsächliche Ausführungsdauer verkleinern, sobald eine Operation eingeplant wurde. Zur Berechnung der Möglichkeiten lässt sich festhalten, dass in der Spitze 390 verschiedene Operationen eingeplant werden könnten, was zu 390! möglichen Kombinationen führt. Dies entspricht einer Zahl mit 846 Stellen.

Zusätzlich zur diesen Möglichkeiten bestehen noch weitere Kombinationsmöglichkeiten durch die Tatsache, dass die Zwischenabstellungen auch noch einen gewissen Spielraum bieten. Die Routensuche gibt lediglich *transfer-area*s vor, in denen die Ladeeinheiten nach einer Transportoperation abgestellt werden sollen. Die tatsächliche Bestimmung des genauen Abstellplatzes eröffnet somit bei einem 40-Fuß-Container mit einer Länge von 12.19 m und der verfügbaren Länge eines Übergabebereiches zwischen zwei Kranen von 86.8 m selbst bei einer Diskretisierung der Abstellfläche eine weitere Fülle von Möglichkeiten. Durch die Stapelung der Ladeeinheiten erhöhen sich auch diese Möglichkeiten nochmals, auch wenn hierbei berücksichtigt werden muss, dass nicht alle Ladeeinheiten stapelbar sind.

4.2.2. Vergleichslösungen

Um die Güte der Ergebnisse des entwickelten Algorithmus zu bewerten, werden Ergebnisse aus einem Greedy-Verfahren (vgl. Bode u. a. (2010) und Bode u. a. (2012)) zum

Vergleich hinzugenommen. Die Ergebnisse aus diesem Verfahren wurden bereits gegen Realdaten eines Containerterminals des kombinierten Verkehrs kalibriert und validiert. Die bestimmten Kranoperationen entsprechen im Mittel den tatsächlich ausgeführten Operationen. Als Vergleichsgrößen lassen sich neben der Anzahl an nicht rechtzeitig abgefertigten Ladeeinheiten (DNF) ebenso die mittlere Rüstdauer und die mittlere Anzahl an Kranspielen pro Ladeeinheit verwenden. Tabelle 4.1 zeigt die Ergebnisse des Greedy-Verfahrens.

Name	**TP 1**	**TP 2**	**TP 3**
mittlere Rüstdauer	40.5408 s	40.7619 s	37.9682 s
mittlere Kranspiele pro Ladeeinheit	1.174962	1.174846	1.153610
Anzahl DNF	25	36	37

Tabelle 4.1.: Ergebnisgrößen der Transportprogramme bei Anwendung des Greedy-Algorithmus aus Bode u. a. (2012)

4.2.3. Umsetzung

Die Implementierung des Multiskalen-Algorithmus und des Containerterminal-Problems erfolgt in der Programmiersprache Java. Die Auswertung innerhalb des (Ko)evolutionären Algorithmus wird parallelisiert vorgenommen, wobei die Auswertung eines einzelnen Individuums aufgrund des zeitabhängigen parallelen SGS seriell ausgeführt wird. Das Preprocessing, das in diesem Fall aus dem Einlesen der Geometrie des Terminals, den Aufbau der einzelnen Ressourcen, das Einlesen des Transportprogramms sowie weiteren Arbeiten zur Parallelisierung des Codes besteht, wird bei der Rechenzeitberechnung nicht berücksichtigt. Diese startet bei der Bestimmung der Fitness der Startpopulation.

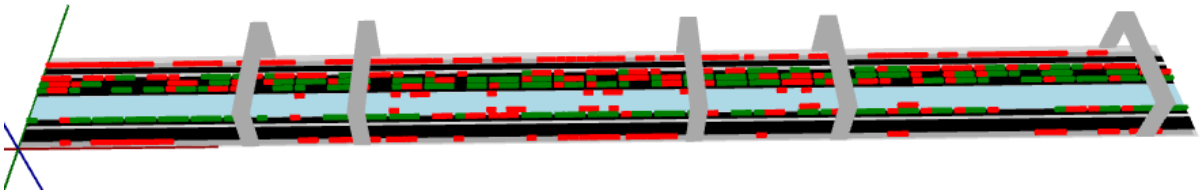


Abbildung 4.19.: 3D-Visualisierung des Betriebs auf dem Containerterminal

Neben der Möglichkeit, diverse Ergebnisplots zu produzieren, bietet das entwickelte Framework weitere visuelle Hilfsmittel, wie beispielsweise eine dreidimensionale Darstellung des simulierten Betriebes auf Basis der bestimmten Lösungen (siehe Abbildung

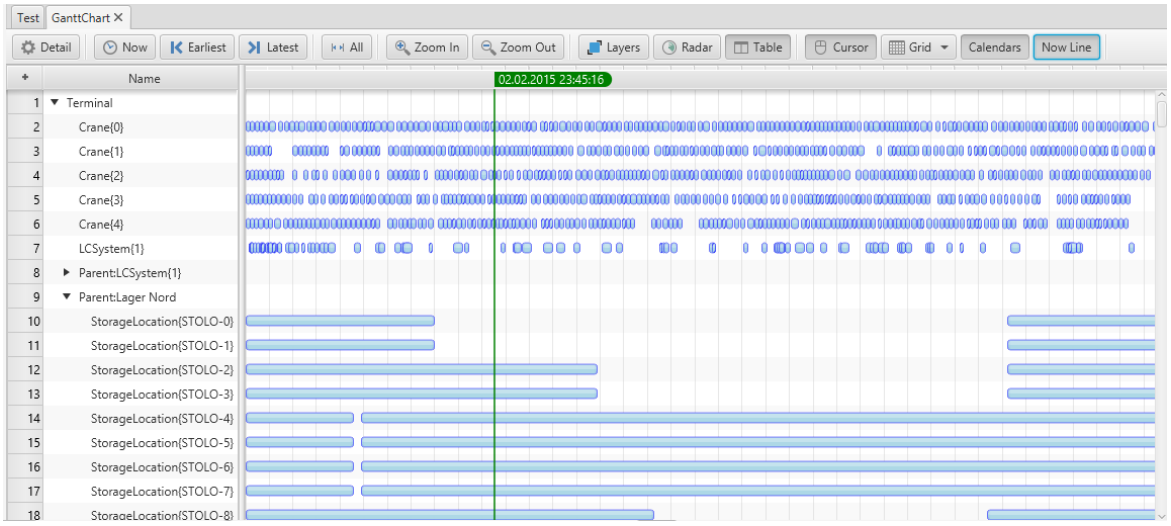


Abbildung 4.20.: Gantt-Chart zur Analyse der Lösungen

4.19) sowie eine Darstellung innerhalb eines interaktiven Gantt-Charts (siehe Abbildung 4.20).

4.2.4. Mikroskopische Optimierung über den gesamten Zeitraum

Um eine Aussage über die Performance des entwickelten Multiskalen-Algorithmus treffen zu können, wurde zunächst eine Optimierung auf mikroskopischer Ebene über den gesamten Zeitraum vorgenommen. Eine Optimierung über einen Optimierungszeitraum von 8 h unter Berücksichtigung von mikroskopischen Kollisionen und Bestimmung von mikroskopischen Abstellplätzen ist sehr aufwändig und benötigt viel Rechenzeit. Im realen Betrieb ist dies aufgrund der ständigen Änderungen an den Nebenbedingungen nicht zu empfehlen. Um allerdings einen Vergleich zu den Lösungen aus Bode u. a. (2012) zu ziehen und eine genauere Vorstellung des Optimums der Probleminstanz zu erlangen, eignet sich eine solche Analyse. Dabei wurde auf die mikroskopische Codierung (siehe Abschnitt 3.4.1) für die Ablaufplanung zurückgegriffen, für die Prozessplanung wurde die prioritätsregelbasierte Codierung (siehe Abschnitt 3.3.2) gewählt.

Die beiden Populationen für die Prozess- und die Ablaufplanung wurden jeweils mit 400 Individuen initialisiert. Für die Population der Prozessplanung wurde eine Mutationsrate von 0.5 gewählt. Bei der Ablaufplanung wurde eine Mutationsrate von 0.4 angesetzt. Die Rekombinationsrate beträgt bei beiden Populationen 1.0. Diese Werte haben sich in einer separaten Parameterstudie als zielführend herausgestellt. Ein genetischer Algorithmus hat zufallsabhängige Aspekte. Ein einmaliges Anwenden des Algorithmus auf eine Probleminstanz bietet daher keine objektiven Ergebnisse. Zu Evaluierungszwecken wurden daher mehrere Durchläufe gestartet, die jeweils über 40 Generationen laufen. Die Ergebnisse dieser Evaluation zeigt Tabelle 4.2. Abbildung 4.21

zeigt den Verlauf der Fitness über die Generationen der ersten fünf Durchläufe. Diese Verläufe sind charakteristisch für Genetische Algorithmen, da zu Beginn in den ersten Generationen deutliche Verbesserungen der Fitness möglich sind und mit zunehmender Generationenanzahl der Algorithmus konvergiert und lediglich kleinere Verbesserungen auftreten.

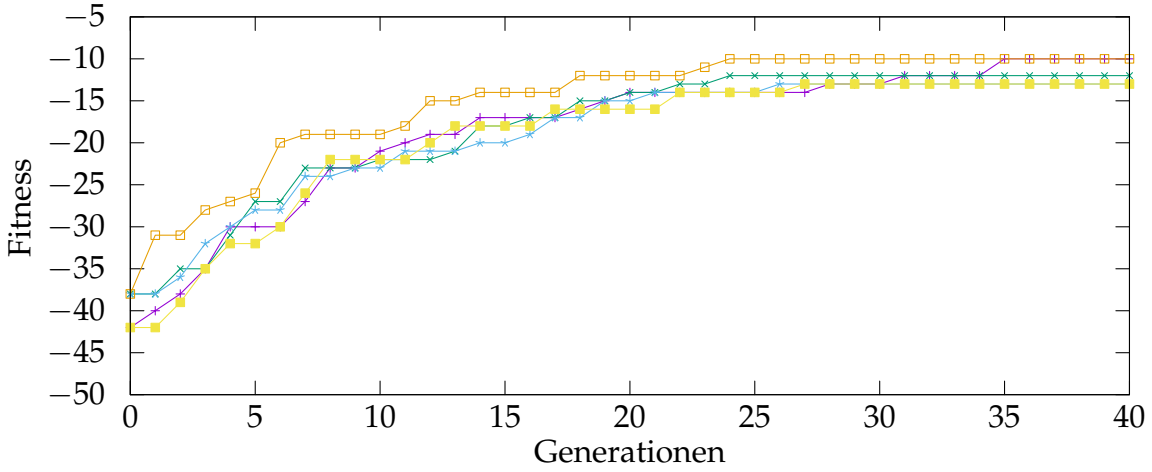


Abbildung 4.21.: Fitnessentwicklung eines holistischen genetischen Algorithmus für das TP 1

Die wichtigste Größe der Zielfunktion ist die Anzahl an nicht abgefertigten Ladeeinheiten, im Weiteren mit DNF bezeichnet. Diese Größe schwankt zwischen 10 und 13. Es deutet sich ein, zumindest lokales, Minimum bei 10 DNF und einer mittleren Rüstdauer von 39.610 s an. Die Rechenzeit, die zur Bestimmung einer Lösung benötigt wird, liegt circa bei 5.5 h. Die Tatsache, dass die Abarbeitung des Gesamtprogramms auf 8 h ausgelegt ist, wobei der letzte Zug aufgrund des Transportprogramms zumeist sogar nach 7 h abgearbeitet ist, führt dazu, dass circa dreiviertel der Zeit, die der tatsächliche Umschlag benötigt, für die Bestimmung einer annehmbaren Lösung mittels einer holistischen Optimierung benötigt wird. Die Güte dieser Lösung ist allerdings im Vergleich zu der des zuvor erwähnten Greedy-Algorithmus deutlich besser. Die Möglichkeiten, auf Änderungen an den Nebenbedingungen zu reagieren, die durch Unsicherheiten in der Ausführung und der Verfügbarkeit der Ladeeinheiten und Ressourcen entstehen, sind allerdings gering.

Exemplarisch lässt sich an der Diversität-Entwicklung der Superpopulation von Durchgang 5 die Konvergenz untersuchen (siehe Abbildung 4.22). Hierbei wird das Diversitätsmaß aus Abschnitt 3.3.6 auf das Superindividuum, bestehend aus Prozess- und Ablaufplanungsindividuum, angewandt. Dadurch, dass zur Generierung eines Superindividuums jeweils das beste Individuum der anderen Population verwendet wird, ist die Kurve nicht streng monoton fallend. Wird ein neues bestes Individuum in der einen Population gefunden, so werden alle nachfolgenden Individuen der anderen Population im darauffolgenden Durchlauf mit diesem kombiniert, was zu einer Auffrischung des gesamten Genpols führt.

Durchlauf	mittlere Rüstdauer	mittlere Kranspiele pro Ladeeinheit	Anzahl DNF	Rechenzeit
1	42.812 s	1.160766	10	05:43:29 h
2	39.423 s	1.158284	12	05:31:58 h
3	39.217 s	1.158284	12	06:01:01 h
4	41.616 s	1.161148	13	05:35:43 h
5	39.610 s	1.157123	10	06:11:13 h
6	41.014 s	1.157037	13	05:44:33 h
7	39.610 s	1.159486	10	06:01:02 h
8	41.844 s	1.158485	10	05:40:23 h
9	39.610 s	1.157123	10	05:35:55 h
10	42.310 s	1.157541	12	05:58:04 h

Intel Core i7-4960X @ 3.60 GHz (4.00 GHz) mit 32 GB RAM

Tabelle 4.2.: Holisitische Optimierung: Ergebnisse des Transportprogramms I

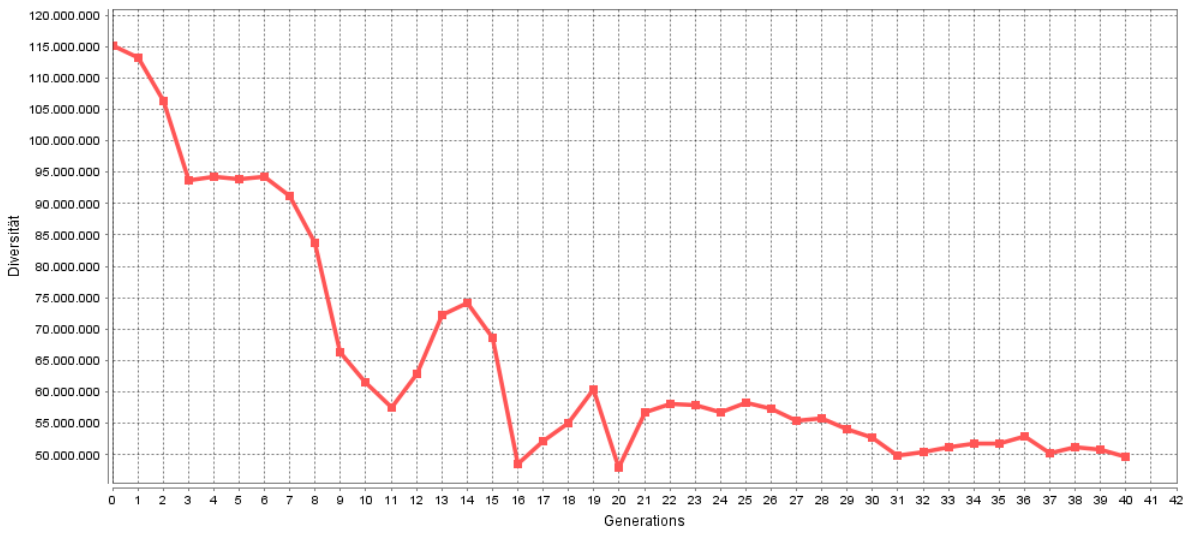


Abbildung 4.22.: Diversitätsentwicklung der Population der Superindividuen

4.2.5. Unscharfe Optimierung auf makroskopischer Ebene

Das vorgestellte Verfahren zur Erzeugung von unscharfen Ablaufplänen unter Berücksichtigung von unscharfen Auslastungsprofilen aus Abschnitt 3.3.9 wurde auf das beschriebene Containerterminal-Problem angewandt.

Dabei wurde zunächst von einer pauschalen Modellierung der Unschärfe ausgegangen (siehe Abbildung 4.23). Diese einfache Umsetzung dient zur Evaluierung der Eignung des Verfahrens. Das Transportprogramm TP1 wurde mit einer geringen pauschalen Unschärfe von $\gamma = 1\,\mathrm{s}$ versehen. Diese Unschärfe betrifft die Operationendauer sowie die Ankunftszeiten der Züge, wodurch die frühsten Startzeiten der Operationen unscharf sind.

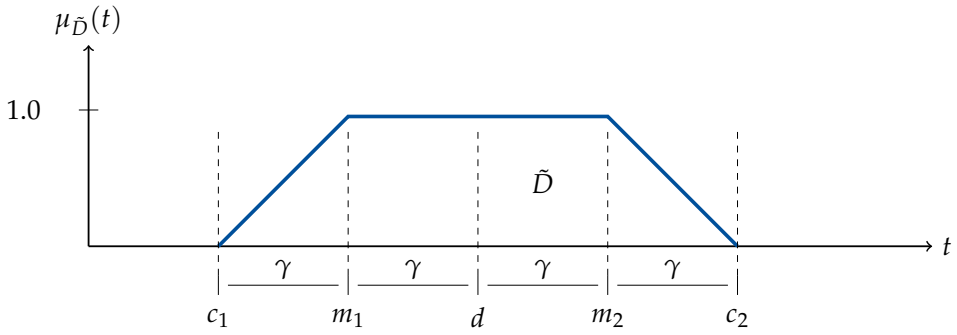


Abbildung 4.23.: Pauschale Modellierung der Unschärfe

Abbildung 4.24 zeigt den unscharfen Ressourcenbedarf für einen der Krane des Containerterminals detailliert pro Operation, Abbildung 4.25 die addierten Auslastungen und somit die unscharfe Gesamtauslastung. Zum Vergleich zeigt Abbildung 4.26 die Auslastung auf makroskopischer Ebene ohne Unschärfe für den selben Ablaufplan.

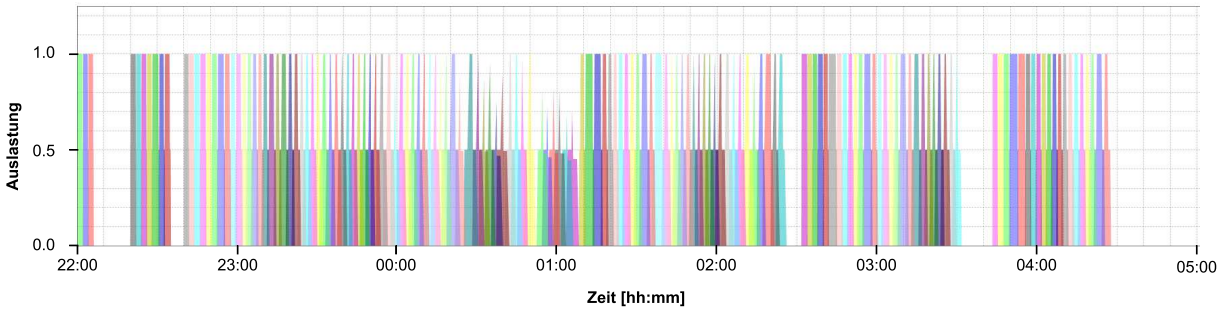


Abbildung 4.24.: Auslastung des Krans 1 bei minimaler Unschärfe. Es wird der unscharfe Ressourcenbedarf jeder einzelnen Operation dargestellt.

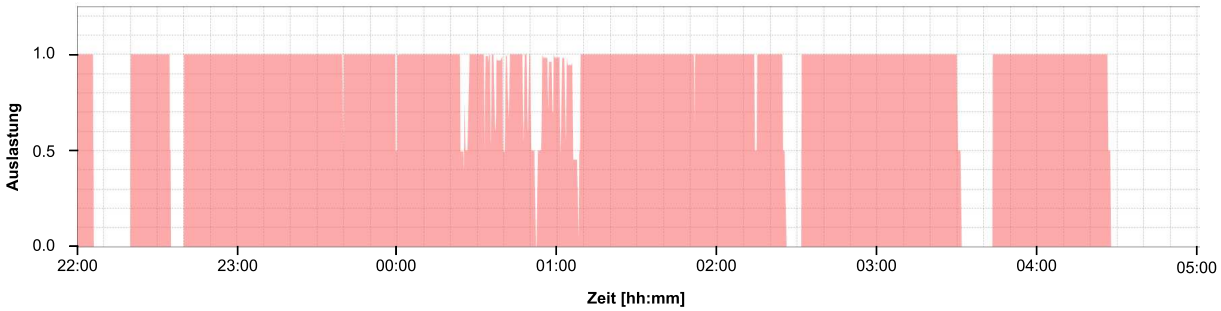


Abbildung 4.25.: Auslastung des Krans 1 bei minimaler Unschärfe. Es wird der aufaddierte unscharfe Ressourcenbedarf dargestellt.

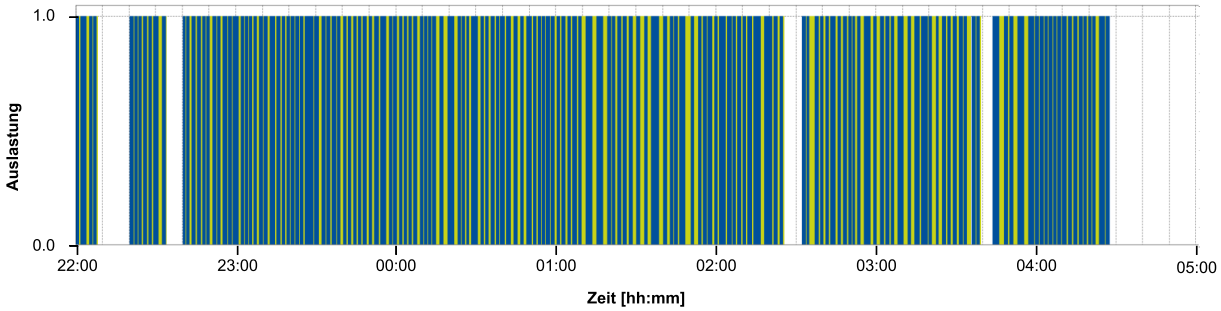


Abbildung 4.26.: Auslastung des Krans 1 ohne Unschärfe. In blau sind Transportoperationen, in grün Rüstfahrten dargestellt.

Es ist zu erkennen, dass sich die Unschärfe im zeitlichen Verlauf zunächst steigert. Gegen 01:10 Uhr und 02:30 Uhr setzt eine Abnahme der Unschärfe ein. Dies ist dadurch zu begründen, dass etwa zu diesen Zeitpunkten neue Operationen zur Verfügung stehen, da neue Züge in den Terminal einfahren. Durch die gewählte Bestimmung des nächsten Einplanzeitpunktes führt dies, weil die Auslastung zu diesen Zeitpunkten etwas unterhalb der Kapazität liegt, dazu, dass die Unschärfe des Einplanzeitpunktes und somit des Startzeitpunktes der entsprechenden Operationen abnimmt.

Es lässt sich festhalten, dass die Auslastung nicht auf einem gleichmäßig hohen Niveau gehalten werden kann, da die entstehenden Verläufe des unscharfen Ressourcenbedarfs der einzelnen Operationen zu komplex sind, um die entstehenden übrigen Kapazitäten effizient auszunutzen (vgl. ebenfalls Abschnitt 3.3.9). Die Modellierung des Auslastungsprofils nach Masmoudi u. Haït (2013) birgt bei den hier betrachteten Aufgabenstellungen augenscheinlich einige Probleme.

Zur Analyse der Lösung wird die Anzahl an Jobs, die nicht rechtzeitig abgefertigt werden konnten, da eine ihrer Operationen einen *agreement index* von 0.0 besitzt, betrachtet (siehe Definition 38). Diese Operationen verletzen somit das Fälligkeitsdatum komplett.

Im hier betrachteten unscharfen Fall ergibt sich eine Anzahl von 21 Jobs, bei denen dieser Fall eintritt. Die scharfe Lösung für den pessimistischen Fall (siehe Abbildung 4.26) weist nur 2 solcher Jobs auf.

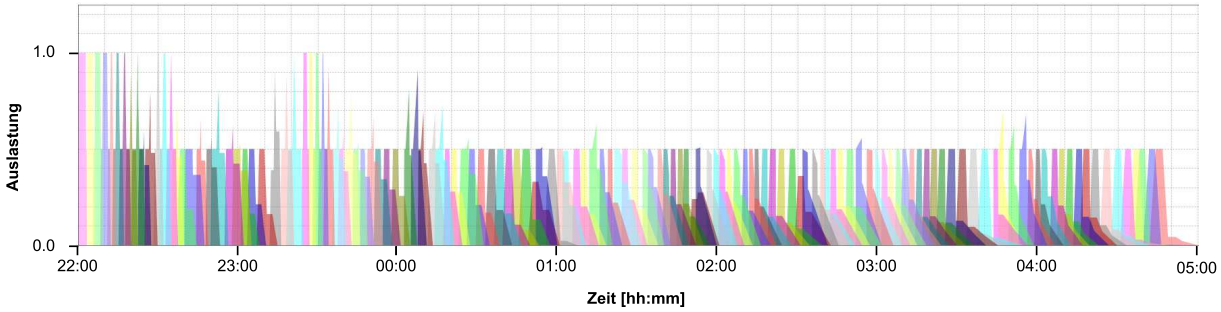


Abbildung 4.27.: Auslastung des Krans 1 bei einer Unschärfe von $\gamma = 30s$

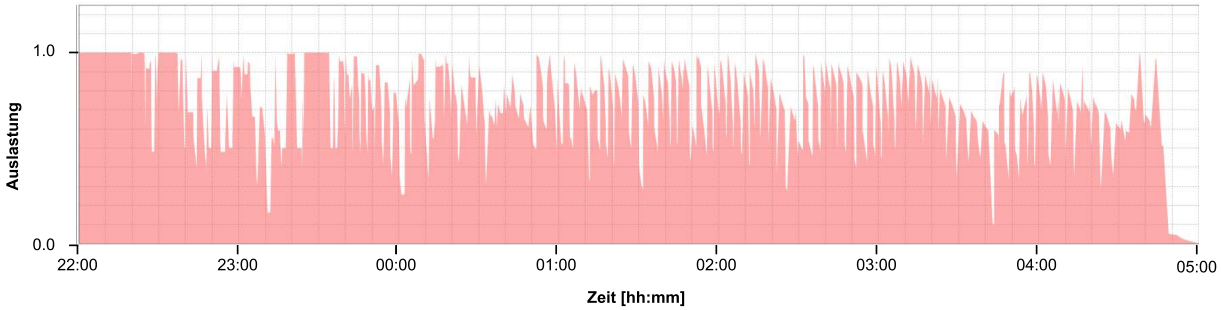


Abbildung 4.28.: Auslastung des Krans 1 bei einer Unschärfe von $\gamma = 30s$

Eine Erhöhung der Unschärfe auf einen Wert von $\gamma = 30s$ zeigt die Problematik noch deutlicher (siehe Abbildungen 4.27 und 4.28). Die entstehenden Peaks in der Gesamtauslastung führen dazu, dass es schwer möglich ist, nachfolgende Operationen so einzuplanen, dass die vorhandene Restkapazität möglichst gut ausgenutzt wird.

Abbildung 4.29 zeigt die entstehende Auslastung einer der Krane aus obigen Beispiel bei der Hinzunahme einer unscharfen Kapazitätsgrenze. Es lässt sich erkennen, dass die zur Verfügung stehenden Kapazitäten besser genutzt werden, dafür steigt der Grad der Verletzung der Kapazitätsgrenze an, sobald aufgrund der gesunkenen Notwendigkeit bei späteren Operationen eine optimale Nutzung der restlichen zur Verfügung stehenden Kapazität nicht mehr möglich ist. Die benötigte Gesamtzeit verringert sich durch die Ausnutzung der zusätzlichen, weniger präferierten Ressourcenkapazität. Dieser Effekt entsteht durch die höhere Gewichtung der Vermeidung von zu spät abgefertigten Ladeeinheiten im Vergleich zur Erfüllung der voll präferierten Kapazitätsbeschränkungen in der betrachteten Zielfunktion.

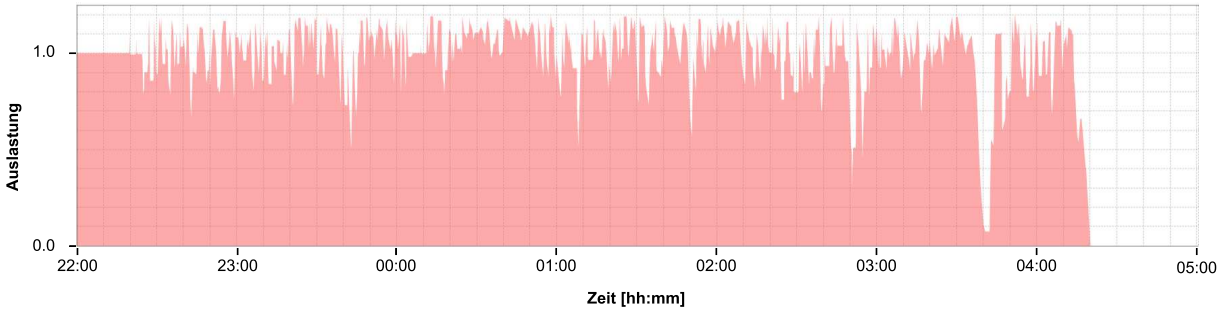


Abbildung 4.29.: Aulastung des Krans 1 bei einer Unschärfe von $\gamma = 30s$ und einer unscharfen Kapazitätsgrenze von $\tilde{r}_k =< 1.0, -\infty, 0.2 >$

Allerdings treten auch bei dieser Modellierung im Verhältnis zum scharfen Fall noch sehr viele Ladeeinheiten auf, deren *agreement index* bei 0.0 liegt. Der Wert liegt bei 18 Jobs im Vergleich zu 2 Jobs im pessimistischen scharfen Fall. Dies lässt darauf schließen, dass die vorgestellte unscharfe Betrachtung für diesen Einsatzzweck unter dem starken Einfluss der Ressourcenbeschränkung nicht zu empfehlen ist.

Betrachtet man die Längsförderanlage, so fällt auf, dass diese weniger kritisch unter diesen Gesichtspunkten ist (siehe Abbildungen 4.30 und 4.31).

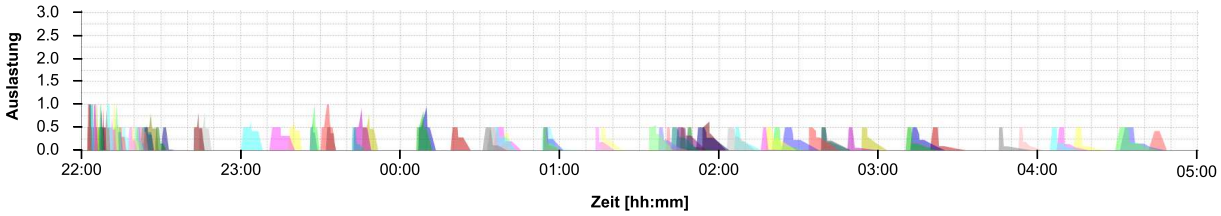


Abbildung 4.30.: Detaillierte Auslastung der Längsförderanlage bei einer Unschärfe von $\gamma = 30s$

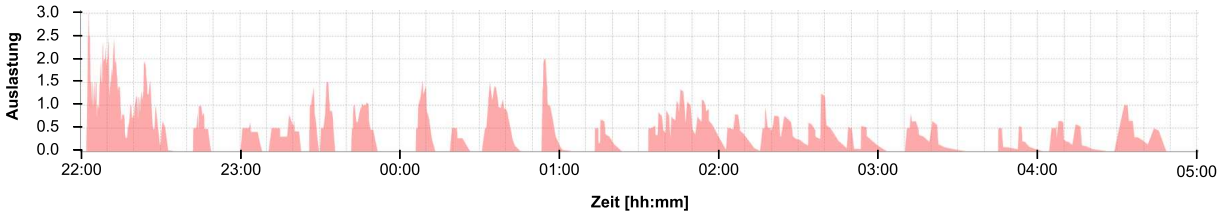


Abbildung 4.31.: Gesamtauslastung der Längsförderanlage bei einer Unschärfe von $\gamma = 30s$

Die Längsförderanlage muss während des gesamten Projektzeitraums verhältnismäßig wenige Operationen bearbeiten. Dadurch wird die volle Ressourcenkapazität zumeist nicht benötigt. Zur Abschätzung der benötigten Anzahl an AGVs auf der Längsförderanlage wäre eine Betrachtung mittels der *Fuzzy-Set*-Theorie sinnvoll einzusetzen, allerdings lässt sich eine unscharfe Betrachtung der Längsförderanlage schwer in ein System integrieren, in dem die übrigen Größen scharf modelliert sind. Die Unschärfe in den Startzeiten der Aufträge der AGVs rührt von den Unschärfen der vorhergegangen Kranoperationen her. Werden diese nicht modelliert, würde nur die Unschärfe in der Dauer der AGV-Transporte berücksichtigt.

4.2.6. Multiskalen-Algorithmus

Im weiteren Verlauf wurde aus zuvor genannten Gründen auf eine unscharfe Modellierung verzichtet. Das Problem wurde mittels des entwickelten Multiskalen-Algorithmus gelöst, der durch die Einführung einer makroskopischen Ebene als reaktive Methode mit zusätzlicher Kontrollinstanz aufgefasst werden kann.

Unter Verwendung der in Abschnitt 4.1.7 definierten Zielfunktionen für die einzelnen Skalen wurden die drei betrachteten Transportprogramme untersucht. Für jedes Transportprogramm wurden mehrere Durchläufe gestartet. Dabei wurden 200 Individuen in 10 Generationen auf mikroskopischer Ebene und 10 Generationen mit jeweils 100 Individuen pro Population auf makroskopischer Ebene verwendet. Tabelle 4.3 zeigt exemplarisch die Varianz in den Ergebnissen für das Transportprogramm TP1.

mittlere Rüstdauer	mittlere Kranspiele pro Ladeeinheit	Anzahl DNF	Rechenzeit
35.734 s	1.198830	4	03:00:12 h
36.420 s	1.198830	4	02:58:58 h
36.900 s	1.191520	4	03:08:50 h
36.628 s	1.192982	4	02:40:31 h
36.633 s	1.194444	4	02:47:00 h
38.104 s	1.195906	4	02:51:20 h
37.785 s	1.190336	5	03:03:24 h
37.167 s	1.191800	5	03:01:31 h

Intel Core i7-4960X @ 3.60 GHz (4.00 GHz) mit 32 GB RAM

Tabelle 4.3.: Mulstikalen-Algorithmus: Ergebnisse des Transportprogramms I (Maximierung der minimalen Pufferzeit)

Zu Vergleichszwecken wurden mehrere Durchgänge mit einer angepassten Zielfunktion auf makroskopischer Ebene betrachtet. Hierzu wird anstelle der Maximierung der minimalen Pufferzeit die Minimierung des *Resource-Leveling*-Index berücksichtigt. Tabelle 4.4 zeigt die Ergebnisse.

Tabelle 4.5 zeigt die besten Lösungen aller Transportprogramme, die in mehreren Durchläufen bestimmt wurden. Eine Parameterstudie bezüglich der genetischen Parameter, beispielsweise der Einfluss der Mutation und der Populationsgröße ist unter B.2 zu finden.

mittlere Rüstdauer	mittlere Kranspiele pro Ladeeinheit	Anzahl DNF	Rechenzeit
37.872 s	1.192250	7	03:04:17 h
36.226 s	1.201767	9	02:44:36 h
36.692 s	1.195266	12	02:42:21 h
36.468 s	1.191679	15	02:53:47 h
35.682 s	1.196136	15	02:50:12 h

Intel Core i7-4960X @ 3.60 GHz (4.00 GHz) mit 32 GB RAM

Tabelle 4.4.: Mulstikalen-Algorithmus: Ergebnisse des Transportprogramms I (Minimierung des *Resource-Leveling*-Indexes)

	mittlere Rüstdauer	mittlere Kranspiele pro Ladeeinheit	Anzahl DNF	Rechenzeit
TP1	35.734 s	1.198830	4	03:00:12 h
TP2	36.840 s	1.204819	20	04:20:54 h
TP3	34.840 s	1.173848	12	04:35:47 h

Intel Core i7-4960X @ 3.60 GHz (4.00 GHz) mit 32 GB RAM

Tabelle 4.5.: Multiskalen-Algorithmus: Beste Ergebnisse der drei Transportprogramme

Optimierungsgrößen

Die Anzahl der Ladeeinheiten, die nicht rechtzeitig abgefertigt werden konnten, konnte im Vergleich zur holisitischen Optimierung signifikant verringert werden. Durch eine Erhöhung der Populationsgröße im Rahmen der holistischen Optimierung wäre es unter Umständen möglich, ebenso gute Ergebnisse zu gewinnen. Dies würde allerdings eine wesentlich längere Berechnungszeit mit sich ziehen, die über den 8 Stunden des tatsächlichen Betriebs liegt. Die bessere Güte der Ergebnisse des Multiskalen-Algorithmus lässt sich unter anderem durch einen kleineren Lösungsraum im Vergleich zur holistischen Optimierung begründen.

Allerdings besteht die Möglichkeit, dass das tatsächliche Optimum durch die Verkleinerung des Lösungsraums nicht mehr erreicht werden kann. Diese Verkleinerung des Lösungsraums ist dadurch gegeben, dass auf makroskopischer Ebene eine Optimierung

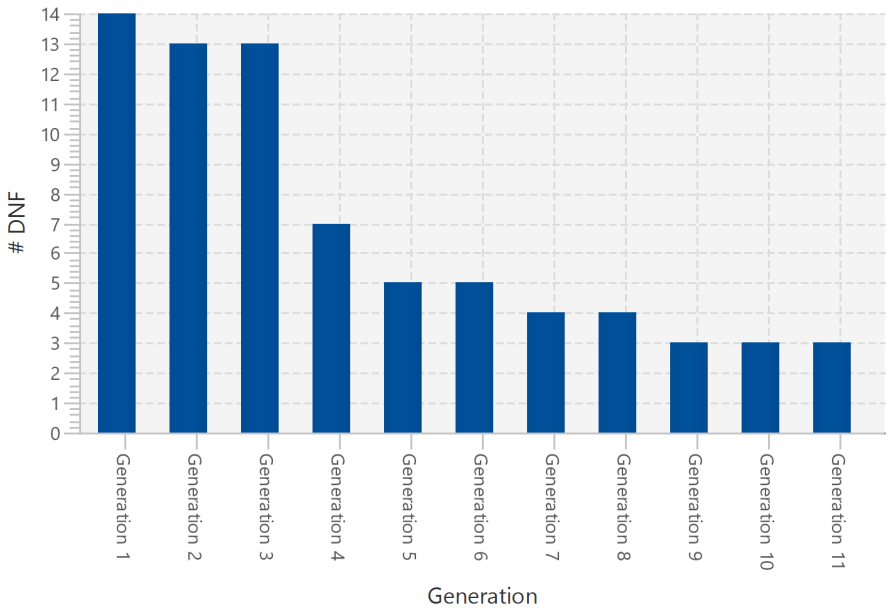
vorgenommen wird und eine Teilmenge an Operationen bestimmt wird, die zu einem Zeitschritt auf mikroskopischer Ebene optimal eingeplant werden soll. Da diese Teilmengenbestimmung auf makroskopischer Ebene durchgeführt wird, besteht die Möglichkeit, dass auf mikroskopischer Ebene bessere Ergebnisse bestimmt werden könnten, wenn Operationen aus nachgelagerten Zeitfenstern des Multiskalen-Algorithmus früher ausgeführt werden würden. Weiterhin lässt sich festhalten, dass die Routenwahl lediglich auf makroskopischer Ebene vorgenommen wird. Somit besteht die Möglichkeit, dass eine andere Routenwahl aus mikroskopischer Sicht zu einer besseren Lösung führen könnte.

Auffällig ist, dass die mittlere Rüstdauer im Vergleich zur holistischen Optimierung verringert werden kann, die Anzahl an mittleren Kranspielen pro Ladeeinheit hingegen steigt. Die Operationenreihenfolge wird dahingehend optimiert, dass die Rüstfahrten zwischen den Operationen geringer ausfallen. Dies führt zu einer Erhöhung der Anzahl an Kranspielen. Weiterhin werden die einzelnen Jobs in mehreren Teiloperationen ausgeführt, als bei den mittels der holistischen Optimierung bestimmten Lösungen. Da sich bei vielen Jobs der Startzug und der Zielzug nur eine geringe Zeitspanne gleichzeitig im Terminal befinden, führt eine Zwischenabstellung der Ladeeinheiten zu besseren Ergebnissen. Bei Direkttransporten der betroffenen Jobs würde das resultierende kleine Ausführungszeitfenster zu einer Verschlechterung der Ergebnisse führen.

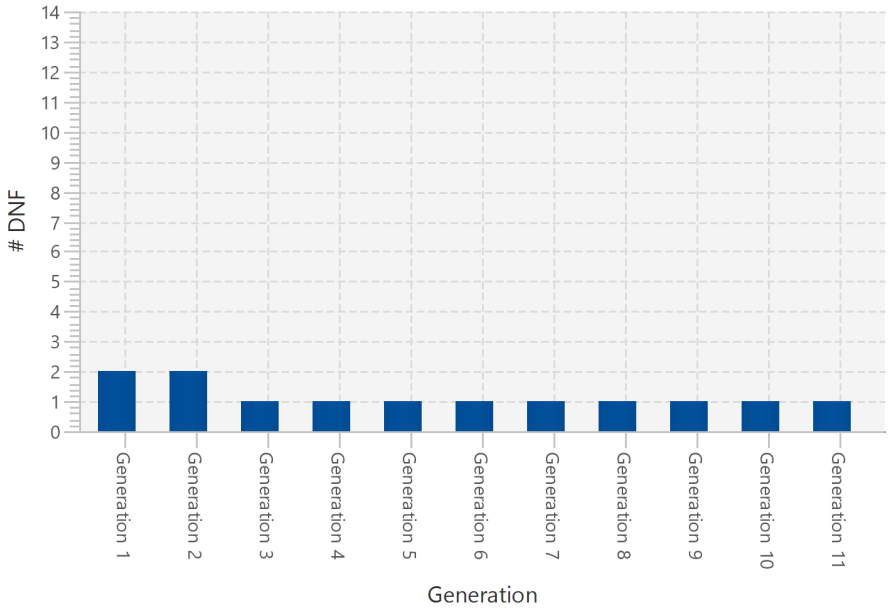
Die Verwendung der minimalen Pufferzeit als Zielgröße führte zu deutlich besseren Ergebnissen als die Minimierung des *Resource-Leveling*-Indexes. Die Güte der Ergebnisse liegt bei der Maximierung der minimalen Pufferzeit ebenfalls deutlich dichter beieinander und streuen weniger.

Abbildung 4.32 zeigt die Entwicklung der Fitness für den ersten Eintrag des Fitnessvektors für die erste und die zweite Iteration auf makroskopischer Ebene bei der Optimierung des Transportprogramms TP1. Zu Beginn, in der ersten Iteration, sinkt die Anzahl an DNF stetig bis auf einen Wert von 5 Ladeeinheiten, die nicht rechtzeitig abgearbeitet werden. Daraufhin folgt die erste Iteration auf mikroskopischer Ebene, bei der die Operationen, die in den ersten 30 Minuten auf makroskopischer Ebene eingeplant wurden, auf mikroskopischer Ebene eingeplant werden. Im Iterationsschritt 2 auf makroskopischer Ebene werden die bereits auf mikroskopischer Ebene eingeplanten Operationen ebenfalls eingeplant und die Lösungen aus der ersten Iteration werden verwendet um auf Basis der neuen Nebenbedingungen eine weitere Optimierung vorzunehmen. Auffällig ist, dass die Anzahl an DNF-Ladeeinheiten schon in der ersten Generation der zweiten Iteration geringer ist, als in der letzten Generation der ersten Iteration. Dies ist zum einen dadurch zu begründen, dass die mikroskopischen Operationen im ersten Zeitfenster in diesem Fall günstigere Dauern haben, als auf makroskopischer Ebene angenommen. Entscheidender ist allerdings der Punkt, dass die Population aus der ersten Iteration als Initiallösung verwendet wird. Dadurch wird häufig von einer guten Lösung ausgehend gestartet. Dies muss allerdings nicht zwangsläufig der Fall sein. Sind große Änderungen an den Nebenbedingungen aufgetreten, so

werden mehrere Generationen benötigt, um ausgehend von der Population der letzten Iteration eine gute Lösung zu finden.



(a)1.Iteration

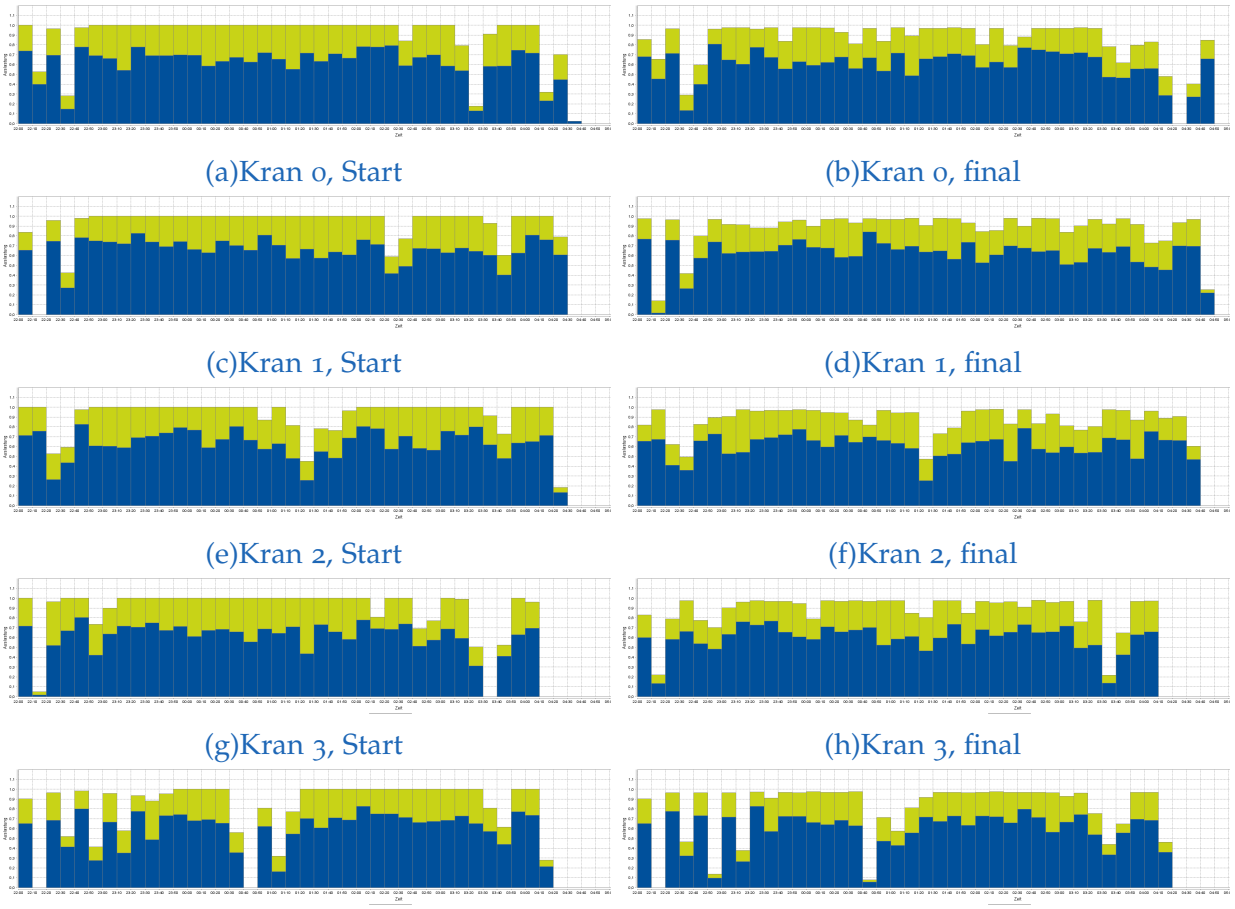


(b)2.Iteration

Abbildung 4.32.: Multiskalen-Algorithmus: Fitnessentwicklung DNF für das TP1

Vergleich der Kranauslastung

Abbildung 4.33 zeigt die Auslastungsprofile der einzelnen Krane zu Beginn des Multiskalen-Algorithmus auf makroskopischer Ebene auf der linken Seite und die finalen Auslastungsprofile nach Beendigung des Verfahrens auf der rechten Seite. Grundlegende Charakteristiken lassen sich wiedererkennen.



(a)Kran 0, Start (b)Kran 0, final

(c)Kran 1, Start (d)Kran 1, final

(e)Kran 2, Start (f)Kran 2, final

(g)Kran 3, Start (h)Kran 3, final

(i)Kran 4, Start (j)Kran 4, final

Abbildung 4.33.: Vergleich der Aulastungsprofile der Krane. Links werden die makroskopischen Auslastungsprofile zu Beginn des Algorithmus dargestellt, rechts die mikroskopischen Auslastungsprofile nach Beendigung des Verfahrens. In blau ist der Anteil der Transportfahrten, in grün der der Rüstfahrten aufgetragen.

Gerade Bereiche, in denen der Terminal wenig ausgelastet ist, sind bereits bei der makroskopischen Voroptimierung zu erkennen, beispielsweise zu Beginn zwischen 22:10 Uhr und 22:40 Uhr sowie gegen Ende, gerade beim Kran 0 gegen 04:10 Uhr oder beim Kran 4 zwischen 00:40 Uhr und 00:50 Uhr. Auffällig ist die Tatsache, dass die makroskopische Auslastung über weite Teile bei 100% liegt, bei der finalen Umsetzung später diese allerdings in den betroffenen Bereichen zumeist knapp unter 100% liegt.

Dies lässt sich dadurch begründen, dass in der detaillierten Betrachtung der Kranbewegungen, unabhängig vom Faktor Mensch und anderen äußeren Einflüssen, häufig keine 100%-ige Auslastung erreicht werden kann, da auf die Ausführungen eines anderen Krans gewartet werden muss. So ist es bei Transporten, aus und in Übergabebereichen zwischen Kranen, häufig sinnvoller zu warten, bis der benachbarte Kran den entsprechenden Bereich freigegeben hat, als eine andere Kranoperation auszuführen. Diese Wartezeiten zählen nicht zu den hier berücksichtigten Rüstfahrten.

Die Tabellen 4.6 4.7 stellen diverse Kenngrößen der einzelnen Krane des TP I auf makroskopischer Ebene zu Beginn des Multiskalen-Algorithmus den Kenngrößen nach Beendigung des gesamten Algorithmus gegenüber.

Es zeigt sich, dass die Anzahl an Operationen, die benötigt werden, zu Beginn des Algorithmus recht gut abgeschätzt wird. Zu Beginn wird von 815 Kranoperationen ausgegangen, nach dem Durchlaufen des gesamten Algorithmus werden tatsächlich 817 Kranoperationen ausgeführt. Weiterhin lässt sich beobachten, dass die mittlere Rüstdauer in Summe verringert werden kann. Dies ist wesentlich auf die Minimierung der Endzeit der einzelnen Intervalle im Rahmen der mikroskopischen Optimierung zurückzuführen.

Kran		Anzahl an Operationen	mittlere Rüstdauer	Standard-abweichung Rüstdauern
Kran 0	Start	169	39 s	19.14 s
	Ende	173	37 s	19.68 s
Kran 1	Start	163	42 s	21.17 s
	Ende	161	37 s	20.35 s
Kran 2	Start	161	36 s	21.19 s
	Ende	161	35 s	19.73 s
Kran 3	Start	157	41 s	19.79 s
	Ende	159	36 s	22.52 s
Kran 4	Start	165	34 s	15.58 s
	Ende	163	37 s	19.82 s

Tabelle 4.6.: Multiskalen-Algorithmus: Vergleich der Kenngrößen der Krane

	mittlere Rüstdauer	Kranspiele pro Ladeeinheit
Start	38.795 s	1.1848
Ende	36.872 s	1.1932

Tabelle 4.7.: Multiskalen-Algorithmus: Vergleich der akkumulierten Kenngrößen der Krane

Zeitliche Verschiebung von Operationen

Von der makroskopischen Vorplanung ausgehend, müssen mikroskopisch zum Teil aufwändigere Umsortierungen in der Einplanreihenfolge vorgenommen werden. Die Betrachtung des Lagers auf makroskopischer Ebene als homogene Einheit führt dazu, dass Operationen, die auf makroskopischer Ebene zu einem bestimmten Zeitpunkt eingeplant wurden, mikroskopisch erst später eingeplant werden.

Abbildung 4.34.: Unterschiede zwischen makroskopischer und mikroskopischer Restkapazität. Aus makroskopischer Sicht ist die Aufnahme eines 40-Fuß-Containers möglich, mikroskopisch kann allerdings kein zusammenhängender Platz gefunden werden.

Abbildung 4.34 verdeutlicht die Problematik anhand eines einfachen Beispiels. Die Zwischenabstellung besteht aus 16 Plätzen à 3.10 m. Grundsätzlich existiert somit Platz für vier 40-Fuß-Container, wenn die Ladeeinheiten nicht gestapelt werden und somit ein Stapelfaktor von 1.0 angenommen wird. Da ebenfalls 20-Fuß-Container berücksichtigt werden, können Lücken entstehen, die dazu führen, dass bei einer makroskopischen Betrachtung für einen weiteren 40-Fuß-Container die Kapazität nicht überschritten wird, da vier Plätze zur Verfügung stehen. Bei der mikroskopischen Betrachtung allerdings wird keine Lageroperation gefunden, da keine zusammenhängende Fläche von vier Plätzen zur Verfügung steht.

Bei der Modellierung der Abstellflächen wurde eine maximale Stapelhöhe von zwei Ladeeinheiten gewählt. Für die makroskopische Lagerbetrachtung ergibt sich die Lagerkapazität über einen pauschalen Stapelfaktor von 1.7, der sich in der Praxis als treffend erwies.

Betrachtet man einen Auszug aus den Ergebnissen, lässt sich dies verdeutlichen. Auf makroskopischer Ebene wurde für eine Transportoperation, die bereits die vorangegangen Rüstfahrt enthält, eine Start und eine Endzeit bestimmt (siehe Tabelle 4.8).

Auf makroskopischer Ebene wurden die bei der Routensuche (siehe 4.1.5) bestimmten *transfer areas* als Zielort einer Transportoperation verwendet. Diese umfasst bei

Start Makro	Start Mikro	Laddeinheit	Zielort	Ressource
22:07:26	23:58:41	ISO138	SubStorage{256 ... 283 }	Crane{0}

Tabelle 4.8.: Zeitlich Verschiebung von Operationen. Auszug aus einem auf makroskopischer Ebene optimierten Ablaufplan und der Umsetzung auf mikroskopischer Ebene im Rahmen des Multiskalen-Algorithmus

der Ladeeinheit ISO138 eine Zwischenabstellungsfläche, die 28 freie Plätze à 3.10 m umfasst und als Austauschbereich zwischen Crane{0} und Crane{1} dient. Bei einem Stapelfaktor von 1.75 ergibt sich eine freie Kapazität von 49 freien Plätzen à 3.10 m. Dieser Transferbereich überschneidet sich mit den beiden *transfer area*s, die der Crane{0} bzw. der Crane{1} benötigen, um Ladeeinheiten zwischenzulagern, die auf einen später eintreffenden Zug verladen werden müssen.

Der Startzug der Ladeeinheit ISO138 ist zwischen 22:00:00 Uhr und 00:16:40 Uhr zur Ent- und Beladung im Bahnhof. Der Zielzug der Ladeeinheit kann zwischen 02:51:40 Uhr und 05:08:20 Uhr be- und entladen werden. Zunächst besteht die Möglichkeit, andere Ladeeinheiten in dem betroffenen Bereich der Zwischenabstellung umzusortieren. Da in dem hier vorliegenden Fall allerdings ein recht großes Zeitfenster für den Umschlag in das betreffende Gebiet zur Verfügung steht, wird die Ladeeinheit erst um 23:58:41 Uhr umgeschlagen. Zuvor sind um 23:57:11 Uhr vier Plätze durch den Transport der Ladeeinheit ISO920, die die Plätze 272, 273, 274 und 275 belegte, frei geworden.

5. Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde auf Basis der Idee von Stefansson u. a. (2006) ein Multiskalen-Algorithmus entwickelt. Dabei wurde ein Schwerpunkt auf stark ressourcen- und zeitbeschränkte Probleme gesetzt, anstatt auf langfristige taktische Planungen. Im Gegensatz zu alleinstehenden *Rolling-Horizon*-Algorithmen wird zusätzlich mindestens eine weitere Detaillierungsebene betrachtet, die als Kontrollinstanz dient. Begleitend zur physischen Ausführung des Projektes wird auf detaillierter mikroskopischer Ebene eine Optimierung unter exakt modellierten Nebenbedingungen vorgenommen. Die mikroskopischen Ergebnisse werden auf die makroskopische Ebene zurückgeführt. Hier wird eine längerfristige Planung des Projekts vorgenommen und ausgehend von zuvor bestimmten Lösungen im Rahmen einer reaktiven Ablaufplanung die aus der mikroskopischen Optimierung veränderten Nebenbedingungen berücksichtigt. Auf makroskopischer Ebene werden überschlägige Aussagen über die Machbarkeit, die Auslastung und den Ressourcenbedarf getroffen, die als Initiallösung für die weitere mikroskopische Optimierung dienen.

Weiterhin wurde durch den Entwurf eines Koevolutionären Algorithmus ein Mittel geschaffen, das die Prozess- und Ablaufplanung als Einheit betrachtet und somit im Rahmen des übergeordneten Multiskalen-Algorithmus eine größere Flexibilität zur Lösung von Problemstellungen dieser Art darstellt. Dabei wurde der Gedanke der *Least-Commitment*-Strategien aufgegriffen, die Ablaufpläne in einer möglichst flexiblen Form repräsentieren. Diese Repräsentation erlaubt es, konkrete Entscheidungen erst zum Ausführungszeitpunkt zu treffen. Dieses Verfahren aus der Ablaufplanung wurde ebenso auf die Prozessplanung übertragen.

Masmoudi u. Haït (2013) haben eine Möglichkeit geschaffen, Ablaufplanprobleme mit unscharfen Zeitinformationen mit Mitteln der Fuzzy-Set-Theorie zu behandeln. Dabei haben sie anders als zuvor in der Literatur die zeitlich unscharfen Informationen in eine unscharfe Auslastung überführt. Die Eignung dieses Verfahrens im Rahmen des Multiskalen-Algorithmus wurde diskutiert. Es stellte sich heraus, dass dieses für stark ressourcenbeschränkte Probleme weniger geeignet ist.

Anhand des komplexen Beispiels der Optimierung des Betriebs auf einem Containerterminal wurde der entwickelte Algorithmus evaluiert. Die Ergebnisse zeigen im Vergleich zu einem an Realdaten validierten und kalibrierten Greedy-Verfahren (siehe Bode u. a. (2012)) signifikante Verbesserungen. Leistungsstarke Hardware vorausgesetzt, kann ein

projektbegleitender Einsatz des entwickelten Algorithmus als *decision support system* gewährleistet werden.

Das betrachtete Problem muss für einen sinnvollen Einsatz des Multiskalen-Algorithmus so beschaffen sein, dass eine Modellierung auf makroskopischer Ebene möglich ist. Die Überprüfung der Einhaltung der Nebenbedingungen auf makroskopischer Ebene muss eine Rechenzeitersparnis im Vergleich zur detaillierten Modellierung aufweisen, die den Einsatz einer weiteren Ebene rechtfertigt. Die Vereinfachungen innerhalb der Modellierung dürfen weiterhin nicht dazu führen, dass wichtige Aspekte bei der makroskopischen Optimierung nicht berücksichtigt werden. Dies gilt beispielsweise im Bezug auf das Anwendungsbeispiel für die Rüstzeiten zwischen Operationen, die adäquat abgebildet werden müssen.

Die unscharfe Auslastungsformulierung birgt viel Potential für weitere Forschungsansätze. Die Modellierung nach Masmoudi u. Haït (2013) bietet eine mathematisch fundierte Möglichkeit, Unsicherheiten abzubilden. Obwohl der gesamte Verlauf der als Fuzzy-Zahl oder Fuzzy-Interval modellierten Unsicherheit berücksichtigt wird, ist der Einfluss der vom Entscheidungsträger zu wählenden Parameter sehr hoch. Für stark ressourcenbeschränkte Probleme führt die Tatsache, dass die unscharfen Ressourcenprofile nur bedingt anpassbar sind, dazu, dass die vorhandene Kapazität nicht optimal genutzt werden kann. In weiteren Arbeiten sollte über alternative Ansätze diskutiert werden, unsichere Auslastungsprofile zu berücksichtigen. Der oben erwähnten Tatsache, dass die Wahl der Parameter, in diesem Fall des Pessimus-Levels, einen solch starken Einfluss auf die Ergebnisse hat, sollte entgegengewirkt werden. Dies würde durch eine gesamtheitliche Berücksichtigung der Unschärfe in der Start- und Endzeit zu jedem Pessimus-Level erfolgen.

Im Rahmen der mikroskopischen Optimierung ist eine Integration einer lokalen Suche möglich um einen hybriden, memetischen Algorithmus zu erhalten. Nach der Anwendung der genetischen Operatoren wird eine lokale Suche durchgeführt, die innerhalb der Nachbarschaft die beste Lösung sucht. Die Nachbarschaft ist dabei gegeben durch die Reihenfolge der Operationen im erweiterten *Activity-On-Node*-Diagramm, das um zusätzliche Kanten erweitert wurde (siehe Definition 8). Diese zusätzlichen Kanten verbinden Operationen miteinander, die auf einer Ressource nacheinander ausgeführt wurden. In diesem erweiterten Graphen wird der kritische Pfad gesucht. Dieser lässt sich wiederum unterteilen in kritische Blöcke. Ein Block beschreibt dabei eine Teilfolge von Operationen des kritischen Pfads, die auf der selben Ressource ausgeführt werden. Innerhalb eines Blockes lassen sich nun Operationen tauschen und die Auswirkungen dieser Tauschvorgänge auf die Fitness untersuchen. Abbildung **??** zeigt das allgemeine Vorgehen.

Appendix

Anhang A.

Studie: Fuzzy-Auslastung

Zum Testen der unscharfen Ablaufplanung wurden die Probleminstanzen der *PSBLIB-Library* verwendet. Diese wurde von Kolisch u. a. (1999) entwickelt und dienen zur Evaluierung von Lösungsverfahren für *Single-* und *Multi-Mode*-Ablaufplanproblemen. Die Probleminstanzen umfassen die technologischen Vorschriften zwischen den Operationen und ihren Ressourcenbedarf (siehe A.2). Um eine Analyse des oben vorgestellten Verfahrens zur Berücksichtigung von Unschärfe zu ermöglichen wird die Operationsdauern d pauschal mit Unschärfe belegt. Die Operationsdauer wird in ein Fuzzy-Intervall überführt, wobei ausgehend vom scharfen Wert d die Bezugspunkte des Intervalls c_1, m_1, m_2 und c_2 im Abstand von jeweils γ hinzugefügt werden (siehe Abbildung 4.23).

A.1. Vergleich zweier Fuzzy-Zahlen

In vielen Problemstellungen ist es nötig, eine Ordnung über der Menge an Fuzzy-Zahlen zu definieren. Beim Lösen von Ablaufproblemen ist dies vor allem notwendig bei der Erzeugung von Prioritätslisten oder die Sicherstellung von Vorgängerbeziehungen. Überlappen sich zwei Fuzzy-Zahlen nicht, ist die Fragestellung trivial. Überlappen sich zwei Fuzzy-Zahlen, so lassen sich grundsätzlich zwei Arten von Überlappung unterscheiden.

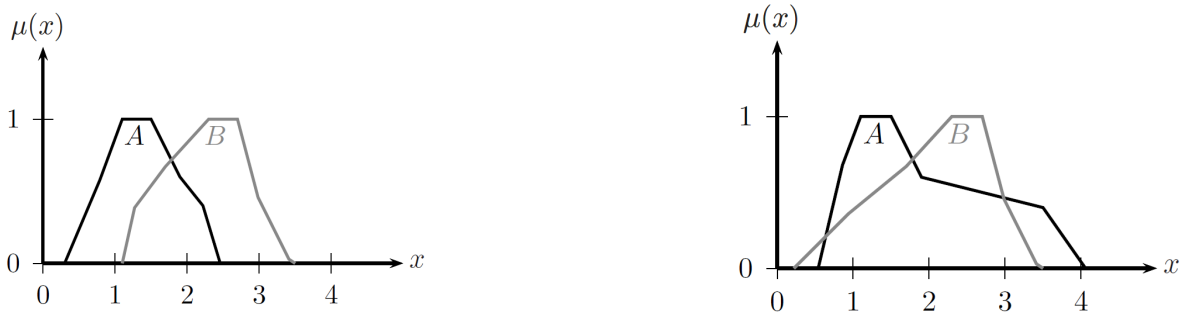
Abbildung A.1a zeigt den trivialen Fall, wenn für zwei verschiedene Fuzzy-Intervalle A und B die untere und obere Grenzen aller α-Cuts von B größer sind als die entsprechenden Grenzen der α-Cuts von A. In einem solchen Fall spricht man davon, dass B streng größer als A ist.

Definition 40: Strenge Ordnungsrelation - Strong Comparison Rule (SCR)

Eine Fuzzy-Zahl $\tilde{A}$ ist streng größer als eine andere Fuzzy-Zahl $\tilde{B}$, wenn gilt:

$$max(\tilde{A}, \tilde{B}) = \tilde{B} \tag{A.1}$$

Im Folgenden wird diese Relation als $\tilde{A} \gg \tilde{B}$, beziehungsweise $\tilde{B} \ll \tilde{A}$ dargestellt.



(a)Trivialer Fall. $\tilde{A}$ ist immer kleiner als $\tilde{B}$

(b)Nicht triviale Überlappung von Fuzzy-Zahlen / Fuzzy Intervallen.

Abbildung A.1.: Unterschiedliche Überlappung von Fuzzy-Zahlen / Fuzzy Intervallen.

Liegt eine Überschneidung wie in A.1b vor, so ist eine Aussage darüber zu treffen, welche der beiden Fuzzy-Zahlen bzw. Intervalle größer als die andere Fuzzy-Zahl ist, nicht trivial.

Für diesen Fall wird in Hapke u. Słowiński (1996) und Hapke u. a. (1999) vorgeschlagen, den Grad $C(\tilde{A} \geq \tilde{B})$ zu bestimmen, der ein Maß angibt, um welchen Wert die Fuzzy-Zahl $\tilde{A}$ größer als die Fuzzy-Zahl $\tilde{B}$ ist.

Definition 41: Grad der Aussage $\tilde{A} \geq \tilde{B}$

Der Grad, zu dem die Aussage $\tilde{A} \geq \tilde{B}$ gilt, lässt sich aus folgenden Ausdruck bestimmen:

$$C(\tilde{A} \geq \tilde{B}) = \frac{1}{2}\left\{S_L(\tilde{A} \geq \tilde{B}) + S_R(\tilde{A} \geq \tilde{B}) - S_L(\tilde{B} \geq \tilde{A}) - S_R(\tilde{B} \geq \tilde{A})\right\}. \tag{A.2}$$

Die Flächen $S_L(\tilde{A} \geq \tilde{B}), S_R(\tilde{A} \geq \tilde{B}), S_L(\tilde{B} \geq \tilde{A})$ und $S_R(\tilde{B} \geq \tilde{A})$ ergeben sich dabei aus den Zugehörigkeitsfunktionen und den entsprechenden α-Cuts. Betrachtet man Abbildung A.2, so stehen die beiden Flächen $S_L(\tilde{A} \geq \tilde{B}), S_R(\tilde{A} \geq \tilde{B})$ mit der Möglichkeit, dass $\tilde{A} \geq \tilde{B}$ gilt, in Relation. Analoges gilt für die Flächen $S_L(\tilde{B} \geq \tilde{A})$ und $S_R(\tilde{B} \geq \tilde{A})$ und der Möglichkeit, dass $\tilde{B} \geq \tilde{A}$ gilt.

Definition 42: Schwache Ordnungsrelation - Weak Comparison Rule (WCR)

Aus dieser Formulierung lässt sich die schwache Vergleichsregel nach Hapke u. Słowiński (1996) definieren:

$$\tilde{A} \geq \tilde{B}, \text{falls} \quad C(\tilde{A} \geq \tilde{B}) \geq 0 \tag{A.3}$$

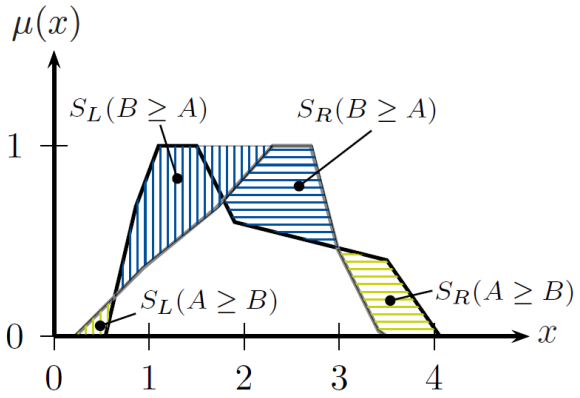


Abbildung A.2.: Vergleich zweier Fuzzy-Zahlen

Die Formulierung A.2 lässt sich überführen in die alternative Form, die Liou u. Wang (1992) beschreiben. Sie splitten das Integral einer Fuzzy-Zahl über deren linken und rechten Funktionsast $f^L_{\tilde{A}}$ und $f^R_{\tilde{A}}$. Zur Bestimmung der beiden Integrale bedient man sich der Inversen $g^L_{\tilde{A}}$ und $g^R_{\tilde{A}}$.

$$I_L(\tilde{A}) = \int_0^1 g^L_{\tilde{A}}(\alpha) d\alpha \tag{A.4}$$

$$I_R(\tilde{A}) = \int_0^1 g^R_{\tilde{A}}(\alpha) d\alpha \tag{A.5}$$

Das totale Integral einer Fuzzy-Zahl ergibt sich aus der gewichteten Summe der beiden Teilintegrale:

$$I_T(\tilde{A}, \gamma) = \gamma \cdot I_L(\tilde{A}) + (1 - \gamma) \cdot I_R(\tilde{A}) \tag{A.6}$$

Anhand dieser Formulierung lässt sich für das Sechspunkt-Fuzzy-Intervall $\tilde{A}$ das Gesamtintegral $I_T(\tilde{A}$ bei einer gleichmäßigen Gewichtung der beiden Teiläste mit $\gamma = 0.5$ bestimmen und recht effizient berechnen:

$$I_L(\tilde{A}) = \frac{1}{2}\left(c_{1\lambda} + m_1 + (c_1 - m_1)\lambda\right) \tag{A.7}$$

$$I_R(\tilde{A}) = \frac{1}{2}\left(c_{2\lambda} + m_2 + (c_2 - m_2)\lambda\right) \tag{A.8}$$

A.2. Beispielhafte Probleminstanz

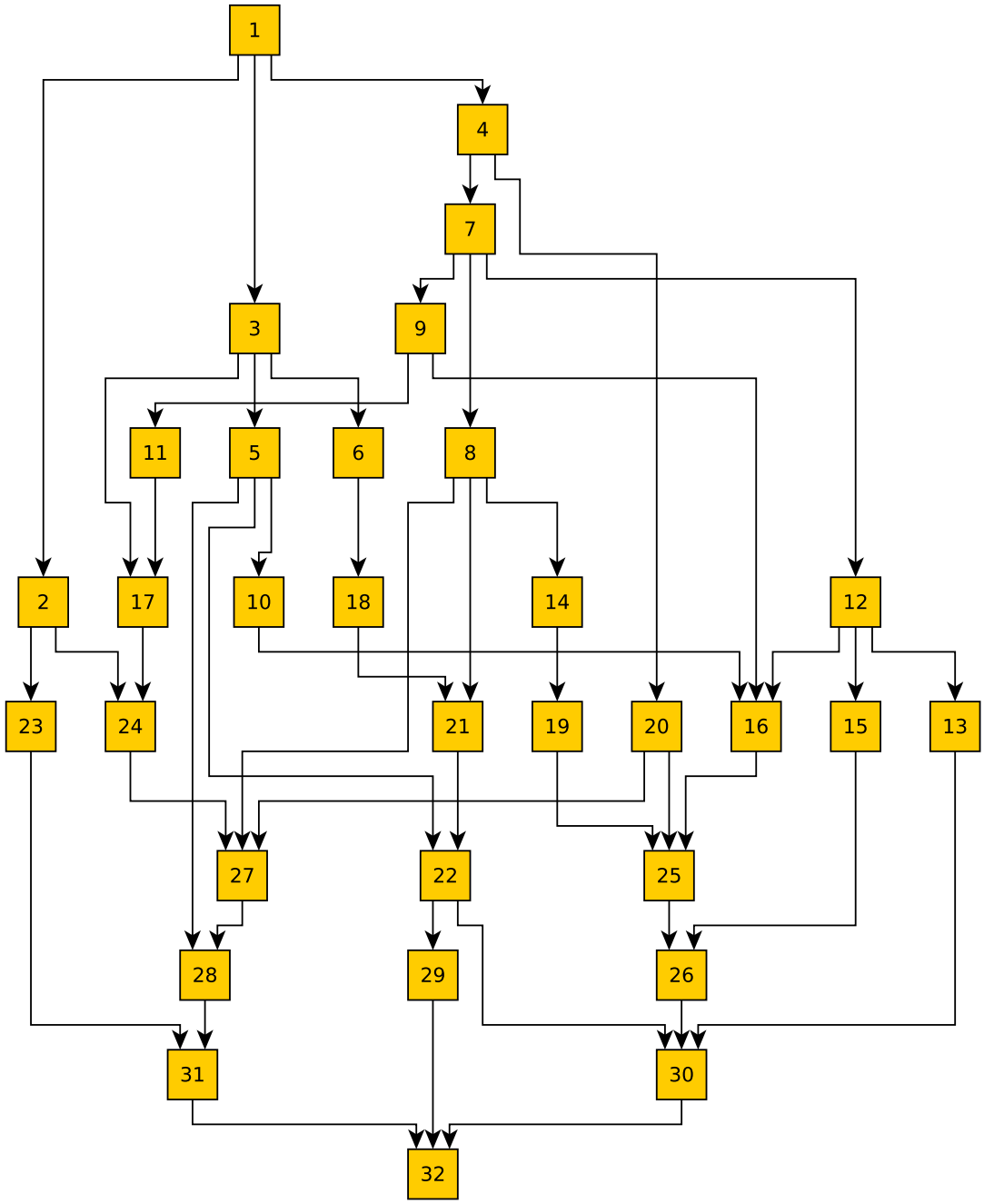


Abbildung A.3.: Activity-On-Node-Diagramm der betrachteten Probleminstanz

A.2.1. Dauer und Ressourcenbedarf der Operationen der betrachteten Probleminstanz

```
******************************************************************************
file with basedata            : j30_17.bas
initial value random generator: 79602564
******************************************************************************
projects                      :  1
jobs (incl. supersource/sink ):  32
horizon                       :  141
RESOURCES
  - renewable                 :  4   R
  - nonrenewable              :  0   N
  - doubly constrained        :  0   D
******************************************************************************
PROJECT INFORMATION:
pronr.  #jobs rel.date duedate tardcost  MPM-Time
    1     30      0       43        0       43
******************************************************************************
PRECEDENCE RELATIONS:
jobnr.    #modes  #successors   successors
   1        1          3           2   3   4
   2        1          2          23  24
   3        1          3           5   6  17
   4        1          2           7  20
   5        1          3          10  22  28
   6        1          1          18
   7        1          3           8   9  12
   8        1          3          14  21  27
   9        1          2          11  16
  10        1          1          16
  11        1          1          17
  12        1          3          13  15  16
  13        1          1          30
  14        1          1          19
  15        1          1          26
  16        1          1          25
  17        1          1          24
  18        1          1          21
  19        1          1          25
  20        1          2          25  27
  21        1          1          22
```

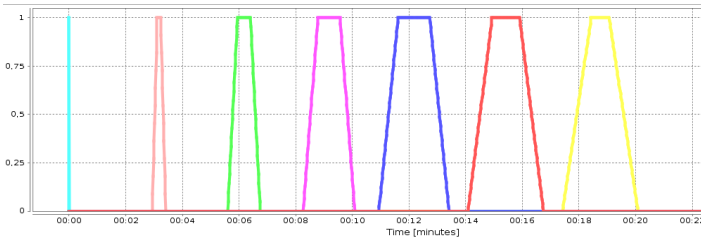
```
  22        1          2          29   30
  23        1          1          31
  24        1          1          27
  25        1          1          26
  26        1          1          30
  27        1          1          28
  28        1          1          31
  29        1          1          32
  30        1          1          32
  31        1          1          32
  32        1          0
************************************************************************
REQUESTS/DURATIONS:
jobnr. mode duration  R 1  R 2  R 3  R 4
------------------------------------------------------------------------
  1      1     0       0    0    0    0
  2      1     1       0    0    0    5
  3      1     1       0    3    0    0
  4      1     1       8    0    0    0
  5      1     7       0    0    2    0
  6      1     6       0    0    0    3
  7      1     4       1    0    0    0
  8      1     5       0    0   10    0
  9      1     8       0    0    3    0
 10      1     7       0    0    0    1
 11      1     8       9    0    0    0
 12      1     1       7    0    0    0
 13      1     2       0    3    0    0
 14      1     3       0    0    0    6
 15      1    10       0    7    0    0
 16      1    10       3    0    0    0
 17      1     2       0    0    3    0
 18      1    10       0    0    4    0
 19      1     1       0    0    0    3
 20      1     1       0    0    7    0
 21      1     7       0    2    0    0
 22      1     9       0    0    0   10
 23      1     9       0    0    7    0
 24      1     4       0    4    0    0
 25      1     4       0    3    0    0
 26      1     1       0    0    4    0
 27      1     1       9    0    0    0
 28      1     8       0    0    0    9
```

```
 29      1      1        0     0     0     1
 30      1      2        0     8     0     0
 31      1      7        0     4     0     0
 32      1      0        0     0     0     0
************************************************************************
RESOURCEAVAILABILITIES:
  R 1  R 2  R 3  R 4
   10    8   13   12
************************************************************************
```

A.3. Entwicklung der Unschärfe im Projektverlauf

Um die Entwicklung der Unschärfe über die Projektdauer zu analysieren, werden sieben Operationen eingeplant, deren Operationsdauer jeweils drei Minuten mit einer Unschärfe von $\gamma = 20s$ beträgt. Die Operationen werden nacheinander auf einer Ressource eingeplant und haben jeweils den Ressourcenbedarf von einer Einheit.

Die Startzeit einer nachfolgenden Operation ergibt sich aus der Endzeit der vorangegangen Operation, folglich der Summe aus deren Operationsstart und Operationsdauer. Durch die Aufsummierung der unscharfen Werte vergrößert sich der Bereich des Trägers der Fuzzyzahl, die die Startzeit beschreibt über den Verlauf des Projekts.



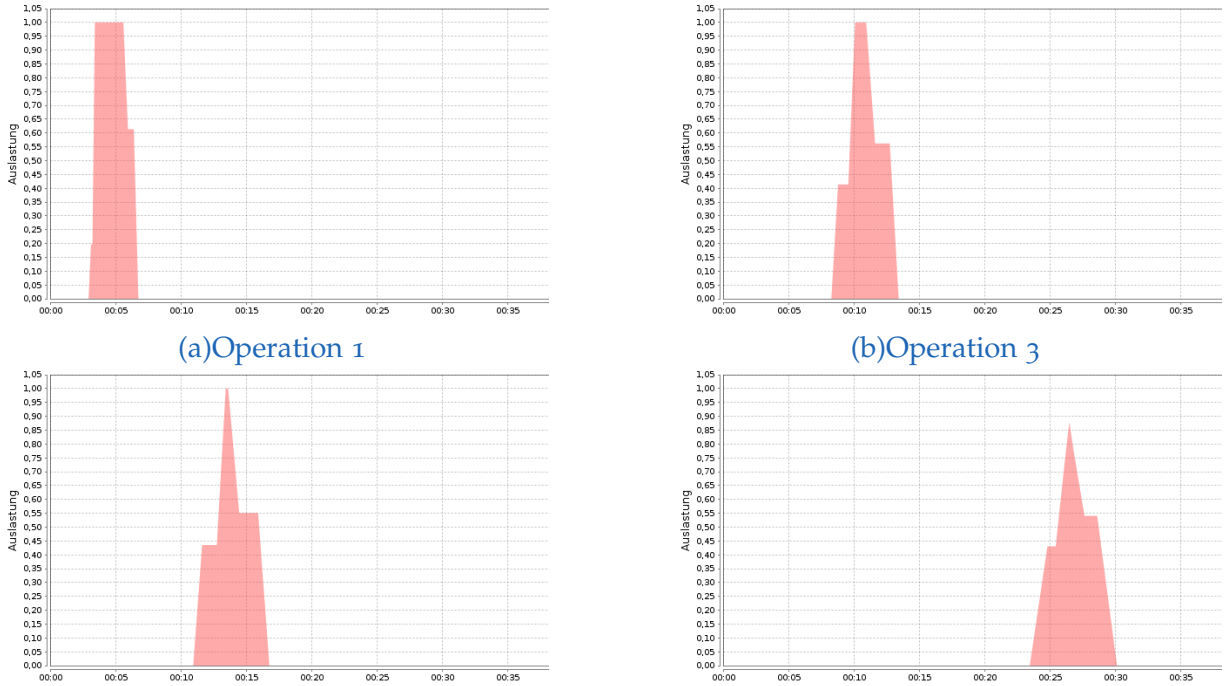
Nr.	c_1	c_2	$c_2 - c_1$
1	23:59:59	00:00:01	2 s
2	00:02:56	00:03:25	29 s
3	00:05:36	00:06:45	69 s
4	00:08:16	00:10:05	109 s
5	00:10:56	00:13:25	149 s
6	00:14:05	00:16:45	160 s
7	00:17:25	00:20:05	160 s

Abbildung A.4.: Beispiel - Entwicklung der unscharfen Startzeiten von aufeinanderfolgenden Operationen

Erweitert man die Probleminstanz um weitere Operationen mit identischem Ressourcenbedarf und identischer Dauer lässt sich beobachten, dass durch die Vergrößerung des Trägers bei den Startzeiten und somit auch bei den Endzeiten sich der Verlauf der unscharfen Auslastung ändert. Zunächst verkleinert sich der Bereich, in dem die Auslastung 100% beträgt, bis dieser lediglich noch einen Peak darstellt und die Auslastung durch die Operation an seiner höchsten Stelle unter 100% liegt. Dadurch vergrößert sich allerdings die Ausdehnung in x-Richtung (siehe Abbildung A.5). Das führt dazu, das im Extremfall mehrere Operationen gleichzeitig eingeplant werden können. Dies entspricht der Anschauung, dass es bei der unscharfen Betrachtung um eine Bewertung der Machbarkeit geht. Es ist nicht wichtig, wann genau die einzelne Operation stattfindet, sondern lediglich, dass es zu einem großen Grad an Glaubwürdigkeit machbar ist, alle Operationen auszuführen.

Durch den immer breiter werdenden Verlauf der Auslastung durch eine einzelne Operation ändert sich auch der Wert von λ_{min} und λ_{max}. Betrachtet man die eingeschränkten Ressourcenprofile und die Extrem-Funktionen, bei denen λ_{min} bzw. λ_{max} für die Bestimmung des Auslastungsprofil gewählt wurde, so wird deutlich, dass die beiden

Funktionsverläufe sich im Verlauf des Projekts annähern. Da das Integral der minimalen und maximalen Auslastung nicht unter- oder überschritten werden darf und die Ausdehnung in x-Richtung anwächst, wird der mögliche Bereich für λ eingeschränkt.



(a)Operation 1

(b)Operation 3

(c)Operation 4

(d)Operation 8

Abbildung A.5.: Beispiel - Zeitliche Entwicklung unscharfer Ressourcenprofile.

A.4. Einfuss der Erhöhung der Unschärfe

Im Folgenden wird die Unschärfe der Dauern schrittweise mittels des Parameter γ erhöht (siehe Abbildung A.6). Durch die größere Unschärfe in den Operationsdauern ergibt sich eine größere Unschärfe in den Startzeiten der nachfolgenden Operationen. Dies führt dazu, dass die Ausdehnung von Plateaus mit hohen Auslastungen, wie bei der vierten Operation aus dem scharfen Fall, die bei 00 : 13 startet, sich schrittweise verringert. Das Auslastungsprofil weist bei einer Unschärfe von $\gamma = 45\,\mathrm{s}$ nur noch einen kleinen Notwendigkeit-Peak auf. Bis auf diese Peaks wird das Auslastungsprofil ausgeglichener. Dies liegt daran, dass bei der unscharfen Betrachtungsweise keine punktuelle Einplanung des benötigten Ressourcenbedarfs vorgenommen wird. Der Ressourcenbedarf verteilt sich entsprechend der possibilitätstheoretischen Modellierung auf den Bereich, in dem die Möglichkeit besteht, dass die Operation ausgeführt wird. Abbildung A.7 zeigt im Vergleich zur vorangehenden Analyse bei einem Start-β von 0.5 die Entwicklung des Ressourcenprofils bei einem Start-β von 1.0.

Abbildung A.6.: Auslastungsprofile der Problemsinstanz unter Unschärfe. Links: Auslastung pro Operation. Rechts: Gesamtauslastung. Start-$\beta = 0.5$. Der Parameter γ verändert sich von oben nach unten: $\gamma_1 = 1\,\text{s}, \gamma_2 = 5\,\text{s}, \gamma_3 = 20\,\text{s}, \gamma_4 = 30\,\text{s}, \gamma_5 = 45\,\text{s}$

Abbildung A.7.: Auslastungsprofile der Problemsinstanz unter Unschärfe. Links: Auslastung pro Operation. Rechts: Gesamtauslastung. Start-$\beta = 1.0$. Der Parameter γ verändert sich von oben nach unten: $\gamma_1 = 1\,\text{s}, \gamma_2 = 5\,\text{s}, \gamma_3 = 20\,\text{s}, \gamma_4 = 30\,\text{s}, \gamma_5 = 45\,\text{s}$

A.5. Variation des Startwertes von λ_L

Bei der Erzeugung eines Ablaufplans wird das Integral des unscharfen Auslastungsprofil einer Operation durch das vom Entscheidungsträger vorgegebenen β bestimmt. Dieser Wert entspricht dem Ressourcenbedarf, der eingeplant werden muss. Die benötigten Ressourcen variieren je nach tatsächlich eintretendem Fall zwischen dem pessimistischen Fall W_Π und dem optimistischen W_N (siehe 3.25 und 3.26).

Bei der Einplanung kann nun zunächst probiert werden, ein symmetrisch Auslastungsprofil einzuplanen oder einen der beiden Teilbereiche zu präferieren. Dies lässt sich über den Startwert des Parameters β_L modifizieren. Bei einem Startwert von $\lambda_L = 0.5$ wird zunächst versucht ein symmetrisches Profil einzuplanen. Ist dies nicht möglich, wird der Parameter schrittweise gesenkt. Um das vom Entscheidungsträger indirekt vorgegebene Integral zu erfüllen, muss dementsprechend der Parameter λ_R berechnet werden. Ein Startwert von $\lambda_L = 1.0$ führt dazu, dass versucht wird den vorderen Teil zu präferieren, eine Operation folglich möglichst früh stattfindet. Analog wird auch hier iterativ verringert, sollte eine Einplanung zunächst nicht möglich sein.

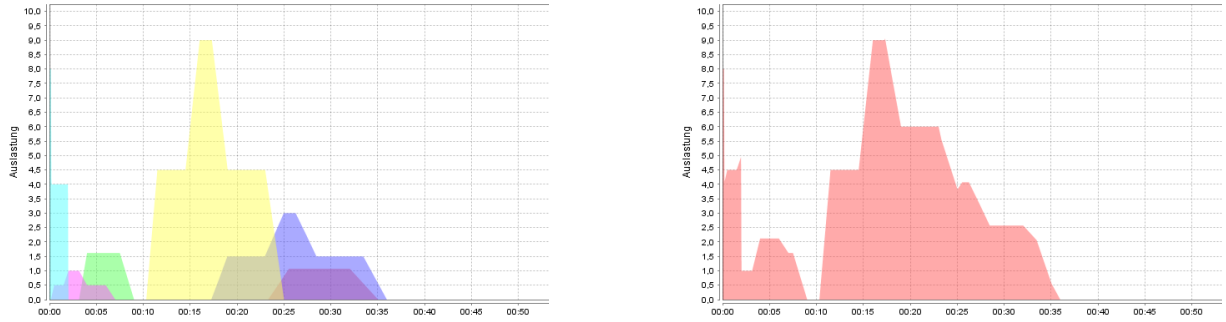


Abbildung A.8.: Beispiel - Unscharfes Auslastungsprofil mit einem Start-λ_L von 0.5 bei $\gamma = 30\,\text{s}$.

Abbildung A.8 zeigt ein aus diesem Vorgehen entstandenes Auslastungsprofil. Die Plateaus seitlich des Plateaus, welches durch die Notwendigkeit entsteht, besitzen die gleiche Höhe. Alternativ dazu wurde zur Erstellung des Auslastungsprofil in Abbildung A.9 versucht möglichst viel des Integrals des Profils auf den vorderen Teil zu verteilen.

Dies führt zur ungleichmäßig hohen Seitenplateaus. Betrachtet man die Gesamtauslastungsprofile, so lässt sich festhalten, dass ein symmetrisches Einplanen einen glatteren Auslastungsverlauf zur Folge hat. Eine asymmetrische Einplanung führt dazu, dass die Peaks im Profil markanter sind und die Auslastung gegen Ende des betrachteten Ausschnittes schneller auf einen niedrigeren Wert fällt. Die zu erwartende Auslastung sinkt dementsprechend stärker gegen Ende.

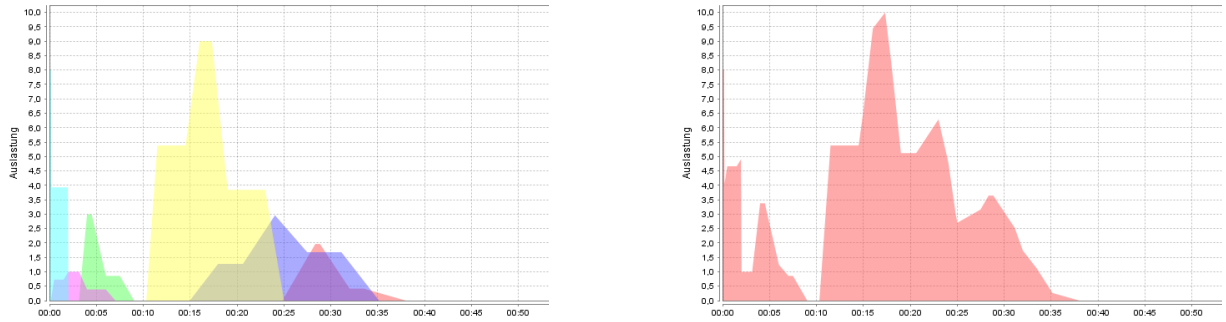


Abbildung A.9.: Beispiel - Unscharfes Auslastungsprofil mit einem Start-λ_L von 1.0 bei $\gamma = 30\,\text{s}$.

A.6. Verkürzung der Gesamtdauer durch Vergrößerung der Unschärfe

Durch eine Vergrößerung der Unschärfe kann es zur Verkürzung der Gesamtdauer kommen. Abbildung A.10 zeigt dieses Phänomen bei den beiden letzten eingeplanten Operationen.

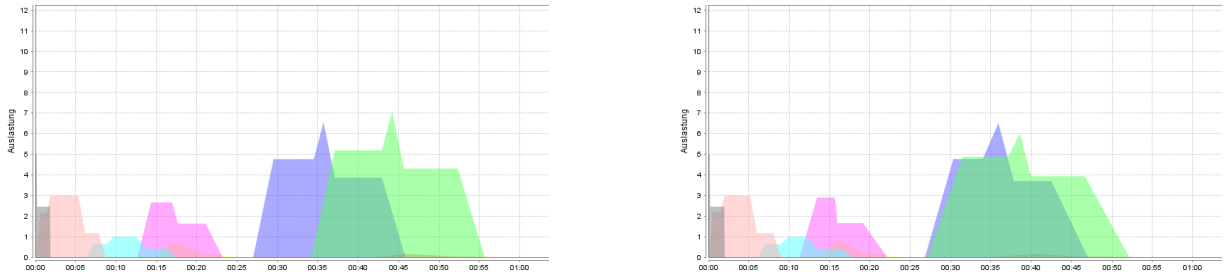


Abbildung A.10.: Beispiel - Verkürzung der Gesamtdauer bei Ressource 4 mit einem Start-β_L von 1.0 bei einer Erhöhung von $\gamma = 25\,\text{s}$ auf $\gamma = 30\,\text{s}$.

Durch die gewählte Form der Modellierung der Unschärfe, besteht auch eine größerer Möglichkeit, dass die Operation früher stattfindet, als bei einer scharfen Modellierung mit einem scharfen Wert d für die Dauer (vgl. Abbildung 4.23). Durch die steigende Unschärfe ist der verringert sich der Wert der Notwendigkeit. Dieses Absenken der Peaks und des gesamten Verlaufs kann dazu führen, dass zwei Operationen im unscharfen Sinn zeitgleich stattfinden können, in dem hier betrachteten Fall die letzten beiden Operationen des Ablaufplans.

Anhang B.

Containerterminal zur Validierung

B.1. Optimierungsergebnisse und Quellcode

Alle berechneten Ergebnisse sowie der Quellcode des in Java geschriebenen Programms und eine genaue Beschreibung der Probleminstanzen des Containerterminal-Problems sind unter https://github.com/matthiasbode/multiskalen zu finden. Nachfolgend sind einige Ergebnisse exemplarisch dargestellt.

B.1.1. Transportprogramme

Die Transportprogramme werden grundsätzlich auf Basis von Expertenannahmen generiert. Sie spiegeln das generell zu erwartende Aufkommen auf einem *Hub* im Rahmen eines *Hub & Spoke*-Systems wider. Um Transportprogramme zu generieren, die ein gewisses Optimierungspotential bergen, wurden einige Veränderungen bezüglich des Anteils an Ladeeinheiten, die zur Stammrelationen gehören, vorgenommen. Die Transportprogramme sind im JSON-Format hinterlegt.


```
{
    "typ": "TwistLockLoadUnit",
    "length": 12.192,
    "origin": {
      "type": "Slot",
      "train": {
        "number": 4
      },
      "number": 35
    },
    "destination": {
      "type": "Slot",
      "train": {
```

```
        "number": 1
      },
      "number": 37
    },
    "hazardous": false,
    "id": "ISO361"
  }
```


Die Abbildungen B.1 und B.2 zeigen exemplarisch die Ent- beziehungsweise Beladungen des Zuges 5 aus Transportprogramm TP1.

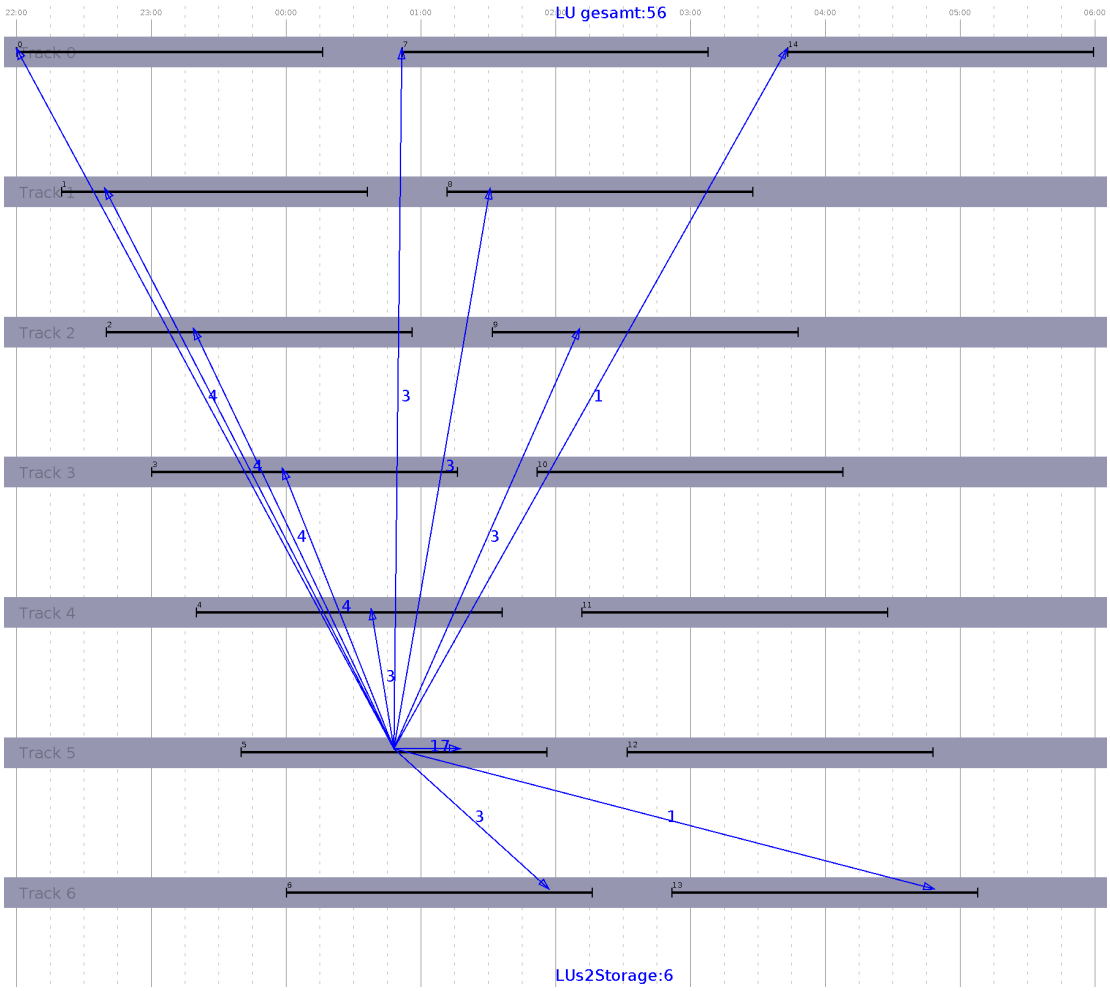


Abbildung B.1.: Exemplarisch für Zug 5 werden die Entladungen angezeigt. Hierbei handelt es sich um die Anzahl an Ladeeinheiten, die auf die anderen Züge umgeladen werden müssen. Beispielsweise werden drei Ladeeinheiten von Zug 5 auf Zug 6 umgeschlagen.

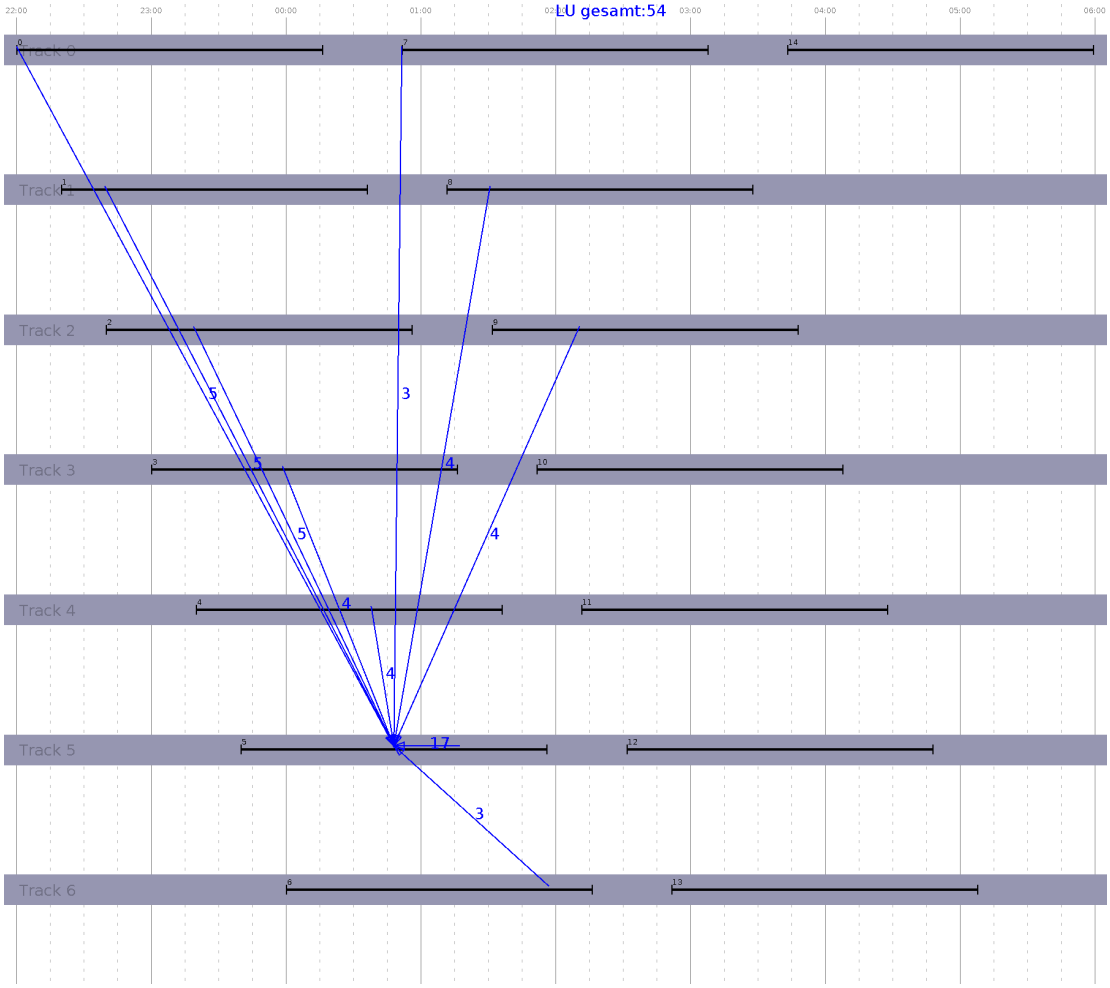


Abbildung B.2.: Exemplarisch für Zug 5 werden die Beladungen angezeigt. Hierbei handelt es sich um die Anzahl an Ladeeinheiten, die von den anderen Zügen auf den Zug 5 umgeladen werden müssen. Beispielsweise werden drei Ladeeinheiten von Zug 6 auf Zug 5 umgeschlagen.

B.1.2. Typische Ergebnisse einer Analyse der Kranfahrten im Rahmen eines 8h Betriebs

```
########################
Krandaten
########################
Crane{0}
Anzahl an Transportoperationen: 163
Mittlere TransportDauer: 87
Mittlere RüstDauer: 38
Standardabweichung RüstDauer: 20.397684182279125
---------------------------
Crane{1}
Anzahl an Transportoperationen: 159
Mittlere TransportDauer: 92
Mittlere RüstDauer: 41
Standardabweichung RüstDauer: 25.724167041908277
---------------------------
Crane{2}
Anzahl an Transportoperationen: 161
Mittlere TransportDauer: 89
Mittlere RüstDauer: 41
Standardabweichung RüstDauer: 25.553217801286788
---------------------------
Crane{3}
Anzahl an Transportoperationen: 150
Mittlere TransportDauer: 89
Mittlere RüstDauer: 40
Standardabweichung RüstDauer: 22.216035627447127
---------------------------
Crane{4}
Anzahl an Transportoperationen: 152
Mittlere TransportDauer: 87
Mittlere RüstDauer: 37
Standardabweichung RüstDauer: 22.676536089094384
---------------------------
mittlere Kranspiele pro LoadUnit: 1.1582840236686391
```

B.2. Studie: Einfluss der genetischen Parameter auf den Multiskalen-Algorithmus

B.2.1. Einfluss der Mutation

Um sicherzustellen, dass der entwickelte genetische Algorithmus auf makroskopischer Ebene keine zu starke Abhängigkeit von der Mutation hat, wurde eine Optimierung auf makroskopischer Ebene mit einer Mutationsrate von 0.0 sowohl bei der Ablaufplanung, als auch bei der Prozessplanung vorgenommen. Die Mutation dient als wichtiger Teil eines genetischen Algorithmus zur Erkundung des Lösungsraum, eine zu starke Abhängigkeit von der Mutation weißt allerdings darauf hin, dass der Algorithmus lediglich durch Zufall eine bessere Lösung in einer späteren Generation bestimmt.

Abbildung B.3 zeigt den Fitnessverlauf für die erste Komponente des Fitnessvektor, die Anzahl an DNF, makroskopischer Ebene mit einer Mutationsrate von 0.4 für die Prozess- und 0.3 für die Ablaufplanung. Im Rahmen weiterer Studien erwies sich diese Parameterwahl als sinnvoll, da bei einer größeren Mutationsrate gute, durch die Rekombination entstandene Lösungen, verschlechterten. Abbildung B.4 zeigt im Vergleich dazu die Fitnessentwicklung komplett ohne Mutation. Bei beiden Durchläufen wurden pro Population je 100 Individuen über 20 Generationen betrachtet. Es zeigt sich, dass die Rekombination alleine durchaus vielversprechende Lösungen hervorbringt, allerdings, wie zu erwarten, unter Umständen in einem lokalen Optimum verweilt. Eine Bereicherung des Genpools durch die Mutation führt somit zur besseren Erkundung des gesamten Lösungsraums.

Dies zeigen auch die Verläufe der Diversität der Population der Superindividuen. Abbildung B.5 zeigt die Entwicklung bei einer Mutationsrate von 0.4 bzw. 0.3. Im Vergleich zu Abbildung B.6 bei einer Mutationsrate von 0.0 lässt sich feststellen, dass die Diversität durch die Mutation erhöht wird.

Wie bereits erwähnt steigt die Güte mit wachsender Mutationsrate zunächst an, sinkt allerdings bei zu hohen Werten wiederum, da gute Lösungen durch die Mutation unter Umständen so verändert werden, dass sie eine schlechtere Güte aufweisen.

Tabelle B.1 zeigt den Einfluss der Mutationsrate der makroskopischen Ebene auf die Ergebnisse des Multiskalen-Algorithmus.

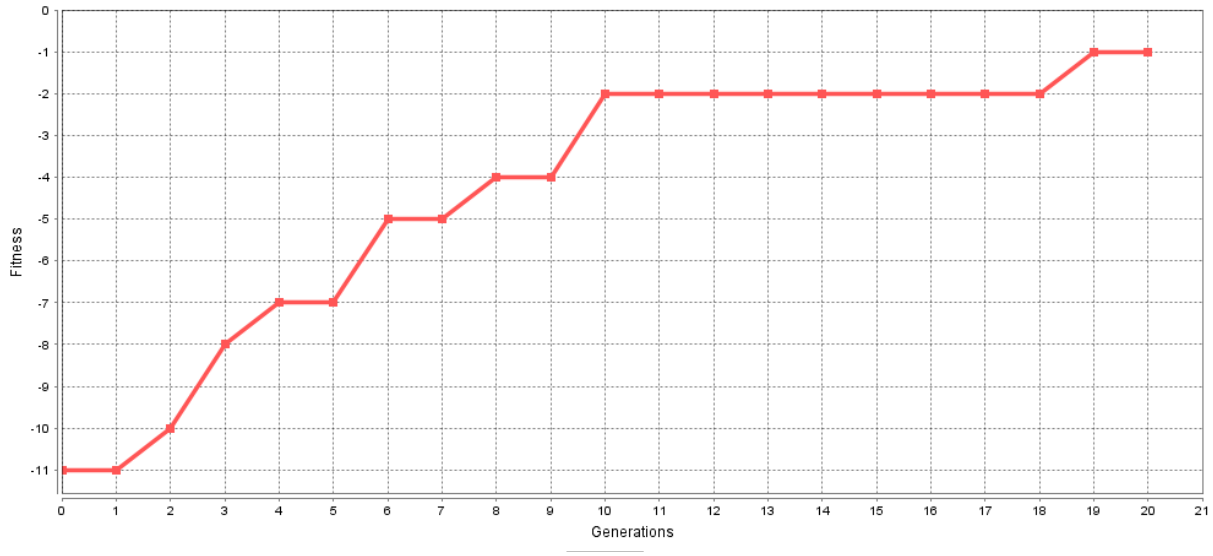


Abbildung B.3.: Fitness-Entwicklung bei einer Mutationsrate von 0.4 bzw. 0.3

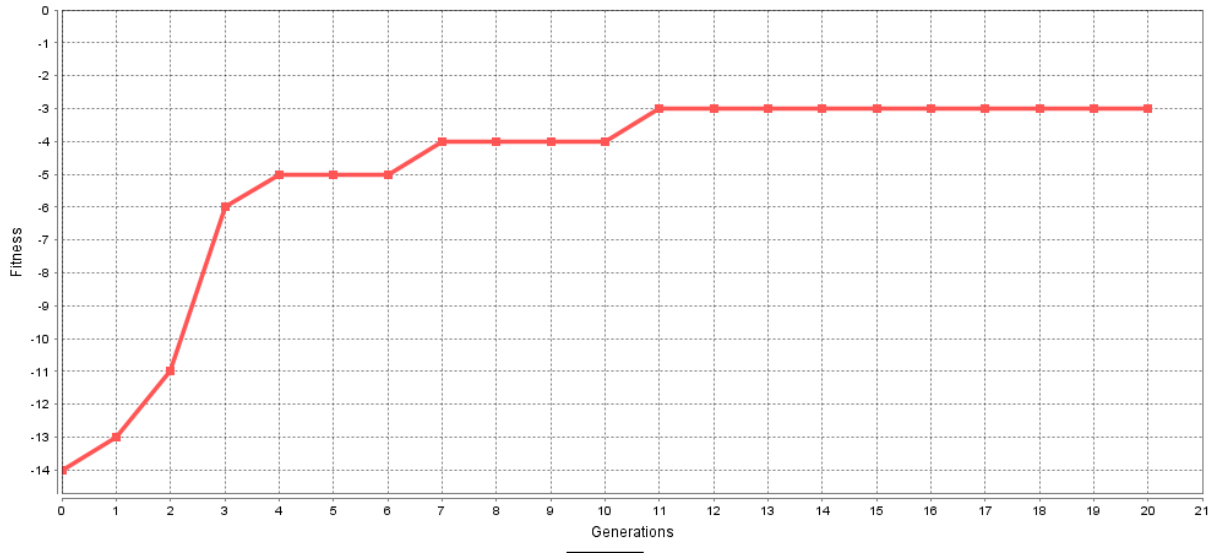


Abbildung B.4.: Fitness-Entwicklung bei einer Mutationsrate von 0.0

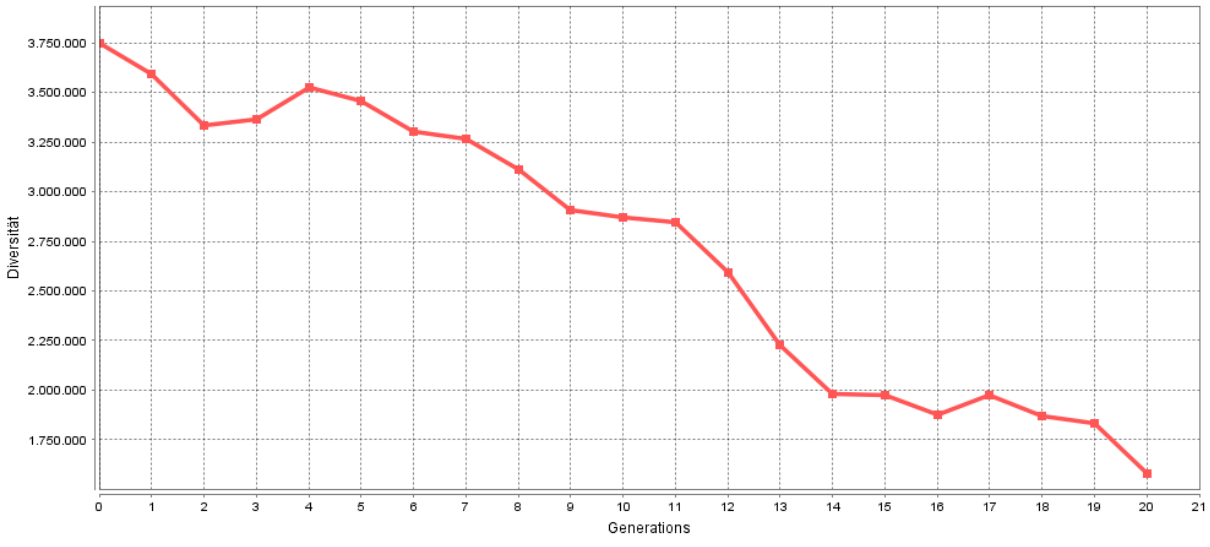


Abbildung B.5.: Diversitätsentwicklung des Superindividuum bei einer Mutationsrate von 0.4 bzw. 0.3

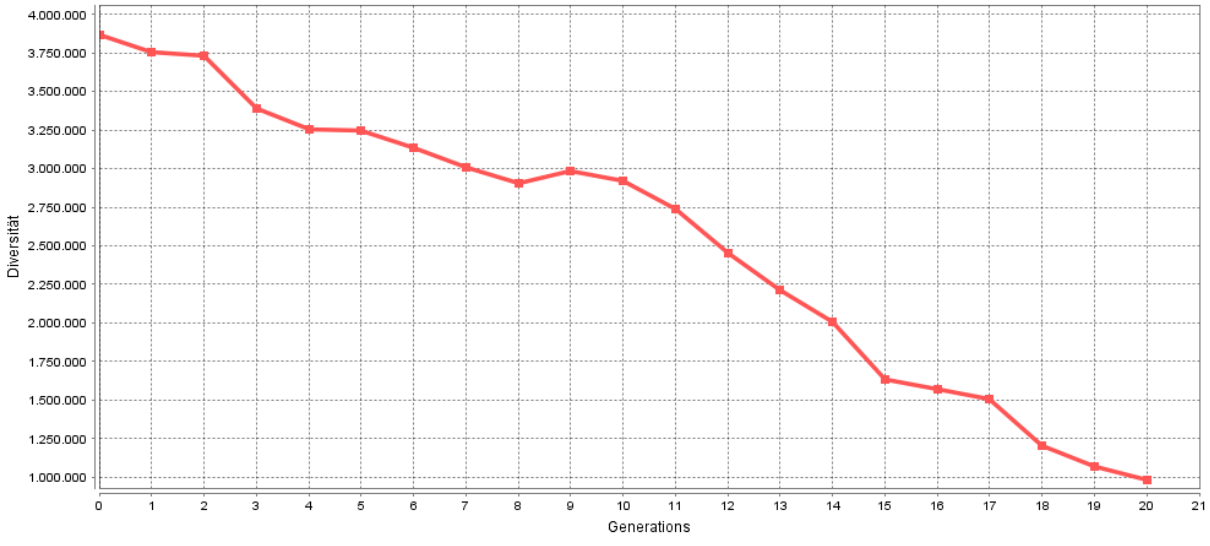


Abbildung B.6.: Diversitätsentwicklung des Superindividuum bei einer Mutationsrate von 0.0

Mutationsrate	mittlere Rüst-dauer	mittlere Kranspiele pro Ladeeinheit	Anzahl DNF
0.0	00:00:35.734 h	1.191190	7
0.1	00:00:36.984 h	1.191444	7
0.3	00:00:35.854 h	1.195707	5
0.5	00:00:36.847 h	1.198998	6
0.7	00:00:34.105 h	1.190208	12

Intel Core i7-4960X @ 3.60 GHz (4.00 GHz) mit 32 GB RAM

Tabelle B.1.: Multiskalen-Algorithmus: Ergebnisse bei unterschiedlicher Mutationsrate (sowohl für die Prozess- als auch für die Ablaufplanung) für das Transportprogramm TP1

Literaturverzeichnis

[Al-Fawzan u. Haouari 2005] Al-Fawzan, M. A. ; Haouari, M.: A bi-objective model for robust resource-constrained project scheduling. In: *International Journal of production economics* 96 (2005), Nr. 2, S. 175–187

[Alcaraz u. a. 2003] Alcaraz, J ; Maroto, C ; Ruiz, R: Solving the Multi-Mode Resource-Constrained Project Scheduling Problem with genetic algorithms. In: *Journal of the Operational Research Society* 54 (2003), Juni, Nr. 6, 614–626. http://dx.doi.org/10.1057/palgrave.jors.2601563. – DOI 10.1057/palgrave.jors.2601563. – ISSN 0160–5682

[Althöfer 1999] Althöfer, I.: Decision Support Systems With Multiple Choice Structure. In: *NUMBERS, INFORMATION AND COMPLEXITY*, 1999, S. 525–540

[Ayache 2005] Ayache, P.-D.: Schedule Generation Schemes for the Job-Shop Problem with Sequence-Dependent Setup Times: Dominance Properties and Computational Analysis. (2005), Nr. 1995, S. 21–52

[Balasubramanian u. Grossmann 2004] Balasubramanian, J. ; Grossmann, I. E.: Approximation to multistage stochastic optimization in multiperiod batch plant scheduling under demand uncertainty. In: *Industrial and Engineering Chemistry Research* 43 (2004), Nr. 14, 3695-3713. www.scopus.com

[Bansal 2011] Bansal, A.: Trapezoidal fuzzy numbers (a, b, c, d): Arithmetic Behavior. In: *International Journal of Physical and Mathematical Sciences* 2 (2011), Nr. 1

[Bertsimas u. a. 2013] Bertsimas, D. ; Litvinov, E. ; Sun, X. A. ; Zhao, J. ; Zheng, T.: Adaptive robust optimization for the security constrained unit commitment problem. In: *Power Systems, IEEE Transactions on* 28 (2013), Nr. 1, S. 52–63

[Bidot 2005] Bidot, J.: A General Framework Integrating Techniques for Scheduling under Uncertainty. (2005)

[Birge u. Louveaux 2011] Birge, J.R. ; Louveaux, F.: *Introduction to stochastic programming*. Springer Science & Business Media, 2011

[Bode u. a. 2010] Bode, M. ; Hofmann, F. ; Berkhahn, V.: Container Transshipment Simulation and Efficiency Analysis. In: Blecker, T. (Hrsg.) ; Huang, G.Q. (Hrsg.) ; Salvador, F. (Hrsg.): *Innovative Process Optimization Methods in Logistics*, 2010, S. 321–335

[Bode u. a. 2012] Bode, M. ; Schiermeyer, C. ; Berkhahn, V.: Job Scheduling using event-discrete Simulation, Pre-Optimisation and Just-In-Time Consideration of Disturbance Factors. In: Gudnason, G. (Hrsg.) ; Scherer, R. (Hrsg.): *eWork an eBusiness in Architecture, Engineering and Construction*, 2012 (ECPPM 2012), S. 227–233

[Bonfill u. a. 2004] Bonfill, A. ; Bagajewicz, M. ; Espuña, A. ; Puigjaner, L.: Risk management in the scheduling of batch plants under uncertain market demand. In: *Industrial & engineering chemistry research* 43 (2004), Nr. 3, S. 741–750

[Bourier 2002] Bourier, G.: Kombinatorik. Version: 2002. http://dx.doi.org/10.1007/978-3-322-96585-1_5. In: *Wahrscheinlichkeitsrechnung und schließende Statistik*. Gabler Verlag, 2002. – DOI 10.1007/978–3–322–96585–1_5. – ISBN 978–3–409–31463–3, 71-82

[Brucker u. Neyer 1998] Brucker, P. ; Neyer, J.: Tabu-search for the multi-mode job-shop problem. In: *Operations-Research-Spektrum* 20 (1998), Nr. 1, 21-28. http://dx.doi.org/10.1007/BF01545525. – DOI 10.1007/BF01545525. – ISSN 0171–6468

[Cai u. Peng 2002] Cai, Z. ; Peng, Z.: Cooperative Coevolutionary Adaptive Genetic Algorithm in Path Planning of Cooperative Multi-Mobile Robot Systems. (2002), S. 61–71

[Civanlar u. Trussell 1986] Civanlar, M. R. ; Trussell, H. J.: Constructing membership functions using statistical data. In: *Fuzzy sets and systems* 18 (1986), Nr. 1, S. 1–13

[Davenport u. a. 2014] Davenport, A. ; Gefflot, C. ; Beck, C.: Slack-based techniques for robust schedules. In: *Sixth European Conference on Planning*, 2014

[Davis 1985] Davis, L.: Applying adaptive algorithms to epistatic domains. In: *IJCAI* Bd. 85, 1985, S. 162–164

[DB Netz AG 2014] *Neubauprojekt MegaHub Lehrte*. http://www.deutschebahn.com/megahub. Version: 2014

[De Jong 1975] De Jong, K. A.: *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* Ann Arbor, MI, USA, Diss., 1975. – AAI7609381

[Deb u. a. 2000] Deb, K. ; A., Samir ; Pratap, A. ; Meyarivan, T.: A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. Version: 2000. http://repository.ias.ac.in/83498/. In: *Parallel Problem Solving from Nature PPSN VI*. 2000. – ISSN 10871357, 849–858

[Della Croce u. a. 1995] Della Croce, F. ; Tadei, R. ; Volta, G.: A genetic algorithm for the job shop problem. In: *Computers & Operations Research* 22 (1995), Januar, Nr. 1, 15–24. http://dx.doi.org/10.1016/0305-0548(93)E0015-L. – DOI 10.1016/0305–0548(93)E0015–L. – ISSN 03050548

[Dell'Amico u. Trubian 1993] Dell'Amico, M. ; Trubian, M.: Applying tabu search to the job-shop scheduling problem. In: *Annals of Operations Research* 41 (1993), Nr. 3, S. 231–252

[Draper u. a. 1994] Draper, D. ; Hanks, S. ; Weld, D.: A Probabilistic Model of Action for Least-commitment Planning with Information Gathering. In: *Proceedings of the Tenth International Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1994 (UAI'94). – ISBN 1–55860–332–8, 178–186

[Dubois u. a. 2003] Dubois, D. ; Fargier, H. ; Fortemps, P.: Fuzzy scheduling : Modelling flexible constraints vs . coping with incomplete knowledge. 147 (2003), S. 231–252

[Dubois u. Prade 1999] Dubois, D. ; Prade, H.: Qualitative possibility theory and its applications to constraint satisfaction and decision under uncertainty. In: *International Journal of Intelligent Systems* 14 (1999), Januar, Nr. 1, 45–61. http://dx.doi.org/10.1002/(SICI)1098-111X(199901)14:1<45::AID-INT4>3.0.CO;2-R. – DOI 10.1002/(SICI)1098–111X(199901)14:1<45::AID–INT4>3.0.CO;2–R. – ISSN 0884–8173

[Eppstein 1998] Eppstein, D.: Finding the k shortest paths. In: *SIAM Journal on computing* 28 (1998), Nr. 2, S. 652–673

[Escudero u. a. 1999] Escudero, L. F. ; Galindo, E. ; Garcia, G. ; Gomez, E. ; Sabau, V.: Schumann, a modeling framework for supply chain management under uncertainty. In: *European Journal of Operational Research* 119 (1999), Nr. 1, S. 14–34

[Falkenauer u. Bouffouix 1991] Falkenauer, E. ; Bouffouix, S.: A genetic algorithm for job shop. In: *Robotics and Automation, 1991. ...* (1991). http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=131689

[Fortemps 1997] Fortemps, P: *Jobshop scheduling with imprecise durations: a fuzzy approach*. http://dx.doi.org/10.1109/91.649907. Version: 1997

[Freundt 2004] Freundt, M.: Ein flexibles Modell für die Bauablaufplanung auf Basis von Graphentheorie und Fuzzy-Zahlen. (2004), Nr. November. ISBN 3832235337

[Gerdes u. a. 2013] Gerdes, I. ; Klawonn, F. ; Kruse, R.: *Evolutionäre Algorithmen: Genetische Algorithmen—Strategien und Optimierungsverfahren—Beispielanwendungen*. Springer-Verlag, 2013

[Gg.-Noell-GmbH 1998] Gg.-Noell-GmbH: *Entwicklung und Erprobung von Umschlagtechnik zum Einsatz in einer Mega-Drehscheibe für den kombinierten Verkehr: Schlußbericht*. 1998 https://books.google.de/books?id=u9hiNQEACAAJ

[Goldberg u. Lingle 1985] Goldberg, D. E. ; Lingle, R.: Alleles, loci, and the traveling salesman problem. In: *Proceedings of the first international conference on genetic algorithms and their applications* Lawrence Erlbaum Associates, Publishers, 1985, S. 154–159

[Guang-yuan u. Zhong 1993] Guang-yuan, W. ; Zhong, Q.: Linear programming with fuzzy random variable coefficients. In: *Fuzzy sets and Systems* 57 (1993), Nr. 3, S. 295–311

[Hapke u. a. 1999] Hapke, M. ; Jaszkiewicz, A. ; Słowiński, R.: Fuzzy Multi-Mode Resource-Constrained Project Scheduling with multiple Objectives. Version: 1999. http://dx.doi.org/10.1007/978-1-4615-5533-9_16. In: Weglarz, J. (Hrsg.): *Project Scheduling* Bd. 14. Springer US, 1999. – DOI 10.1007/978–1–4615–5533–9_16. – ISBN 978–1–4613–7529–6, 353-380

[Hapke u. Słowiński 1996] Hapke, M. ; Słowiński, R.: Fuzzy priority heuristics for project scheduling. In: *Fuzzy Sets and Systems* 83 (1996), November, Nr. 3, 291–299. http://dx.doi.org/10.1016/0165-0114(95)00338-X. – DOI 10.1016/0165–0114(95)00338–X. – ISSN 01650114

[Herroelen u. Leus 2005] Herroelen, W. ; Leus, R.: Project scheduling under uncertainty: Survey and research potentials. In: *European Journal of Operational Research* 165 (2005), September, Nr. 2, 289–306. http://dx.doi.org/10.1016/j.ejor.2004.04.002. – DOI 10.1016/j.ejor.2004.04.002. – ISSN 03772217

[Holland 1975] Holland, J.H.: *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. University of Michigan Press, 1975 http://books.google.de/books?id=JE5RAAAAMAAJ. – ISBN 9780472084609

[Horn 1997] Horn, J.: *The nature of niching: Genetic algorithms and the evolution of optimal, cooperative populations*, Citeseer, Diss., 1997

[Jasperneite 2012] Jasperneite, O. J.and N. J.and Niggemann: Systemkomplexität in der Automation beherrschen. In: *atp edition-Automatisierungstechnische Praxis* 54 (2012), Nr. 09, S. 36–45

[Jee u. a. 2007] Jee, K.-W. ; McShan, D. L. ; Fraass, B. A.: Lexicographic ordering: intuitive multicriteria optimization for IMRT. In: *Physics in Medicine and Biology* 52 (2007), Nr. 7, S. 1845

[Jeong u. a. 2007] Jeong, S.-J. ; Lee, C.-G. ; Bookbinder, J. H.: The European freight railway system as a hub-and-spoke network. In: *Transportation Research Part A: Policy and Practice* 41 (2007), Juli, Nr. 6, 523–536. http://dx.doi.org/10.1016/j.tra.2006.11.005. – DOI 10.1016/j.tra.2006.11.005. – ISSN 09658564

[Keng u. Y. Y. Yun 1989] Keng, Naiping ; Y. Y. Yun, D.: A Planning/Scheduling Methodology for the Constrained Resource Problem. In: *In Proceedings IJCAI'89*, 1989, S. 998–1003

[Kobylański u. Kuchta 2007] Kobylański, P. ; Kuchta, D.: A note on the paper by M. A. Al-Fawzan and M. Haouari about a bi-objective problem for robust resource-constrained project scheduling. In: *International Journal of Production Economics* 107 (2007), Juni, Nr. 2, 496–501. http://dx.doi.org/10.1016/j.ijpe.2006.07.012. – DOI 10.1016/j.ijpe.2006.07.012. – ISSN 09255273

[Kolisch u. a. 1999] Kolisch, R. ; Schwindt, C. ; Sprecher, A.: Benchmark Instances for Project Scheduling Problems. Version: 1999. http://dx.doi.org/10.1007/978-1-4615-5533-9_9. In: Weglarz, J. (Hrsg.): *Project Scheduling* Bd. 14. Springer US, 1999. – DOI 10.1007/978–1–4615–5533–9_9. – ISBN 978–1–4613–7529–6, 197-212

[Kouvelis u. a. 2000] Kouvelis, P. ; Daniels, R. L. ; Vairaktarakis, G.: Robust scheduling of a two-machine flow shop with uncertain processing times. In: *Iie Transactions* 32 (2000), Nr. 5, S. 421–432

[Ladjici u. a. 2014] Ladjici, A.A. ; Tiguercha, A. ; Boudour, M.: Nash Equilibrium in a two-settlement electricity market using competitive coevolutionary algorithms. In: *International Journal of Electrical Power & Energy Systems* 57 (2014), S. 148–155

[Leon u. a. 1994] Leon, J. V. ; Wu, D. S. ; Storer, R. H.: Robustness measures and robust scheduling for job shops. In: *IIE transactions* 26 (1994), Nr. 5, S. 32–43

[Leu u. Yang 1999] Leu, S.-S. ; Yang, C.-H.: GA-Based Multicriteria Optimal Model for Construction Scheduling. In: *Journal of Construction Engineering and Management* 125 (1999), Dezember, Nr. 6, 420–427. http://dx.doi.org/10.1061/(ASCE)0733-9364(1999)125:6(420). – DOI 10.1061/(ASCE)0733–9364(1999)125:6(420). – ISSN 0733–9364

[Li u. Ierapetritou 2008] Li, Z. ; Ierapetritou, M.: Process scheduling under uncertainty: Review and challenges. In: *Computers and Chemical Engineering* 32 (2008), S. 715–727. http://dx.doi.org/10.1016/j.compchemeng.2007.03.001. – DOI 10.1016/j.compchemeng.2007.03.001. – ISBN 0098–1354

[Li u. Ierapetritou 2010] Li, Z. ; Ierapetritou, M. G.: Rolling horizon based planning and scheduling integration with production capacity consideration. In: *Chemical Engineering Science* 65 (2010), November, Nr. 22, 5887–5900. http://dx.doi.org/10.1016/j.ces.2010.08.010. – DOI 10.1016/j.ces.2010.08.010. – ISSN 00092509

[Lin u. Chong 2015] Lin, Y.-K. ; Chong, C. S.: Fast GA-based project scheduling for computing resources allocation in a cloud manufacturing system. In: *Journal of Intelligent Manufacturing* (2015), S. 1–13

[Liou u. Wang 1992] Liou, T.-S. ; Wang, M.-J. J.: Ranking fuzzy numbers with integral value. In: *Fuzzy Sets and Systems* 50 (1992), Nr. 3, S. 247–255. http://dx.doi.org/10.1016/0165-0114(92)90223-Q. – DOI 10.1016/0165–0114(92)90223–Q. – ISSN 01650114

[Liu 2006] Liu, B.: A survey of credibility theory. In: *Fuzzy Optimization and Decision Making* 5 (2006), Nr. 4, S. 387–408

[Liu u. Liu 2002] Liu, B. ; Liu, Y.-K.: Expected value of fuzzy variable and fuzzy expected value models. In: *Fuzzy Systems, IEEE Transactions on* 10 (2002), Nr. 4, S. 445–450

[Ludewig 2002] Ludewig, J.: Modelle Im Software Engineering - Eine Einführung Und Kritik. In: *Modellierung in Der Praxis - Modellierung für Die Praxis*, GI, 2002 (Modellierung 2002). – ISBN 3–88579–342–3, 7–22

[Maravelias u. Sung 2009] Maravelias, C. T. ; Sung, C.: Integration of production planning and scheduling: Overview, challenges and opportunities. In: *Computers and Chemical Engineering* 33 (2009), S. 1919–1930. http://dx.doi.org/10.1016/j.compchemeng.2009.06.007. – DOI 10.1016/j.compchemeng.2009.06.007. – ISBN 0098–1354

[Masmoudi u. Haït 2013] Masmoudi, M. ; Haït, A.: Project scheduling under uncertainty using fuzzy modelling and solving techniques. In: *Engineering Applications of Artificial Intelligence* 26 (2013), Januar, Nr. 1, 135–149. http://dx.doi.org/10.1016/j.engappai.2012.07.012. – DOI 10.1016/j.engappai.2012.07.012. – ISSN 09521976

[Mattfeld 1996] Mattfeld, D. C.: *Evolutionary search and the job shop : investigations on genetic algorithms for production scheduling*. 1996 (November). – ix, 152 p. S. http://dx.doi.org/10.1007/978-3-662-11712-5. http://dx.doi.org/10.1007/978-3-662-11712-5. – ISBN 3790809179

[Méndez u. a. 2006] Méndez, C. A. ; Cerdá, J. ; Grossmann, I. E. ; Harjunkoski, I. ; Fahl, M.: State-of-the-art review of optimization methods for short-term scheduling of batch processes. In: *Computers & Chemical Engineering* 30 (2006), Mai, Nr. 6-7, 913–946. http://dx.doi.org/10.1016/j.compchemeng.2006.02.008. – DOI 10.1016/j.compchemeng.2006.02.008. – ISSN 00981354

[Merz 2000] Merz, P.: *Memetic Algorithms for Combinatorial Optimization Problems*, Diss., 2000

[Möller u. Beer 2004] Möller, B. ; Beer, M.: *Fuzzy randomness: uncertainty in civil engineering and computational mechanics*. Springer Science & Business Media, 2004

[Mori u. Tseng 1997] Mori, M. ; Tseng, C.C.: A genetic algorithm for multi-mode resource constrained project scheduling problem. In: *European Journal of Operational Research* 100 (1997), Juli, Nr. 1, 134–141. http://dx.doi.org/10.1016/S0377-2217(96)00180-4. – DOI 10.1016/S0377–2217(96)00180–4. – ISSN 03772217

[Mulvey u. a. 1995] Mulvey, J. M. ; Vanderbei, R. J. ; Zenios, S. A.: Robust optimization of large-scale systems. In: *Operations research* 43 (1995), Nr. 2, S. 264–281

[Negoita u. a. 1978] Negoita, C. ; Zadeh, L. ; Zimmermann, H.: Fuzzy sets as a basis for a theory of possibility. In: *Fuzzy sets and systems* 1 (1978), S. 3–28

[Nowicki u. Smutnicki 1996] Nowicki, E. ; Smutnicki, C.: A fast tabu search algorithm for the permutation flow-shop problem. In: *European Journal of Operational Research* 91 (1996), Nr. 1, S. 160–175

[Ocker 2010] Ocker, D.: *Unscharfe Risikoanalyse strategischer Ereignisrisiken*. Lang, Peter, Gmbh, Internationaler Verlag Der Wissenschaften, 2010

[Osman u. Potts 1989] Osman, I. ; Potts, C.: Simulated annealing for permutation flow-shop scheduling. In: *Omega* 17 (1989), Nr. 6, S. 551–557

[Ovacik u. Uzsoy 1995] Ovacik, I. M. ; Uzsoy, R.: Rolling horizon procedures for dynamic parallel machine scheduling with sequence-dependent setup times. In: *International Journal of Production Research* 33 (1995), Nr. 11, S. 3173–3192. http://dx.doi.org/10.1080/00207549508904867. – DOI 10.1080/00207549508904867. – ISSN 0020–7543

[Ozdamar 1999] Ozdamar, L.: A genetic algorithm approach to a general category project scheduling problem. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 29 (1999), Nr. 1, 44–59. http://dx.doi.org/10.1109/5326.740669. – DOI 10.1109/5326.740669. – ISSN 1094–6977

[Potter u. De Jong 1994] Potter, M. A. ; De Jong, K. A.: A cooperative coevolutionary approach to function optimization. In: *Parallel problem solving from nature—PPSN III*. Springer, 1994, S. 249–257

[Qassim 2012] Qassim, R.: Integrated Process Planning and Scheduling and Multimode Resource Constrained Project Scheduling: Ship Block Assembly Application. In: *cdn.intechopen.com* (2012). http://cdn.intechopen.com/pdfs/36416/InTech-Integrated_process_planning_and_scheduling_and_multimode_resource_constrained_project_scheduling_ship_block_assembly_application.pdf

[Radcliffe 1991] Radcliffe, N. J.: Forma analysis and random respectful recombination. In: *Proceedings of the 4^{th} International Conference on Genetic Algorithms* (1991), S. 222–229

[REFA Verband für Arbeitsstudien und Betriebsorganisation e. V. 1984] REFA Verband für Arbeitsstudien und Betriebsorganisation e. V.: *Methodenlehre des Arbeitsstudiums*. 7. Hanser, 1984. – ISBN 3–446–14234–7

[Rixen u. Kopfer 1994] Rixen, I. ; Kopfer, H.: Ein Genetischer Algorithmus für das Job-Shop-Scheduling-Problem von. In: *University of Bremen, Chair of Logistics, Bremen* (1994). http://www.econbiz.de/archiv/hb/uhb/logistik/genetischer_algorithmus_jssp.pdf

[Rotter 2004] Rotter, H.: New operating concepts for intermodal transport: The mega hub in Hanover/Lehrte in Germany. In: *Transportation Planning and Technology* 27 (2004), Nr. 5, 347-365. http://dx.doi.org/10.1080/0308106042000273022. – DOI 10.1080/0308106042000273022

[Rudolph 1996] Rudolph, G.: Convergence of evolutionary algorithms in general search spaces. In: *Evolutionary Computation, 1996., Proceedings of ...* (1996), 50–54. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=542332. ISBN 0780329023

[Sacerdoti 1975] Sacerdoti, E. D.: The Nonlinear Nature of Plans. In: *Proceedings of the 4th International Joint Conference on Artificial Intelligence - Volume 1*. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1975 (IJCAI'75), 206–214

[Sadeh u. a. 1993] Sadeh, N. ; Otsuka, S. ; Schnelbach, R.: Predictive and reactive scheduling with the Micro-Boss production scheduling and control system. In: *Proceedings, IJCAI-93 Workshop on Knowledge-Based Production Planning, Scheduling and Control*, 1993

[Sakawa u. Kubota 2000] Sakawa, M. ; Kubota, R.: Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms. In: *European Journal of Operational Research* 120 (2000), Nr. 2, S. 393–407. http://dx.doi.org/10.1016/S0377-2217(99)00094-6. – DOI 10.1016/S0377–2217(99)00094–6. – ISSN 03772217

[Schulz 2002] Schulz, K.: *(Un-)sicherheiten bei der Okosystemmodellierung*. Vorlesungsskript, 2002

[Schwefel 1975] Schwefel, H.-P.: *Evolutionsstrategie und numerische Optimierung*, Technische Universität Berlin, Diss., 1975

[Shao u. a. 2009] Shao, X. ; Li, X. ; Gao, L. ; Zhang, C.: Integration of process planning and scheduling—A modified genetic algorithm-based approach. In: *Computers & Operations Research* 36 (2009), Juni, Nr. 6, 2082–2096. http://dx.doi.org/10.1016/j.cor.2008.07.006. – DOI 10.1016/j.cor.2008.07.006. – ISSN 03050548

[Smith 2007] Smith, J. E.: Coevolving memetic algorithms: a review and progress report. In: *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society* 37 (2007), Februar, Nr. 1, 6–17. http://www.ncbi.nlm.nih.gov/pubmed/17278554. – ISSN 1083–4419

[Spears 1995] Spears, W. M.: Adapting Crossover in Evolutionary Algorithms. In: *Evolutionary programming*, 1995, S. 367–384

[Stefansson u. a. 2006] Stefansson, H ; Shah, N ; Jensson, P: Multiscale planning and scheduling in the secondary pharmaceutical industry. In: *AIChE journal* 52 (2006), Nr. 12, 4133–4149. http://dx.doi.org/10.1002/aic. – DOI 10.1002/aic

[Stork 2001] Stork, F.: Stochastic resource-constrained project scheduling. (2001)

[Syswerda 1991] Syswerda, G.: Order-Based Genetic Algorithms and the Graph Coloring Problem. In: *Handbook of genetic algorithms* (1991)

[T. 1972] T., Robert: Depth first search and linear graph algorithms. In: *SIAM Journal on Computing* (1972)

[Till u. a. 2007] Till, J. ; Sand, G. ; Urselmann, M. ; Engell, S.: A hybrid evolutionary algorithm for solving two-stage stochastic integer programs in chemical batch scheduling. In: *Computers & Chemical Engineering* 31 (2007), Nr. 5–6, 630 - 647. http://dx.doi.org/http://dx.doi.org/10.1016/j.compchemeng.2006.09.003. – DOI http://dx.doi.org/10.1016/j.compchemeng.2006.09.003. – ISSN 0098–1354

[Turksen 1991] Turksen, I.B.: Measurement of membership functions and their acquisition. In: *Fuzzy Sets and Systems* 40 (1991), Nr. 1, 5 - 38. http://dx.doi.org/http://dx.doi.org/10.1016/0165-0114(91)90045-R. – DOI http://dx.doi.org/10.1016/0165–0114(91)90045–R. – ISSN 0165–0114

[Van Laarhoven u. a. 1992] Van Laarhoven, P. ; Aarts, E. ; Lenstra, J. K.: Job shop scheduling by simulated annealing. In: *Operations research* 40 (1992), Nr. 1, S. 113–125

[Vent 1973] Vent, W.: Rechenberg, Ingo, Evolutionsstrategie — Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. 170 S. mit 36 Abb. Frommann-Holzboog-Verlag. Stuttgart 1973. Broschiert. In: *Feddes Repertorium* 86 (1973), Nr. 5, 337–337. http://dx.doi.org/10.1002/fedr.19750860506. – DOI 10.1002/fedr.19750860506. – ISSN 1522–239X

[Weicker 2007] Weicker, K.: *Evolutionäre Algorithmen (2. Auflage)*. Stuttgart : Teubner, 2007

[Wright 1932] Wright, S.: The Roles of Mutation, Inbreeding, Crossbreeding, and Selection in Evolution. In: *Proceedings of the Sixth International Congress of Genetics 1*, 1932, S. 356–366

[Yen 1971] Yen, J. Y.: Finding the k shortest loopless paths in a network. In: *management Science* 17 (1971), Nr. 11, S. 712–716

[Zadeh 1965] Zadeh, L.A.: Fuzzy sets. In: *Information and Control* 8 (1965), Nr. 3, 338 - 353. http://dx.doi.org/http://dx.doi.org/10.1016/S0019-9958(65)90241-X. – DOI http://dx.doi.org/10.1016/S0019–9958(65)90241–X. – ISSN 0019–9958

[Zhao u. Liu 2006] Zhao, S. ; Liu, Y.: GA-based resource leveling optimization for construction project. In: *Machine Learning and ...* (2006), Nr. August, 13–16. http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4028460. ISBN 1424400600

[Zimmermann 2000] Zimmermann, H.-J.: An application-oriented view of modeling uncertainty. In: *European Journal of Operational Research* 122 (2000), April, Nr. 2, 190–198. http://dx.doi.org/10.1016/S0377-2217(99)00228-3. – DOI 10.1016/S0377–2217(99)00228–3. – ISSN 03772217