Driftorientierte Einsteuerung von Aufträgen auf Variantenfließlinien in der Automobilindustrie

Der Fakultät für Wirtschaftswissenschaften der
Universität Paderborn
zur Erlangung des akademischen Grades
Doktor der Wirtschaftswissenschaften
- Doctor rerum politicarum -
vorgelegte Dissertation
von
Dipl.-Math. oec Christian Franz
geboren am 11. Oktober 1985 in Iserlohn

2014

Inhaltsverzeichnis

Abbildungsverzeichnis

Algorithmenverzeichnis

Tabellenverzeichnis

1 Einleitung und Motivation

Das anhaltende Wachstum der Produktvielfalt in der Automobilindustrie bringt viele Herausforderungen mit sich. Insbesondere im Premiumsegment erreicht die Anzahl der aus Kombination von Modellen, Typen und Ausstattungen bestellbaren Fahrzeuge immer neue Rekorde. Die unterschiedlichen Wünsche der Kunden führen dazu, dass aktuell kaum zwei vollkommen gleiche Autos produziert werden. Um die Montagelinien unter diesen Umständen gut auslasten zu können, müssen diese auf eine Vielzahl unterschiedlicher Fahrzeuge ausgelegt sein.

Aufgrund der beschriebenen Heterogenität, der auf einer Montagelinie produzierten Fahrzeuge, hat die Sequenz, in der diese montiert werden, einen großen Einfluss auf die Arbeitsbelastung der Werker. Eben diese Belastung in einem geeigneten Fenster zu halten, ist ein entscheidender Faktor für eine kostengünstige und qualitativ hochwertige Produktion. Wenn die Arbeitsbelastung der Werker gering ist, ist dies gleichbedeutend mit einer geringen Auslastung der Montagelinie, was zu hohen Kosten pro Fahrzeug führt. Andererseits ist auch eine Überlastung der Werker zu vermeiden, da diese durch geeignete Maßnahmen kompensiert werden müsste. Schnellere – und damit oft qualitativ schlechtere – Arbeit der Werker, Einsatz von Unterstützern, Nacharbeit und dadurch zusätzlich entstehende Kosten oder ein Stillstand des Bandes und eine damit verbundene Senkung des Produktionsvolumens sind mögliche Konsequenzen von Überlastungen, die allesamt negative Auswirkungen haben.

Mit dem Ziel die Werkerbelastung möglichst ausgeglichen zu gestalten, wird die Reihenfolge der Fahrzeuge schon einige Tage vor Produktionsstart geplant. Wegen kurzfristig auftretender Probleme – wie z. B. fehlender Teile – kann diese Sequenz selten wie geplant bearbeitet werden. Einige Aufträge müssen auf Warteposition zurückgestellt werden und später, wenn die Teile nachgeliefert worden sind, müssen neue Termine

für deren Produktion bestimmt werden. Die Aufgabe der Einsteuerung liegt nun darin, trotz aller unvorhergesehenen Schwierigkeiten eine neue und möglichst gute, d. h. Überlastungen vermeidende, Sequenz von Fahrzeugen in die Montage einzusteuern.

Die große Herausforderung ist dabei der Umgang mit der Dynamik des Systems. Zum einen können zu jeder Zeit unvorhergesehene Ereignisse Änderungen an der geplanten Produktionsreihenfolge erzwingen. Zum anderen muss innerhalb eines Takts darauf reagiert werden können, da die Produktion unabhängig von Problemen in anderen Prozessen nicht angehalten werden soll.

Das Hauptziel dieser Arbeit ist es, Methoden zu entwickeln, mit denen die Einsteuerung schnell und sicher auf alle möglichen Ereignisse reagieren kann. Dabei liegt der Fokus darauf, die Aufträge in einer Reihenfolge in die Montage zu leiten, die zu möglichst wenigen Überlastungssituationen bei den Werkern führt.

Diese Arbeit gliedert sich in acht Kapitel, von denen die ersten vier einen Überblick über die betrachtete Thematik geben sollen. Es folgen im Hauptteil drei Kapitel, in denen verschiedene Facetten des Problems analysiert werden, bevor im letzten Kapitel die Resultate zusammengefasst werden.

Im folgenden Kapitel wird die Problemstellung beschrieben und der Hintergrund erläutert. Danach wird in Kapitel 3 auf existierende Forschungsergebnisse in diesem Bereich eingegangen. Davon lassen sich die zu leistenden Arbeiten herleiten, die neben einer Abgrenzung der betrachteten Probleme in Kapitel 4 ausgeführt werden.

Der Kern der Arbeit beschäftigt sich zunächst mit einem statischen Problem in Kapitel 5. Dabei soll eine Menge von Aufträgen in eine gegebene Sequenz eingegliedert werden. Danach wird in Kapitel 6 das Problem ganzheitlich mitsamt seiner Dynamik betrachtet. Eine explizite Analyse der Anzahl der benötigten Unterstützer, die mit einer Betrachtung einzelner Schichten einhergeht, wird in Kapitel 7 durchgeführt.

Zuletzt werden die Ergebnisse in Kapitel 8 zusammengefasst und ein kurzer Ausblick auf offene Fragen gegeben.

2 Problemstellung

In diesem Kapitel wird das untersuchte Problem zunächst in den Produktions- und den Planungsprozess eingeordnet. Danach werden die verwendeten Objekte definiert und die Problemstellung abgegrenzt. Nach einer Nennung von grundlegenden Annahmen wird die Wahl der Zielfunktion erläutert.

2.1 Einordnung in den Produktions- und Planungsprozess

Für eine exakte Einordnung der Fragestellung werden zwei Perspektiven benötigt. Zum einen kann das Problem in dem Produktionsprozess lokalisiert werden. Zum anderen muss es im Rahmen des Planungsprozesses abgegrenzt werden.

Zunächst wird der Produktionsprozess betrachtet. Diese Arbeit beschäftigt sich mit der Reihenfolge, in der die Aufträge in die Montage eingesteuert werden. Die Herstellung und die Bearbeitung der Karosserien im Rohbau und in der Lackiererei werden nicht näher betrachtet. Ausgangspunkt für das vorliegende Problem ist der Bestand der Karosserien im Sortierer. Dieser ist ein großes Lager, in dem die lackierten Karosserien zwischengelagert werden und er dient als Puffer, dem die Karosserien in beliebiger Reihenfolge entnommen werden können. Es wird realistisch angenommen, dass dieser groß genug ist, um die Produktionsmenge einer halben bis ganzen Schicht zu lagern, sodass hierdurch keine direkte Limitierung entsteht. Es kann natürlich vorkommen, dass benötigte Karosserien nicht verfügbar sind. Die Ursache dafür ist jedoch nicht die Art oder Größe des Sortierers, sondern liegt in betriebsbedingten Störungen bei vorausgegangenen Prozessen, die dann in Form von Sperrungen in die Problemstellung

eingehen.

Der Kern dieser Arbeit beschäftigt sich mit der Frage, in welcher Reihenfolge die Karosserien aus dem Sortierer entnommen werden und anschließend durch die Montagelinie laufen. Mit der Auslagerung werden die Karosserien fest mit einem Auftrag verknüpft und durchlaufen die komplette Montage in unveränderbarer Reihenfolge. Im Anschluss an die Montage werden alle relevanten Systeme eines jeden Fahrzeugs geprüft. Dabei findet eine Verteilung der Fahrzeuge auf mehrere Prüfstände statt. Diese Verteilung und die Reihenfolge, in der die Fahrzeuge schließlich den Prüfbereich verlassen, werden in dieser Arbeit ebenfalls nicht näher betrachtet. Relevant für die Bewertung eines Verfahrens zur Festlegung der Produktionsreihenfolge in der Montage bzw. für eine Montagesequenz sind die Arbeitsbelastungen der Werker entlang der Endmontagelinie.

Nun wird das Problem aus planungstechnischer Sicht beschrieben. Die relevanten Prozesse sind in Abbildung 2.1 grafisch dargestellt.

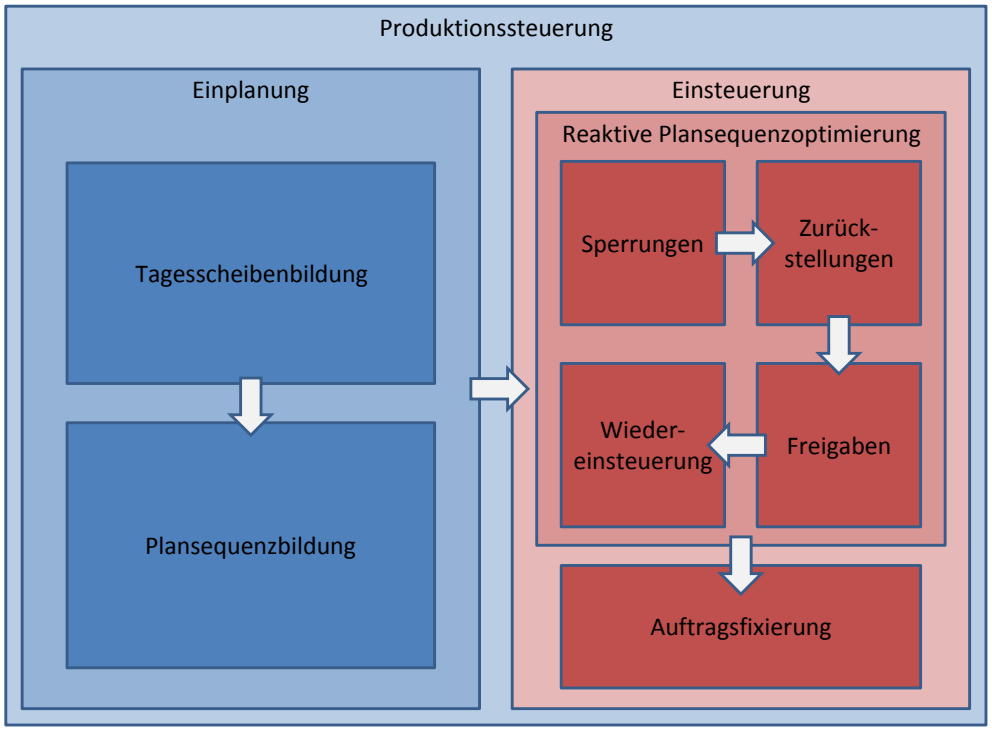


Abbildung 2.1: Darstellung der Prozesse der Produktionssteuerung

Schon einige Tage vor Montagebeginn werden im Rahmen der Tagesscheibenbildung

die Aufträge bestimmt, die an einem bestimmten Tag in die Montage einlaufen sollen. In einem weiteren Schritt wird eine sinnvolle Reihenfolge dieser Aufträge ermittelt. Diese beiden Prozesse bilden die Einplanung und das Resultat hieraus ist eine Plansequenz, die die Grundlage für das in dieser Arbeit untersuchte Problem bildet.

Im Optimalfall kann diese Reihenfolge unverändert übernommen werden und im Rahmen der Einsteuerung müssen keine weiteren Entscheidungen getroffen werden. In der Praxis sind jedoch Abweichungen von der Plansequenz an der Tagesordnung. Aufgrund von Lieferausfällen und damit verbundener fehlender Teile oder wegen Problemen bei den vorgelagerten Prozessen und fehlender Karosserien können Fahrzeuge häufig nicht wie geplant produziert werden. Der kurzfristige Umgang mit diesen Änderungen und die Entscheidung darüber, welches Fahrzeug tatsächlich in jedem Takt neu in die Montage einläuft, erfolgt im Rahmen der Einsteuerung, die das Untersuchungsobjekt dieser Arbeit bildet. Diese Arbeit beschränkt sich dabei auf eine reaktive Optimierung der Produktionssequenz. D. h. die Plansequenz wird nur infolge von nicht geplanten Ereignissen, die eine Veränderung der Sequenz erzwingen, modifiziert. Die genauen Abläufe innerhalb der Einsteuerung werden im nächsten Abschnitt detailliert beschrieben.

2.2 Beschreibung der relevanten Objekte und Begriffe

Die wesentlichen Objekte der Untersuchungen in dieser Arbeit sind *Aufträge*, die in eine optimale Produktionsreihenfolge gebracht werden sollen. Da in dieser Arbeit der praktische Bezug zur Automobilindustrie permanent vorhanden ist, wird der Begriff *Fahrzeug* auch als Synonym für einen Auftrag verwendet.

Ausgangspunkt für die in dieser Arbeit analysierte Problemstellung ist das Auftreten von unvorhergesehenen Ereignissen, die dazu führen, dass bestimmte Aufträge vorübergehend nicht montiert werden können. Diese werden in Form von *Sperrungen* beschrieben. Die Ursache von Sperrungen kann ein Problem in vorgelagerten Prozessen sein, das z. B. dazu führt, dass eine benötigte Karosserie nicht verfügbar ist. Außerdem können nicht rechtzeitig gelieferte Teile zu Materialengpässen führen, aus denen

ebenfalls Sperrungen resultieren können. Wenn diese Sperrungen dazu führen, dass ein geplanter Termin für den Montagebeginn nicht eingehalten werden kann, erfolgt eine *Zurückstellung* des betroffenen Auftrags. An dieser Stelle sei darauf hingewiesen, dass gesperrte Aufträge nicht zwangsläufig zurückgestellt werden. Wenn eine Sperrung vor dem geplanten Einsteuerungstermin aufgehoben wird, wird der entsprechende Auftrag nicht zurückgestellt, sondern wie geplant bearbeitet. Eine *Freigabe* eines zurückgestellten Auftrags erfolgt, wenn die Ursache für die jeweilige Sperrung nicht mehr vorliegt. Bei jeder Freigabe von Fahrzeugen kann ein wichtiges Teilproblem formuliert werden, das als *Wiedereinsteuerung* bezeichnet wird. Dabei werden den freigegebenen Aufträgen, in Abhängigkeit von den Umständen zum jeweiligen Zeitpunkt, neue Positionen in der Plansequenz zugeordnet.

Die beschriebenen Abläufe verdeutlichen die dynamische Natur des Problems. In jedem Takt können neue unvorhergesehene Ereignisse geschehen und in jedem Takt muss eine Entscheidung darüber getroffen werden, welches Fahrzeug als nächstes produziert wird. Diese Entscheidung wird als *Auftragsfixierung* bezeichnet und markiert das Ende der Einsteuerung. Aufträge, die einmal fixiert worden sind, durchlaufen die gesamte Montagelinie in der entsprechenden Reihenfolge.

Aus den eben genannten Prozessen lassen sich Klassen ableiten, in die die Aufträge eingeordnet werden können. Die Menge aller zu produzierenden Fahrzeuge kann auf folgende Weise in fünf Mengen partitioniert werden:

1. Die Fahrzeuge, die bereits an einer Position in der Produktionssequenz *fixiert* worden sind. Diese können schon in die Montage eingelaufen sein oder die jeweilige Karosserie wartet auf die Auslagerung aus dem Sortierer. Ein Verschieben dieser Fahrzeuge ist nicht mehr möglich.

2. Die Fahrzeuge, die wie *geplant* auf das Einlaufen in die Montage warten.

3. Die Fahrzeuge, die *gesperrt* sind. Diese Fahrzeuge befinden sich noch in der Plansequenz, aber zurzeit sind nicht alle notwendigen Teile verfügbar, um diese Fahrzeuge zu montieren und solange können diese Fahrzeuge nicht an einer Position der Produktionssequenz fixiert werden.

4. Die Fahrzeuge, die *zurückgestellt* sind. Diese Fahrzeuge waren gesperrt, als sie

fixiert werden sollten und konnten deshalb nicht wie geplant ihre Position in der Produktionssequenz einnehmen. Die Ursache der jeweiligen Sperrung ist noch nicht behoben.

5. Die Fahrzeuge, die zurückgestellt waren, aber jetzt wieder für die Montage *freigegeben* sind. Die Ursache der jeweiligen Sperrung ist behoben.

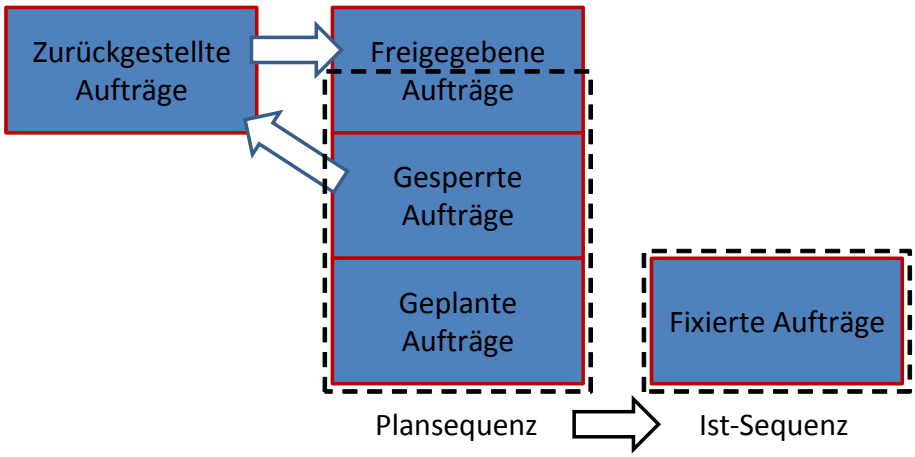


Abbildung 2.2: Schematische Einteilung der Aufträge

Entsprechend der Abbildung 2.2 befinden sich alle Fahrzeuge zu jedem Zeitpunkt in einer Sequenz oder ungeordnet in einer weiteren Menge von Aufträgen. Die durch das rechte gestrichelte Rechteck markierte Sequenz ist die *Ist-Sequenz*, in der alle fixierten Aufträge in ihrer Produktionsreihenfolge angeordnet sind. Die andere Sequenz, die durch das linke gestrichelte Rechteck abgegrenzt ist, ist die *Plansequenz*, die alle geplanten und gesperrten, aber nicht zurückgestellten Aufträge enthält. Zusätzlich können je nach verwendeter Methode auch freigegebene Aufträge wieder neue Plätze in dieser Plansequenz erhalten haben. Die zurückgestellten Aufträge bilden eine eigenständige Menge von ungeordneten Fahrzeugen. Ebenso befinden sich die freigegebenen Fahrzeuge, die noch nicht wieder in die Plansequenz eingeordnet worden sind, in einer weiteren ungeordneten Menge von Fahrzeugen.

Die Fixierung eines Auftrags muss regelmäßig - entsprechend der Produktionsabstände - erfolgen. Der zeitliche Abstand zwischen zwei Fahrzeugen wird als *Taktzeit* bezeichnet. Diese gibt somit auch die Zeiten der Auftragsfixierungen vor. Es wird in der Folge angenommen, dass die Geschwindigkeit, mit der sich ein Montageband bewegt,

konstant ist. Somit können Zeit- und Längenmaße äquivalent verwendet werden. Da die Taktzeit die zentrale Größe ist, die im gesamten Montageprozess einheitlich ist, wird in der Folge die Verwendung von Zeitmaßen bevorzugt.

Die Produktionszeiten lassen sich in *Schichten* unterteilen. Dies sind Produktionsintervalle, in denen eine gewisse Menge an Werkern arbeitet. Üblicherweise unterteilt sich ein Tag in drei Schichten, in denen gearbeitet werden kann. Somit dauert eine Schicht acht Stunden abzüglich Pausen. Aus der verbleibenden Arbeitszeit und der Taktzeit lässt sich folglich die Anzahl der Produktionstakte einer Schicht und somit die Stückzahl der eingesteuerten Fahrzeuge bestimmen.

Die Fahrzeuge durchlaufen auf ihrem Weg durch die Montage mehrere *Montagebänder*, die in aufeinanderfolgende *Stationen* aufgeteilt sind. Die Gesamtheit aller Bänder oder Stationen wird als *Montagelinie* bezeichnet. Eine beispielhafte Darstellung eines Abschnitts eines Montagebands ist in Abbildung 2.3 zu finden. An jeder Station können mehrere *Arbeitsplätze* angesiedelt sein. Jedem Arbeitsplatz sind bestimmte *Arbeitsvorgänge*, deren Ausführung eine festgelegte Arbeitszeit benötigt, zugewiesen. Diese Arbeitsvorgänge werden dort von einem oder mehreren *Werkern* verrichtet und können sich in Abhängigkeit von den jeweiligen Aufträgen unterscheiden. An Arbeitsplätzen mit mehreren Werkern können diese entweder gemeinsam (*Teamstation*) oder abwechselnd (*Mehrtakter*) arbeiten. Abwechselnd bedeutet in diesem Zusammenhang, dass sie die Fahrzeuge aufteilen und trotzdem gleichzeitig, aber an verschiedenen Fahrzeugen arbeiten.

Alle nachfolgend genannten Begriffe, die sich auf eine räumliche Größe beziehen, werden - wie zuvor beschrieben - mit einer Zeiteinheit bewertet. Dieser Wert entspricht stets dem jeweiligen Zeitäquivalent, den das Band bei gegebener Geschwindigkeit benötigt, um sich die jeweilige Strecke zu bewegen. Die Länge einer Station entspricht also der Taktzeit. Folglich trifft an jeder Station ein Fahrzeug pro Takt ein.

Die Werker dürfen innerhalb eines festgelegten *Arbeitsbereichs* arbeiten. Dieser besteht aus dem *Stationsbereich* und dem *Driftbereich*. Die Länge des Stationsbereichs entspricht dem Produkt der Anzahl der abwechselnd arbeitenden Werker und der Taktzeit. Die Länge des Driftbereichs wird individuell für jeden Arbeitsplatz festgelegt. Dabei muss gewährleistet werden, dass die Tätigkeiten auch im Driftbereich

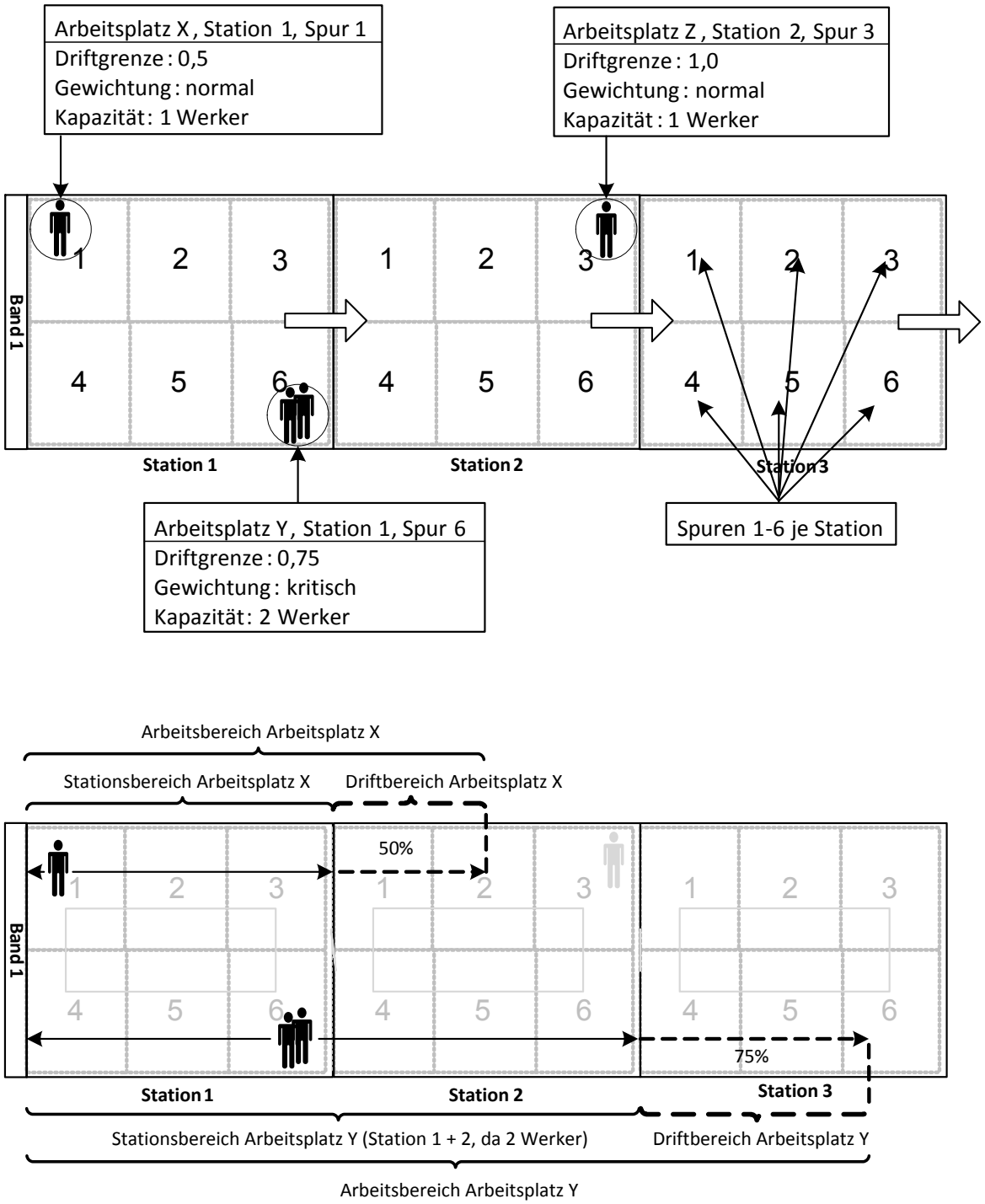


Abbildung 2.3: Schematische Darstellung von einem Montagebandabschnitt (vgl. [Alt09])

ausgeführt werden können und dadurch keine Behinderung anderer Arbeitsplätze vorliegt. Wenn Arbeiten innerhalb des Driftbereichs erledigt werden, wird dies als *Drift* bezeichnet. Das Ende des Driftbereichs wird *Driftgrenze* genannt. Diesem kann ein Zeitwert zugeordnet werden, der der Länge des gesamten Arbeitsbereichs entspricht. Wenn nicht alle Arbeitsvorgänge innerhalb des Arbeitsbereichs abgearbeitet werden können, wird dies als *Driftüberschreitung* bezeichnet. Da dies nicht zulässig ist, werden in diesen Fällen sämtliche Arbeitsvorgänge des entsprechenden Auftrags an dem Arbeitsplatz von einem *Unterstützer* oder *Springer* übernommen. Der eigentliche Werker kann dann direkt an seinem nächsten Fahrzeug arbeiten.

Die Springer sind vielseitig qualifizierte Arbeitskräfte, die an einer Menge von Arbeitsplätzen aushelfen können. Dementsprechend sind die Arbeitsplätze in Gruppen eingeteilt, innerhalb derer Springer überall einspringen können. Wenn ein Springer nicht benötigt wird, kann er anderen organisatorischen oder vorbereitenden Tätigkeiten nachgehen, die hier nicht näher betrachtet werden.

Wenn ein Springer am Montageband aushilft, wird von einem *Springereinsatz* gesprochen. Obwohl dieser eine Driftüberschreitung verhindert, werden bei der Bewertung von Produktionssequenzen die beiden Ausdrücke synonym verwendet.

2.3 Problemdifferenzierung

Das vorliegende Problem kann auf zwei unterschiedliche Weisen angegangen werden. Zum einen kann der Fokus auf das als Wiedereinsteuerung bezeichnete Teilproblem gelegt werden. In diesem Fall ist ein statisches Optimierungsproblem zu lösen, welches bei jeder Freigabe von Fahrzeugen unabhängig voneinander neu aufgestellt wird.

Alternativ dazu können die beschriebenen Prozesse gemeinsam und über einen Produktionszeitraum hinweg in ihrer dynamischen Struktur erfasst und ein dynamisches Optimierungsproblem formuliert werden.

Das statische Problem reduziert sich darauf, aus der Plansequenz und den freigegebenen Aufträgen eine neue Plansequenz zu bilden, die möglichst optimal ist. Dabei wird

ein Horizont, also eine Anzahl an Takten, vorgegeben, innerhalb derer die Aufträge eingegliedert werden müssen.

Im statischen Problem kann auf das Betrachten von *Fälligkeitsterminen*, d. h. Positionen, an denen ein Fahrzeug spätestens fixiert werden muss, verzichtet werden. Die Fahrzeuge, aus denen eine Sequenz gebildet werden muss, sind gegeben und damit ist implizit ein Fälligkeitstermin für alle Fahrzeuge verbunden. Dieser entspricht der letzten betrachteten Position.

Diese Sichtweise kann aufgeweicht werden, indem unterschiedliche Fälligkeitstermine erlaubt werden. Dabei kann entweder das Platzieren eines wiedereinzusteuernden Auftrags auf eine der ersten Positionen erzwungen werden oder es kann erlaubt werden, ein Fahrzeug gar nicht in das betrachtete Intervall zu integrieren, wenn der Fälligkeitstermin weit in der Zukunft liegt. Dazu kann die Menge der freigegebenen Fahrzeuge in zwei Gruppen unterteilt werden. Ein Teil der Aufträge muss direkt in die Plansequenz integriert werden. Der andere Teil kann integriert werden oder wird ansonsten auf eine neue Wiedereinsteuerung in einem späteren Takt verschoben. Diese Varianten werden jedoch bei der Untersuchung des statischen Problems in dieser Arbeit nicht verfolgt.

Mit den beschriebenen Veränderungen des Problems ist der Übergang zum dynamischen Problem fließend. Dieses stellt ein Online-Problem mit Echtzeitcharakter dar. Eine Vereinfachung dieses Problems ist eine rollierende Anwendung des statischen Problems, das in jedem Takt gelöst wird. Allerdings stellt dies nur eine sehr beschränkte Variante im Umgang mit der Dynamik dar.

Im Verlauf einer Schicht können Fahrzeuge gesperrt werden. Zu Beginn eines jeden Takts können neue Sperrdaten verfügbar sein, die in die Entscheidung über die nächste Auftragsfixierung am Ende des Takts einbezogen werden müssen. Die gesperrten Fahrzeuge können, solange diese Sperrung besteht, nicht fixiert werden, sondern werden, wenn sie die erste Stelle in der Plansequenz erreicht haben, zurückgestellt. Wenn Sperrungen wieder aufgehoben werden, können die zuvor zurückgestellten Fahrzeuge im Sinne des statischen Problems wieder in die Sequenz eingefügt werden. Für das dynamische Problem stellt die Einhaltung der Fälligkeitstermine ein wichtiges Kriterium dar.

Die Analyse des dynamischen Problems ist nicht auf ein bestimmtes Produktionsintervall beschränkt, sondern sollte über Schichten und Tage hinweg gehen. Dadurch wird es zusätzlich möglich, relevante Größen - wie die Anzahl der Werker in einer Schicht - zu bestimmen, sodass dieses Problem hinsichtlich zusätzlicher Zielfunktionen betrachtet werden kann.

Unabhängig von der Zielsetzung beim dynamischen Problem muss die Entscheidung über die Fixierung eines Auftrags getroffen werden. Die relevanten Größen können sich in jedem Takt ändern. Dennoch muss nach jedem Takt ein Fahrzeug bestimmt werden, das unter den aktuellen Umständen die Produktionssequenz sinnvoll fortsetzt.

2.4 Grundlegende Annahmen

Die Zuordnung von Werkern und Arbeitsvorgängen zu den Arbeitsplätzen und die Abhängigkeiten untereinander werden hier nicht näher betrachtet. Diese Problematik wird als vorher abgehandelt angenommen und deren Ergebnisse gehen als Eingangsgrößen in das vorliegende Problem ein. Wenn zwischen zwei aufeinander folgenden Arbeitsplätzen eine Abhängigkeit besteht, die Überlagerungen der Tätigkeiten ausschließt, so geht diese Information lediglich über die Driftgrenzen in das betrachtete Problem ein. So führt eine unzulässige Behinderung an der direkt folgenden Station dazu, dass ein Arbeitsplatz praktisch keinen Driftbereich zugeteilt bekommt. Auf eine explizite Betrachtung der Abhängigkeiten zwischen einzelnen Arbeitsplätzen wie bei Bautista und Suarez [BS09] wird in dieser Arbeit verzichtet.

Die Arbeitsorganisation bildet eine wesentliche Rahmenbedingung für die Formulierung des Problems. Es wird hier die Annahme getroffen, dass sämtliche Arbeiten, die nicht von den standardmäßigen Werkern innerhalb des Arbeitsbereichs erledigt werden können, durch Springer ausgeglichen werden. Die Möglichkeiten eines Bandstopps oder der Nacharbeit werden nicht betrachtet. Bei dem Einsatz eines Springers wird davon ausgegangen, dass dieser stets den kompletten Arbeitsaufwand an dem jeweiligen Arbeitsplatz erledigt. Eine Übernahme von einem Teil der Arbeitsvorgänge oder ein gleichzeitiges gemeinsames Arbeiten mit dem standardmäßigen Werker ist nicht vorgesehen. Die gewählte Organisation wird als *Skip policy* (s. [BKS11]) bezeichnet.

Außerdem wird eine Voraussetzung für den Einsatz eines Springers unterstellt. Der Einsatz muss erzwungen sein. Kein Springer darf freiwillig eingesetzt werden, wenn der standardmäßig vorgesehene Werker das jeweilige Fahrzeug eigenhändig innerhalb des Arbeitsbereichs bearbeiten könnte. Ein Verstoß gegen diese Bedingung kann zwar in einigen Fällen zu einer Verbesserung der Zielfunktion führen. Allerdings ist eine Umsetzung von freiwillig vorgezogenen Springereinsätzen in der Praxis nicht oder nur schwer umsetzbar. Der Einsatz von Unterstützern stellt das letzte Mittel dar, um einen Bandstopp zu verhindern und soll nur eingesetzt werden, wenn keine Alternativen zur Verfügung stehen. Deshalb soll das freiwillige Vorziehen von Springereinsätzen unterbunden werden.

Es wird von einer über den gesamten Produktionszeitraum konstanten Taktzeit ausgegangen. Schwankungen, die in der Realität vorkommen können und die die Produktionsmenge einer Schicht maßgeblich beeinflussen, werden nicht betrachtet. Insbesondere ist die Reduzierung der Bandgeschwindigkeit zur Entlastung der Werker, die eine Erhöhung der Taktzeit mit sich bringt, in dieser Arbeit keine Option.

Boysen et al. [BFS09] definieren die Taktzeit als durchschnittliche Arbeitszeit aller Fahrzeuge und begründen damit die Existenz von über- und unterlastenden Fahrzeugen. Hier muss angemerkt werden, dass der so berechnete Wert lediglich eine obere Schranke für die beliebig festzusetzende Taktzeit ist. Eine gewisse Reduzierung der Auslastung von 100 % - bei Berechnung der Taktzeit als Durchschnitt der Arbeitszeiten - ist üblich und auch notwendig, um die Überlastungen nicht zu groß werden zu lassen bzw. die Anzahl der Überlastungen so gering zu halten, dass diese von einer begrenzten Anzahl an Springern abgefangen werden können.

Zwischen den Arbeitsplätzen gibt es keinen Puffer, sodass ein Umordnen während des Montageprozesses nicht vorgesehen ist. Die eingesteuerte Sequenz durchläuft folglich sämtliche Arbeitsplätze in einer gleichbleibenden Reihenfolge.

Zusätzlich wird davon ausgegangen, dass auftretende Sperrungen eindeutig sind. Die Möglichkeit eine Sperrung aufgrund eines fehlenden Teils zu ignorieren und das Teil in Nacharbeit einzubauen, wird in dieser Arbeit nicht betrachtet. Eine solche Entscheidung könnte jedoch - auch bei Einsatz eines, der in dieser Arbeit entwickelten Verfahren - von einem Experten getroffen werden. Dieser würde bei auftretenden Problemen

entscheiden, ob sich daraus eine Sperrung ableitet. In das untersuchte Problem gehen somit nicht fehlende Teile, sondern nur tatsächliche Sperrungen als relevante Größe ein.

Darüber hinaus gibt es eine wesentliche Grundbedingung an die Produktionssequenz. In dieser sollen alle Nachfolgebeziehungen zwischen den Aufträgen aus der geplanten Sequenz erhalten bleiben, soweit dies möglich ist. Möglich ist die Beibehaltung für alle Aufträge, die nicht zurückgestellt werden und soll für diese durch eine geeignete Bedingung erzwungen werden. Dies ist insbesondere aus Sicht des Logistikbereichs wichtig, da häufig Teile *Just-in-Sequence* geliefert werden. Ein Verstoß gegen diese Bedingung kann zu einem unnötigen Umsortieraufwand der Bauteile führen und im Extremfall sogar zu einem Übersteigen des verfügbaren Platzes für die umzusortierenden Teile. Auf eine konkrete Betrachtung der Auswirkungen von Sequenzänderungen auf den Logistikbereich wird in dieser Arbeit verzichtet. Daher ist diese Annahme von zentraler Bedeutung.

Zuletzt wird eine gravierende Randbedingung genannt, die durch den operativen Charakter des Problems begründet wird. Eine große Herausforderung bei dem vorliegenden Problem ist die beschränkte Rechenzeit. Alle Entscheidungen müssen innerhalb eines Takts getroffen werden. D. h. beim statischen Problem muss die neue Plansequenz in einer Zeit, die kleiner als die Taktzeit ist, berechnet werden. Beim dynamischen Problem muss zumindest das nächste zu fixierende Fahrzeug am Ende eines jeden Takts feststehen. Da ein umgesetzter Algorithmus als entscheidungsunterstützendes Werkzeug eingesetzt werden kann, also auch zusätzliche Zeit für menschliche Entscheidungen oder Informationsweitergaben berücksichtigt werden sollten, ist die verfügbare Rechenzeit deutlich unter der Taktzeit anzusetzen. In dieser Arbeit wird dabei von einer maximal verfügbaren Rechenzeit von 30 s pro Takt ausgegangen.

2.5 Wahl der Zielfunktion

Das allgemeine Ziel des Optimierungsproblems ist es, eine Produktionssequenz zu erzeugen, bei der die Auslastung der normalen Werker möglichst hoch ist. Dies ist gleichbedeutend mit einem möglichst kleinen Anteil der Arbeiten, der von Springern

erledigt werden muss. Die Frage nach der Zielfunktion ist nun, wie dieser Anteil der Arbeiten von Springern ausgedrückt wird.

Die Beantwortung dieser Frage soll zunächst losgelöst von in der Literatur vorhandenen Ansätzen erfolgen und sich stattdessen an den praktischen Anforderungen orientieren. Insbesondere müssen dazu die Kostentreiber bestimmt und modelliert werden. Dazu werden zunächst die Größen betrachtet, die sich einfach bestimmen lassen, bevor danach komplexere Werte analysiert werden.

Die aus dieser Sicht erste relevante Größe ist die Summe aller Springerarbeitszeiten. Diese scheint sinnvoll, wenn von einem Springer jeweils die Restarbeiten übernommen werden, die der normale Werker nicht mehr innerhalb seines Arbeitsbereichs erledigen kann. In diesem Fall kann die Berechnung der Springereinsatzzeiten mittels linearer Gleichungen durchgeführt werden. Wie zuvor erwähnt, wird in dieser Arbeit jedoch die Skip policy angenommen, sodass eine diskrete Betrachtung der Springereinsätze notwendig wird.

Ein Schwachpunkt der reinen Betrachtung der Springerarbeitszeit ist, dass der zusätzliche Aufwand des Unterbrechens anderer Tätigkeiten, die Bewegung zum Arbeitsort und eventuelle arbeitsvorbereitende Tätigkeiten in dieser Zeit nicht berücksichtigt werden. Um diesem Aufwand Rechnung zu tragen, sollte ein fixer Belastungsanteil für jeden Einsatz eines Springers aufgeschlagen werden oder direkt die Anzahl der Springereinsätze als Zielfunktion verwendet werden.

Der Vorteil der bisher genannten Zielgrößen ist, dass bei der Berechnung das genaue zeitliche Auftreten der Springereinsätze vernachlässigt werden kann. Die Bedeutung eines Springereinsatzes ist insbesondere unabhängig von der jeweiligen Schicht, in der dieser anfällt und welche in den meisten Fällen nicht der Schicht entspricht, in der die Entscheidung zur Einsteuerung getroffen werden muss. Deshalb kann eine solche Bewertung für jede Produktionssequenz unabhängig von ihrer Länge bestimmt werden und besitzt stets die volle Aussagekraft.

Es bleibt zu klären, ob neben der Verwendung der Anzahl der Springereinsätze auch die Dauer dieser Einsätze in die Zielfunktion integriert werden sollte. Dazu kann zunächst angemerkt werden, dass wegen der verwendeten Skip policy alle Springereinsatzzeiten

größer als die Kapazität eines Arbeitsplatzes sind (für kürzere Arbeiten kann keine Überschreitung der Driftgrenze auftreten). Deshalb sind die Unterschiede zwischen möglichen Unterstützereinsätzen an einem Arbeitsplatz zumeist relativ gering. Dies steht im krassen Gegensatz zu der Berechnung bei Verwendung der *Side-by-side policy* (vgl. [BKS11]), bei der auch Einsätze von Sekundenbruchteilen auftreten können, da hier die Springer nur bei der Bearbeitung eines Teils der Tätigkeiten des Auftrags helfen, um eine Driftüberschreitung zu verhindern.

Eine mögliche Weiterentwicklung der Zielfunktion ist, nicht die Anzahl der Springereinsätze, sondern die Anzahl der benötigten Springer für diese Einsätze zu betrachten. Vereinfacht kann dies mittels der maximalen Anzahl an gleichzeitigen Springereinsätzen abgebildet werden. Für eine genaue Bewertung ist jedoch eine detaillierte Kenntnis der Praxissituation notwendig. Bei langen Montagelinien, die für die Automobilindustrie typisch sind, ist es nicht möglich, dass eine einzelne Person an allen Arbeitsplätzen aushelfen kann. Deshalb werden die Linien in Bereiche aufgeteilt, innerhalb derer ein Springer an allen Arbeitsplätzen einspringen kann. So kann für jeden Bereich isoliert eine benötigte Anzahl an Springern ermittelt werden, die gleichzeitig eingesetzt werden. Die Summe der Springer ergibt dann den Zielfunktionswert.

Diese Größe scheint auf den ersten Blick die tatsächlichen Kosten, die von der Ungleichverteilung der Arbeitsbelastungen verursacht werden, direkter als die vorher genannten Werte zu beschreiben. Jedoch müssen dabei einige zusätzliche Aspekte berücksichtigt werden.

Zum einen muss beachtet werden, dass bei der Reduzierung eines Springers nicht die gesamten Lohnkosten als Einsparung gesehen werden können, da die Springer während der Zeiten, in denen sie nicht am Band aushelfen müssen auch anderen Tätigkeiten nachgehen können. Der Nutzen einer Reduzierung der benötigten Anzahl an Springern ist also nicht so direkt ermittelbar, wie es auf den ersten Blick scheint. Deshalb muss bei jeder praktischen Anwendung dieser Zielgröße geprüft werden, ob diese bei der jeweiligen Arbeitsorganisation sinnvoll ist.

Außerdem ist die Anzahl der verfügbaren Springer in der Praxis schon gegeben. Die Personalplanung ist zum Zeitpunkt der Einsteuerung schon abgeschlossen. Von daher sollte die Anzahl der verfügbaren Springer eher als Nebenbedingung gesehen werden

und nicht als zu optimierende Größe. Bei Verwendung einer Nebenbedingung, die die Anzahl der Springer begrenzt, muss allerdings garantiert werden, dass es stets eine zulässige Lösung gibt. So könnte in Extremfällen erlaubt werden, zusätzliche Aufträge zu verschieben, sodass die Fahrbarkeit der Sequenz in der Montage zu Lasten eines zusätzlichen Aufwands in der Logistik gesichert wird.

Ein weiteres Problem bei Verwendung dieser Größe als Zielfunktion ist, dass sie nur in den Momenten mit den kritischsten Zuständen relevant wird. Während einer entspannten Produktionsphase, in der relativ wenige Springereinsätze nötig sind, könnten Springer so öfter als nötig eingesetzt werden ohne die Zielfunktion zu beeinflussen.

Unabhängig von den beschriebenen Optimierungszielen ist für die Zielfunktion ebenfalls entscheidend, über welchen Zeitraum diese berechnet wird. In dem zuerst untersuchten statischen Problem ist dieser auf ein relatives kurzes Intervall beschränkt. Von daher erscheint eine Fokussierung auf die Springereinsätze sinnvoll. Bei der anschließenden Betrachtung des dynamischen Problems, das im Allgemeinen nicht auf einen bestimmten Zeithorizont beschränkt ist, wird zunächst die Zielfunktion vom statischen Problem einfach auf ein längeres Intervall von etwa einer Schicht ausgeweitet. Nun ist es jedoch auch möglich bei genauer Betrachtung der zeitlichen Verteilung die Anzahl der Springer zu ermitteln.

Zudem muss die Konsistenz mit der Einplanungsoptimierung sichergestellt werden. Deshalb wird die in dieser Arbeit gewählte Zielfunktion von der zu Grunde liegenden Problemstellung aus der Praxis bestimmt. Die verfügbaren Eingangsdaten, die sich aus einem Optimierungsverfahren für die Einplanung ergeben, wurden auf Basis der reinen Betrachtung der Anzahl der Springereinsätze erstellt. Deshalb wird auch im hier untersuchten Folgeprozess, der Einsteuerung, dieselbe Zielfunktion verwendet und auf eine Berechnung der genauen Zeiten der Springereinsätze verzichtet. Der Fokus dieser Arbeit liegt deshalb auf der Minimierung der Anzahl der Springereinsätze. Die in dieser Arbeit vorgestellten Methoden lassen sich jedoch alle ohne großen Aufwand erweitern, sodass auch die Springerarbeitszeiten in die Zielfunktion einfließen könnten. So ließen sich die Ergebnisse übertragen, wenn bei einer anderen Anwendung auch die Zeiten in der Einplanung beachtet würden.

Ein Kapitel wird sich zusätzlich mit der Möglichkeit beschäftigen, die genaue An-

zahl der benötigten Springer zu berechnen. Dies stellt jedoch keinen Widerspruch zur Einplanung mit der anderen Zielfunktion dar. Die beiden Zielfunktionen sind ähnlich genug, sodass bei allen Testdaten die Anzahl der benötigten Werker schon bei der Einplanung ausreichend minimiert wurde. Da diese Zahl erst durch Sperrungen deutlich ansteigt, kann bei der Einplanung auf eine Minimierung in diese Richtung verzichtet werden. Die Zahl der benötigten Unterstützer bei Realisierung der Plansequenz besitzt nach wenigen Sperrungen nur noch geringe Bedeutung für die Qualität der resultierenden Ist-Sequenz.

Eine ergänzende Anmerkung ist, dass sämtlichen erwähnten Zielfunktionen Gewichtungen hinzugefügt werden können. Dadurch kann bewertet werden, wie schwierig der Einsatz eines Springers an einem bestimmten Arbeitsplatz ist. Diese Erweiterung ist für die Anwendbarkeit in der Praxis von großer Bedeutung und wird deshalb auch in einigen Tests in dieser Arbeit berücksichtigt.

Außerdem sind bei der Optimierung des dynamischen Prozesses, der immer nur stückweise betrachtet wird, auch alternative Zielfunktionen denkbar, die darauf abzielen, dass insbesondere das Ende der betrachteten Sequenz robust gegenüber weiteren Sperrungen ist und die in Zukunft zu erwartenden Unterstützereinsätze durch die entsprechende Steuerung nicht negativ beeinflusst werden. Solche temporär verwendeten Hilfszielfunktionen werden bei ihrem Auftreten in dieser Arbeit erläutert.

3 Literaturüberblick

Diese Zusammenfassung der vorhandenen Veröffentlichungen unterteilt sich in zwei Abschnitte. Zunächst wird die existierende Forschung im betrachteten Themengebiet zusammengestellt. Danach werden Ressourcen für Methoden genannt, die bei der Entwicklung neuer Verfahren genutzt werden, die allerdings nicht im unmittelbaren Zusammenhang mit dem analysierten Problem stehen.

3.1 Inhaltlich relevante Literatur

Bei der Optimierung von Variantenfließlinien gibt es zwei Hauptplanungsaufgaben (s. [BSJ94], [EBS10]), die sich gegenseitig beeinflussen. Einerseits muss das mittelfristige bzw. strategische (s. [Mey04]) *Assembly Line Balancing Problem* gelöst werden. Dabei wird entschieden, wie sämtliche Arbeitsvorgänge, die an einem Fahrzeug erledigt werden können, verteilt werden. Dies beinhaltet die räumliche Platzierung von Arbeitsplätzen, die Verteilung der Arbeitsvorgänge auf die Arbeitsplätze, die Bestimmung der Anzahl der Werker an den jeweiligen Arbeitsplätzen und die Festlegung der Taktzeit. In dieser Arbeit wird allerdings ausschließlich der kurzfristige Horizont betrachtet, in dem das *Sequencing Problem* gelöst werden muss, d. h. die Frage, in welcher Reihenfolge die Fahrzeuge montiert werden.

In der Literatur haben sich für dieses Sequencing Problem zwei Klassen von Zielfunktionen gebildet (s. [BFS09]): *Work Overload* und *Just-in-time Objectives*.

Durch die verschiedenen Varianten mit diversen unterschiedlich kombinierten Sonderausstattungen, die auf einer Linie gefertigt werden, haben die Werker je nach Auftrag verschiedene Arbeitsbelastungen an einem Arbeitsplatz. Dabei ist üblicherweise die

Belastungszeit von manchen Aufträgen größer und von manchen kleiner als die Taktzeit. Wenn jedoch mehrere arbeitsintensive Aufträge hintereinander bearbeitet werden müssen, summiert sich die Drift und es kann Überlast entstehen. Bei der ersten Klasse von Zielfunktionen soll dies durch eine geeignete Sequenz, in der sich die arbeitsreichen und arbeitsarmen Aufträge abwechseln, vermieden werden.

Die Just-in-time Zielfunktionen versuchen den Materialbedarf ausgeglichen zu gestalten. Ausgehend von einem durchschnittlichen Auftrag wird ein Sollbedarf ermittelt, von dem der reale Bedarf möglichst gering abweichen soll. Der Materialbedarf wird jedoch in dieser Arbeit nicht weiter betrachtet. Die Auswirkungen der Sequenz auf die Logistik gehen lediglich in die Nebenbedingung ein, die garantiert, dass die Reihenfolge im Vergleich zur Planung nicht unnötig verändert werden darf.

Zu den Problemen rund um die Reihenfolgebildung von Aufträgen auf Variantenfließlinien existieren in der Literatur drei Ansätze: Das *Level-Scheduling*, das *Car-Sequencing* und das *Mixed-Model-Sequencing*. Klassifizierungsschemata zu allen drei Ansätzen finden sich bei Boysen et al. [BFS09], mitsamt diverser Quelleneinordnungen. Im folgenden Kapitel wird diese Arbeit gemäß einem dieser Schemata klassifiziert.

Der Level-Scheduling Ansatz versucht eine möglichst ausgeglichene Sequenz zu erzeugen. Dabei wird ein fiktiver Durchschnittsauftrag berechnet und der Abstand zwischen den tatsächlichen Werten und hintereinander produzierten Durchschnittsaufträgen minimiert. In welcher Hinsicht dabei der durchschnittliche Auftrag gebildet wird, ist offen. Boysen et al. [BFS07] unterscheiden die Orientierung an Varianten, Arbeitsbelastungen und Materialverbrauch. In der Literatur dominieren dabei die Ansätze zur ausgeglichenen Variantenfertigung oder zum nivellierten Materialverbrauch. Von daher ist dieser Ansatz für die vorliegende Arbeit von geringer Bedeutung. Es gibt allerdings auch die Möglichkeit diese Methode für die Arbeitsbelastung anzuwenden, wie z. B. bei Gujjula und Günther [GG10], die jedoch eine Kombination mit dem Mixed-Model-Sequencing Ansatz vorschlagen. Hier dient die Idee des Level-Scheduling lediglich als Hilfe für eine heuristische Lösung. Außerdem ist das Level-Scheduling mit dem Car-Sequencing Ansatz kombiniert untersucht worden. So wurde bei Yavuz [Yav13] die *Iterated Beam Search* zur gemeinsamen Optimierung angewandt.

Eine Kritik an diesem Ansatz ist, dass nicht das eigentliche Ziel fokussiert wird. Wie bei

Boysen et al. [BFS07] erwähnt, stellt eine ausgeglichene Belastungsverteilung entlang der Produktionssequenz lediglich ein Ersatzziel dar. Eine gewisse Über- und Unterlast ist in der Realität aufgrund der unterschiedlichen Aufträge unvermeidbar und nicht zwingend problematisch. Im Rahmen festgelegter Driftbereiche können die Arbeitsbelastungen ohne negativen Einfluss auf die Ziele schwanken. Von daher bildet dieser Ansatz die Zielfunktion zu ungenau ab. Zu diesem Ergebnis kam auch Boysen [Boy05], der in experimentellen Studien zeigte, dass die folgenden beiden, alternativen Ansätze deutlich bessere Ergebnisse hervorbringen.

Zum Car-Sequencing Ansatz finden sich mit Abstand die meisten Quellen, z. B. Parrello et al. [PKW86] oder Dincbas et al. [DSVH88]. Die Idee hinter diesem Ansatz ist, durch das Aufstellen von Regeln implizit die Überlastungen zu minimieren. Diese Regeln haben üblicherweise die Form $H_o : N_o$, was bedeutet, dass in jeder Folge von N_o aufeinander folgenden Aufträgen maximal H_o Aufträge mit einem bestimmten Merkmal existieren dürfen. Diese Regeln werden für besonders arbeitsintensive Arbeitsvorgänge aufgestellt, da durch eine Häufung dieser Arbeitsvorgänge Überlastungen unvermeidbar wären. Das Optimierungsziel ist es dann, eine Sequenz zu finden, die alle Regeln einhält bzw. möglichst wenige verletzt. Dieser Ansatz steht auch aktuell noch im Fokus der Forschung. Eine industrielle Fallstudie, anhand der diverse Lösungsmethoden vorgestellt und getestet worden sind, ist bei Solnon et al. [SCNA08] zu finden. Neue Methoden verknüpfen Ideen des *Constraint Programming* mit dem *Boolean Satisfiability Problem* und übertragen diese auf den Car-Sequencing Ansatz (s. [AHME+13]).

Wie schon das Level-Scheduling ist auch dies nur ein Hilfskonstrukt. Das eigentliche Ziel ist das gleiche wie beim dritten Ansatz, allerdings werden lediglich Heuristiken verwendet, die dazu führen sollen, dieses möglichst gut zu erreichen. Auch das Car-Sequencing hat sich in Tests im Vergleich mit dem Mixed-Model-Sequencing als schlechter herausgestellt (s. [GRB11]).

Der letzte Ansatz ist das Mixed-Model-Sequencing, das auch in dieser Arbeit analysiert wird. Die erste Formulierung dieses Ansatzes in der Literatur findet sich bei Wester und Kilbridge [WK64]. Bei diesem Ansatz werden detailliert alle Belastungszeiten und ausgehend davon die Belastungen an jedem Arbeitsplatz berechnet. Dadurch kann die Bewegung jedes einzelnen Werkers entlang seiner Station und die Überschreitungen

von Grenzen bewertet werden. Deshalb ist dieser Ansatz exakter als die anderen, allerdings auch komplexer in seinen Berechnungen. Außerdem müssen die exakten Belastungszeiten von jedem Auftrag an jedem Arbeitsplatz vorliegen, damit dieser Ansatz verwendet werden kann.

Ein Überblick über frühe Beiträge findet sich bei Yano und Bolat [YB89]. Für neuere Referenzen sei erneut auf Boysen et al. [BFS09] verwiesen. Deren Übersicht verdeutlicht, dass die Forschung von der Modellbildung und der Entwicklung heuristischer Methoden dominiert wird. Aufgrund der Komplexität dieses Ansatzes lassen sich exakte Methoden nur auf eingeschränkte Probleme oder kleine Instanzen anwenden und dienen dann eher als Referenzmethoden anstatt als praktikable Anwendungsansätze für den realen Einsatz.

Das Mixed-Model-Sequencing Problem ist NP-schwer. Dies wurde von Tsai [Tsa95] für eine Variante mit einem einzelnen Arbeitsplatz und dem Ziel der Minimierung der Überlastungszeit gezeigt. Auch die in dieser Arbeit verwendete Zielfunktion führt beim Sequenzierungsproblem zu einem NP-schweren Problem (s. [BKS11]). Da dieses Sequenzierungsproblem als Spezialfall des Resequenzierungsproblems aufgefasst werden kann, überträgt sich die Eigenschaft auch darauf.

Die allgemeine Problemstellung ist auf verschiedene Weisen abgekürzt worden. So wurde das Mixed-Model-Sequencing Problem mit MSP (s. [SKD98]), MMSP (s. [BC11]) oder einfach MM (s. [BKS11]) bezeichnet. Eine Benennung der Zielfunktion folgt in den meisten Fällen, welche entweder ein W oder WO für die überlastenden Arbeitszeiten (work overload) oder OS für die Anzahl der Überlastungssituationen (overload situations) sein kann. Da der Fokus dieser Arbeit auf die Resequenzierung gelegt wird und die zweite Art der Zielfunktionen im Mittelpunkt steht, kann für das im Kapitel 5 betrachtete Problem die Bezeichnung MMRP-OS (mixed-model resequencing problem) verwendet werden.

Häufig wird bei der Analyse dieses Problems angenommen, dass es eine begrenzte Anzahl an Modellen gibt, die mehrmals produziert werden müssen (z. B. [SKD98], [BC11], [BKS11]). Auf diese Sichtweise wird in dieser Arbeit verzichtet, da in der Realität selten mehrere identische Fahrzeuge produziert werden. Dieses Szenario kann bei den Modellen in den genannten Veröffentlichungen als ein Spezialfall angesehen

werden. Jedoch führt dieser dazu, dass Methoden, die die Existenz mehrerer identischer Aufträge ausnutzen, nicht verwendet werden können.

Außerdem werden bei den meisten Publikationen weitere Annahmen getroffen, die nicht der Praxis entsprechen. So muss z. B. bei der Berechnung der Werkerpositionen nach Springereinsätzen genau beachtet werden, welche Arbeit in der Realität von wem übernommen wird. In den meisten Veröffentlichungen unterstützen die Springer bei Bedarf die eigentlichen Werker, indem sie einen Teil der Arbeit übernehmen. Für den betrachteten Praxisfall ist eine solche Modellierung verzerrend, da ein Unterstützer hier immer einen kompletten Auftrag bei Bedarf übernimmt. Für diese Variante wurde von Boysen et al. [BKS11] die Verwendung der Bezeichnung Skip policy vorgeschlagen.

Nun wird der Überblick über die verwendeten Zielfunktionen vervollständigt. In den meisten Veröffentlichungen wird versucht die Summe aller Überlastungszeiten zu minimieren. Bei Boysen et al. [BKS11] werden Schwächen dieses Konzepts herausgestellt und der Vorteil der Skip policy sowie der damit verbundenen Betrachtung der Anzahl der Überlastungssituationen wird erläutert. Dies wird durch Altemeier et al. [AHKD10] unterstützt, die auf eine Korrelation zwischen der Anzahl der Unterstützereinsätze und den zusätzlichen Kosten durch Qualitätseinbußen und Nacharbeit hinweisen. Eine weiterführende Betrachtung der Zielfunktion in Bezug auf die Anzahl der benötigten Unterstützer ist bei Gujjula und Günther [GG09c] zu finden. Allerdings werden dabei die Einsätze der Unterstützer in einer Art und Weise geplant, die bei den in dieser Arbeit getroffenen Annahmen nicht umsetzbar ist. Die Überlastungssituationen werden antizipiert und die nötigen Springereinsätze optimiert und dabei gegebenenfalls auch früher als nötig platziert.

Eine Erweiterung der allgemeinen Modelle ist von Bautista und Suàrez [BS09] präsentiert worden. In ihrem Modell wird der Einfluss der Arbeitsplätze untereinander betrachtet. Dies ist nötig, wenn eine Bearbeitung des Werkstücks über die Stationsgrenze hinaus stattfindet, welche zu einer Behinderung der Arbeiten an nachfolgenden Arbeitsplätzen führt. In dieser Arbeit wird auf eine solche explizite Betrachtung der Abhängigkeiten von Arbeitsplätzen verzichtet, da diese schon implizit über Driftgrenzen in das Problem aufgenommen werden.

Nach der Differenzierung der Ansätze kann nun auch entsprechend der Verwendung

eines Ansatzes unterschieden werden. Der größte Teil des Forschungsaufwands wurde für allgemeine Sequenzierungsprobleme betrieben. Diese Arbeit beschäftigt sich jedoch mit dem Spezialfall der Resequenzierung, also der Bildung einer neuen Sequenz aus einer vorhandenen Sequenz nach gewissen Vorgaben. Ein Framework zur Strukturierung der Resequenzierungsprobleme auf Variantenfließlinien sowie einen umfangreichen Literaturüberblick liefern Boysen et al. [BSW12]. Bei einer allgemeineren Betrachtung von Reschedulingproblemen kann auf eine breitere Literaturbasis zurückgegriffen werden. Ein Überblick über Strategien, Policen und Methoden findet sich bei Vieira et al. [VHL03].

Es wird jedoch deutlich, dass sich der überwiegende Teil der Forschung im Bereich der Resequenzierung auf Variantenfließlinien in der Automobilindustrie mit der Umsortierung der Aufträge vor der Lackierung befasst. Nur wenige Forscher haben sich mit der Resequenzierung vor der Endmontage beschäftigt. Von diesen wiederum wird zumeist der Car-Sequencing Ansatz verwendet (z. B. [CS97], [IS03] und [BGR11]). Dieser Ansatz wird aktuell weiter untersucht, so wird bei Boysen und Zenker [BZ13] eine spezielle Art von Puffern in diesem Kontext betrachtet. Die dort untersuchten *Selectivity Banks*, die nur eine eingeschränkte Umsortierung der Aufträge erlauben, werden mit einer Zerlegung des Problems in zwei Phasen und der Anwendung verschiedener heuristischer Verfahren gesteuert.

Die einzigen Untersuchungen zur Resequenzierung vor der Montage nach dem Mixed-Model-Sequencing Ansatz finden sich bei Gujjula und Günther [GG09a,GG09b]. Dabei liegt der Fokus in einem Beitrag [GG09b] auf den Just-in-Sequence Anlieferungen. In einem gewissen Rahmen dürfen alle Fahrzeuge verschoben werden und die Kosten für den Logistikaufwand werden gegengerechnet. Auf eine solche Bewertung wird in dieser Arbeit verzichtet und stattdessen werden stärkere Beschränkungen für die Sequenzänderung verwendet, um dem logistischen Aufwand Rechnung zu tragen. Dies entspricht dem Ansatz aus ihrer anderen Veröffentlichung [GG09a]. Dieser Ansatz bildet eine wesentliche Grundlage für weitere Untersuchungen in der vorliegenden Arbeit. Es wird ein statisches Resequenzierungsproblem beschrieben und als gemischt-ganzzahliges lineares Programm (MILP) modelliert, welches in vielen Aspekten mit den Annahmen in dieser Arbeit übereinstimmt. Außerdem wird ein heuristischer Suchalgorithmus vorgeschlagen. Jedoch sind die durchgeführten Tests auf künstlich generierte Instanzen

mit eingeschränkter Problemgröße begrenzt.

Zur Steuerung des dynamischen Prozesses der Einsteuerung nach dem Mixed-Model-Sequencing Ansatz von Aufträgen über ein komplettes Produktionsintervall wurde keine Veröffentlichung gefunden. Choi und Shin [CS97] haben einen Ansatz entwickelt, der nach Level-Scheduling- und Car-Sequencing-Kriterien optimiert und eine dynamische Steuerung einer realen Montagelinie ermöglicht.

3.2 Methodisch relevante Literatur

Die Analyse der Literatur über die gegebene Problemstellung hat gezeigt, dass noch eine Lücke in der Forschung existiert. Um das Problem zu lösen, werden Verfahren benötigt, die im Zusammenhang mit dem Resequenzierungsproblem bislang nicht verwendet wurden. Deshalb wird nun ein Überblick über Literatur ergänzt, die wegen ihrer Methoden herangezogen wurde, aber keinen inhaltlichen Bezug hat.

Das untersuchte Problem kann als *Online-Optimierungsproblem* bezeichnet werden. Solche Probleme definieren sich darüber, dass sich relevante Werte während der Bearbeitung des Problems ändern können. Ein Algorithmus muss zur Lösung gewisse Entscheidungen treffen, ohne zukünftige Ereignisse zu kennen (vgl. [BEY98]). Für das dynamische Problem kann also nur auf allgemeine Ansätze zur Online-Optimierung aus der Literatur zurückgegriffen werden.

Dabei kann zunächst geklärt werden, welche Paradigmen für dieses Problem verwendet werden. Da mit Freigabe eines Auftrags alle Informationen über diesen - insbesondere die Bearbeitungszeiten - bekannt sind, wird das *Clairvoyant* Paradigma (vgl. [Mas03]) verwendet. Dieses besagt, dass die Bearbeitungszeiten eines Auftrags in dem Moment, in dem dieser eingeplant werden soll, feststehen. Außerdem wird das *Time Stamp Model* (vgl. [GKR$^+$01]) verwendet, bei dem im Gegensatz zum *Sequence Model* nach dem Eintreffen von Aufträgen mit der Entscheidung gewartet werden darf und eingegliederte Aufträge auch wieder neu platziert werden dürfen, solange die Bearbeitung noch nicht begonnen hat.

Grötschel et al. [GKR+01] unterscheiden vier Grundstrategien für Online-Algorithmen: Die Anfragen werden in der Reihenfolge, in der sie auftreten, beantwortet (FIFO); es wird immer der aktuell beste Auftrag ausgewählt (GREEDY); bei jedem neuen Ereignis wird alles neu optimiert (REPLAN); eine Lösung des statischen Offline-Problems wird verwendet und neue Aufträge werden gesammelt und danach erneut in Betrachtung eines neuen statischen Problems optimiert und der Rest wird dabei unverändert gelassen (IGNORE).

Das vorliegende Problem stellt ein *kombinatorisches Optimierungsproblem unter Unsicherheit* dar. Bei einem kombinatorischen Optimierungsproblem ist es das Ziel, aus einer endlichen Menge, die gewissen Bedingungen genügt, ein Element zu finden, das optimal bezüglich einer Zielfunktion ist (vgl. [PS82]). Bei dem hier untersuchten Problem sind die Randbedingungen insoweit fixiert, dass alle möglichen Aufträge eines Produktionsintervalls vorab bekannt sind. Lediglich das Auftreten von Sperrungen ist nicht vorhersehbar. Ungeachtet der Dynamik des Problems, die die Unsicherheit nur schrittweise auflöst, können daher Verfahren für diese Art von Optimierungsproblemen betrachtet werden. Die zwei üblichen Ansätze zur exakten Lösung von kombinatorischen Optimierungsproblemen unter Unsicherheit sind das *Stochastic Programming* und die *Robust Optimization* (s. [LLMS09]).

Bei dem Stochastic Programming Ansatz wird eine Wahrscheinlichkeitsverteilung der unsicheren Variablen angenommen und die Zielfunktion als Erwartungswert ausgedrückt (vgl. [SDR09]). Die Belegung der Variablen kann dabei auch schrittweise in mehreren Stufen erfolgen, wie in der vorliegenden Anwendung. Für das hier analysierte Problem ist eine Formulierung einer sinnvollen Verteilung nur schwer möglich und eine Berechnung der Erwartungswerte würde an der Komplexität scheitern.

Die Idee der robusten Optimierung ist, eine Lösung zu suchen, die für jede mögliche Realisierung der unsicheren Variablen zulässig ist (s. [BBC11]). Üblicherweise impliziert diese Methode, dass die schlechteste mögliche Realisierung den Zielfunktionswert bestimmt. Beim vorliegenden Problem sind jedoch nicht alle theoretisch möglichen Realisierungen der unsicheren Variablen realistisch. Daher beschränkt sich die Anwendung dieses Ansatzes auf eine Sicherstellung der Zulässigkeit von Modellen und Sequenzen unter allen theoretisch möglichen Bedingungen.

Metaheuristische Ansätze zur Lösung von allgemeinen kombinatorischen Optimierungsproblemen können bei Blum und Roli [BR03] gefunden werden. Metaheuristiken werden als allgemeine Ansätze auf einem hohen Abstraktionslevel, die die Verwendung von einfachen Heuristiken kombinieren, beschrieben. Dabei sind insbesondere die in dieser Arbeit verwendeten Ansätze, das *Simulated Annealing*, die *Tabu Search* und die *Variable Neighborhood Search*, relevant.

Bianchi et al. [BDGG06] geben einen Überblick über Metaheuristiken, die insbesondere bei stochastischen kombinatorischen Optimierungsproblemen angewandt werden können. Dabei werden *Ant Colony Optimization*, *Evolutionary Computation*, Simulated Annealing, Tabu Search und *Stochastic Partitioning Methods* beschrieben. Eine genaue Erläuterung der in dieser Arbeit verwendeten Metaheuristiken erfolgt später an den entsprechenden Stellen.

Bei einem Vergleich mit bisher in der Praxis verwendeten Verfahren im Rahmen eines Entscheidungsunterstützungssystems kann man von einem Wandel vom *Reactive Planning* hin zum *Incremental Planning* sprechen. Hierbei wird eine vorausschauende Planung durchgeführt, welche im Gegensatz zur einfachen Reaktion auf die momentane Situation oder dem *Deliberative Planning*, der die gesamte Plansequenz verändern würde (vgl. [SPG+97]), steht.

4 Einordnungen und Forschungsbedarf

Die untersuchte Problemstellung wird in diesem Kapitel zunächst in ein Mixed-Model-Sequencing Schema und ein allgemeines Resequenzierungsschema eingeordnet und es werden Bezeichnungen für die analysierten Facetten des Problems gegeben. Danach werden aus der vorhandenen Literatur und der Praxis der Forschungsbedarf abgeleitet und Zielstellungen für diese Arbeit formuliert.

4.1 Problemeinordnung

Das vorliegende Problem kann gemäß dem Schema von Boysen et al. [BFS09] im Rahmen der Mixed-Model-Sequencing Probleme klassifiziert werden.

Dabei unterteilen sich die Arbeitsplätze in geschlossene und nach rechts geöffnete: $\alpha_1 = closed; open^{right}$. Wie weit die Arbeitsplätze nach rechts geöffnet sind, wird durch die Driftgrenzen individuell festgelegt. Arbeitsüberlastungen werden durch Unterstützer abgebaut und beeinflussen somit nicht die folgenden Produktionsschritte: $\alpha_2 = \circ$. Die Belastungszeiten werden als deterministisch angesehen: $\alpha_3 = \circ$. Gleichzeitige Arbeit mehrerer Werker ist grundsätzlich möglich, wird aber nicht explizit betrachtet, da Abhängigkeiten durch geeignete Driftgrenzen ausgeschlossen werden: $\alpha_4 = \circ$. Rüstzeiten werden nicht berücksichtigt, $\alpha_5 = \circ$, ebenso wie parallele Arbeitsplätze: $\alpha_6 = \circ$.

Die Anzahl der Arbeitsplätze ist nicht exakt festgelegt: $\beta_1 = \circ$. Sie liegt in allen verwendeten Testinstanzen jedoch im dreistelligen Bereich. Die einzelnen Arbeitsplätze

sind nicht homogen, sondern unterscheiden sich hinsichtlich des Driftbereichs oder der Anzahl der an ihnen arbeitenden Werker: $\beta_2 = div$. Die Fahrzeuge werden in einem gleichbleibenden Takt in die Montage eingeleitet: $\beta_3 = \circ$. Die Rücklaufzeiten der Werker werden nicht gesondert berücksichtigt, da sie schon implizit bei den Belastungszeiten der einzelnen Fahrzeuge erfasst werden: $\beta_4 = \circ$. Ebenso wird keine besondere Anordnung der Montagebänder betrachtet: $\beta_5 = \circ$. Dennoch spielt diese letztlich bei der Gruppenbildung von den Unterstützerkräften durchaus eine Rolle.

Die in dieser Arbeit verwendete Zielfunktion stellt eine Ableitung von der bei der Einführung dieses Klassifizierungsschemas genannten Arbeitsüberlastung dar. Der Kern der Arbeit beschäftigt sich mit der Anzahl der Unterstützereinsätze, was durch $\gamma = nos$ (*number of work overload situations*) beschrieben wird.

Diese Annahmen führen zu folgender Bezeichnung des grundlegenden Modells:

$$\left(closed; open^{right} | div | nos\right)$$

Des Weiteren kann das untersuchte Problem in das Resequenzierungsframework von Boysen et al. [BSW12] eingeordnet werden. In dieser Arbeit werden nur Umsortierungen in Betracht gezogen, die durch unvorhergesehene Ereignisse erzwungen werden. Es werden also nur reaktive Resequenzierungen analysiert.

Der in Kapitel 2 beschriebene Sortierer bietet einen Zugriff auf alle eingelagerten Karosserien und kann folglich als AS/RS (*automated storage and retrieval system*) Puffer bezeichnet werden. Neben den physischen Umsortierungen sind jedoch ebenfalls virtuelle Umsortierungen möglich. Diese nehmen jedoch aufgrund der Modellvielfalt nur eine untergeordnete Rolle ein und werden nicht explizit untersucht.

Die vorliegende Problemstellung kommt aus dem operativen Bereich und beschreibt in erster Linie einen dynamischen Prozess. Lediglich in Kapitel 5 wird das Problem auf einen einzelnen Zeitpunkt zu einem statischen Problem reduziert.

Die Zielfunktion ist von der Arbeitsbelastung abhängig und basiert auf dem Mixed-Model-Sequencing Ansatz. Dabei werden heuristische Ansätze die wesentlichen Lösungsmethoden darstellen.

Im vorangegangenen Kapitel wurden verschiedene Bezeichnungen für das Mixed-Model-Sequencing Problem genannt. Davon abgeleitet, können die in dieser Arbeit untersuchten Problemstellungen benannt werden. Das statische Teilproblem im nächsten Kapitel kann als MMRP-OS (*mixed-model resequencing problem with the objective to minimize the number of overload situations*) bezeichnet werden. Für die dynamische Betrachtung in Kapitel 6 kann auch die Problembezeichnung um den dynamischen Aspekt ergänzt werden: DMMRP-OS. Die alternative Zielfunktion in Kapitel 7, die Minimierung der Anzahl der benötigten *utility worker*, kommt in der Abkürzung DMMRP-UW zum Ausdruck.

Im Gegensatz zu den meisten Modellen in der Literatur wird der Spezialfall, dass von einer Fahrzeugart mehrere identische Fahrzeuge produziert werden müssen, nicht explizit behandelt. Dies schließt natürlich nicht aus, dass es Fälle gibt, in denen die Belastungszeiten zweier Fahrzeuge identisch sind. Allerdings besitzen die in dieser Arbeit betrachteten Aufträge individuelle Belastungszeiten.

Die beschriebene Problemstellung ist nicht auf die Automobilindustrie beschränkt. In der Elektronikindustrie spielt z. B. bei der Herstellung von Leiterplatten die Sequenzierung und Resequenzierung eine wichtige Rolle (vgl. [SJJN99]). Allerdings werden in dieser Arbeit viele spezifische Annahmen getroffen, sodass die direkte Anwendung der entwickelten Algorithmen auf den Automobilbereich beschränkt bleibt. Grundsätzliche Ideen können bei entsprechenden Anpassungen vermutlich auch auf die Steuerung anderer Herstellungsprozesse übertragen werden.

4.2 Ziele der Arbeit

Das Hauptziel dieser Arbeit ist es, das beschriebene Problem so praxisnah wie möglich zu modellieren und danach Lösungsansätze zu finden, die praxistauglich sind und möglichst gute Ergebnisse hervorbringen. Dabei ist es ein Anliegen, den bisher - gerade in der Praxis - wenig beachteten Ansatz des Mixed-Model-Sequencing auch für komplexe Instanzen bei der Resequenzierung und mit geringer Rechenzeit nutzbar zu machen.

Zunächst ist es das Ziel dieser Arbeit, eine wenig beachtete Zielfunktion zu etablieren. Es soll nun die Anzahl der Springereinsätze und nicht, wie im überwiegenden Teil der in der Literatur vorhandenen Ansätze, die Überlastzeit minimiert werden. Diese Entscheidung ist nicht unerheblich, da von linearen Variablen zu diskreten gewechselt wird, was die Berechnungen deutlich erschwert. Die bisherigen Ansätze in der Literatur begnügen sich hauptsächlich mit Zielfunktionen, die lineare Funktionen von der Drift oder von der Unterstützerarbeitszeit sind. Allerdings ist es in der Praxis ein erheblicher Unterschied, ob ein Springer dreimal eine Minute an einem Fahrzeug aushilft oder einmal drei Minuten. Um der Belastung, die allein durch den Springereinsatz und nicht durch die eigentliche Zeit entsteht, Rechnung zu tragen, beschäftigt sich die vorliegende Arbeit mit dieser Zielfunktion.

Diese Zielfunktion soll als erstes im statischen Problem verwendet werden. Für dieses Problem soll ein Modell aufgestellt werden. Dieses Modell soll entsprechend der bei der Problemstellung genannten Aspekte weiterentwickelt werden, um alle relevanten Eigenschaften des Praxisproblems abzudecken. Danach sollen Algorithmen entwickelt werden, die dieses Problem in einer für die Praxis zulässigen Zeit bearbeiten können. Die Ergebnisse, die durch die verschiedenen Algorithmen erzeugt werden, können dann miteinander verglichen werden. Wenn möglich, sollten optimale Lösungen ermittelt werden, damit eine objektive Bewertung der entwickelten Methoden möglich ist.

Die größte Lücke in der Literatur ist jedoch das dynamische Problem, zu dem in der hier beschriebenen Form keine Veröffentlichungen gefunden werden konnten. Um diese zu schließen, sollen Strategien entwickelt werden, wie die für das statische Problem entwickelten Algorithmen im Produktionsverlauf angewandt werden können. An dieser Stelle sollen die Algorithmen, die das statische Problem gut gelöst haben, weiterentwickelt und an das dynamische Umfeld angepasst werden. Für den sich jeden Takt wiederholenden Einsatz der statischen Wiedereinsteuerung soll getestet werden, welche Aufträge in jedem Takt verschoben werden dürfen. Hier soll geprüft werden, welche der Strategien für Online-Probleme in diesem Kontext sinnvoll eingesetzt werden können und ob Strategien kombiniert werden können und ob deren Wahl vom jeweiligen Zustand des Systems abhängig gemacht werden kann. Das ist erforderlich, da eine optimale Lösung des statischen Problems in jedem Takt keine optimale Lösung des dynamischen Problems garantiert.

Eine Herausforderung wird dabei sein, die Qualität der produzierten Lösungen zu bewerten. Das in der Wissenschaft verwendete Qualitätsmaß für Online-Algorithmen ist die Kompetitivität (vgl. [KMRS88]). Diese ist jedoch im vorliegenden Problem nicht verwendbar. Zum einen verhindert die Art der Zielfunktion eine Aussagekraft, da sich stets Beispiele generieren lassen, in denen der optimale Zielfunktionswert 0 ist, die aber die Schwächen des Algorithmus ausnutzen, sodass er einen positiven Zielfunktionswert und somit ein Kompetitivitätsverhältnis von unendlich hätte. Andererseits erreicht eine exakte Analyse der Algorithmen schnell die Grenzen von Rechenzeit und Speicherplatz. Die optimale Lösung erhielte man, wenn der gesamte Tag, mitsamt allen Sperrungen und Freigaben, bekannt wäre und dafür eine optimale Sequenz berechnet würde. Dies ist jedoch in keiner akzeptablen Zeit möglich. Deshalb sollten alternative Bewertungsmethoden verwendet werden, die über einen reinen Vergleich der entwickelten Algorithmen hinausgehen. Die Berechnung von Schranken könnte hier ein sinnvolles Mittel sein.

Zusätzlich soll überprüft werden, ob das beschriebene Problem verändert werden kann, um die Kosten beim Praxiseinsatz direkter abbilden zu können. Dabei ist es das Ziel, eine detailliertere Betrachtung der relevanten Größen vorzunehmen und die theoretische Umsetzbarkeit solcher Verfeinerungen zu überprüfen. Ausgehend von diesen Überlegungen können Potentiale für weitere Optimierungsprobleme ermittelt werden.

5 Das statische Problem

In diesem Kapitel wird das in der Problembeschreibung als statisches Problem bezeichnete Szenario analysiert. Dabei werden im ersten Abschnitt Modelle entwickelt, die das Problem mit unterschiedlichen Detaillierungen erfassen. Nach der Formulierung eines grundlegenden Modells wird dieses auf Arbeitsplätze mit mehreren Arbeitern erweitert. Zusätzlich wird der Aspekt des freiwilligen Einsetzens von Unterstützern betrachtet.

Danach werden Lösungsansätze für das formulierte Problem beschrieben. Diese gliedern sich in exakte und heuristische Verfahren. Zuletzt werden die vorgestellten Verfahren getestet und bewertet.

5.1 Modellierungen

In diesem Abschnitt wird zunächst ein Basismodell formuliert. Anschließend wird dieses dahingehend erweitert, dass auch Arbeitsplätze, an denen sich mehrere Werker abwechseln, erfasst werden können. Danach werden die Modelle so angepasst, dass - wie in der Praxis - keine Springer freiwillig eingesetzt werden. Ein besonderer Fokus wird dabei darauf gelegt, wie flexibel die Werker arbeiten und unter welchen Bedingungen ein Vorziehen von Springereinsätzen zu keiner Verbesserung führen kann.

Das Ziel dieses Abschnitts ist es, das Problem so zu modellieren, dass es auch von Standardsoftware für Optimierungsprobleme bearbeitet werden kann. Dazu werden MILPs mit diversen Anpassungen formuliert.

5.1.1 Basismodell

Der bisherige Fokus der Forschung über die Produktionsreihenfolge nach dem Mixed-Model-Sequencing Ansatz liegt auf dem allgemeinen Sequenzierungsproblem. Ein Modell, das dabei die Arbeitsbelastung betrachtet, wurde von Yano und Rachamadugu [YR91] beschrieben. Eine weitere Formulierung und Referenzen von anderen Ansätzen finden sich bei Boysen et al. [BFS09]. Ein Sequenzierungsmodell, das wie in dieser Arbeit die Anzahl der Springereinsätze als Zielfunktion verwendet, wurde von Boysen et al. [BKS11] formuliert.

Beim Literaturüberblick in Kapitel 3 wurde erwähnt, dass der ähnlichste Resequenzierungsansatz, der in der Literatur gefunden wurde, von Gujjula und Günther [GG09a] stammt. Daher lehnt sich das Basismodell, welches hier als erstes vorgestellt wird, stark an deren Modell an. Der entscheidende Unterschied liegt in der Zielfunktion. In dieser Arbeit wird in erster Linie die Anzahl der Springereinsätze betrachtet, anstatt die tatsächliche Zeit der Unterstützereinsätze zu analysieren. In Tabelle 5.1 werden die im Modell verwendeten Bezeichnungen definiert.

$$\min \sum_{j \in J} \sum_{l \in A} s_{jl} \tag{5.1a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{5.1b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{5.1c}$$

$$M \cdot s_{jl} + t_{jl} \geq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{5.1d}$$

$$o_{jl} = t_{jl} - c \qquad \forall j \in J,\ l \in A \tag{5.1e}$$

$$p_{1l} = sp_l + o_{1l} \qquad \forall l \in A \tag{5.1f}$$

$$p_{jl} = p_{j-1,l} + o_{jl} \qquad \forall j \in J,\ j \geq 2,\ l \in A \tag{5.1g}$$

$$p_{jl} \geq c \qquad \forall j \in J,\ l \in A \tag{5.1h}$$

$$p_{jl} \leq d_l \qquad \forall j \in J,\ \in A \tag{5.1i}$$

$$\sum_{j \in J} j \cdot x_{ij} + 1 \leq \sum_{j \in J} j \cdot x_{i'j} \qquad \forall i, i' \in I^P \text{mit } i < i' \tag{5.1j}$$

$$t_{jl} \geq 0 \qquad \forall j \in J,\ l \in A \tag{5.1k}$$

Mengen	
$I^W = \{1, ..., w\}$	Menge aller wiedereinzusteuernden Fahrzeuge
$I^P = \{w+1, ..., n\}$	Menge aller Planfahrzeuge
$I = I^W \cup I^P$	Menge aller Fahrzeuge
$J = \{1, ..., n\} = I$	Menge aller Positionen in der Produktionssequenz
$A = \{1, ..., L\}$	Menge aller Arbeitsplätze
Parameter	
b_{il}	Belastungszeit von Fahrzeug i an Arbeitsplatz l
c	Taktzeit
d_l	Driftgrenze an Arbeitsplatz l
sp_l	Startposition des Werkers an Arbeitsplatz l (mindestens so groß wie die Taktzeit c)
M	eine große Zahl (größer als jede Belastungszeit)
Entscheidungsvariablen	
x_{ij}	binäre Variable $\begin{cases} 1, \text{ wenn Fahrzeug } i \text{ an Position } j \text{ gesetzt} \\ 0, \text{ sonst} \end{cases}$
s_{jl}	binäre Variable $\begin{cases} 1, \text{ wenn ein Springer beim Fahrzeug an Position } j \text{ an Arbeitsplatz } l \text{ eingesetzt wird} \\ 0, \text{ sonst} \end{cases}$
Hilfsvariablen	
t_{jl}	Belastungszeit des Fahrzeugs an Position j an Arbeitsplatz l für den normalen Werker
o_{jl}	Überlastung des Werkers durch das Fahrzeug an Position j an Arbeitsplatz l
p_{jl}	Werkerposition nach Bearbeitung des Fahrzeugs an Position j an Arbeitsplatz l

Tabelle 5.1: Notation im Basismodell

$$x_{ij} \in \{0,1\} \qquad \forall i \in I, j \in J \quad (5.1l)$$

$$s_{jl} \in \{0,1\} \qquad \forall j \in J, l \in A \, (5.1m)$$

Das in diesem Modell beschriebene Ziel ist, die Anzahl der Springereinsätze zu minimieren (5.1). Die Bedingungen (5.1b) und (5.1c) garantieren, dass jedem Auftrag genau eine Position und umgekehrt auch jeder Position genau ein Auftrag zugeordnet wird. Bedingung (5.1d) sichert die Zuordnung jeder Belastung auf entweder den Werker ($t_{jl} > 0$) oder einen Springer ($s_{jl} > 0$). Die Differenz von der Belastung des Werkers und der Taktzeit bestimmt die Überlast (5.1e). Dabei entsprechen negative Werte einer Unterlast. In Takten mit Unterlast wird der Werker folglich entlastet und kann sich in seinem Arbeitsbereich zurückbewegen. In (5.1f) werden die Werkerpositionen nach dem ersten Auftrag und in (5.1g) allgemein nach jedem weiteren Takt berechnet. Bedingung (5.1h) stellt die Annahme dar, dass nicht schon vor dem Stationsbereich an einem Auftrag gearbeitet werden darf. Damit diese Bedingung erfüllt werden kann, ist es gelegentlich nötig, die Belastungszeit eines Werkers t_{jl} künstlich zu erhöhen. Die Erhöhung entspricht dann der Freizeit des Werkers, die er hat, bevor das nächste Fahrzeug den Stationsbereich erreicht. Bedingung (5.1i) verhindert den Einsatz des normalen Werkers in Überlastungssituationen. Die Einhaltung der relativen Reihenfolge unter den Aufträgen, die schon in der Plansequenz waren, wird durch (5.1j) garantiert. Auch bei Einsatz eines Springers darf die Belastung des eigentlichen Werkers nicht negativ werden (5.1k). Schließlich werden (x_{ij}) und (s_{jl}) in (5.1l) und (5.1m) als binäre Variablen definiert.

Dieses Modell beschreibt das vorliegende Problem in seiner simpelsten Struktur und wird im Folgenden erweitert, um es an verschiedene Gegebenheiten aus der Praxis anzupassen.

5.1.2 Mehrtakter

Das zuvor beschriebene Modell setzt voraus, dass an jedem Arbeitsplatz ein einzelner Werker arbeitet. Allerdings ist es in der Realität häufig unvermeidbar, an einem Arbeitsplatz Tätigkeiten durchzuführen, die im Durchschnitt länger als die Taktzeit dauern. Dies würde einen einzelnen Werker systematisch überlasten und kann da-

Parameter	
nw_l	Anzahl der Werker an Arbeitsplatz l
sp_{lk}	Startposition des k-ten Werkers an Arbeitsplatz l
Mengen	
$W_l = \{1, \ldots, nw_l\}$	Menge der Werkernummern an Arbeitsplatz l

Tabelle 5.2: Notation im Mehrtaktermodell

durch abgefangen werden, dass mehrere Werker eingesetzt werden. Wenn diese als Team gleichzeitig arbeiten, kann man das Team theoretisch als eine Person betrachten, die mit mehrfacher Geschwindigkeit arbeitet und die Modellierung kann übernommen werden. Interessanter ist es jedoch, wenn sich mehrere Werker abwechseln. Solche Arbeitsplätze werden als *Mehrtakter* bezeichnet und sie müssen gesondert modelliert werden.

Die simpelste Art die Modellierung anzupassen ist eine alternierende Aufteilung der Fahrzeuge auf die Werker. Bei einem zweitaktigem Arbeitsplatz bearbeitet dann der erste Werker alle Aufträge mit ungerader Positionsnummer und der zweite Werker alle Aufträge mit gerader Positionsnummer. Auch eine Überlast bei einem der beiden Werker wird wie zuvor durch einen Springereinsatz ausgeglichen. Dadurch kann sich der Werker, dessen Auftrag vom Springer übernommen wird, zurückarbeiten. Der andere Werker bleibt davon vollkommen unbeeinflusst.

Zur Formulierung dieser Variante werden, im Vergleich zum Basismodell, Änderungen wie in Tabelle 5.2 beschrieben, verwendet.

$$\min \sum_{j \in J} \sum_{l \in A} s_{jl} \tag{5.2a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{5.2b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{5.2c}$$

$$M \cdot s_{jl} + t_{jl} \geq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{5.2d}$$

$$o_{jl} = t_{jl} - c \cdot nw_l \qquad \forall j \in J,\ l \in A \tag{5.2e}$$

$$p_{kl} = sp_{lk} + o_{kl} \qquad \forall l \in A,\ k \in W_l \quad (5.2f)$$

$$p_{jl} = p_{j-nw_l,l} + o_{jl} \qquad \forall j \in J,\ j > nw_l,\ l \in A \quad (5.2g)$$

$$p_{jl} \geq c \cdot nw_l \qquad \forall j \in J,\ l \in A \quad (5.2h)$$

$$p_{jl} \leq d_l \qquad \forall j \in J,\ l \in A \quad (5.2i)$$

$$\sum_{j \in J} j \cdot x_{ij} + 1 \leq \sum_{j \in J} j \cdot x_{i'j} \qquad \forall i, i' \in I^P \text{mit } i < i' \quad (5.2j)$$

$$t_{jl} \geq 0 \qquad \forall j \in J,\ l \in A \quad (5.2k)$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \quad (5.2l)$$

$$s_{jl} \in \{0, 1\} \qquad \forall j \in J, l \in A \quad (5.2m)$$

Im Vergleich zum Basismodell (5.1) wird nun die verfügbare Arbeitszeit pro Fahrzeug auf $c \cdot nw_l$ erhöht und somit die Überlastung mit diesem Wert berechnet (5.2e). Für jeden der nw_l Werker an einem Arbeitsplatz wird ausgehend von der individuellen Startposition sp_{lk} die Werkerposition nach dem ersten Fahrzeug gemäß (5.2f) berechnet. Ebenso werden die folgenden Werkerpositionen nun in Schritten der Länge nw_l ermittelt (5.2g), da ein Werker jedes nw_l-te Fahrzeug bearbeitet. Die letzte Änderung wird bei der Bedingung (5.2h) vorgenommen, die weiterhin eine Arbeit vor Beginn des Arbeitsbereichs unterbindet.

Diese Modellierung der Mehrtakter ist die Grundlage der weiteren Analysen in dieser Arbeit. Dennoch soll an dieser Stelle eine alternative Modellierung dargestellt werden, die bei arbeitsorganisatorischer Umsetzung zusätzliches Optimierungspotential bietet. Dazu müssen die Werker flexibler eingesetzt werden können. Inwieweit dies in der Praxis Anwendung finden kann, ist vom Unternehmen oder sogar vom Standort innerhalb eines Unternehmens abhängig. Die Analyse der Arbeitsorganisation geht jedoch über den Rahmen dieser Arbeit hinaus, deshalb wird nun lediglich eine alternative Modellierung vorgestellt, die mögliche Verbesserung aber nicht weiter analysiert.

Die folgende alternative Modellierung von Mehrtaktern ermöglicht auch ein Vertauschen der Werker untereinander. So könnte statt eines Springereinsatzes der andere Werker einspringen, wenn er zuvor einen besonders kurzen Auftrag bearbeitet hätte.

Variablen	
w_{jlk}	binäre Variable $\begin{cases} 1, \text{ wenn Werker } k \text{ in Takt } j \text{ an Arbeitsplatz } l \text{ den} \\ \quad \text{Auftrag übernimmt} \\ 0, \text{ sonst} \end{cases}$
p_{jlk}	individuell für jeden Werker bestimmte Werkerposition
t_{jlk}	individuell für jeden Werker bestimmte Belastungszeit

Tabelle 5.3: Weitere Notation im Mehrtaktermodell

Hierfür sind weitere Definitionen nötig, die in Tabelle 5.3 genannt werden.

$$\min \sum_{j \in J} \sum_{l \in A} s_{jl} \tag{5.3a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{5.3b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{5.3c}$$

$$M \cdot (1 - w_{jlk}) + t_{jlk} \geq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.3d}$$

$$p_{1lk} = t_{jlk} + sp_{lk} - c \qquad \forall l \in A,\ k \in W_l \tag{5.3e}$$

$$p_{jlk} = t_{jlk} + p_{j-1,l,k} - c \qquad \forall j \in J,\ j \geq 2,\ l \in A,\ k \in W_l \tag{5.3f}$$

$$p_{jlk} \geq c \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.3g}$$

$$p_{jlk} \leq d_l \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.3h}$$

$$\sum_{k \in W_l} w_{jlk} + s_{jl} = 1 \qquad \forall j \in J,\ l \in A \tag{5.3i}$$

$$\sum_{j \in J} j \cdot x_{ij} + 1 \leq \sum_{j \in J} j \cdot x_{i'j} \qquad \forall i, i' \in I^P \text{mit } i < i' \tag{5.3j}$$

$$t_{jlk} \geq 0 \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.3k}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{5.3l}$$

$$s_{jl} \in \{0, 1\} \qquad \forall j \in J, l \in A \tag{5.3m}$$

$$w_{jlk} \in \{0, 1\} \qquad \forall j \in J, l \in A, k \in W_l \tag{5.3n}$$

Die Aufteilung der Arbeiten auf einen normalen Werker oder einen Springer wird durch die Gleichungen (5.3i) vorgenommen. Die Belastungen jedes einzelnen Werkers

sowie deren Positionen werden nun jeden Takt individuell berechnet. Dies geschieht durch die Nebenbedingungen (5.3d) bis (5.3g), die von den vorherigen Modellen an diese Vorgehensweise angepasst wurden.

Abschließend soll an dieser Stelle ergänzend erwähnt werden, dass die hier beschriebenen Modellierungen von Mehrtaktern nicht auf das Problem der Einsteuerung beschränkt sind. Dieselbe Idee lässt sich - und sollte bei einer praktischen Verwendung dieses Ansatzes - auch auf die Planung und damit auf allgemeine Sequenzierungsprobleme übertragen werden.

5.1.3 Vorgezogene Springereinsätze

Bisher wurde nur bestimmt, wann ein Unterstützer einspringen muss. Um das Modell realitätsnäher zu gestalten, muss allerdings zusätzlich festgelegt werden, wann ein Unterstützer einspringen darf. Genauer gesagt, sollen diese beiden Ereignisse identisch sein. Es darf nur Springereinsätze geben, wenn diese zwingend erforderlich sind. Diese Bedingung entspringt Gegebenheiten aus der Praxis, in der Unterstützer erst gerufen werden, wenn ein Werker seine Arbeit nicht im vorgegebenen Rahmen erledigen kann. Dieses Mittel stellt die letzte Möglichkeit dar einen Bandstopp zu verhindern und wird erst gewählt, wenn es aufgrund des aktuellen Belastungszustands am Band keine Alternative gibt.

Der folgende Satz liefert die Begründung, warum eine Beachtung dieses Aspekts überhaupt relevant ist:

Satz 1. *Vorgezogene Springereinsätze können zu einer Verringerung der Anzahl an Springereinsätzen führen.*

Beweis. Dieser Satz kann mittels eines einfachen Beispiels bewiesen werden: Es wird ein einzelner Arbeitsplatz betrachtet. Die Taktzeit beträgt 10 s und die erlaubte Driftzeit weitere 20 s. Die Aufträge besitzen entsprechend ihrer Montagereihenfolge die Bearbeitungszeiten 30 s, 11 s und 21 s.

Zunächst werden Springer nur eingesetzt, wenn dies zwingend notwendig ist. Die da-

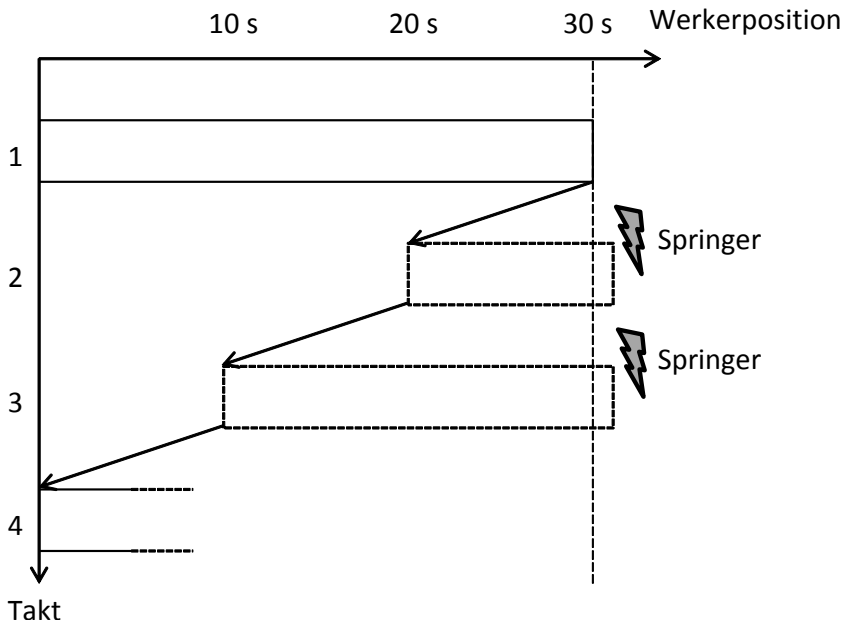


Abbildung 5.1: Werkerbewegungen ohne vorgezogene Springer

raus resultierenden Bewegungen des normalen Werkers zeigt Abbildung 5.1. Er kann das erste Fahrzeug innerhalb des Arbeitsbereichs bearbeiten, driftet dabei aber so weit, dass am zweiten Fahrzeug ein Springer eingesetzt werden muss. Der normale Werker kann sich um die Taktzeit zurückarbeiten und hat immer noch eine Verspätung von 10 s am dritten Fahrzeug. Wenn er dieses Fahrzeug bearbeiten würde, verließe er jedoch mit den zusätzlichen 21 s den Arbeitsbereich. Folglich gibt es einen weiteren Springereinsatz, also insgesamt zwei.

Alternativ kann beim ersten Fahrzeug ein Springer eingesetzt werden, obwohl dies nicht nötig ist. Der normale Werker kann danach - wie in Abbildung 5.2 dargestellt - die anderen beiden Aufträge bearbeiten. Also ist diese Variante mit nur einem Springereinsatz aus Sicht der verwendeten Zielfunktion vorteilhaft.

□

Nun werden zunächst Bedingungen hergeleitet, unter denen eine Verbesserung durch das freiwillige Vorziehen von Springereinsätzen nicht möglich ist. Dies erleichtert die Analyse für den Fall, dass die Bedingungen in der Praxis tatsächlich erfüllt sind. Für den Fall, dass diese Bedingungen nicht erfüllt sind, muss das Problem tiefer gehend analysiert werden. Dies wird in dieser Arbeit mit alternativen Modellen geschehen,

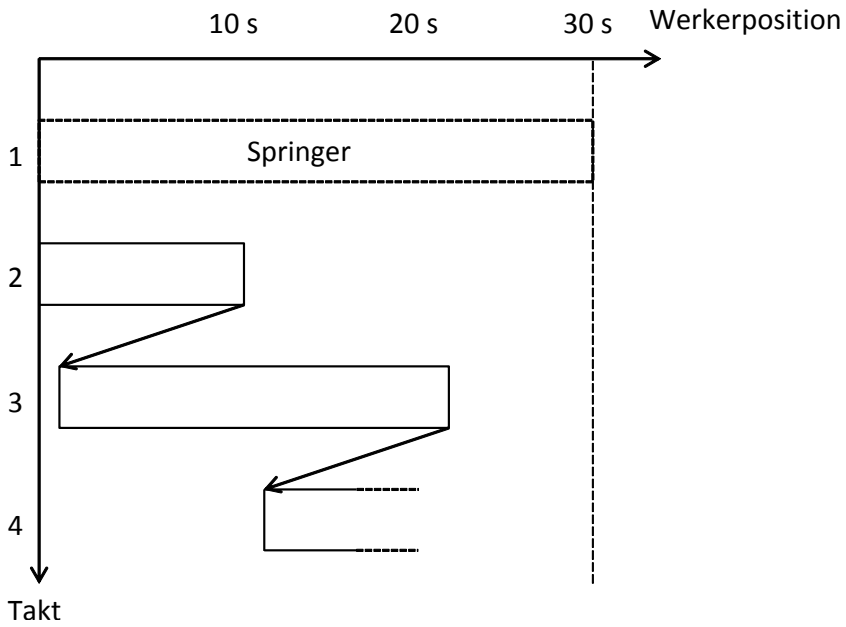


Abbildung 5.2: Werkerbewegungen mit vorgezogenem Springer

in denen das freiwillige Vorziehen von Springereinsätzen durch geeignete Nebenbedingungen verhindert wird.

Unter bestimmten Annahmen wurde von Boysen et al. [BKS11] für ein Umfeld ohne Mehrtakter gezeigt, dass eine Unterscheidung wie hier beschrieben unnötig ist, da stets eine Optimallösung existiert, die keine Springereinsätze vorzieht. Unter Optimallösung wird dabei eine optimale Allokation von Werkern und Springern zu den Arbeiten verstanden. Die entscheidende Bedingung zur Gültigkeit des Satzes bei ihnen ist, dass der Arbeitsbereich nicht größer als die doppelte Taktzeit ist.

Für diese Arbeit wird diese Bedingung nun an Mehrtakter angepasst, mit dem Ziel die Aussage zu erhalten. Dabei kann für die simple Modellierung von Mehrtaktern, bei der alle Werker unabhängig voneinander betrachtet werden, die Aussage fast ohne Veränderung übernommen werden, da für jeden Werker die gleichen Bedingungen gelten wie für einen einzeln arbeitenden Werker. Lediglich die Taktzeit muss durch die Kapazität, d. h. dem Produkt aus der Taktzeit und der Anzahl der Werker, ersetzt werden. Somit lautet die Bedingung, unter der der Satz gültig bleibt, in diesem Fall: Der Arbeitsbereich ist nicht größer als die doppelte Kapazität. Mit dieser Änderung kann jeder Werker wie im Basismodell behandelt werden. Diese Formulierung ist auch

im Basismodell korrekt, da für einen Arbeitsplatz mit einem einzelnen Werker die Kapazität der Taktzeit entspricht und stellt folglich eine einfache Erweiterung dar.

Komplizierter wird die Formulierung einer entsprechenden Aussage bei der flexiblen Verwendung der Werker, die sich gegenseitig helfen können und somit ihre Positionen vertauschen. Die Bedingung, dass der Arbeitsbereich nicht größer als die doppelte Kapazität sein darf, genügt nun nicht mehr, um die Existenz einer optimalen Lösung ohne freiwilliges Vorziehen von Springereinsätzen zu garantieren. Dies wird durch folgendes Gegenbeispiel demonstriert:

Beispiel 1. *Die Taktzeit beträgt 10 s. Es arbeiten zwei Werker, die zu Beginn die Startpositionen 0 bzw. 15 s besitzen. Der Driftbereich beträgt 25 s, sodass der Arbeitsbereich mit 35 s unterhalb der doppelten Kapazität liegt. Es sollen Aufträge mit den Bearbeitungszeiten von 34, 30, 18, 21 und 32 s in dieser Reihenfolge produziert werden.*

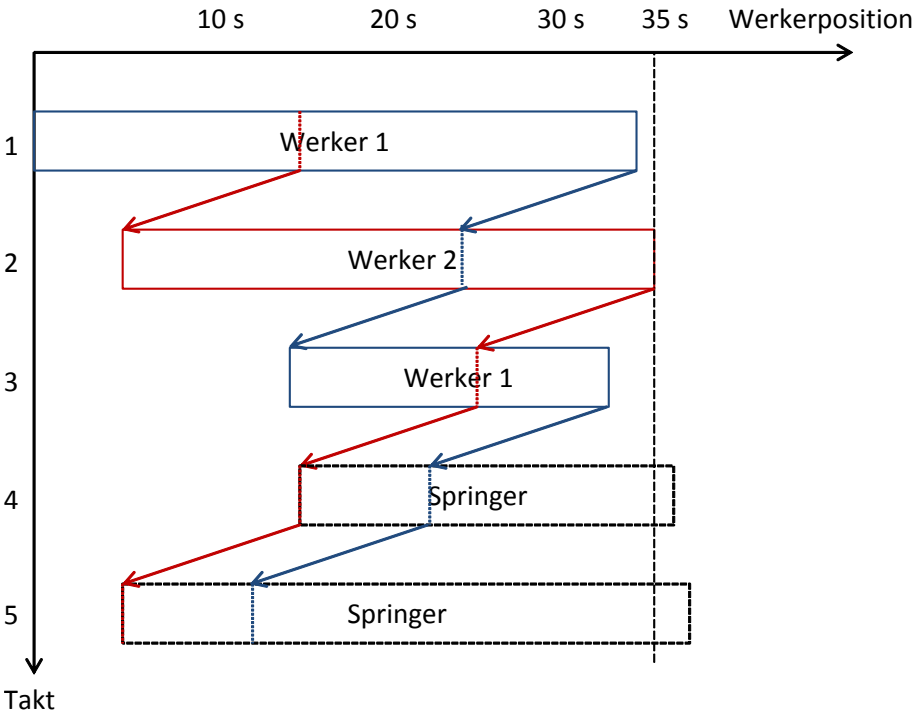


Abbildung 5.3: Werkerbewegungen ohne vorgezogene Springer

Wenn die ersten drei Fahrzeuge von den regulären Werkern übernommen werden, d. h. der erste Werker bearbeitet das erste und dritte Fahrzeug, driften beide Werker so weit, dass sie weder das vierte noch das fünfte Fahrzeug ohne Überschreiten der Driftgrenze

bearbeiten können. Abbildung 5.3 zeigt die schematische Darstellung der Werkerbewegungen. Deshalb sind nun zwei Springereinsätze nötig.

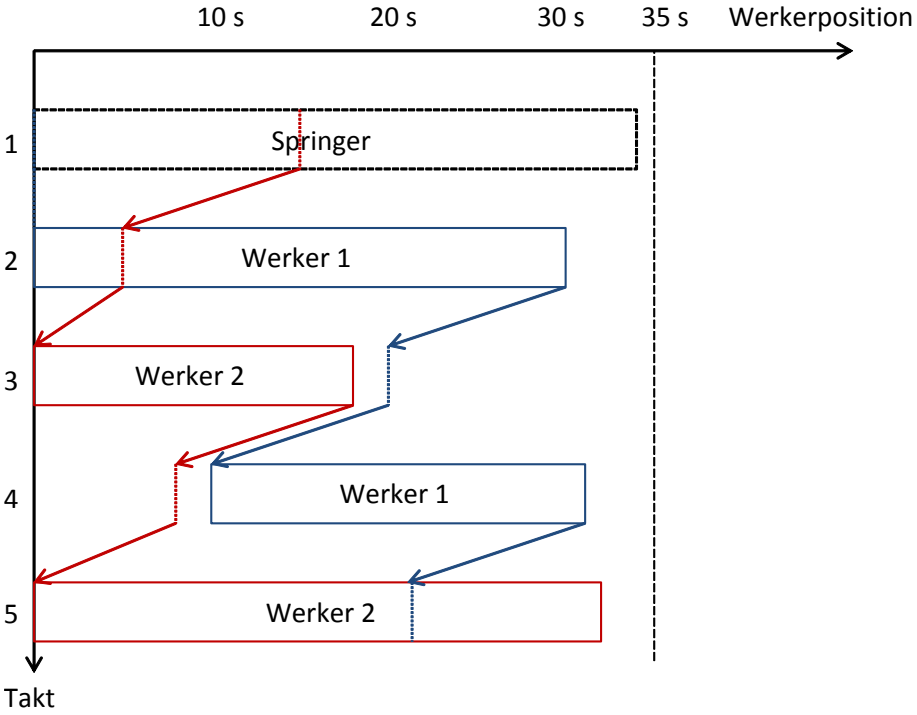


Abbildung 5.4: Werkerbewegungen mit vorgezogenem Springer

Das freiwillige Vorziehen eines Springereinsatzes zum ersten Fahrzeug führt dazu, dass die normalen Werker - wie in Abbildung 5.4 dargestellt - die restlichen vier Fahrzeuge abwechselnd ohne Überschreitung der Driftgrenze bearbeiten können. Dabei ist zu beachten, dass es zu einer Vertauschung der ursprünglichen Reihenfolge der Werker kommt. Der erste Werker übernimmt bei seinem ersten Einsatz ein Fahrzeug, dass eigentlich der zweite Werker bearbeiten würde, der jedoch erst verspätet mit der Bearbeitung beginnen könnte. Somit kann mit dieser Steuerung ein Springereinsatz vermieden werden.

Es wurde also gezeigt, dass unter der Annahme, dass die Werker ihre Positionen tauschen können, eine stärkere Bedingung nötig ist, um die Möglichkeit einer Verbesserung der Zielfunktion durch Vorziehen eines Springers auszuschließen. Deshalb wird nun die folgende Bedingung betrachtet:

Der Arbeitsbereich ist nicht größer als die Summe aus Kapazität und Taktzeit.

Formal kann diese Bedingung durch

$$d_l \leq (nw_l + 1) \cdot c \qquad \forall\, l \in A \quad (5.4)$$

ausgedrückt werden. Diese Bedingung impliziert direkt, dass kein Auftrag eine Bearbeitungszeit haben darf, die größer als die Summe aus Kapazität und Taktzeit ist.

Mit dieser Bedingung werden zunächst zwei Lemmata und schließlich ein Satz, der der zuvor genannten Aussage ähnelt, bewiesen. Grundlage für die folgenden Aussagen ist das Szenario, in dem immer der Werker mit der niedrigsten Werkerposition den nächsten Auftrag übernimmt.

Lemma 1. *Es sei ein Mehrtakterarbeitsplatz mit k Werkern gegeben. Bei Einsatz eines Unterstützers kann in den nächsten k Takten von den normalen Werkern an der linken Stationsgrenze gestartet werden.*

Beweis. Es sei t der Takt, in dem ein Unterstützer eingesetzt wird. Die Werkerpositionen der normalen Werker in diesem Takt seien durch $x(t) = [x_1(t), ..., x_k(t)]$ gegeben. Die Einträge in diesem Vektor seien geordnet nach ihrem letztem Einsatz vor Takt t, d. h. nach dem Takt, in dem der letzte von ihnen bearbeitete Auftrag angesetzt war.
$\Rightarrow$ Werker i, $(i = 1, \ldots, k)$ hat spätestens im Takt $t - 1 - k + i$ mit seinem letzten Auftrag begonnen. Der Arbeitsbereich sei durch $s \leq (k + 1) \cdot c$ gegeben.
$\Rightarrow x_i(t) \leq s - (k + 1 - i) \cdot c$
$\Rightarrow x_i(t + i) \leq \max\{s - (k + 1 - i) \cdot c - i \cdot c, 0\} = \max\{s - (k + 1) \cdot c, 0\} = 0$
$\Rightarrow$ Die Werker können in der in diesem Beweis genutzten Reihenfolge eingesetzt werden, ohne mit aus vergangenen Aufträgen übertragener Drift vorbelastet zu sein. Eine Vertauschung der Werker wäre zwar möglich, wenn mehrere Werker eine Position von 0 hätten, dies würde jedoch implizieren, dass der eigentliche Werker danach den nächsten Auftrag ohne Startdrift übernehmen könnte.
$\Rightarrow$ Es steht in den k Takten nach Takt t stets mindestens ein Werker an der linken Stationsgrenze bereit.

□

Lemma 2. *Es sei ein Mehrtakterarbeitsplatz mit k Werkern gegeben. Es seien $x(t) = [x_1(t), ..., x_k(t)]$ und $y(t) = [y_1(t), ..., y_k(t)]$ zwei Vektoren mit möglichen Werkerpositionen nach einem beliebigen, aber fixierten Takt t. Außerdem gelte für alle Werker i,*

dass $x_i(t) \leq y_i(t)$. Dann gilt auch für alle anderen Takte $t' \geq t$ und alle Werker i, solange weder bei x noch bei y ein Springereinsatz erzwungen wird und nachdem die Vektoreneinträge in x und y entsprechend ihrer Größe geordnet wurden:

$$x_i(t') \leq y_i(t')$$

Beweis. Es wird Takt $t+1$ betrachtet. Angenommen es gibt weder bei x noch bei y einen Springereinsatz, so gilt: $x_1(t+1) = \max\{x_1(t) + b_{t+1} - c, 0\}$ und $y_1(t+1) = \max\{y_1(t) + b_{t+1} - c, 0\}$.
$\Rightarrow x_1(t+1) \leq y_1(t+1)$
Nach diesem Takt können die Einträge in den Vektoren wieder entsprechend ihrer Größe geordnet werden. Die neuen Vektoren werden mit x' und y' bezeichnet. Es seien $x_1(t+1) = x'_m$ und $y_1(t+1) = y'_n$.
Fall I: $m \geq n$
Für $i = 1, \ldots, n-1$ und $i = m+1, \ldots, k$ folgt $x'_i \leq y'_i$ direkt aus den Voraussetzungen.
Für $i = n, \ldots, m$ gilt: $x'_i \leq x'_m \leq y'_n \leq y'_i$.
Fall II: $m < n$
Für $i = 1, \ldots, m-1$ und $i = n+1, \ldots, k$ folgt $x'_i \leq y'_i$ direkt aus den Voraussetzungen.
Für $i = m+1, \ldots, n$ gilt: $x'_i \leq y'_{i-1} \leq y'_i$.
Schließlich gilt für $i = m$: $x'_i \leq x'_{i+1} \leq y'_i$.
Folglich gilt die Voraussetzung nach einem Takt erneut. Die Behauptung folgt nun mit Induktion.

□

Satz 2. *Unter der oben genannten Bedingung* (5.4) *gilt: Es gibt immer eine optimale Allokation von Werkern und Springern zu den Arbeiten, bei der keine Springereinsätze freiwillig vorgezogen werden.*

Beweis. Es wird ein einzelner Arbeitsplatz mit k Werkern betrachtet. Dabei wird angenommen, es gebe eine optimale Lösung, in der ein Springereinsatz freiwillig vorgezogen wird und der Zielfunktionswert kleiner ist als in dem Szenario, in dem nur erzwungene Springereinsätze zugelassen werden. Es sei t der erste Takt, in dem ein solcher Springereinsatz stattfindet. Außerdem sei $t+i$ der Takt, in dem ein Springereinsatz durch Überschreitung der Driftgrenze erzwungen wäre, wenn nicht vorher freiwillig ein

Springer eingesetzt worden wäre. Diesen Takt muss es geben, sonst wäre eine Lösung mit freiwilligem Springereinsatz nicht optimal. $\lightning$

Mit Lemma 1 folgt, dass in den Takten $t + i + 1$ bis $t + i + k$ die normalen Werker an der linken Stationsgrenze starten können und somit auch keine Springereinsätze nötig sind. Wenn in der optimalen Lösung hier erneut Springereinsätze stattfinden, kann die Beweisführung weiter nach hinten verlagert werden bis hinter den nächsten erzwungenen Springereinsatz. Also kann ohne Beschränkung der Allgemeinheit angenommen werden, dass auch in der optimalen Lösung kein weiterer Springereinsatz bis zum Takt $t + i + k$ stattfindet.

Da in dieser Lösung auch die Takte $t + i + 1$ bis $t + i + k$ von den normalen Werken übernommen werden, können die Werkerpositionen hier nicht kleiner sein als bei der Variante mit ausschließlich erzwungenen Springereinsätzen, in der die Werker diese Aufträge alle an der linken Stationsgrenze starten können. D. h. für die nach ihrer Größe sortierten Werkerpositionen nach Takt $t + i + k$ gilt, dass paarweise verglichen die Werte der Methode mit ausschließlich erzwungenen Springereinsätzen nie größer sind als die jeweils anderen.

Mit Lemma 2 folgt nun, dass der nächste erzwungene Springereinsatz nicht ohne Springereinsatz bei der anderen Variante existieren kann. D. h. entweder diese Variante ist auch optimal oder es folgt ein neuer Springereinsatz nur bei der freiwilligen Variante. Dann können die Beweisschritte jedoch einfach wiederholt werden bis zum Ende der gesamten Sequenz. Es kann jedoch nie passieren, dass in der Variante mit ausschließlich erzwungenen Springereinsätzen nach einem Takt ein Springereinsatz mehr als bei der anderen Variante notwendig ist.

Diese Beweisführung gilt unabhängig für alle Arbeitsplätze, da sich diese nicht gegenseitig beeinflussen.

□

Nun sind also Bedingungen bekannt, unter denen die zuvor formulierten Modelle verwendet werden können, auch wenn keine Springereinsätze freiwillig vorgezogen werden sollen. Diese sollten also bei der Optimierung einer neuen Montagelinie überprüft werden, um bei Einhaltung die weitere Analyse zu vereinfachen.

Die beschriebenen Bedingungen treffen in der Realität - wie z. B. bei den in dieser Arbeit verwendeten Daten - nicht auf sämtliche Arbeitsplätze zu. Deshalb wird nun das Modell dahingehend erweitert, dass ein freiwilliges Vorziehen von Springereinsätzen unterbunden wird. Dazu ist es notwendig, dass nicht nur untere Schranken für die Werkerpositionen beachtet werden, sondern die exakten Positionen. Dazu werden in den folgenden Modellen die Belastungen der Werker immer nach oben und unten exakt abgeschätzt und daraus bestimmt, ob ein Werker eine Wartezeit hat, bevor das nächste Fahrzeug die Station erreicht.

Für diese Berechnungen wird eine zusätzliche Variable benötigt[1].

$$r_{jl} : \text{binäre Variable} \begin{cases} 1, \text{ wenn der Werker an Station } l \text{ in Takt } j+1 \text{ nicht an der} \\ \quad \text{linken Stationsgrenze mit der Arbeit beginnen kann} \\ 0, \text{ sonst} \end{cases}$$

Außerdem wird eine weitere Konstante eingeführt:

m : kleine reelle Zahl, kleiner als der kleinste verwendete Teil einer Zeiteinheit bei den Belastungszeiten

Zur Reduzierung der Variablennamen übernimmt p_{-1l} die Rolle von sp_l.

Zunächst wird das Modell ohne Mehrtakter erweitert.

$$\min \sum_{j \in J} \sum_{l \in A} s_{jl} \tag{5.5a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{5.5b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{5.5c}$$

$$M \cdot s_{jl} + t_{jl} \geq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{5.5d}$$

$$t_{jl} \leq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{5.5e}$$

$$t_{jl} \leq M \cdot (1 - s_{jl}) \qquad \forall j \in J,\ l \in A \tag{5.5f}$$

[1]Die grundlegende Idee hinter der Veränderung des Modells wurde von einem unveröffentlichten Modell von Rico Gujjula übernommen, das nach privater Korrespondenz vorliegt.

$$o_{jl} = t_{jl} - c \qquad \forall j \in J,\ l \in A \quad (5.5g)$$

$$p_{jl} \geq p_{j-1,l} + o_{jl} \qquad \forall j \in J,\ l \in A \quad (5.5h)$$

$$p_{jl} \leq p_{j-1,l} + o_{jl} + (1 - r_{jl}) \cdot M \qquad \forall j \in J,\ l \in A \quad (5.5i)$$

$$p_{jl} \leq r_{jl} \cdot M + c \qquad \forall j \in J,\ l \in A \quad (5.5j)$$

$$p_{jl} \geq c \qquad \forall j \in J,\ l \in A \quad (5.5k)$$

$$p_{jl} \leq d_l \qquad \forall j \in J,\ l \in A \quad (5.5l)$$

$$\sum_{j \in J} j \cdot x_{ij} + 1 \leq \sum_{j \in J} j \cdot x_{i'j} \qquad \forall i, i' \in I^P \text{mit } i < i' \quad (5.5m)$$

$$s_{jl} \cdot (d_l + c + m) \leq p_{j-1,l} + o_{jl} + c \qquad \forall j \in J,\ l \in A \quad (5.5n)$$

$$t_{jl} \geq 0 \qquad \forall j \in J,\ l \in A \quad (5.5o)$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \quad (5.5p)$$

$$s_{jl} \in \{0, 1\} \qquad \forall j \in J, l \in A \quad (5.5q)$$

$$r_{jl} \in \{0, 1\} \qquad \forall j \in J, l \in A \quad (5.5r)$$

Wenn in Takt j an Arbeitsplatz l ein Springer eingesetzt wird, führen die Bedingungen (5.5f) und (5.5o) dazu, dass für die Belastung des normalen Werkers $t_{jl} = 0$ gilt. Ohne Springereinsatz führen die Bedingungen (5.5d) und (5.5e) dazu, dass die reale Belastung exakt übergeben wird $t_{jl} = \sum_{i \in I} x_{ij} \cdot b_{il}$. Aufgrund dieser Bedingungen wird immer die genaue Werkerposition für jeden Takt ermittelt.

Dabei garantieren bei Freizeit des Werkers (5.5i) und (5.5k), dass $r_{jl} = 0$ gesetzt wird und zusammen mit (5.5j), dass schließlich auch die Position des Werkers auf c gesetzt wird. Wenn der Werker am nächsten Fahrzeug ohne Pause weiterarbeiten muss, wird wegen (5.5h) und (5.5j) $r_{jl} = 1$ gesetzt, sodass mit (5.5i) $p_{jl} = p_{j-1,l} + o_{jl}$ gilt.

Mit dieser exakten Berechnung der Werkerpositionen kann genau bestimmt werden, wann ein Einsatz des normalen Werkers zu einer Überschreitung der Driftgrenze führen würde. Die Ungleichungen (5.5n) garantieren, dass nur in diesen Fällen ein Springer eingesetzt werden darf.

Durch ergänzen der zuvor beschriebenen Nebenbedingungen kann auch das einfache Mehrtaktermodell auf die gleiche Weise erweitert werden, um das freiwillige Vorziehen von Springereinsätzen zu unterbinden.

Bei dem Modell mit vertauschbaren Werken unterscheiden sich die notwendigen Anpassungen etwas. Deshalb wird dieses Modell in der erweiterten Form genannt. Dazu wird eine Variable erneut angepasst.

$$r_{jlk} : \text{binäre Variable} \begin{cases} 1, \text{ wenn der } k\text{-te Werker an Station } l \text{ in Takt } j+1 \text{ nicht an} \\ \quad \text{der linken Stationsgrenze mit der Arbeit beginnen kann} \\ 0, \text{ sonst} \end{cases}$$

$$\min \sum_{j \in J} \sum_{l \in A} s_{jl} \tag{5.6a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{5.6b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{5.6c}$$

$$M \cdot (1 - w_{jlk}) + t_{jlk} \geq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6d}$$

$$t_{jlk} \leq \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6e}$$

$$t_{jlk} \leq M \cdot w_{jlk} \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6f}$$

$$p_{jlk} \geq t_{jlk} + p_{j-1,l,k} - c \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6g}$$

$$p_{jlk} \leq t_{jlk} + p_{j-1,l,k} - c + (1 - r_{jlk}) \cdot M \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6h}$$

$$p_{jlk} \leq r_{jlk} \cdot M + c \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6i}$$

$$p_{jlk} \geq c \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6j}$$

$$p_{jlk} \leq d_l \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6k}$$

$$\sum_{k \in W_l} w_{jlk} + s_{jl} = 1 \qquad \forall j \in J,\ l \in A \tag{5.6l}$$

$$\sum_{j \in J} j \cdot x_{ij} + 1 \leq \sum_{j \in J} j \cdot x_{i'j} \qquad \forall i, i' \in I^P \text{mit } i < i' \tag{5.6m}$$

$$s_{jl} \cdot (d_l + c + m) \leq p_{j-1,l,k} + t_{jlk} \qquad \forall j \in J,\ l \in A, k \in W_l \tag{5.6n}$$

$$t_{jlk} \geq 0 \qquad \forall j \in J,\ l \in A,\ k \in W_l \tag{5.6o}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{5.6p}$$

$$s_{jl} \in \{0, 1\} \qquad \forall j \in J, l \in A \tag{5.6q}$$

$$w_{jlk} \in \{0, 1\} \qquad \forall j \in J, l \in A, k \in W_l \tag{5.6r}$$

$$r_{jlk} \in \{0,1\} \qquad \forall j \in J, l \in A, k \in W_l \quad (5.6s)$$

Mit diesen Anpassungen wird ein freiwilliges Vorziehen von Springern verhindert.

Dennoch gibt es ein weiteres Problem in diesem Modell. Es ist möglich, dass ein Werker freiwillig einen Auftrag nicht bearbeitet, obwohl er könnte. Stattdessen wartet er, bis ein anderer Werker seine Arbeit beendet hat und dann diesen Auftrag (verspätet) übernimmt. Der wartende Werker kann dann den nächsten Auftrag ohne Verspätung übernehmen. Die Wirkweise dieses Vorgehens wird durch ein Beispiel illustriert.

Beispiel 2. *Es wird ein Zweitakterarbeitsplatz betrachtet. Die Taktzeit ist 10 s, die zusätzlich erlaubte Drift beträgt 30 s. Also ergibt sich insgesamt ein Arbeitsbereich von 40 s. Ein Werker ist direkt verfügbar, der andere noch 15 s beschäftigt. Es sollen Aufträge mit den Bearbeitungszeiten von 25, 10 und 36 s bearbeitet werden.*

Bei der Variante, dass ein Werker, der Zeit hat, einen Auftrag übernehmen muss, übernimmt der erste Werker den ersten und der zweite Werker den zweiten Auftrag, sodass beide bis 25 s arbeiten müssen. Danach kann keiner den nächsten Auftrag übernehmen, ohne die Driftgrenze zu überschreiten. Folglich ist ein Springereinsatz nötig, wie in Abbildung 5.5 dargestellt ist.

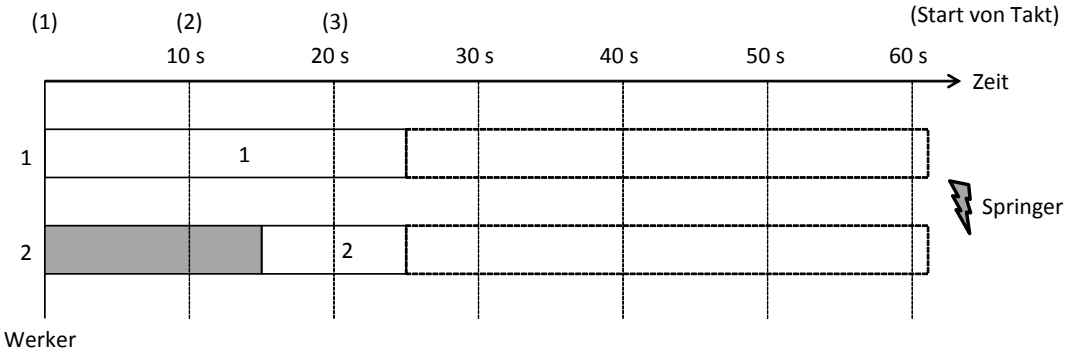


Abbildung 5.5: Auftragsverteilung ohne freiwilliges Warten

Falls das Warten auf einen späteren Auftrag erlaubt ist, kann der erste Werker auf den zweiten Auftrag warten, damit er mit der Arbeit an diesem direkt beim Eintreten in die Station beginnen kann. Abbildung 5.6 verdeutlicht, dass dann auch der zweite Werker noch genug Zeit hat, um den ersten Auftrag zu erledigen. Der erste Werker wird durch das Auslassen des ersten Auftrags so früh mit dem zweiten fertig, dass er

mit dem dritten rechtzeitig starten kann, um diesen vor der Driftgrenze zu beenden. Somit kann der Springereinsatz verhindert werden.

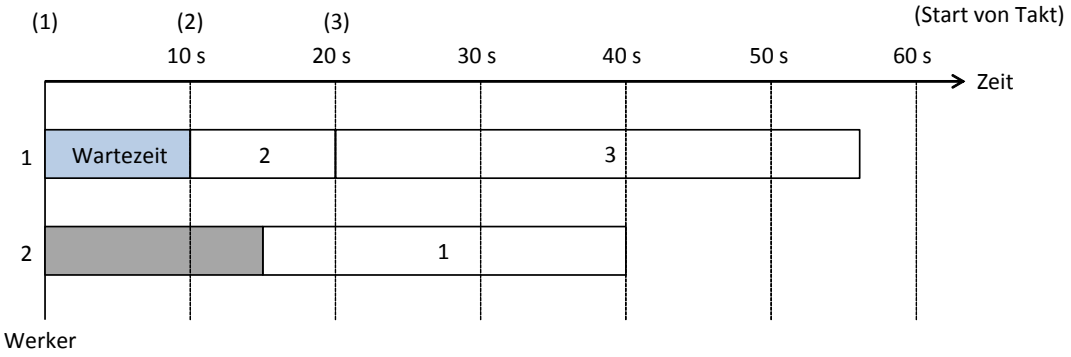


Abbildung 5.6: Auftragsverteilung mit freiwilligem Warten

In der Praxis ist ein solches Vorgehen nicht umsetzbar, daher sollte ein Modell entwickelt werden ohne freiwilliges Warten der Werker. Das vorherige Modell müsste dazu um Nebenbedingungen ergänzt werden, die garantieren, dass immer einer der Werker mit der niedrigsten Position den nächsten Auftrag übernimmt. Eine solche Nebenbedingung kann wie folgt aussehen:

$$M \cdot w_{jlk} - M \leq p_{j-1,l,k'} - p_{j-1,l,k} \qquad \forall j \in J, l \in A, k, k' \in W_l, k \neq k'$$

Durch eine solche Berechnung der individuellen Werkerpositionen mit der Möglichkeit, die Reihenfolge der Werker bei Mehrtaktern zu vertauschen, steigt natürlich auch der Rechenaufwand. Wie hier als MILP formuliert, ist eine Lösung von realitätsnahen Problemen in akzeptabler Zeit nahezu unmöglich. Jedoch sollte dieser Ansatz nicht komplett ignoriert werden, da bei einer iterativen Berechnung der Zielfunktion für eine gegebene Sequenz der Rechenaufwand nicht um ein Vielfaches im Vergleich zum ersten Ansatz steigt. Für eine solche Variante kann die Wahl des Werkers nach seiner Position eine sinnvolle Methode sein, um den Werkern ihre Aufträge zuzuordnen.

Da der Schwerpunkt dieser Arbeit auf der Anwendung von heuristischen Verfahren liegt, kann eine Formulierung mit nichtlinearen Nebenbedingungen verwendet werden. Der Vorteil dieser alternativen Beschreibung des Problems liegt in erster Linie in der intuitiveren Art und damit in der besseren Verständlichkeit für den Leser. Deshalb wird nun das Modell, das für die im weiteren Verlauf der Arbeit vorgestellten Verfahren

zugrunde liegt, in einer anderen Form als die bisherigen Modelle vorgestellt. Dabei übernimmt nun zur Vereinfachung die Variable p_{jl} für $j = -nw_l+1, \ldots, 0$ die Rolle der Startpositionen der einzelnen Werker. Es gilt, dass die Startpositionen eine Vorarbeit ausschließen, also $p_{jl} \geq nw_l \cdot c$ für $j = -nw_l + 1, \ldots, 0$. Ebenso erhält t_{jl} eine etwas andere Bedeutung. Die Berechnung dieser Größe als Belastungszeit des Fahrzeugs an Position j an Arbeitsplatz l geschieht nun unabhängig davon, ob ein Springer eingesetzt wird oder nicht.

$$\min \sum_{j \in J} \sum_{l \in A} s_{jl} \tag{5.7a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{5.7b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{5.7c}$$

$$t_{jl} = \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{5.7d}$$

$$s_{jl} = \begin{cases} 1, \text{ wenn } p_{j-nw_l,l} - nw_l \cdot c + t_{jl} > d_l \\ 0, \text{ sonst} \end{cases} \qquad \forall j \in J,\ l \in A \tag{5.7e}$$

$$p_{jl} = \max\left(p_{j-nw_l,l} - nw_l \cdot c + (1 - s_{jl}) \cdot t_{jl}, nw_l \cdot c\right) \qquad \forall j \in J,\ l \in A \tag{5.7f}$$

$$\sum_{j \in J} j \cdot x_{ij} + 1 \leq \sum_{j \in J} j \cdot x_{i'j} \qquad \forall i, i' \in I^P \text{mit } i < i' \tag{5.7g}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{5.7h}$$

Die Zielfunktion sowie die ersten beiden und letzten beiden Nebenbedingungen sind aus den vorherigen Modellen bekannt. Die Gleichungen (5.7d) wandeln die Belastungszeiten der Fahrzeuge in Belastungszeiten der einzelnen Positionen um. Die Verwendung von Springern wird durch Bedingung (5.7e) eindeutig aus den Werkerpositionen und den Belastungszeiten abgeleitet. Die Gleichungen (5.7f) berechnen die Werkerpositionen nach jedem Takt. Im Fall einer geringen Arbeitsbelastung im aktuellen Takt, die zu einer Pause vor dem nächsten Auftrag führt, wird diese schon auf die Werkerposition zugeschlagen. Deshalb entspricht sie mindestens der Kapazität am jeweiligen Arbeitsplatz.

5.2 Lösungsansätze

Nun werden Lösungsansätze vorgestellt, die auf die zuvor modellierten Probleme angewandt werden können. Zunächst werden exakte Verfahren genannt, die garantiert Optimallösungen bestimmen können. Danach wird auf heuristische Methoden, die auch unter realen Bedingungen eingesetzt werden können, eingegangen.

5.2.1 Exakte Lösungsverfahren

Zur exakten Lösung des vorliegenden Problems wurden zwei Varianten verfolgt. Zum einen wurde die Anwendbarkeit von Standardsolvern auf verschiedene kleine Probleminstanzen getestet. Zum anderen wurde ein einfacher Brute-Force Algorithmus im Vergleich dazu analysiert.

5.2.1.1 Standardsolver

Die Verwendung von Programmlösern aus gängiger Optimierungssoftware für MILPs stellte sich als wenig erfolgversprechend für den Praxiseinsatz heraus. Sowohl IBM ILOG CPLEX als auch Gurobi konnten die meisten Testinstanzen des vorliegenden Problems nicht in akzeptabler Zeit lösen[2]. Selbst für das Wiedereinsteuern eines einzigen Auftrags in eine Sequenz von 20 weiteren Aufträgen benötigte CPLEX schon mehrere Sekunden (siehe Tabelle 5.4 am Ende dieses Abschnitts). Schon ab zwei wiedereinzusteuernden Aufträgen wird die vorgegebene Zeit von 30 Sekunden überschritten. Bei diesen Tests wurde das erste in dieser Arbeit vorgestellte Modell für Mehrtakter (5.2) verwendet, das auch das freiwillige Vorziehen von Springern erlaubt.

Aufgrund dieser Ergebnisse können die erwähnten Programmlöser lediglich als Mittel zur Berechnung von Schranken eingesetzt werden. Dies ist allerdings mit einem erheblichen Zeitaufwand verbunden, da Instanzen mit vier oder fünf wiedereinzusteuernden Aufträgen mehrere Stunden Rechenzeit benötigen und auch diese Zeit oft nicht ausreicht, um eine Optimallösung zu bestimmen. Dabei hängt es von der jeweiligen

[2]Bei Verwendung eines Intel®Core™2 Duo Prozessors mit 2,53 GHz.

Instanz ab, ob CPLEX oder Gurobi schneller eine Lösung berechnet bzw. in vorgegebener Zeit die bessere Lösung oder die beste untere Schranke produziert[3].

Das größte Hindernis bei der Verwendung eines Programmlösers ist die Tatsache, dass sogar für eine gegebene Sequenz die Verteilung der Arbeit auf Werker und Springer ein nicht-triviales Problem darstellt. In Abschnitt 5.1.3 wurde gezeigt, dass dabei unter Umständen Springer eingespart werden können, wenn diese früher als zwingend notwendig eingesetzt werden. Folglich sind zwei voneinander abhängige Optimierungsprobleme (Reihenfolgebildung und Arbeitsverteilung) kombiniert zu lösen, obwohl in der Realität nur das erste zu lösen ist und sich die Lösung des zweiten als direkte Konsequenz daraus ergibt. Die einfache iterative Berechnung der Werkerpositionen lässt sich nicht durch lineare Gleichungen ausdrücken. Das führt dazu, dass eine große Anzahl an binärer Variablen nötig wird, die zu einer enormen Komplexität und Rechenzeit führen. Eine Formulierung wie in Modell (5.6), die das Problem auf die Reihenfolgebildung reduziert, ist aufgrund der gestiegenen Variablenzahl noch langsamer in der Lösungsberechnung. Deshalb wird die Verwendung solcher Programme in dieser Arbeit nicht weiter verfolgt.

5.2.1.2 Brute-Force

An dieser Stelle wird das trivialste aller Verfahren beschrieben. Im Rahmen einer *erschöpfenden Suche* werden nacheinander alle möglichen Sequenzen gebildet und für jede einzelne Sequenz der Zielfunktionswert iterativ bestimmt, es wird also eine *vollständige Enumeration* des Lösungsraumes durchgeführt. Dabei wird keine intelligente Suchstrategie verwendet und keinerlei Wissen über das Problem ausgenutzt. Der Vorteil ist, dass dieses Verfahren einfach und die Rechenzeit sehr gut prognostizierbar ist. Der offensichtliche Nachteil ist, dass diese Rechenzeit mit wachsender Problemgröße exponentiell ansteigt.

Für den Praxiseinsatz können mit dieser Methode nur kleine Instanzen gelöst werden. Bei der beschriebenen Problemstellung lassen sich in der vorgegebenen Zeit auf einer typischen Montagelinie maximal drei Fahrzeuge in eine Sequenz von 20 optimal

[3]Dies ist ein Ergebnis einer am DS&OR Lab, Universität Paderborn durchgeführten Studie.

eingliedern. Dieses Problem besitzt insgesamt 10626 Lösungen, die getestet werden müssen. Die Bewertung einer Lösung benötigt unter den Testbedingungen[4] eine knappe Millisekunde, sodass das beschriebene Problem in unter zehn Sekunden optimal gelöst werden kann.

Auch für mittelgroße Probleminstanzen, die nicht in der durch die Praxis vorgegebenen Zeit gelöst werden können, kann diese Methode dennoch dazu verwendet werden, um eine optimale Lösung zu bestimmen. Diese dient dann als Benchmark für andere Verfahren, damit deren Lösungsqualität bewertet werden kann.

Für große Probleminstanzen ist dieses Verfahren jedoch komplett unbrauchbar, da die benötigte Rechenzeit - wie beschrieben - mit wachsender Problemgröße rapide ansteigt und ein vorzeitiges Abbrechen der Lösungssuche bei einem solchen Verfahren zu einem nicht aussagekräftigen Ergebnis führt.

5.2.1.3 Vergleich der Ansätze

Wie die beschriebenen Aspekte nahelegen, ist der Brute-Force Ansatz in allen getesteten Instanzen deutlich schneller als die Verwendung eines Standardsolvers. Die folgende Tabelle 5.4 enthält die Rechenzeiten der jeweiligen Verfahren für das Wiedereinsteuern von einer bestimmten Anzahl an Fahrzeugen in eine Sequenz von 20 Fahrzeugen. Dabei wurde die Suche nach einer Stunde abgebrochen, wenn noch kein Optimum bestimmt wurde.

$\lvert I^W \rvert$	Brute-Force	CPLEX
1	< 1	$4 - 9$
2	< 1	$49 - 524$
3	$6 - 7$	$874 - > 3600$
4	$165 - 170$	> 3600
5	> 3600	> 3600

Tabelle 5.4: Rechenzeiten zur Bestimmung von Optimallösungen (in s)

Neben dem klaren Schnelligkeitsvorteil des Brute-Force Ansatzes zeigt sich ein weiterer Vorteil. Die benötigte Rechenzeit kann gut abgeschätzt werden, da die Varianz und

[4] Verwendung eines Intel®Core™ 2 Duo Prozessors mit 2,53 GHz.

damit die Abhängigkeit von der Probleminstanz sehr gering ist. Dagegen ist es bei der Verwendung von CPLEX unmöglich die Rechenzeit vorherzusagen.

Von der Entwicklung von anspruchsvolleren Verfahren, die eine Weiterentwicklung des Brute-Force Ansatzes darstellen, wird abgesehen. Zwar könnte durch Ausnutzung der Problemstruktur und durch Speicherung von Zwischenwerten die Zielfunktionsberechnung für ähnliche Sequenzen verkürzt werden. Allerdings sind die Problemgrößen, die in der Praxis gelöst werden müssen, zu groß, als dass ein solcher Ansatz vielversprechend wäre. Deshalb wird im nächsten Kapitel auf die Verwendung von Heuristiken eingegangen.

5.2.2 Heuristische Lösungsverfahren

Nachdem analysiert wurde, wie und unter welchen Voraussetzungen eine optimale Lösung ermittelt werden kann, werden nun Heuristiken betrachtet. Darunter versteht man „Optimierungsmethoden, die versuchen problemspezifisches Wissen auszunutzen und für die keine Garantie besteht, dass sie die optimale Lösung finden“ [Rot11].

Es wird zunächst eine konstruktive Heuristik beschrieben und insbesondere auf eine Schwäche, die als Anomalie bezeichnet wird, eingegangen. Danach werden zwei lokale Suchverfahren genannt und Möglichkeiten aufgezeigt, wie diese in Metaheuristiken angewandt werden können. Zum Abschluss dieses Abschnitts wird eine Variante erläutert, wie mit einer großen Anzahl gleichzeitig wiedereinzusteuernder Aufträge umgegangen werden kann.

5.2.2.1 Konstruktive Heuristik

Die einfachste Art mehrere Objekte in eine gegebene Sequenz einzugliedern ist es, diese nacheinander an eine Position zu setzen. Genauso wird auch hier vorgegangen, um schnell eine Lösung zu erhalten. Ein Fahrzeug nach dem anderen kann an die jeweils im aktuellen Zustand optimale Position gesetzt werden. Dazu muss lediglich für jede mögliche Position die gesamte Sequenz bewertet werden. Das Verfahren wird als *sequentielles Einsteuern* bezeichnet und durch Algorithmus 5.1 formal beschrieben.

Algorithmus 5.1: Sequentielles Einsteuern (SE)

```
Initialisierung;
p := |I^P|;
für jedes Fahrzeug i ∈ I^W tue
    f* := ∞;
    für alle Positionen j ∈ {1, ..., p + 1} tue
        setze Fahrzeug i an die Position j der aktuellen Plansequenz;
        berechne den Zielfunktionswert f(j) für die Sequenz;
        wenn f(j) < f* dann
            j* := j;
            f* := f(j);
        Ende
        entferne Fahrzeug i aus der Sequenz;
    Ende
    setze Fahrzeug i an die Position j*;
    p := p + 1;
Ende
```

Dieser Algorithmus stellt ein konstruktives Verfahren dar. Offensichtlich bleibt nach jedem Fahrzeug, das eingefügt wird, eine Sequenz erhalten, die gegen keine Randbedingung verstößt, sodass nach dem Einfügen aller Fahrzeuge stets eine zulässige Lösung für das vorliegende Problem entsteht.

Im Folgenden wird ein Problem dieser Heuristik beschrieben, das als Scheduling Anomalie bezeichnet werden kann. Anschließend wird die einzige Steuerungsgröße, die beim sequentiellen Einfügen existiert - die Reihenfolge, in der die Fahrzeuge eingegliedert werden - näher betrachtet.

5.2.2.1.1 Anomalien

Es wird nun ein Phänomen beschrieben, welches potentielle Schwierigkeiten bei der sequentiellen Einsteuerung verdeutlicht. Um dies möglichst isoliert betrachten zu können, wird das Problem auf einfache Arbeitsplätze ohne Mehrtakter reduziert. Ebenso wird eine Annahme getroffen, damit keine Verzerrung durch mögliches Vorziehen von Springereinsätzen eintreten kann: Die Arbeitsbereiche sind begrenzt durch die doppelte Taktzeit.

Bei der Lösung des statischen Problems treten auch unter diesen Bedingungen Probleme auf, die als *Scheduling Anomalie* bezeichnet werden können. Diese sind in erster Linie aus dem Bereich des *Multiprocessor Scheduling* und bei *Bin Packing* Problemen bekannt. Zuerst wurden sie von Graham [Gra69] als *Timing Anomalies* erwähnt und analysiert. Das Prinzip hinter diesem Phänomen ist, dass eine Veränderung einer Bedingung oder Ausgangsgröße, die das Problem augenscheinlich erleichtern sollte, zu einer Verschlechterung der Zielfunktion führt.

Als variable Größe für die Wiedereinsteuerung wird hier die Größe des Intervalls betrachtet, in das die Fahrzeuge eingegliedert werden. Eine Vergrößerung dieses Intervalls entspricht einer Relaxierung einer Nebenbedingung. Durch eine solche Vergrößerung des Lösungsraums kann der optimale Zielfunktionswert natürlich nicht verschlechtert werden.

Allerdings kann bei realen Problemgrößen häufig keine optimale Lösung berechnet werden. Deshalb werden an dieser Stelle Heuristiken betrachtet. Bei der ersten zuvor beschriebenen Heuristik, dem sequentiellen Einfügen der Aufträge an die jeweils beste Position, kann eine Vergrößerung des Intervalls dazu führen, dass sich der Zielfunktionswert verschlechtert. Dies wird mit folgendem Beispiel demonstriert:

Beispiel 3. *Es sind zwei Arbeitsplätze gegeben. Die Taktzeit beträgt 10 s. Ebenso besitzen beide Arbeitsplätze einen Driftbereich von weiteren 10 s. Die Bearbeitungszeiten der geplanten und der wiedereinzusteuernden Fahrzeuge kann den beiden Tabellen 5.5 und 5.6 entnommen werden.*

Fahrzeug	1	2	3	4	5	6
Arbeitsplatz 1	8	20	0	10	13	8
Arbeitsplatz 2	0	17	15	14	14	13

Tabelle 5.5: Bearbeitungszeiten der Fahrzeuge in der Plansequenz (in s)

Fahrzeug	1	2	3
Arbeitsplatz 1	18	12	15
Arbeitsplatz 2	18	0	0

Tabelle 5.6: Bearbeitungszeiten der wiedereinzusteuernden Fahrzeuge (in s)

Im Szenario A ist ein Einfügen vor dem ersten Fahrzeug der Plansequenz und zwischen allen Fahrzeugen der Plansequenz erlaubt. Lediglich nach dem letzten Fahrzeug der

Plansequenz darf kein weiteres Fahrzeug mehr eingegliedert werden. Im Szenario B ist zusätzlich auch ein Eingliedern am Ende der Plansequenz erlaubt.

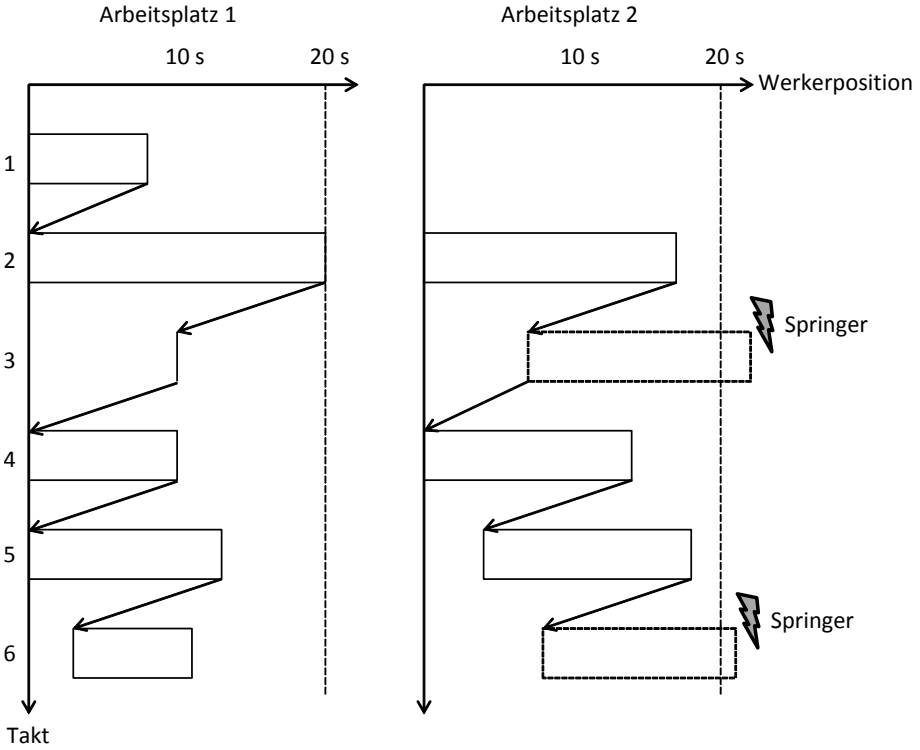


Abbildung 5.7: Werkerbewegungen bei der Plansequenz

In der Ausgangssituation gibt es in der Plansequenz zwei Driftüberschreitungen am zweiten Arbeitsplatz, und zwar beim dritten und sechsten Fahrzeug, wie in Abbildung 5.7 dargestellt ist.

Beim Durchführen der sequentiellen Einsteuerung im Szenario A wird das erste Fahrzeug ganz nach vorne gesetzt, so entsteht nur eine zusätzliche Driftüberschreitung am ersten Arbeitsplatz am zweiten Job der Plansequenz. Danach wird das zweite Fahrzeug vor das dritte Fahrzeug der Plansequenz gesetzt, was zu einer Reduzierung von einer Driftüberschreitung am zweiten Arbeitsplatz führt. Das dritte wiedereinzusteuernde Fahrzeug wird dann vor das fünfte der Plansequenz gesetzt, um noch die letzte Driftüberschreitung am zweiten Arbeitsplatz zu verhindern. Somit gibt es am Ende der Wiedereinsteuerung gemäß Abbildung 5.8 insgesamt eine Driftüberschreitung am ersten Arbeitsplatz.

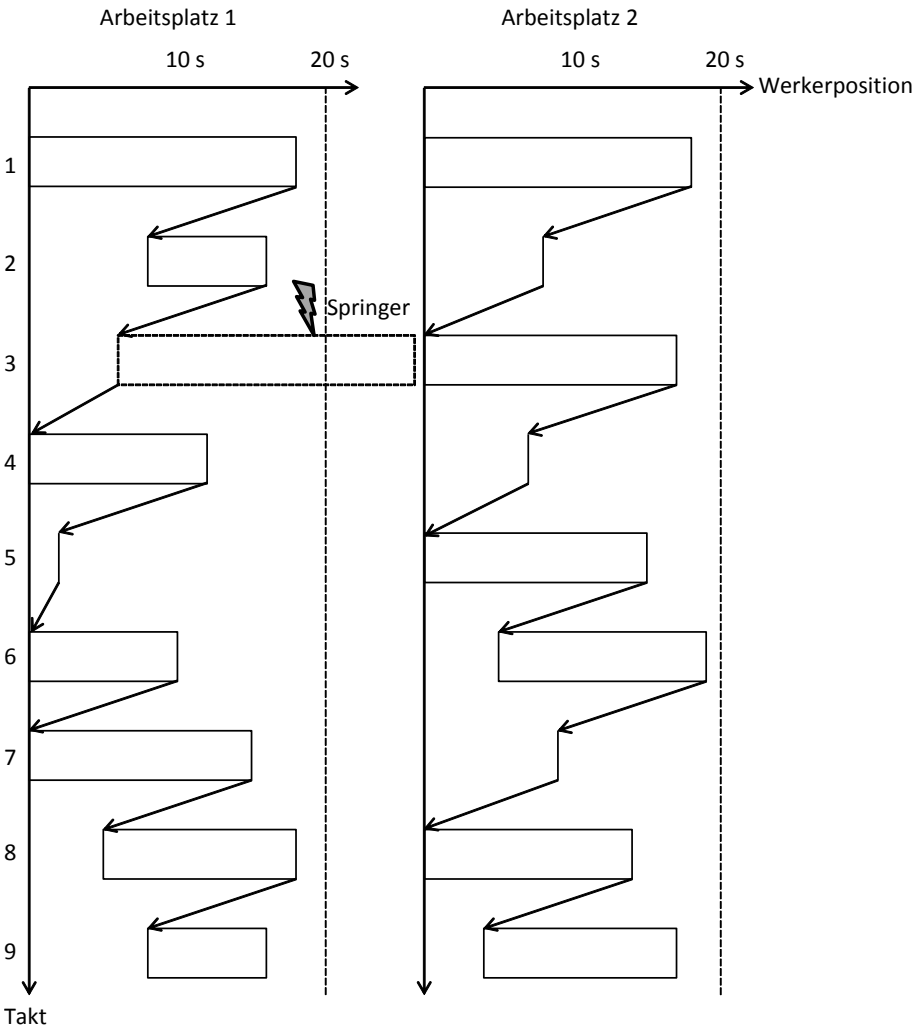


Abbildung 5.8: Werkerbewegungen im Szenario A

Alternativ kann im Szenario B auch die letzte Position verwendet werden. Das wird auch direkt vom ersten Fahrzeug genutzt, das hier ohne zusätzliche Driftüberschreitung eingefügt werden kann. Das zweite Fahrzeug kann danach keine Driftüberschreitung verhindern und wird an die erste Position gesetzt, damit wenigstens keine weiteren Driftüberschreitungen entstehen. Zuletzt kann das dritte Fahrzeug an eine beliebige Position mit immer den gleichen Konsequenzen gesetzt werden: Es entsteht eine Driftüberschreitung am ersten Arbeitsplatz. Insgesamt ergeben sich also drei Driftüberschreitungen (s. Abbildung 5.9).

Das Beispiel kann erweitert werden, indem der zweite Arbeitsplatz beliebig oft wieder-

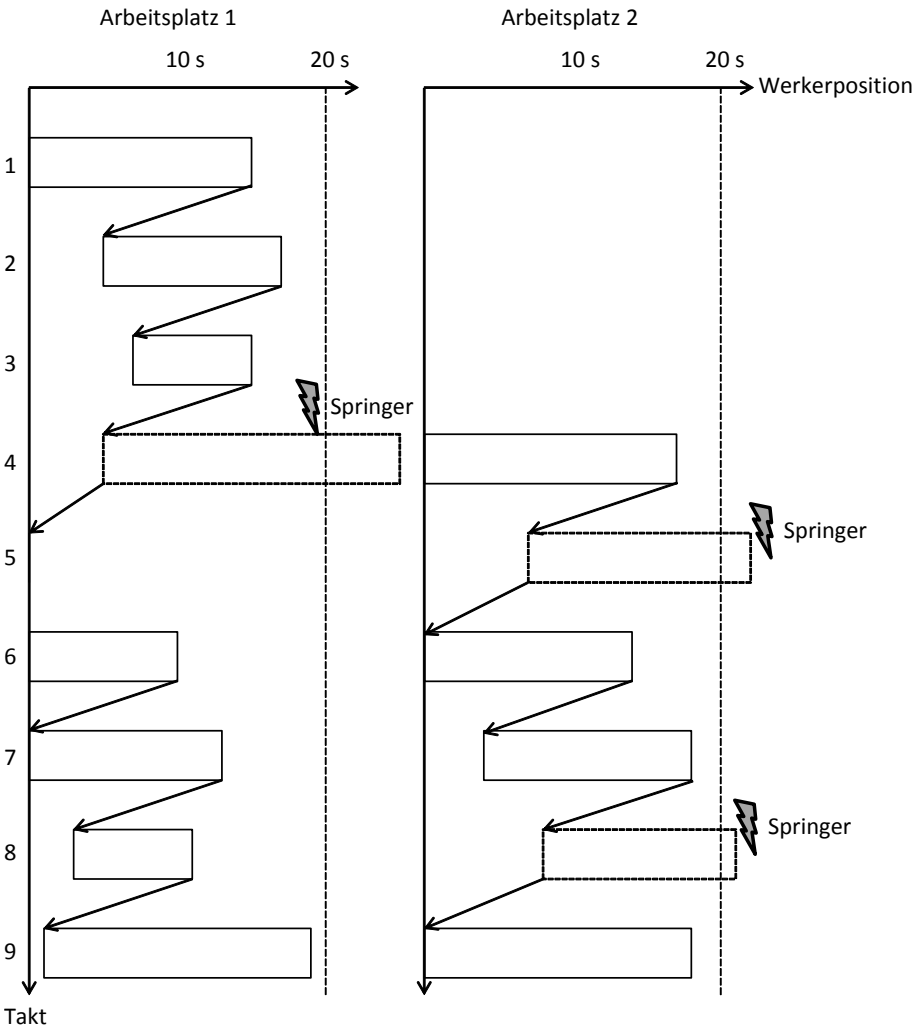


Abbildung 5.9: Werkerbewegungen im Szenario B

holt wird. So kann ein Beispiel generiert werden, bei dem $2 \cdot$Anzahl der Arbeitsplätze-2 zusätzliche Driftüberschreitungen im Szenario B gegenüber A entstehen.

Die -2 entsteht dadurch, dass für das erste Fahrzeug eine Verbesserung gegenüber der ersten Variante entstehen soll, da sonst die Wahl der Position wieder von einer Heuristik abhängig wird.

Offensichtlich ist dies auch eine enge Schranke, da das erste Fahrzeug keine zusätzlichen Driftüberschreitungen erzeugen kann.

Ein wichtiger Aspekt, der bei diesem Beispiel auffällt, ist die Schwäche der Zielfunk-

tion. Diese ist diskret und beachtet nicht das sukzessive Aufbauen von Drift bis zur anschließenden Driftüberschreitung, sondern nur die Anzahl der Driftüberschreitungen. So könnte im Beispiel vom zweiten und dritten wiedereinzusteuernden Auftrag Drift am zweiten Arbeitsplatz abgebaut werden. In der Kombination wäre eine ganze Driftüberschreitung abgebaut, jedoch kann ein einzelner Auftrag dies nicht leisten. Deshalb ist es unter Berücksichtigung der Zielfunktion egal, ob Drift abgebaut wird oder nicht. Diese Überlegungen führen mit zu einer Verwendung von ergänzenden Teilen der Zielfunktion, was später im Rahmen der Analyse des dynamischen Problems im Abschnitt 6.4.1 weiter ausgeführt wird.

5.2.2.1.2 Einfügungsreihenfolge

Der einzige Ansatzpunkt, der im Rahmen des sequentiellen Einfügens existiert, ist die Reihenfolge, in der die Fahrzeuge in die Plansequenz integriert werden. Dabei sind viele Auswahlmöglichkeiten denkbar. Im Sinne einer Orientierung am realen Problem wäre eine Sortierung in der Reihenfolge, in der die Fahrzeuge ursprünglich geplant waren, sinnvoll. Das bedeutet, dass die Fahrzeuge, deren Plantermine schon am längsten überschritten sind, zuerst Positionen in der aktuellen Sequenz erhalten. Allerdings wäre der Einfluss einer solchen Entscheidung auf die Zielfunktion rein zufällig. Unter der Annahme, dass bei verspäteten Fahrzeugen einzig die Einhaltung eines Fälligkeitstermins relevant ist und ansonsten die genaue Position zwischen Plantermin und Fälligkeitstermin nicht bewertet wird, wäre dieser Ansatz genauso gut, wie ein zufälliges Auswählen der Reihenfolge.

Um eine sinnvolle Reihenfolge zu bestimmen, sollten also die Belastungszeiten eines Auftrags oder sogar die durch einen Auftrag entstehenden Driftüberschreitungen verwendet werden. Die vielversprechendste Idee ist dabei, zuerst die Fahrzeuge auszuwählen, die nur schwer in die Plansequenz eingefügt werden können. Nun stellt sich die Frage, welche Fahrzeuge schwer einzusteuern sind. Ein Ansatz besteht darin, die Fahrzeuge nach der Summe ihrer Belastungszeiten zu ordnen und die Fahrzeuge mit der größten Belastungssumme zuerst einzufügen, da diese mit größerer Wahrscheinlichkeit zusätzliche Driftüberschreitungen erzeugen.

Ein alternativer Ansatz betrachtet direkt die durch das Eingliedern eines Fahrzeugs an

eine exakte Position entstehenden Driftüberschreitungen. Parallel zum vorhergehenden Ansatz können nun die Fahrzeuge, die viele Driftüberschreitungen erzeugen, zuerst eingesteuert werden mit der Hoffnung, dass die anderen Fahrzeuge die Überlastungen anschließend ausgleichen können.

Eine Weiterentwicklung dieser Idee ist, nicht nur die Driftüberschreitungen an der besten Position für jedes Fahrzeug zu ermitteln, sondern für alle Positionen. Der entscheidende Wert ist dann nicht der absolute Wert, sondern die Differenz in der Anzahl der Driftüberschreitung von der besten Position zu den anderen Positionen. Ein Fahrzeug, das immer eine ähnliche Anzahl an Driftüberschreitungen erzeugt, egal an welche Position es gesetzt wird, hat eine niedrige Priorität beim Einsteuern. Zuerst sollten Fahrzeuge in die Plansequenz eingegliedert werden, die an eine Position deutlich besser passen als an alle anderen. Die Fahrzeuge, die in Bezug auf ihre guten Positionen am unflexibelsten sind, sollten zuerst eingesteuert werden, da die anderen Aufträge sehr wahrscheinlich trotzdem eine gute Position zugewiesen bekommen können.

In Tests konnte sich keine Strategie als dominant erweisen. Die optimale Reihenfolge, in der die Fahrzeuge eingefügt werden sollten, hängt von der Testinstanz ab. Die durchschnittlichen Ergebnisse unterscheiden sich nur marginal, sodass für die initiale Konstruktion einer Startlösung ein möglichst schnelles Verfahren gewählt werden sollte. Deswegen wird der Algorithmus SE ohne besondere Betrachtung der Reihenfolge der wiedereinzusteuernden Fahrzeuge verwendet. In den durchgeführten Tests sind die Fahrzeuge nach ihrer ursprünglichen Position in der Plansequenz sortiert.

5.2.2.2 Lokale Suchverfahren

Die im letzten Abschnitt vorgestellten Ansätze zum sequentiellen Einfügen dienen in erster Linie dazu schnell eine akzeptable Lösung zu finden. Dies sollte jedoch als Startlösung gesehen werden, die mit anderen Methoden verbessert werden kann. Diese versuchen nicht das Problem von Grund auf anders zu lösen, sondern verwenden nur kleine Veränderungen der Sequenz. In diesem Abschnitt werden zwei Methoden vorgestellt, nämlich das Versetzen eines Auftrags und das Vertauschen von zwei Aufträgen. In einem ähnlichen Kontext wurden diese Veränderungen der Sequenz auch bei Okamura und Yamashina [OY79] verwendet, bei denen sie aber aufgrund einer anderen

Zielfunktion durch eine Heuristik gezielt gesteuert wurden.

5.2.2.2.1 Versetzen von Aufträgen

Ein wiedereingesteuerter Auftrag wird aus der vollständigen Sequenz wieder entfernt und neu an die im aktuellen Zustand beste Position eingesetzt.

Zunächst wird eine Nachbarschaft zu einer gegebenen Sequenz definiert. Dabei kann für jeden einzelnen (wiedereingesteuerten) Auftrag eine Nachbarschaft definiert werden und je nach Anwendung der Suche kann die Vereinigung aller dieser Nachbarschaften als die Nachbarschaft der ursprünglichen Sequenz bezeichnet werden.

Es sei π eine Sequenz und i ein Auftrag in dieser Sequenz. Dann bezeichnet $N_1(\pi, i)$ die Nachbarschaft dieses Auftrags. Diese enthält alle Sequenzen, die durch Verschieben von Auftrag i generiert werden können. Entsprechend wird die gesamte Nachbarschaft, die durch das Verschieben eines beliebigen wiedereingesteuerten Auftrags erreicht werden kann, mit $N_1(\pi, I^W) = \cup_{i \in I^W} N_1(\pi, i)$ bezeichnet.

Für die lokale Suche gibt es nun verschiedene Vorgehensweisen. Es kann eine Nachbarschaft nach der anderen durchsucht werden und zum lokalen Optimum gewechselt werden. Dies kann iterativ solange durchgeführt werden bis keine Nachbarschaft mehr eine Verbesserung enthält. Dabei kann die Reihenfolge der Nachbarschaften bzw. wiedereingesteuerten Aufträge fest oder zufällig gewählt werden.

Alternativ kann die gesamte Nachbarschaft durchsucht werden. Dafür ist allerdings ein deutlich größerer Rechenaufwand nötig, bevor überhaupt eine Position verändert werden kann. Diese Variante wird in dieser Arbeit nicht weiter verfolgt.

Da später zwei verschiedene Varianten des Verschiebens von Aufträgen aufgegriffen werden, werden diese nun formal definiert. Zunächst wird das iterative Versetzen im Algorithmus 5.2 formal beschrieben, bei dem alle Fahrzeuge regelmäßig verschoben werden. Die wiedereingesteuerten Fahrzeuge werden hier der Reihe nach betrachtet. Dabei folgt auf das letzte Fahrzeug wieder das erste. Diese Prozedur kann solange durchgeführt werden, bis eine komplette Rotation durch alle wiedereingesteuerten Fahrzeuge ohne eine Verbesserung bleibt.

Algorithmus 5.2: Iteratives Versetzen (IV)

Initialisierung;
Startsequenz: π;
beste gefundene Sequenz: $\pi_{opt} := \pi$;
$counter := 0$;
solange $counter < |I^W|$ **tue**
  nehme nächstes Fahrzeug $i \in I^W$;
  finde $\pi^* \in N_1(\pi, i)$ mit $f(\pi^*) = \min_{\pi' \in N_1(\pi,i)} f(\pi')$;
  wenn $\pi^* = \pi$ **dann**
    $counter := counter + 1$;
  sonst
    $counter := 0$;
    $\pi := \pi^*$;
  Ende
Ende
$\pi_{opt} := \pi$;

Alternativ zu diesem Verfahren kann das Fahrzeug, das verschoben wird, zufällig bestimmt werden, was als randomisiertes Versetzen (Algorithmus 5.3) bezeichnet wird.

Algorithmus 5.3: Randomisiertes Versetzen (RV)

Initialisierung;
Startsequenz: π;
beste gefundene Sequenz: $\pi_{opt} := \pi$;
solange *Abbruchkriterium nicht erfüllt ist* **tue**
  wähle zufälliges Fahrzeug $i \in I^W$;
  finde $\pi^* \in N_1(\pi, i)$ mit $f(\pi^*) = \min_{\pi' \in N_1(\pi,i)} f(\pi')$;
  $\pi := \pi^*$;
  wenn $f(\pi) < f(\pi_{opt})$ **dann**
    $\pi_{opt} := \pi$;
  Ende
Ende

5.2.2.2.2 Vertauschen von Aufträgen

Zwei wiedereingesteuerte Aufträge tauschen ihre Positionen, wenn dies zu einer Verbesserung der Zielfunktion führt.

Hier könnte analog zum Versetzen eine Nachbarschaft für jedes Auftragspaar, das vertauscht werden kann, definiert werden. Allerdings würden alle diese Nachbarschaften nur eine einzige Sequenz enthalten. Deshalb ist es sinnvoll an dieser Stelle nur eine Nachbarschaft für die Menge aller wiedereinzusteuernden Aufträge zu definieren. Diese enthält alle Sequenzen, die durch das Vertauschen von zwei Aufträgen entstehen können und wird mit $N_2(\pi)$ bezeichnet. Diese Nachbarschaft wird beim lokalen Vertauschen (Algorithmus 5.4) verwendet.

Algorithmus 5.4: Lokales Vertauschen (LV)

```
Initialisierung;
Startsequenz: π;
beste gefundene Sequenz: π_opt := π;
solange Abbruchkriterium nicht erfüllt ist tue
    finde π* ∈ N_2(π) mit f(π*) = min_{π'∈N_2(π)} f(π');
    π := π*;
    wenn f(π) < f(π_opt) dann
        π_opt := π;
    Ende
Ende
```

5.2.2.3 Metaheuristiken

Die größte Herausforderung und zugleich die größte Schwäche der lokalen Suchverfahren ist das Verlassen von lokalen Optima. Bei reiner Verwendung eines lokalen Suchverfahrens kann ein lokales Optimum nie wieder verlassen werden. Die Suche würde bei dieser Lösung terminieren. Mit dem Ziel, dieses Problem zu überwinden, können die lokalen Suchverfahren in Heuristiken - sogenannte Metaheuristiken - eingebettet werden. An dieser Stelle werden drei Verfahren vorgestellt, die in dem vorliegenden Problem eingesetzt werden können.

5.2.2.3.1 Simulated Annealing

Die Idee hinter Simulated Annealing beruht auf einer Analogie zur thermischen Mechanik. Bei der Abkühlung von Metallen sorgt eine langsame Abkühlung dafür, dass die Atome sich so ordnen können, dass ein energiearmer Zustand erreicht wird.

Die erste Beschreibung von Simulated Annealing ist bei Kirkpatrick et al. [KGV83] zu finden. Für eine detaillierte Beschreibung des Hintergrundes und der Anwendung dieses Verfahrens sei an dieser Stelle außerdem auf Laarhoven und Aarts [LA87] sowie Černý [Če85] verwiesen. Es wurde ebenfalls bei dem allgemeinen Mixed-Model-Sequencing Problem von McMullan und Frazier [MF00] erfolgreich angewandt.

Für das vorliegende Problem wurde ein Simulated Annealing Algorithmus mit der Verwendung des Versetzen-Operators implementiert. Die erste Entscheidung, die getroffen wurde, ist, wie viel Kontrolle über den Optimierungsprozess behalten werden soll bzw. wie viele zufällige Elemente verwendet werden. Jeder Schritt innerhalb des Algorithmus entspricht dem testweisen Versetzen eines Auftrags in der Produktionsreihenfolge mit einer anschließenden Entscheidung, ob die neue Reihenfolge angenommen wird. Dabei müssen vorab zwei Entscheidungen getroffen werden:

1. Welcher Auftrag wird verschoben?
2. Auf welche Position wird dieser Auftrag gesetzt?

Da nur eine sehr begrenzte Rechenzeit zur Verfügung steht, ist es notwendig, dass jeder Auftrag hinreichend oft betrachtet wird. Wenn die Antwort auf die erste Frage durch einen Zufallsgenerator bestimmt wird, kann es passieren, dass so selten versucht wird, einzelne Aufträge auf eine neue Position zu setzen, dass allein durch diesen systematischen Aspekt ein Erreichen einer guten Lösung verhindert wird. Daher werden die Aufträge immer nacheinander durchiteriert, sodass alle gleich oft die Chance haben, auf einer anderen Position platziert zu werden.

Bei der Beantwortung der zweiten Frage wird auf die Stärke des Zufalls vertraut. Dem jeweils betrachteten Auftrag wird eine zufällige neue Position zugewiesen. Die neue Reihenfolge muss dann hinsichtlich ihres Zielfunktionswertes ausgewertet werden. Jedoch ist im Gegensatz zu dem später im Rahmen der Tabu Search verwendeten Ansatz pro Iteration nur eine einzige Reihenfolge zu bewerten.

Die eigentliche Anwendung der Idee hinter dem Simulated Annealing geschieht erst bei der Entscheidungsfindung, ob die neue Sequenz akzeptiert wird. Dabei wird in Abhängigkeit der Zielfunktionswerte, der Temperatur und einer weiteren Zufallsgröße ermittelt, ob die neue Reihenfolge für den Optimierungsprozess weiterverwendet wird

oder ob die alte Reihenfolge zunächst beibehalten wird.

Wie vorher beschrieben, ist die zu minimierende Zielfunktion die Anzahl der benötigten Unterstützereinsätze in der vorliegenden Sequenz. Bei jeder Verschiebung eines Auftrags, die zu einer Verringerung dieser Anzahl führt, wird die neue Sequenz auf jeden Fall weiterverwendet. Falls bei Positionierung des betrachteten Auftrags an der neuen Position Δ mehr Springereinsätze benötigt werden, wird die neue Sequenz nur mit der Wahrscheinlichkeit $e^{-\frac{\Delta}{T}}$ angenommen. Dabei bezeichnet T die Temperatur, die entsprechend eines Abkühlungsschemas im Verlauf der Anwendung des Algorithmus immer weiter sinkt.

Ein grundlegender Simulated Annealing Algorithmus wird in Algorithmus 5.5 beschrieben. Dabei bezeichnet $N(.)$ eine beliebige Nachbarschaft und $f(.)$ ist weiterhin die Zielfunktion.

Algorithmus 5.5: Allgemeines Simulated Annealing

```
Initialisierung;
Startlösung: π;
beste gefundene Lösung: π_opt := π;
solange Abbruchkriterium nicht erfüllt ist tue
    wähle π' ∈ N(π) zufällig aus einer Nachbarschaft;
    Δ := f(π') − f(π);
    generiere eine zwischen 0 und 1 gleichverteilte Zufallszahl r ~ U(0,1);
    wenn e^(−Δ/T) > r dann
        π := π';
        wenn f(π) < f(π_opt) dann
            π_opt := π;
        Ende
    Ende
    reduziere T entsprechend eines Abkühlungsschemas;
Ende
```

5.2.2.3.2 Variable Neighborhood Search

Eine weitere Metaheuristik, die zur Lösung des hier betrachteten Problems verwendet werden kann, ist die *Variable Neighborhood Search (VNS)*, die von Mladenović und Hansen [MH97] vorgestellt wurde. Die Grundidee dieser Metaheuristik liegt in der

Suche in wechselnden Nachbarschaften $N_k(.)$. Diese kann auf unterschiedliche Arten durchgeführt werden. Flexible Aspekte sind insbesondere die Wahl der nächsten Nachbarschaft und die Akzeptanz von Zwischenlösungen mit schlechtem Zielfunktionswert.

Ein grundlegender Variable Neighborhood Search Algorithmus wird in Algorithmus 5.6 beschrieben.

Algorithmus 5.6: Allgemeine Variable Neighborhood Search

```
Initialisierung;
Startlösung: π;
beste gefundene Lösung: π_opt := π;
solange Abbruchkriterium nicht erfüllt ist tue
    wähle π' ∈ N_k(π) zufällig;
    berechne lokales Optimum π* nach lokaler Suche ausgehend von π';
    wenn f(π*) < f(π) dann
        π := π*;
        wenn f(π) < f(π_opt) dann
            π_opt := π;
        Ende
    sonst
        wähle eine neue Nachbarschaft N_k';
    Ende
Ende
```

5.2.2.3.3 Tabu Search

Wie schon bei den Modellierungen beschrieben wurde, stammt das ähnlichste Modell zu dem hier verwendeten von Gujjula und Günther [GG09b]. In ihrem Paper schlagen sie einen heuristischen Ansatz, der sich der Tabu Search bedient, vor. Deshalb soll auch an dieser Stelle ein ähnlicher Ansatz vorgestellt werden.

In der Literatur wurde Tabu Search zuerst von Glover [Glo86] erwähnt. Eine gute Einführung in dieses Thema findet sich z. B. bei Gendreau [Gen03]. Im Folgenden wird zunächst der allgemeine Ansatz beschrieben.

Um lokale Optima, die durch lokale Suchverfahren gefunden werden, zu überwinden und die Zielfunktion weiter zu verbessern, wird eine temporäre Verschlechterung der

Zielfunktion in Kauf genommen. Entscheidend für die Effizienz eines Algorithmus ist es, ein Kreisen zwischen wenigen Lösungen zu verhindern. Dies geschieht hier mittels sogenannter Tabu-Listen. Bestimmte schon untersuchte Lösungen, werden tabu, d. h. bei der Suche dürfen sie nicht erneut betrachtet werden. Üblicherweise werden Lösungen für eine vorher bestimmte Anzahl an Transformationen in einer Tabu-Liste gespeichert, weshalb dabei auch von einem Kurzzeitgedächtnis (vgl. [Gen03]) gesprochen wird.

Ein grundlegender Tabu Search Algorithmus wird in Algorithmus 5.7 beschrieben.

Algorithmus 5.7: Allgemeine Tabu Search

Initialisierung;
Startlösung: π;
beste gefundene Lösung: $\pi_{opt} := \pi$;
solange *Abbruchkriterium nicht erfüllt ist* **tue**
  finde $\pi^* \in N(\pi) \backslash \{tabu\}$ mit $f(\pi^*) = \min_{\pi' \in N(\pi) \backslash \{tabu\}} f(\pi')$;
  $\pi := \pi^*$;
  wenn $f(\pi) < f(\pi_{opt})$ **dann**
    $\pi_{opt} := \pi$;
  Ende
  update Tabu-Liste *tabu*;
Ende

Dazu existieren in der Literatur diverse Erweiterungen und Verfeinerungen der Tabu Search, die in Abhängigkeit vom jeweiligen Problem sinnvoll sein können. An dieser Stelle wird lediglich das Konzept der Intensivierung und Diversifizierung der Suche genannt, da dieses auch in den implementierten Algorithmen verwendet wird. Die Intensivierung dient dazu die Lösungssuche in einem vielversprechenden Bereich des Lösungsraumes zu vertiefen. Gendreau [Gen03] unterstreicht jedoch vor allem die Wichtigkeit einer geeigneten Diversifizierung. Diese verfolgt das gegensätzliche Ziel, möglichst viele Bereiche des Lösungsraumes zu untersuchen.

Eine Kombination der letzten beiden Metaheuristiken - wie sie auch in dieser Arbeit erfolgen wird - wurde z. B. von Cordeau et al. [CLP08] auf das Car-Sequencing Problem angewandt.

5.2.2.3.4 Getestete Heuristiken

Aus dieser vorgestellten Auswahl an Metaheuristiken werden drei verschiedene Algorithmen abgeleitet, die nach Kalibrierung mit einem separaten Datensatz durch die im folgenden Abschnitt beschriebenen Tests analysiert werden.

Der erste Algorithmus bedient sich der Idee des Simulated Annealing und wendet diese auf die Verwendung des Versetzen-Operators an. Die anderen beiden Algorithmen bauen auf der Variable Neighborhood Search auf. Dabei kombinieren sie diese zusätzlich mit Elementen der Tabu Search.

Begrenzt wird die Suche durch eine vorzugebende maximale Rechenzeit t_{max}, die sich an der zur Verfügung stehenden Taktzeit orientiert. In allen nun vorgestellten Algorithmen bezeichnet t die Laufzeit.

Bei der Kalibrierung des Simulated Annealing Algorithmus erwies sich die Verwendung einer exponentiell sinkenden Temperatur als sinnvoll. Deshalb wurde folgender Algorithmus 5.8 ausgewählt, um ihn mit den Alternativen zu vergleichen.

Die weiteren beiden Algorithmen verwenden in ihrem Kern das randomisierte Verschieben von Aufträgen. Eine besondere Intensivierung der Suche findet statt, wenn bei einem Suchschritt in diesem Kern eine verbesserte Lösung gefunden worden ist. Dies führt dazu, dass die Anzahl der durchzuführenden Suchschritte in diesem Bereich wieder auf den Ausgangswert zurückgesetzt wird. Zur Diversifizierung der Suche wird in der alternativen Nachbarschaft ein Suchschritt vollzogen, nämlich ein Vertauschen von zwei Aufträgen. Der erste dieser beiden Algorithmen (Algorithmus 5.9) arbeitet dabei mit einer sehr beschränkten Verwendung von Tabulisten. Es wird lediglich die aktuelle Sequenz tabu gesetzt. Folglich wird in jedem Schritt eine Veränderung der Sequenz erzwungen.

Eine Weiterentwicklung dieses Verfahren nutzt tatsächlich Tabulisten. In einigen Tests wurde ein Kreisen zwischen wenigen Lösungen beobachtet. Dies soll mit Tabulisten $tabu_i$, die immer die letzten sechs betrachteten Positionen vom jeweiligen Auftrag i enthalten, verhindert werden. Diese Variante ist in Algorithmus 5.10 dargestellt.

Bei der Formulierung dieses Algorithmus ist zu beachten, dass die Tabulisten $tabu_i$

Algorithmus 5.8: Simulated Annealing (SA)

Initialisierung;
rufe Algorithmus SE auf;
Startsequenz: π;
beste gefundene Sequenz: $\pi_{opt} := \pi$;
Starttemperatur: $T := T_0$;
solange $t < t_{max}$ **tue**
  für jedes $i \in I^W$ **tue**
    wähle $\pi' \in N_1(\pi, i)$ zufällig;
    $\Delta := f(\pi') - f(\pi)$;
    generiere eine zwischen 0 und 1 gleichverteilte Zufallszahl $r \sim U(0, 1)$;
    $T := T_0 \cdot e^{-10 \cdot \frac{t}{t_{max}}}$;
    wenn $e^{-\frac{\Delta}{T}} > r$ **dann**
      $\pi := \pi'$;
      wenn $f(\pi) < f(\pi_{opt})$ **dann**
        $\pi_{opt} := \pi$;
      Ende
    Ende
  Ende
Ende

nicht alle verbotenen Sequenzen beinhalten, sondern nur alle Positionen, an die i nicht gesetzt werden darf. Die verwendete Schreibweise ist somit etwas salopp, wird aber dadurch gerechtfertigt, dass eine tabu gesetzte Position alle Sequenzen, in denen diese Position eingenommen wird, implizit beschreibt.

5.2.2.4 Teileinsteuerung

Zum Abschluss dieses Abschnitts wird ein Verfahren vorgestellt, das verwendet werden kann, wenn die zuvor genannten Heuristiken aufgrund zu hoher Komplexität einer Probleminstanz keine guten Ergebnisse liefern. Wenn dieses Problem durch das gleichzeitige Einfügen vieler Aufträge entsteht, kann die Problemlösung auseinander gezogen werden. Die Aufträge werden in mehrere Gruppen aufgeteilt und diese Gruppen werden nacheinander entsprechend der verwendeten Methode eingesteuert. Im statischen Problem ist dieses innerhalb eines Takts zu bewältigen und deshalb muss die verfüg-

Algorithmus 5.9: Variable Neighborhood Search (VNS)

```
Initialisierung;
rufe Algorithmus SE auf;
Startsequenz: π;
beste gefundene Sequenz: π_opt := π;
counter := 0;
solange t < t_max tue
    solange counter < 5 tue
        wähle neues Fahrzeug i ∈ I^W zufällig;
        finde π* ∈ N_1(π, i)\{π} mit f(π*) = min_{π'∈N_1(π,i)\{π}} f(π');
        counter := counter + 1;
        wenn f(π*) < f(π) dann
            counter := 0;
            wenn f(π*) < f(π_opt) dann
                π_opt := π*;
            Ende
        Ende
        π := π*;
    Ende
    finde π* ∈ N_2(π)\{π} mit f(π*) = min_{π'∈N_2(π)\{π}} f(π');
    π := π*;
    wenn f(π) < f(π_opt) dann
        π_opt := π;
    Ende
Ende
```

bare Rechenzeit auf die Anzahl der Gruppen aufgeteilt werden.

Nachdem eine Gruppe eingegliedert ist, werden die entsprechenden Aufträge fixiert und die nächste Gruppe von Aufträgen wird in die neue Sequenz eingesteuert. Dadurch geht einerseits Optimierungspotential verloren, da am Ende, wenn alle Aufträge in der Sequenz sind, nur noch die der letzten Gruppe verschoben werden dürfen. Andererseits kann man durch eine geeignete Wahl der Gruppengröße erreichen, dass die Teilschritte annähernd optimal gelöst werden.

Mit Blick auf diese Methode ist es wichtig zu analysieren, welche Problemgröße ausreichend gut gelöst werden kann, d. h. wie viele Fahrzeuge mit der gewählten Methode gleichzeitig in eine Sequenz eingefügt werden können. Dieser Wert kann als Anhalts-

Algorithmus 5.10: Variable Tabu Search (VTS)

Initialisierung;
rufe Algorithmus SE auf;
Startsequenz: π;
beste gefundene Sequenz: $\pi_{opt} := \pi$;
$counter := 0$;
solange $t < t_{max}$ **tue**
  solange $counter < 5$ **tue**
    wähle neues Fahrzeug $i \in I^W$ zufällig;
    finde $\pi^* \in N_1(\pi, i) \backslash \{tabu_i\}$ mit $f(\pi^*) = \min_{\pi' \in N_1(\pi,i)\backslash\{tabu_i\}} f(\pi')$;
    $counter := counter + 1$;
    wenn $f(\pi^*) < f(\pi)$ **dann**
      $counter := 0$;
      wenn $f(\pi^*) < f(\pi_{opt})$ **dann**
        $\pi_{opt} := \pi^*$;
      Ende
    Ende
    $\pi := \pi^*$;
    update $tabu_i$;
  Ende
  finde $\pi^* \in N_2(\pi) \backslash \{\pi\}$ mit $f(\pi^*) = \min_{\pi' \in N_2(\pi)\backslash\{\pi\}} f(\pi')$;
  $\pi := \pi^*$;
  wenn $f(\pi) < f(\pi_{opt})$ **dann**
    $\pi_{opt} := \pi$;
  Ende
Ende

punkt für die Wahl einer geeigneten Gruppengröße herangezogen werden.

Diese Idee wird in dieser Arbeit nicht weiter ausgeführt. Dennoch sei darauf hingewiesen, dass mit dieser Methode in einigen Tests Verbesserungen erzielt werden konnten. Außerdem wird im folgenden Kapitel über die dynamischen Verfahren ein Ansatz vorgestellt, in dem nicht alle freigegebenen Fahrzeuge gleichzeitig in eine Sequenz eingegliedert werden müssen, was einer Aufteilung der freigegebenen Fahrzeuge auf mehrere Takte entspricht.

5.3 Bewertung der Ansätze

Aufgrund der Komplexität des vorliegenden Problems kann keine sichere Ergebnisqualität einzelner Verfahren garantiert werden. Eine Bewertung ist nur in vergleichender Weise möglich, sodass nun die Heuristiken relativ zueinander bewertet werden. Dazu wird zunächst beschrieben, welche Tests hierfür durchgeführt wurden. Anschließend werden die Resultate der Tests präsentiert und schließlich Folgerungen daraus abgeleitet.

5.3.1 Testbeschreibung

Die Grundlage der verwendeten Testinstanzen bilden reale Fahrzeugdaten, d. h. das Verhältnis der Belastungszeiten der verschiedenen Fahrzeuge zueinander ist auch in der Praxis wiederzufinden. Diese Fahrzeugdaten wurden in unterschiedliche Reihenfolgen gebracht, damit auch verschiedene Szenarien, die in der Realität vorkommen können, in den Tests abgebildet werden. Als Ausgangsbasis für die Testinstanzen wurden optimierte Plansequenzen für komplette Produktionstage gebildet. Bei jedem Test wurde jedoch nur ein kleiner Abschnitt dieser Plansequenzen betrachtet.

In der Hälfte der Tests wurde die Plansequenz unverändert gelassen und es wurde versucht weitere Fahrzeuge in diese Sequenz optimal einzusortieren. Dies entspricht dem Praxisfall, dass gesperrte Fahrzeuge freigegeben werden, aber ansonsten die zukünftige Sequenz nicht durch weitere Sperrungen beeinträchtigt ist. In der anderen Hälfte der Tests wurde vor dem Einfügen auch die Plansequenz verändert, in dem Sinne, dass einige Fahrzeuge hieraus gelöscht wurden, was eine zusätzliche Sperrung in der Praxis simuliert.

Eine Analyse dieser beiden Szenarien ist wichtig, da der Anspruch an den Algorithmus etwas unterschiedlich ist. Im ersten Fall muss beim Eingliedern versucht werden, möglichst wenige zusätzliche Driftüberschreitungen zu erzeugen. Dagegen ist beim zweiten Fall häufig das Ziel, die Driftüberschreitungen, die durch das Entfernen einzelner Aufträge entstanden sind, wieder abzubauen und somit kritische Stellen in der Sequenz zu entschärfen.

Zudem wurden verschiedene Intervalllängen - von 20 und 60 Fahrzeugen - getestet, in die Fahrzeuge integriert werden sollten. Die Anzahl der Fahrzeuge, die in diese Teilsequenzen eingefügt werden sollten, variierte von zwei bis zehn. Für sämtliche zuvor beschriebenen Kombinationsmöglichkeiten wurden sechs Testinstanzen generiert. Sämtliche Heuristiken, die mit einem zufälligen Verfahren arbeiten, wurden anhand ihres Durchschnitts nach 20 Berechnungen jeder Testinstanz bewertet.

Den Heuristiken stand zur Optimierung eine Rechenzeit von 30 s zur Verfügung. Zum Vergleich wurden für die kleineren Testinstanzen, d. h. denen mit bis zu maximal 5 wiedereinzusteuernden Fahrzeugen, die optimalen Lösungen berechnet. Dazu wurde die vollständige Enumeration aller möglichen Lösungen verwendet. Für die übrigen Testinstanzen konnte diese Berechnung nicht in einer angemessenen Zeit durchgeführt werden. Sämtliche Verfahren wurden in C# implementiert und die Tests wurden auf einem Intel® Xeon® Prozessor mit 2,66 GHz pro Kern und 22 GB RAM unter dem Betriebssystem Microsoft Windows Server 2003 durchgeführt. Mehrere Instanzen wurden dabei parallel auf je einem Kern berechnet. Lediglich für die Berechnung der optimalen Lösungen der mittelgroßen Instanzen wurde die Enumeration der Sequenzen auf mehrere Kerne aufgeteilt.

5.3.2 Testergebnisse

Die Testergebnisse werden nach der Anzahl der wiedereinzusteuernden Aufträge klassifiziert präsentiert. Dabei werden jeweils vier Szenarien unterschieden. Die Zahl 20 oder 60 bezieht sich auf die Größe des Intervalls, in das die Fahrzeuge integriert wurden. Der Zusatz „optimiert“ verweist auf die Verwendung einer optimierten Sequenz ohne weitere Sperrungen, während „gesperrt“ die Instanzen mit zusätzlichen Sperrungen der Plansequenz markiert.

Zunächst wurden kleine Probleminstanzen getestet, die sich mittels Brute-Force Ansatz in kurzer Zeit optimal lösen lassen. So können die heuristischen Verfahren sinnvoll bewertet werden. In Abbildung 5.10 sind dazu die Ergebnisse der Tests, bei denen zwei oder drei Fahrzeuge wiedereingesteuert wurden, dargestellt. Die Werte wurden dabei normalisiert. Die Optimallösung, die durch Verwendung vom Brute-Force Ansatz be-

stimmt wurde, entspricht dem Wert 100.

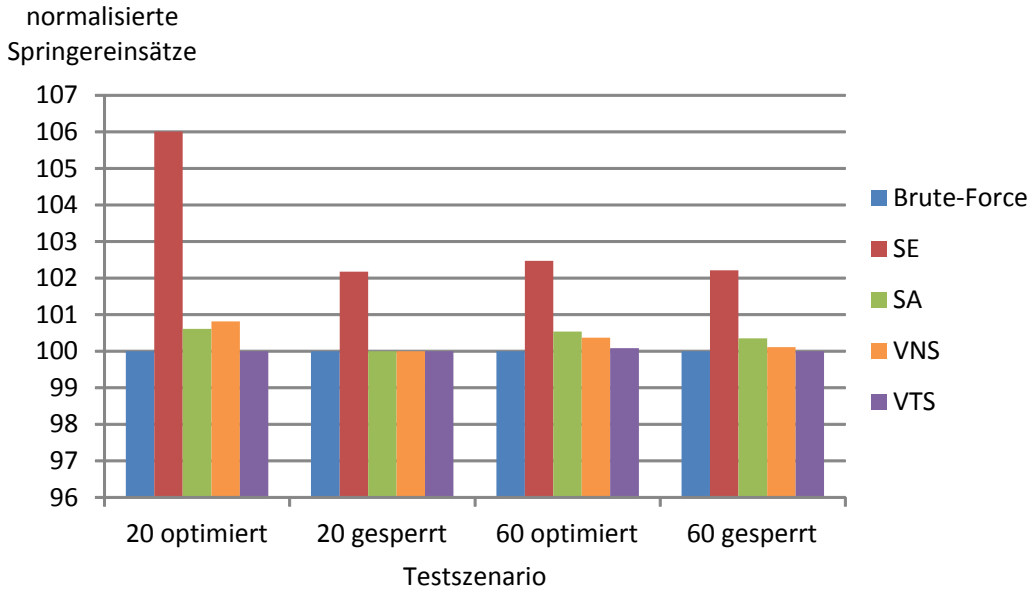


Abbildung 5.10: Vergleich der Algorithmen bei 2-3 eingefügten Fahrzeugen

Die Resultate zeigen, dass das sequentielle Einfügen selbst bei kleinen Instanzen deutliches Verbesserungspotential offen ließ. Alle Heuristiken erreichten die Optimallösung im Schnitt bis auf einen Abstand von unter einem Prozent. Die besten Ergebnisse wurden vom VTS Algorithmus berechnet. Diese Heuristik fand in 22 von 24 Testinstanzen stets die Optimallösung. In den restlichen beiden Instanzen wurde zumindest in einigen Testläufen eine optimale Sequenz erzeugt, aber nicht immer.

Für mittelgroße Testinstanzen stehen auch Optimallösungen als Vergleich zur Verfügung. Allerdings wurden diese mit mehreren Stunden Rechenzeit bestimmt. Der Brute-Force Ansatz ist somit keine alternative Variante für die Praxis, sondern nur ein Mittel zur Berechnung einer theoretischen Schranke. Die Ergebnisse des Einfügens von vier oder fünf Fahrzeugen sind in Abbildung 5.11 dargestellt.

Bei diesen mittelgroßen Instanzen war der Abstand der Initiallösungen (SE) zu den Optima deutlich größer als bei den kleinen Instanzen, nämlich im Schnitt über 8 %. Der SA Algorithmus konnte diese Lücke um etwa die Hälfte schließen. Der VNS Algorithmus war der beste bei diesen Instanzen mit einem Abstand von unter einem halben

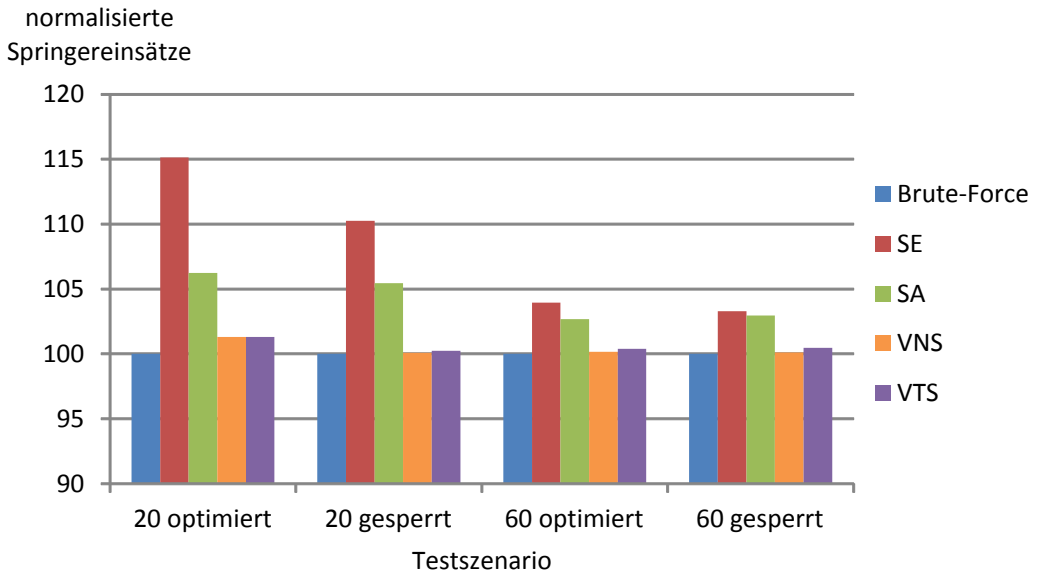


Abbildung 5.11: Vergleich der Algorithmen bei 4-5 eingefügten Fahrzeugen

Prozent zu den Optimallösungen, während der VTS Algorithmus etwas schlechtere Ergebnisse produzierte.

In den letzten Tests wurde die Zahl der wiedereinzusteuernden Fahrzeuge nochmal erhöht, sodass nun acht oder zehn Fahrzeuge in die Plansequenz integriert werden sollten. Für diese großen Instanzen war es nicht mehr möglich, in akzeptabler Zeit optimale Lösungen zu bestimmen. Deswegen werden nur die Ergebnisse der Heuristiken gegenübergestellt. Die Normierung verschiebt sich nun dahingehend, dass der Algorithmus VNS den Vergleichswert darstellt und seine Ergebnisse den Wert 100 zugewiesen bekommen. Die Resultate der anderen Heuristiken im Vergleich hierzu können Abbildung 5.12 entnommen werden.

Auffallend sind bei den großen Instanzen im Wesentlichen zwei Dinge. Zum einen wurde der Abstand zwischen dem VNS Algorithmus und den anderen Heuristiken größer. So enthielten die mit dem SA Ansatz erzeugten Sequenzen von 5 bis zu über 15 % mehr Driftüberschreitungen und auch der VTS Algorithmus war nun deutlich schlechter als die Variante ohne Tabulisten. Zum anderen konnte sich der SA Ansatz kaum noch von der Initiallösung absetzen. In einigen Instanzen fand er nicht eine Verbesserung der Initiallösung.

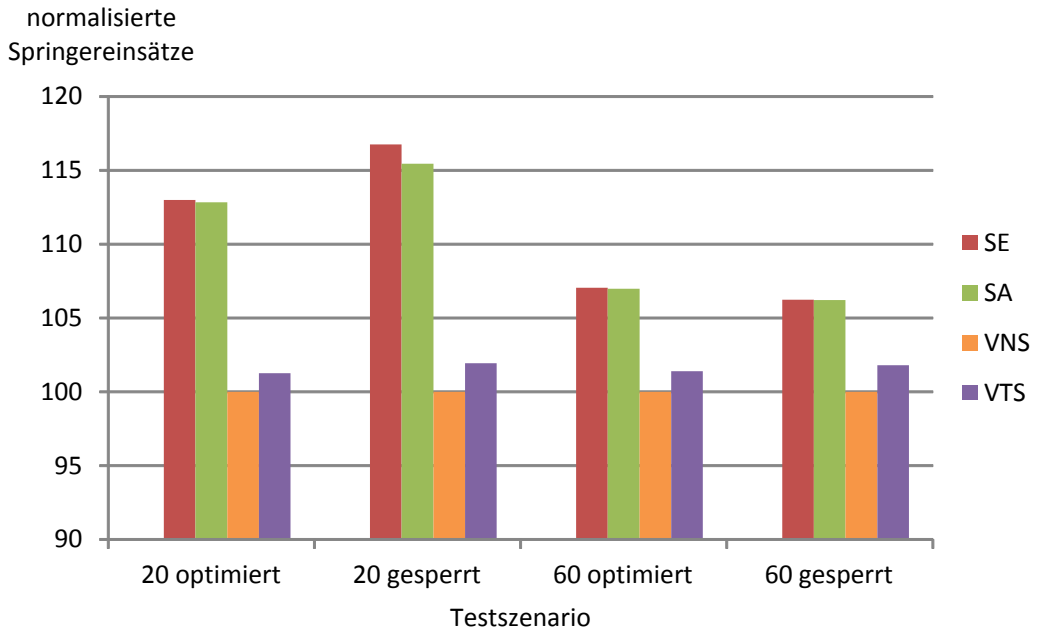


Abbildung 5.12: Vergleich der Algorithmen bei 8-10 eingefügten Fahrzeugen

5.3.3 Folgerungen

Die besten Sequenzen werden vom VNS und vom VTS Algorithmus bestimmt. Die Tabulisten beim VTS Algorithmus beweisen ihre Stärke bei den kleinen Instanzen, bei denen fast immer eine optimale Lösung gefunden wird. Bei größeren Instanzen ist jedoch der VNS Algorithmus ohne Tabulisten besser. Zwischenergebnisse der Algorithmen zeigen, dass ohne Verwendung von Tabulisten das Risiko des Kreisens zwischen wenigen Lösungen bei den kleinen Instanzen besteht. Die Tabulisten ermöglichen es, lokale Minima zu verlassen und führen zu einem erweiterten Suchraum mit besseren Ergebnissen. Dieser Vorteil verschwindet jedoch bei größeren Probleminstanzen, da hier das Risiko des Kreisens grundsätzlich sinkt. Zudem steigt bei der Verwendung von Tabulisten das Risiko kein lokales Optimum zu finden. Deshalb kann aus den Tests abgeleitet werden, dass ein Algorithmus mit Tabulisten, die von der Anzahl der variablen Aufträge abhängen, verwendet werden sollte.

In den folgenden Kapiteln wird deshalb auf dem VTS Algorithmus aufgebaut. Dabei werden jedoch die Tabulisten im Gegensatz zu diesem Kapitel nur bei zwei oder drei einzufügenden Aufträgen verwendet und ansonsten wie im VNS Algorithmus vorgegan-

gen. Zusätzlich wird am Ende eine lokale Suche mit dem IV Algorithmus angehangen, sodass garantiert ein lokales Minimum gefunden wird. Die in jedem Takt zur Verfügung stehende Rechenzeit wird im nächsten Kapitel begrenzt, damit umfangreiche Tests durchführbar sind. Somit ist diese zusätzliche Suchprozedur kein Problem für die Laufzeit. Das hier beschriebene Verfahren wird in der Folge als VTS+ Algorithmus bezeichnet.

Der Simulated Annealing Ansatz liefert akzeptable Ergebnisse bei kleinen Problemgrößen. Mit ansteigender Zahl an wiedereinzusteuernden Fahrzeugen wird die Verbesserung gegenüber der Initiallösung allerdings immer geringer. Durch das rein zufällige Verschieben von Aufträgen in diesem Algorithmus kann es viele Iterationen dauern, bis eine Verbesserung gefunden wird. Die Tests zeigen, dass bei den großen Testinstanzen manchmal nicht ein verbessernder Schritt vollzogen wird, die verfügbare Zeit also für diesen Algorithmus deutlich zu gering ist. Deshalb ist dieser Ansatz für die Praxis nicht sinnvoll, wenn viele Fahrzeuge gesperrt und wieder freigegeben werden.

Des Weiteren ist ein Vergleich zwischen den Testinstanzen mit einer kurzen Plansequenz von 20 Fahrzeugen und einer langen Plansequenz von 60 Fahrzeugen interessant. Vor allem ab den mittelgroßen Instanzen wird deutlich, dass die Initiallösung einen größeren Abstand zur optimalen oder der besten bekannten Lösung hat, wenn die Plansequenz, in die wiedereingesteuert werden soll, kurz ist. Dies lässt sich in erster Linie mit der Verwendung von relativen Größen erklären, die ansteigen, wenn eine kürzere Sequenz und damit verbunden kleinere Zielfunktionswerte betrachtet werden. Dennoch gibt es bei den Instanzen mit kurzen Plansequenzen einige auffällig schlechte Ergebnisse des sequentiellen Einsteuerns. Eine Ursache kann sein, dass bei kleinen Intervallen die Abhängigkeit zwischen den wiedereinzusteuernden Aufträgen stärker ist. Folglich kann hier durch gemeinsame Betrachtung der wiedereingesteuerten Aufträge und damit verbunden durch zusätzliche Verschiebungen erst das gesamte Optimierungspotential genutzt werden, während bei großen Intervallen für viele Aufträge eine Position in der Plansequenz gefunden werden kann, die unabhängig von anderen wiedereingesteuerten Aufträgen gut ist.

6 Das dynamische Problem

In diesem Kapitel wird der gesamte Prozess der Einsteuerung untersucht. Sperrungen und Freigaben werden nicht wie im vorangegangenen Kapitel als gegebene Eingangsgrößen angesehen, sondern sie treten dynamisch im Verlauf eines Produktionsintervalls auf. Im Folgenden wird die Methodik dieses Kapitels erläutert, bevor das dynamische Problem modelliert wird. Es werden Lösungsstrategien entwickelt, die die zuvor vorgestellten Algorithmen für das statische Teilproblem verwenden. Diese werden anschließend getestet. Ausgehend von den Testergebnissen werden diese bewertet und Schlussfolgerungen für den Einsatz in der Praxis gezogen.

6.1 Methodik

Im Literaturüberblick wurde das dynamische Problem als kombinatorisches Online-Optimierungsproblem klassifiziert. Diese Einordnung begründet sich mit dem schrittweisen Ablauf des realen Geschehens. In jedem Takt muss eine Entscheidung getroffen werden und ein Fahrzeug in die Montage eingesteuert werden. Es erscheint daher sinnvoll, diese Eigenschaft auch für die Modellierung und die weitere Problembetrachtung zu verwenden.

Für die Gestaltung von exakten Lösungsstrategien gibt es im Wesentlichen zwei Möglichkeiten. Im Sinne einer stochastischen Optimierung können die dynamisch auftretenden Sperrungen durch Zufallsvariablen modelliert werden. Da eine genaue Analyse von Wahrscheinlichkeitsverteilungen und die Berechnung von Erwartungswerten in diesem Kontext viel zu komplex sind, wird auf eine weitergehende stochastische Betrachtung jedoch verzichtet.

Entscheidend für die Zuverlässigkeit der Algorithmen ist, dass diese in jedem Fall zulässige Sequenzen erzeugen. Angelehnt an das Gebiet der robusten Optimierung werden das Problem und dabei vor allem die Sperrungen so formuliert, dass garantiert eine zulässige Sequenz gefunden werden kann. Ebenso sollen die Algorithmen so aufgebaut werden, dass tatsächlich nur zulässige Sequenzen erzeugt werden. Bei der Analyse wird jedoch ebenfalls aus Komplexitätsgründen darauf verzichtet, sämtliche möglichen Sperrungen einer Instanz zu betrachten und eine Worst-Case-Analyse zur Ermittlung des schlechtesten möglichen Zielfunktionswerts durchzuführen. Es wird lediglich eine kleine Anzahl an Sperrszenarien betrachtet und die dabei konstruierten Sequenzen bewertet.

Das Ziel für die Praxis sollte sein, alle realistischen Sperrszenarien möglichst gut abzudecken. Da sich die Menge der realistischen Szenarien aber nicht abgrenzen lässt, stellt die beschriebene Methodik einen geeigneten Weg zur Bewertung dar. Auf diese Weise lässt sich zum einen sicherstellen, dass die entwickelten Strategien mit allen Szenarien umgehen können. Zum anderen beschränkt sich die weitere Bewertung der Strategien auf die Ergebnisse in einer realistischen Auswahl von Testszenarien.

6.2 Modellierung

In diesem Abschnitt werden die statischen Modelle an den dynamischen Kontext angepasst. Dabei können einige Elemente der statischen Modelle nahezu unverändert übernommen werden, andere werden an die neuen Umstände angepasst und einige kommen ganz neu hinzu. Die Zielfunktion bleibt die gleiche, einzig die Summe wird über ein längeres Intervall - z. B. eine komplette Schicht - gebildet. Die Berechnung der Springereinsätze wird nun analog zum Modell (5.7) vorgenommen. In den dynamischen Modellen werden die Springereinsätze ausschließlich iterativ berechnet. Dies ist zum einen intuitiv und zum anderen gut an die Problemstruktur angepasst. Diese Vorgehensweise garantiert außerdem, dass keine Springereinsätze freiwillig vorgezogen werden können. Die wesentlichen Unterschiede zum statischen Ansatz sind die nun differenziertere Betrachtung der Aufträge und die daran geknüpften Bedingungen, wie aus diesen eine Sequenz gebildet werden darf.

6.2.1 Annahmen

Grundlage der Einsteuerung bleibt weiterhin eine Plansequenz von jetzt $2n$ Aufträgen. Diese darf in ihrer Reihenfolge vorerst nicht verändert werden. Aus dieser Sequenz können aufgrund von Sperrungen jederzeit Fahrzeuge herausfallen. Außerdem können die gesperrten Aufträge in jedem Takt wieder freigegeben werden, sodass sie wieder in die Plansequenz integriert werden müssen. Eine wesentliche Bedingung dabei ist, dass jeder Auftrag einen Fälligkeitstermin hat, bis zu dem er in die Montage eingeleitet werden soll.

Dabei wird für die Modellbildung die Annahme getroffen, dass zu Beginn des betrachteten Produktionszeitraums keine Sperrungen vorliegen und auch keine ehemals zurückgestellten Aufträge in der Plansequenz sind oder auf eine Eingliederung in diese warten. Diese Annahme wird auch von allen in dieser Arbeit verwendeten Testszenarien eingehalten. Dies dient lediglich der einfacheren Formulierbarkeit der Problemstellung und Generierung der Testinstanzen, ein Wegfall dieser Bedingung würde die Nutzbarkeit der entwickelten Verfahren in keiner Weise einschränken.

Außerdem wird angenommen, dass bei einem betrachteten Produktionsintervall der Länge n höchstens n Fahrzeuge gesperrt werden. Diese Schranke ist willkürlich gewählt und sollte in der Praxis niemals erreicht werden. Sie soll lediglich für das theoretische Modell garantieren, dass die Betrachtung von $2n$ Fahrzeugen in jedem Fall ausreicht, um die Produktion des gesamten Intervalls abzudecken.

Die letzte Annahme ist aus praktischer Sicht selbstverständlich, soll aber der Vollständigkeit halber aufgeführt werden. Die hier eingeführten Fälligkeitstermine dürfen nicht vor den Positionen gemäß Planreihenfolge liegen. Die direkte Konsequenz hieraus ist, dass alle nicht gesperrten Fahrzeuge garantiert vor ihrem Fälligkeitstermin eingesteuert werden.

6.2.2 Basismodell

Bevor das erste dynamische Modell aufgestellt wird, werden die neuen bzw. abgewandelten Variablen in Tabelle 6.1 definiert. Im Vergleich zum statischen Modell, muss

Mengen	
$I = \{1, ..., 2n\}$	Menge aller Fahrzeuge
$J = I$	Menge aller Positionen
Parameter	
due_i	Fälligkeitstermin von Fahrzeug i
β_{ij}	binäre Variable $\begin{cases} 1, \text{ wenn Fahrzeug } i \text{ in Takt } j \text{ nicht gesperrt ist} \\ 0, \text{ sonst} \end{cases}$
Hilfsvariablen	
α_{ij}	binäre Variable $\begin{cases} 1, \text{ wenn Fahrzeug } i \text{ in Takt } j \text{ nicht zurückgestellt ist} \\ 0, \text{ sonst} \end{cases}$

Tabelle 6.1: Notation in den dynamischen Modellen

das hier modellierte Problem schrittweise gelöst werden, d. h. alle Sperrinformationen sind erst zu Beginn eines Takts mit dem entsprechenden Index j bekannt und ebenso müssen am Ende des Takts j alle Variablen mit einem Index j belegt werden. Also sind nach einem Takt t alle Konstanten und Variablen mit einem Index über J für $j = 1, \ldots, t$ fixiert.

$$\min \sum_{j=1}^{n} \sum_{l \in A} s_{jl} \tag{6.1a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{6.1b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{6.1c}$$

$$t_{jl} = \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{6.1d}$$

$$s_{jl} = \begin{cases} 1, \text{ wenn } p_{j-1,l} - c + t_{jl} > d_l \\ 0, \text{ sonst} \end{cases} \qquad \forall j \in J,\ l \in A \tag{6.1e}$$

$$p_{jl} = \max\left(p_{j-1,l} - c + (1 - s_{jl}) \cdot t_{jl}, c\right) \qquad \forall j \in J,\ l \in A \tag{6.1f}$$

$$x_{ij} \leq \beta_{ij} \qquad \forall i \in I,\ j \in J \tag{6.1g}$$

$$\alpha_{ij} = \begin{cases} 1, \text{ wenn } \exists k < i : \sum_{t=1}^{j} x_{kt} + (1 - \alpha_{kt}) = 0 \\ \beta_{ij}, \text{ sonst} \end{cases} \qquad \forall i \in I,\ j \in J \quad (6.1h)$$

$$\sum_{j=1}^{t} x_{ij} - x_{i'j} \geq 0 \qquad (6.1i)$$
$$\forall i, i' \in I \text{ mit } i < i' \text{ und } \textstyle\sum_{j=1}^{t} 2 - \alpha_{ij} - \alpha_{i'j} = 0, t \in J$$

$$\sum_{j=1}^{t} x_{ij} \geq \alpha_{it} \qquad (6.1j)$$
$$\forall i \in I, t = due_i, \ldots, n$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \quad (6.1k)$$

Die Zielfunktion sowie die Nebenbedingungen (6.1b) bis (6.1f) und (6.1k) sind aus dem statischen Problem bekannt. An dieser Stelle ist lediglich anzumerken, dass die Zielfunktion über die ersten n Fahrzeuge gebildet wird, aber dennoch alle $2n$ Fahrzeuge auf $2n$ Positionen verteilt werden müssen.

Die Bedingung (6.1g) verhindert, dass gesperrte Fahrzeuge eingesteuert werden.

Die Gleichungen (6.1h) ermöglichen eine Unterscheidung zwischen einer Sperrung und einer tatsächlichen Zurückstellung. D. h. ein Fahrzeug, das laut β gesperrt ist, wird erst zurückgestellt ($\alpha = 0$), wenn alle nicht zurückgestellten Fahrzeuge, die in der Plansequenz vor dem betrachteten Fahrzeug sind, schon eingesteuert wurden. Dies garantiert, dass kein Fahrzeug beliebig verschoben werden darf (vgl. Bedingung (6.1i)), wenn vor dem ursprünglichen Einsteuerungstermin eine Sperrung vorlag, die allerdings nie zu einer tatsächlichen Zurückstellung geführt hat.

Die Einhaltung der Vorrangbeziehungen zwischen nicht zurückgestellten (an dieser Stelle ist die Unterscheidung von gesperrt und zurückgestellt essentiell) Fahrzeugen wird durch die Ungleichungen (6.1i) erzwungen.

Ein Überschreiten der Fälligkeitstermine ist laut Bedingung (6.1j) nur bei einer tatsächlichen Zurückstellung erlaubt. Nachdem die Sperrung eines überfälligen Auftrags aufgehoben wird, dieser also freigegeben ist, muss dieser sofort eingesteuert werden.

Ein Problem dieses ersten Modells besteht darin, dass möglicherweise keine zulässige Lösung existiert. Dies kann zum einen durch zu lang anhaltende Sperrungen - und damit auch Zurückstellungen - geschehen, zum anderen durch die sehr restriktive Bedingung mit den Fälligkeitsterminen. Der Zwang, einen verspäteten Auftrag direkt nach seiner Freigabe einzusteuern, führt bei gleichzeitiger Freigabe zweier (oder mehrerer) Aufträge dazu, dass wegen Bedingung (6.1j) beide Aufträge die nächste Position zugewiesen bekommen müssen, was zu einem Widerspruch mit (6.1c) führen würde.

Zunächst wird durch eine Annahme das erste Problem gelöst. Da für die Zielfunktion nur die n ersten Positionen relevant sind, kann danach auch auf Sperrungen verzichtet werden. Es gelte also:

$$\beta_{ij} = 1 \qquad \forall i \in I,\ j = n+1, \ldots, 2n \quad (6.2)$$

Daraus folgt wegen (6.1h) auch:

$$\alpha_{ij} = 1 \qquad \forall i \in I,\ j = n+1, \ldots, 2n \quad (6.3)$$

Mit dieser Annahme kann also die zweite Hälfte der Positionen stets in einer Reihenfolge entsprechend der Plansequenz besetzt werden, um eine zulässige Lösung zu bilden.

Nach dieser Anpassung kann nur noch das zweite erwähnte Problem die Existenz einer zulässigen Lösung verhindern. Zur Lösung dieses Konflikts wird nun eine Weiterentwicklung dieses Modells betrachtet.

Es wird eine Hilfsvariable a_{it} verwendet, die alle Aufträge markiert, die bis zu einem Zeitpunkt t mindestens in einem Takt zurückgestellt waren. Das Modell wird also ergänzt um:

$$a_{it} = \begin{cases} 0, \text{ wenn } \sum_{j=1}^{t} \alpha_{ij} = t \\ 1, \text{ sonst} \end{cases}$$

Daraus kann man die Anzahl A_t der Aufträge ableiten, die vor einem Zeitpunkt t

zurückgestellt waren und bis zum Zeitpunkt t noch nicht wieder eingesteuert wurden:

$$A_t = \sum_{i \in I} \max \left(a_{it} - \sum_{j=1}^{t-1} x_{ij}, 0 \right)$$

Mit diesen Definitionen kann schließlich Bedingung (6.1j) durch folgende ersetzt werden:

$$\sum_{j=1}^{t+A_t-1} x_{ij} \geq \sum_{s=t}^{t+A_t-1} \alpha_{is} - A_t + 1 \qquad \forall i \in I, t = due_i, \ldots, n$$

Da immer ein bestimmter Auftrag i betrachtet wird, der jedoch in den entscheidenden Fällen immer einer der Aufträge ist, die von A_t gezählt werden, muss in allen Summen, in denen A_t verwendet wird, 1 subtrahiert werden.

Die Konsequenz dieser Bedingung ist, dass ein Auftrag i in den nächsten A_t Takten eingesteuert werden muss, wenn er in all diesen Takten nicht zurückgestellt ist, also $\alpha_{is} = 1$ gilt. Nur in diesem Fall ist die rechte Seite der Ungleichung gleich 1, ansonsten ist sie 0 oder negativ.

Für den Fall $A_t = 1$ reduziert sich die Bedingung auf die alte Bedingung (6.1j). Nun muss noch der Sonderfall $A_t = 0$ abgedeckt werden. In diesem Fall gibt es keine Aufträge, die noch in die Plansequenz eingeordnet werden müssen. Deshalb ist an dieser Stelle keine Bedingung erforderlich.

Die komplette Formulierung des geänderten Modells ist also:

$$\min \sum_{j=1}^{n} \sum_{l \in A} s_{jl} \tag{6.4a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{6.4b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{6.4c}$$

$$t_{jl} = \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{6.4d}$$

$$s_{jl} = \begin{cases} 1, \text{ wenn } p_{j-1,l} - c + t_{jl} > d_l \\ 0, \text{ sonst} \end{cases} \qquad \forall j \in J,\ l \in A \quad (6.4e)$$

$$p_{jl} = \max\left(p_{j-1,l} - c + (1 - s_{jl}) \cdot t_{jl}, c\right) \qquad \forall j \in J,\ l \in A \quad (6.4f)$$

$$x_{ij} \leq \beta_{ij} \qquad \forall i \in I,\ j \in J \quad (6.4g)$$

$$\alpha_{ij} = \begin{cases} 1, \text{ wenn } \exists k < i : \sum_{t=1}^{j} x_{kt} + (1 - \alpha_{kt}) = 0 \\ \beta_{ij}, \text{ sonst} \end{cases} \qquad \forall i \in I,\ j \in J \quad (6.4h)$$

$$\sum_{j=1}^{t} x_{ij} - x_{i'j} \geq 0 \qquad (6.4i)$$

$$\forall i, i' \in I \text{ mit } i < i' \text{ und } \textstyle\sum_{j=1}^{t} 2 - \alpha_{ij} - \alpha_{i'j} = 0, t \in J$$

$$a_{it} = \begin{cases} 0, \text{ wenn } \sum_{j=1}^{t} \alpha_{ij} = t \\ 1, \text{ sonst} \end{cases} \qquad \forall i \in I,\ t \in J \quad (6.4j)$$

$$A_t = \sum_{i \in I} \max\left(a_{it} - \sum_{j=1}^{t-1} x_{ij}, 0\right) \qquad \forall t \in J \quad (6.4k)$$

$$\sum_{j=1}^{t+A_t-1} x_{ij} \geq \sum_{s=t}^{t+A_t-1} \alpha_{is} - A_t + 1 \qquad (6.4l)$$

$$\forall i \in I, t = due_i, \ldots, n \text{ und } A_t \geq 1$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \ (6.4m)$$

Für dieses Modell kann unter den vorher in diesem Abschnitt genannten Annahmen eine Existenzaussage formuliert werden. Die relevanten Annahmen, dass höchstens n Fahrzeuge gesperrt werden und die Fälligkeitstermine nicht vor den Planterminen liegen dürfen, können formal wie folgt formuliert werden:

$$\sum_{i \in I} \min\left(2n - \sum_{j \in J} \beta_{ij}, 1\right) \leq n \qquad (6.5)$$

$$due_i \geq i \qquad \forall i \in I \quad (6.6)$$

Satz 3. *Das Modell* (6.4) *besitzt für jede Instanz, die die Annahmen* (6.2), (6.5) *und* (6.6) *erfüllt, eine zulässige Lösung.*

Beweis. Im Modell sind die Bedingungen (6.4d), (6.4e), (6.4f), (6.4h), (6.4j) und (6.4k) lediglich Berechnungsvorschriften, die von gegebenen oder in vergangenen Takten ermittelten Größen abhängen. Diese sind also der Definition nach erfüllt. Es bleibt folglich zu zeigen, dass eine Belegung der x Variablen existiert, die nicht gegen die restlichen Bedingungen verstößt. Dazu wird die folgende Strategie betrachtet. Es wird immer das Fahrzeug mit der vordersten Position in der Plansequenz, welches nicht gesperrt ist, eingesteuert. Das bedeutet, in Takt t wird das $x_{i^*t} = 1$ gesetzt mit $i^* = \min_{i \in I : \beta_{it}=1 \text{und} \sum_{s=1}^{t-1} x_{is}=0} i.$

Ein solches Fahrzeug gibt es in den ersten n Takten immer, wegen der Annahme (6.5), dass maximal n von den $2n$ Fahrzeugen gesperrt werden. Ab Takt $n+1$ gibt es laut Annahme (6.2) keine Sperrungen mehr, also kann in allen Takten ein Fahrzeug gemäß der genannten Strategie gefunden werden.

Aus der Definition der Strategie folgt direkt, dass auch die Bedingungen (6.4b), (6.4c), (6.4g) und (6.4m) eingehalten werden.

Da Bedingung (6.4i) nur für zwei nie zurückgestellte Fahrzeuge aktiv wird, diese aber laut der gewählten Strategie immer entsprechend ihrer Reihenfolge in der Plansequenz ausgewählt werden, ist diese Bedingung ebenfalls erfüllt.

Bleibt als letztes die Einhaltung von Nebenbedingung (6.4l) zu zeigen. Sei i der betrachtete Auftrag und $t \geq due_i$ der betrachtete Zeitpunkt. Wenn $\alpha_{it} = 0$, dann ist die Bedingung unabhängig von der Belegung von (x_{ij}) stets erfüllt. Ebenso wenn $\sum_{j=1}^{t-1} x_{ij} = 1$. Also sei nun $\alpha_{it} = 1$ und $\sum_{j=1}^{t-1} x_{ij} = 0$. Jetzt werden zwei Fälle unterschieden.

Fall 1: i war nie zurückgestellt (dies ist nur mit $t = due_i = i$ möglich), dann wird mit der Strategie $x_{it} = 1$ gesetzt und die Bedingung ist erfüllt.

Fall 2: i war zu einem früheren Zeitpunkt zurückgestellt, ist aber in t wieder freigegeben. Für alle nicht von A_t erfassten und noch nicht fixierten Aufträge i' gilt: $i' > i$,

also können nach der verwendeten Strategie nur Fahrzeuge, die in A_t erfasst wurden, in den nächsten Takten eingesteuert werden. Folglich wird i, wie durch die Bedingung erfordert, innerhalb der nächsten A_t Takte eingesteuert, vorausgesetzt dieser Auftrag wird nicht erneut zurückgestellt. In dem Fall wäre jedoch die rechte Seite der Bedingung ≤ 0 und sie somit automatisch erfüllt.

□

Dieser Satz rechtfertigt eine Analyse des Problems in Hinblick auf robuste Methoden. Eine Strategie, die stets eine zulässige Sequenz erzeugt, wurde im Beweis vorgestellt. Andere Strategien müssen die gleiche Eigenschaft besitzen, dass unabhängig von den eintretenden Sperrungen die Erzeugung einer zulässigen Sequenz in jedem Takt aufrecht erhalten wird. Ebenfalls ermöglicht dieser Satz die einfache Generierung von Testinstanzen. Dabei muss lediglich auf die Einhaltung der genannten Annahmen geachtet werden. Auf diese Weise liefern die Tests zulässige und vergleichbare Lösungen.

Das Modell (6.4) kann noch weiter verändert werden, um eine flexiblere Einsteuerung von verspäteten Fahrzeugen zu ermöglichen. Dazu wird ein zusätzlicher Parameter verwendet:

k : Anzahl der Takte, nach denen ein verspätet freigegebenes Fahrzeug spätestens eingesteuert werden muss

Ein Ersetzen der Nebenbedingung (6.4l) durch die Bedingung

$$\sum_{j=1}^{t+A_t-1+k} x_{ij} \geq \sum_{s=t}^{t+A_t-1+k} \alpha_{is} - A_t - k + 1 \qquad \forall i \in I, t = due_i, \ldots, n,\ A_t \geq 1 \quad (6.7)$$

verhindert, dass Fahrzeuge, deren Fälligkeitstermine bei Freigabe schon in der Vergangenheit liegen oder in wenigen Takten erreicht werden, sofort eingesteuert werden, ohne die Auswirkungen auf die Zielfunktion zu berücksichtigen. Die Annahme dahinter ist, dass eine ohnehin vorliegende Verspätung bei zusätzlicher Verzögerung um wenige Minuten nicht wesentlich schlimmer wird. Allerdings kann die Belastung in der Montage durch Gewährung eines kleinen Spielraums bei der Einsteuerung deutlich reduziert werden. Dazu bestimmt k die Länge des Intervalls, in das zurückgestellte Fahrzeuge eingefügt werden müssen, die erst kurz vor bzw. nach ihrem Fälligkeitstermin due_i

freigegeben wurden.

Es sei an dieser Stelle angemerkt, dass die neue Bedingung (6.7) eine Relaxierung zur ursprünglichen Bedingung (6.4l) darstellt. Die Aussage von Satz 3 bleibt also erhalten.

6.2.3 Mehrtakter

Wie im statischen Fall kann auch dieses Modell erweitert werden, indem auch Mehrtakter durch geeignete Bedingungen repräsentiert werden. Dazu wird erneut die Bezeichnung nw_l verwendet, die die Anzahl der Werker an dem jeweiligen Arbeitsplatz l beschreibt. Wie in Modell (5.7) übernimmt nun die Variable p_{jl} für $j = -nw_l+1, \ldots, 0$ die Rolle der Startpositionen der einzelnen Werker. Bei der Analyse des dynamischen Problems wird lediglich die simple Modellierung von Mehrtaktern verwendet, bei der alle Werker unabhängig voneinander arbeiten und sich nicht gegenseitig aushelfen können. Mit dieser Änderung kann folgendes Modell formuliert werden:

$$\min \sum_{j=1}^{n} \sum_{l \in A} s_{jl} \tag{6.8a}$$

$$\text{s. t.} \sum_{j \in J} x_{ij} = 1 \qquad \forall i \in I \tag{6.8b}$$

$$\sum_{i \in I} x_{ij} = 1 \qquad \forall j \in J \tag{6.8c}$$

$$t_{jl} = \sum_{i \in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{6.8d}$$

$$s_{jl} = \begin{cases} 1, \text{ wenn } p_{j-nw_l,l} - nw_l \cdot c + t_{jl} > d_l \\ 0, \text{ sonst} \end{cases} \qquad \forall j \in J,\ l \in A \tag{6.8e}$$

$$p_{jl} = \max\left(p_{j-nw_l,l} - nw_l \cdot c + (1 - s_{jl}) \cdot t_{jl}, nw_l \cdot c\right) \qquad \forall j \in J,\ l \in A \tag{6.8f}$$

$$x_{ij} \leq \beta_{ij} \qquad \forall i \in I,\ j \in J \tag{6.8g}$$

$$\alpha_{ij} = \begin{cases} 1, \text{ wenn } \exists k < i : \sum_{t=1}^{j} x_{kt} + (1 - \alpha_{kt}) = 0 \\ \beta_{ij}, \text{ sonst} \end{cases} \qquad \forall i \in I,\ j \in J \tag{6.8h}$$

$$\sum_{j=1}^{t} x_{ij} - x_{i'j} \geq 0 \tag{6.8i}$$
$$\forall i, i' \in I \text{ mit } i < i' \text{ und} \sum_{j=1}^{t} 2 - \alpha_{ij} - \alpha_{i'j} = 0, t \in J$$

$$a_{it} = \begin{cases} 0, \text{ wenn } \sum_{j=1}^{t} \alpha_{ij} = t \\ 1, \text{ sonst} \end{cases} \qquad \forall i \in I,\ t \in J \tag{6.8j}$$

$$A_t = \sum_{i \in I} \max\left(a_{it} - \sum_{j=1}^{t-1} x_{ij}, 0\right) \qquad \forall t \in J \tag{6.8k}$$

$$\sum_{j=1}^{t+A_t-1+k} x_{ij} \geq \sum_{s=t}^{t+A_t-1+k} \alpha_{is} - A_t - k + 1 \tag{6.8l}$$
$$\forall i \in I, t = due_i, \ldots, n \text{ und } A_t \geq 1$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{6.8m}$$

Da sich in diesem Modell lediglich die Berechnung der Werkerpositionen verändert hat, überträgt sich die Existenzaussage hierauf.

Korollar 1. *Das Modell* (6.8) *besitzt für jede Instanz, die die Annahmen* (6.2), (6.5) *und* (6.6) *erfüllt, eine zulässige Lösung.*

Eine wichtige Anmerkung ist, dass mit diesen Modellen nicht die Anzahl der in einer realen Schicht benötigten Springereinsätze berechnet wird, sondern vielmehr die Anzahl der Springereinsätze, die an Fahrzeugen entstehen, die im Laufe einer Schicht eingesteuert werden. Da an dieser Stelle der genaue zeitliche Bedarf eines Unterstützers nicht relevant ist, ist dieses Vorgehen unproblematisch und aufgrund der praktikablen Berechnung sinnvoll. Eine genaue Betrachtung, wann ein Springereinsatz nötig ist, wird im nächsten Kapitel in Verbindung mit einer Modellierung der Springer je Schicht durchgeführt.

Zur verbesserten Anpassungen an die Praxis ist eine zusätzliche Erweiterung dieser Modelle möglich. Bisher wurde angenommen, dass jeder Unterstützereinsatz die gleichen Folgen hat, d. h. unter den gleichen Voraussetzungen möglich ist und die gleichen Kosten verursacht. In der Praxis gibt es jedoch Arbeitsplätze, an denen es z. B. aufgrund der technischen Gegebenheiten nicht möglich ist, dass ein Unterstützer gleichzei-

tig mit dem normalen Werker arbeitet. Ebenso ist es vorstellbar, dass an bestimmten Arbeitsplätzen der Unterstützereinsatz nur unter schwierigen Bedingungen oder von besonderen Experten durchführbar ist, was die Kosten eines Unterstützereinsatzes erhöhen würde. Deshalb wird im abschließenden Modell die Zielfunktion um eine Gewichtung w_l erweitert:

$$\min \sum_{j=1}^{n} \sum_{l \in A} w_l \cdot s_{jl} \tag{6.9}$$

In den Tests, die diese Erweiterung der Zielfunktion verwenden, werden die Arbeitsplätze in zwei Gruppen aufgeteilt. Dabei werden die Arbeitsplätze, an denen der Einsatz von Springern unbedingt vermieden werden soll, stärker gewichtet als die anderen Arbeitsplätze.

6.3 Lösungsstrategien

In diesem Abschnitt werden Strategien entwickelt, die die Einsteuerung durch einen kompletten Tag hindurch leiten können. Zunächst wird dabei auf die in der Literatur beschriebenen, grundsätzlichen Ansätze für allgemeine Online-Optimierungsprobleme eingegangen und diese werden auf ihre Tauglichkeit für das vorliegende Problem untersucht. Daraus werden Algorithmen abgeleitet, die in den folgenden Abschnitten getestet werden.

6.3.1 Allgemeine Ansätze

Grötschel et al. [GKR$^+$01] unterscheiden die folgenden vier Grundstrategien für Online-Algorithmen, deren mögliche Anwendung im betrachteten Kontext dargestellt wird.

1. FIFO

Sobald ein Auftrag freigegeben wird, wird dieser an eine Position in der Plansequenz eingefügt und danach wie jeder andere Auftrag aus der Sequenz behandelt. Diese Vari-

ante führt zu schnellen Entscheidungen und größtmöglicher Planungssicherheit für die Logistik. Aus praktischen Gesichtspunkten ist dies ein akzeptabler Ansatz, allerdings wird viel Potential für Verbesserungen verschenkt. Ein Auftrag, der kurzfristig an eine Position eingefügt worden ist, kann in den meisten Fällen ohne Zusatzaufwand für die Logistik erneut verschoben werden. Gerade wenn nach der Eingliederung in die Plansequenz neue Sperrungen vorgenommen werden müssen oder neue Aufträge freigegeben werden, ist die alte Position nicht mehr gut und eine Umsortierung kann zu einer Verbesserung führen.

2. GREEDY

Ein gieriger Ansatz betrachtet nur die Gegenwart. In dem hier betrachteten Kontext kann dies bedeuten, dass immer der Auftrag in die Montage eingeleitet wird, der am wenigsten Springereinsätze verursacht. Dabei kann zu jedem Zeitpunkt ein wiedereinzusteuernder Auftrag oder der nächste Auftrag der Plansequenz gewählt werden. Dieser Ansatz besitzt mehrere Schwachpunkte. Der größte ist die fehlende Betrachtung der Zukunft. Ein Fahrzeug, das ohne Springereinsätze montiert werden kann, kann trotzdem für folgende Fahrzeuge extrem hohe Driftpositionen der Werker hinterlassen und somit in den Folgetakten zu einer enormen Anzahl an Springereinsätzen führen. Außerdem droht einigen Fahrzeugen, die viele Sonderausstattungen und damit relativ hohe Belastungszeiten an vielen Stationen besitzen, eine lange Wartezeit. Sie werden schließlich nur durch Zwang eingesteuert und führen zwangsläufig zu starken Überlastungen in der Montage.

3. REPLAN

Bei diesem Ansatz bleiben alle einmal zurückgestellten Fahrzeuge immer variabel. Einmal in die Plansequenz integriert, können sie bei jedem neuen Ereignis - wie einer Sperrung - oder auch in jedem Takt neu positioniert werden. Dieser Ansatz bietet die größte Flexibilität und als statische Momentaufnahme auch das größte Optimierungspotential in jedem Takt. Die Gefahr bei diesem Ansatz besteht darin, dass Fahrzeuge bei jedem neuen Optimierungslauf immer weiter nach hinten gesetzt werden, was nach neuen unvorhergesehen Ereignissen zu Problemen führen kann.

4. IGNORE

Dies kann als freiere Interpretation des FIFO Ansatzes gesehen werden. Die Eingliederung in die Plansequenz muss nicht strikt in der Reihenfolge der Freigabe erfolgen. Wenn dies jedoch geschehen ist, werden die Aufträge im Gegensatz zum REPLAN Ansatz auch bei späteren Ereignissen, die eine Veränderung begründen könnten, in der Plansequenz an der gewählten Position verbleiben.

6.3.2 Konkrete Ausgestaltung der Ansätze

Zunächst wird der aus dem vorherigen Kapitel abgeleitete VTS+ Algorithmus 6.1 formal beschrieben, welcher eine Kernprozedur in den neu entwickelten Algorithmen darstellt.

Algorithmus 6.1: Variable Tabu Search+ (VTS+)

```
Initialisierung;
wenn |I^W| < 4 dann
  | rufe Algorithmus VTS auf;
sonst
  | rufe Algorithmus VNS auf;
Ende
rufe Algorithmus IV auf;
```

Bei diesem Verfahren ist anzumerken, dass die wiedereinzusteuernden Fahrzeuge in ein vorgegebenes Intervall der Plansequenz eingefügt werden müssen. Die Länge dieses Intervalls entspricht bei der Anwendung im weiteren Verlauf dieser Arbeit der Größe k, die aus dem Modell (6.8) bekannt ist. Die Übereinstimmung der Länge des betrachteten Intervalls und des erlaubten Verzögerns spät freigegebener Fahrzeuge vereinfacht die konkrete Formulierung einiger Algorithmen.

Ausgangspunkt für die Untersuchungen dynamischer Verfahren ist ein existierender regelbasierter Ansatz, der zwischen den Kategorien GREEDY und FIFO einzuordnen ist. Freigegebene Aufträge werden hierbei nach dem FIFO-Prinzip getestet und dann bei ausreichender Eignung an die nächste Position in die Sequenz eingesteuert. Ein Auftrag passt an die Position, wenn in Abhängigkeit der vorherigen und nachfolgenden

Aufträge bestimmte Werte in vorgegebenen Grenzen liegen. Falls eine Einsteuerungsbedingung nicht erfüllt ist, wird der Auftrag erst im nächsten Takt wieder betrachtet. Im Folgenden wird dieser Ansatz RULE genannt.

Bei diesem Verfahren wird eine vorgegebene Anzahl an Aufträgen in der Vergangenheit gemeinsam mit dem aktuell untersuchten Auftrag betrachtet. Wenn die Anzahl der durch diese Aufträge verursachten Überlastungen höher als ein vorgegebener Grenzwert ist, wird der untersuchte Auftrag im aktuellen Takt nicht in die Ist-Sequenz überführt. Falls der Grenzwert eingehalten wird, wird der Test mit den zukünftigen Aufträgen laut Plansequenz wiederholt. Falls auch hier der Grenzwert eingehalten wird, wird der Auftrag fixiert. Von diesem Vorgehen wird nach einer vorgegeben Zeit in Abhängigkeit vom Fälligkeitstermin abgewichen. Wenn der Fälligkeitstermin nahe oder überschritten ist, wird lediglich die Vergangenheit überprüft. Eine Einhaltung des Grenzwertes hier reicht dann aus, um den Auftrag zu fixieren.

Der zweite getestete Ansatz entspricht im Wesentlichen dem FIFO-Prinzip. Immer wenn neue Aufträge freigegeben werden, werden diese möglichst optimal in die bestehende Plansequenz integriert. Dazu werden die im vorhergehenden Kapitel beschriebenen Methoden verwendet. Diese Aufträge werden danach wie alle anderen Aufträge der Plansequenz behandelt, d. h. auch bei neuen Freigaben bleibt ihre relative Position zueinander unverändert. Je nach Wahl des Algorithmus, der zur Lösung des statischen Problems in einem Takt mit freigegebenen Aufträgen verwendet wird, werden diese Algorithmen DSE (dynamisches sequentielles Einsteuern; Algorithmus 6.2) bzw. DVTS (dynamische Variable Tabu Search; Algorithmus 6.3) genannt.

Algorithmus 6.2: Dynamisches SE (DSE)

```
Initialisierung;
für alle Takte j tue
    wenn Fahrzeuge I^W in Takt j freigegeben werden dann
        rufe Algorithmus SE auf;
    Ende
Ende
```

Eine Erweiterung dieses Ansatzes kann unter der Rubrik REPLAN eingeordnet werden. Bei diesem Algorithmus 6.4 wird in jedem Takt, in dem keine Fahrzeuge in die Plansequenz integriert werden müssen, erneut die Variable Tabu Search Prozedur auf-

Algorithmus 6.3: Dynamische VTS (DVTS)

```
Initialisierung;
für alle Takte j tue
    wenn Fahrzeuge I^W in Takt j freigegeben werden dann
        rufe Algorithmus VTS+ auf;
    Ende
Ende
```

gerufen und die zuvor eingegliederten Fahrzeuge dürfen verschoben werden, wenn dies die gesamte Sequenz verbessert. Dabei ist die Anwendung dieser Prozedur auf die Fahrzeuge beschränkt, die noch weit genug von ihrem Fälligkeitstermin entfernt sind, d. h. eine Verschiebung innerhalb des betrachteten Intervalls der Länge k kann sie nicht hinter ihren Fälligkeitstermin schieben. In Takt j darf kein Fahrzeug i mit $due_i \leq j+k$ verschoben werden.

Algorithmus 6.4: DVTS+REPLAN

```
Initialisierung;
für alle Takte j tue
    wenn Fahrzeuge I^W in Takt j freigegeben werden dann
        rufe Algorithmus VTS+ auf;
    sonst
        finde wiedereingesteuerte Fahrzeuge P^W mit Abstand zu ihrem
        Fälligkeitstermin in der Plansequenz;
        rufe Algorithmus VTS+ mit P^W ohne die Startprozedur SE auf;
    Ende
Ende
```

Alle bisher genannten Algorithmen haben gewisse Schwächen, auf die nun näher eingegangen wird, mit dem Ziel einen alternativen Ansatz zu entwickeln. Bei der Freigabe von mehreren Fahrzeugen mit stark unterschiedlichen Fälligkeitsterminen stehen die Algorithmen vor einer großen Herausforderung. Entweder alle Fahrzeuge werden in ein relativ kleines Intervall eingegliedert, damit auch der früheste Fälligkeitstermin eingehalten werden kann. Dies würde aber für die anderen Aufträge eine enorme Einschränkung des Optimierungspotentials bedeuten, was lediglich durch eine erneute Verschiebung der Aufträge in den Folgetakten ausgeglichen werden kann. Oder die

Fahrzeuge werden in individuelle Intervalle eingegliedert, was jedoch zu Veränderungen in den Algorithmen führen würde, da der Vertauschungsoperator nur unter besonderen Bedingungen angewandt werden kann. Außerdem steigt bei langen Intervallen die Rechenzeit in jedem Optimierungsschritt, was problematisch werden kann.

Deshalb wird nun ein Ansatz vorgestellt, der im Grundsatz die IGNORE-Strategie verwendet und dabei die Aufträge individuell betrachtet. Die Idee hinter diesem Ansatz ist es, die Termintreue zu berücksichtigen. Fahrzeuge, deren Plantermin schon weit überschritten ist, müssen unbedingt schnell eingesteuert werden. Fahrzeuge, die erst vor wenigen Takten zurückgestellt und wieder freigegeben worden sind, müssen, wenn es keinen guten Platz gibt, nicht direkt wiedereingesteuert werden.

Für die Entwicklung eines solchen Algorithmus muss zunächst geklärt werden, welche Position gut genug ist. Hier könnte eine simple Bedingung lauten, dass die Zielfunktion nicht verschlechtert werden darf.

Bei dem Vergleich der Zielfunktionswerte vor und nach dem Einfügen eines Auftrags ist zu beachten, dass zwei unterschiedlich lange Sequenzen miteinander verglichen werden. Intuitiv sollten hier die relativen Werte verglichen werden, jedoch führt ein solcher Ansatz bei diskreten Werten, die nur geringe Abweichungen haben, zu Problemen. Dies kommt insbesondere bei der Gewichtung von einzelnen Arbeitsplätzen zum Tragen, was durch folgendes Beispiel verdeutlicht wird. Ursprünglich wird eine Sequenz von 50 Fahrzeugen betrachtet, bei denen zehn Driftüberschreitungen vorkommen, von denen eine mit einem großen Faktor gewichtet ist. Die neue Sequenz mit 51 Fahrzeugen ist dann offensichtlich immer besser, wenn keine zweite stark gewichtete Driftüberschreitung hinzukommt. Das Gewicht der einen Driftüberschreitung wird nun auf 51 statt 50 Fahrzeuge verteilt. Die durch das zusätzliche Fahrzeug gewonnene Verbesserung kann also bei hinreichend großer Gewichtung eine große Anzahl an zusätzlichen normal gewichteten Driftüberschreitungen ausgleichen. Daher muss bei der Entscheidung auf die verschiedenen Größenordnungen und die Art der Werte (diskret oder kontinuierlich) geachtet werden und daraus eine geeignete Vergleichsmethode (absolut oder relativ) abgeleitet werden.

Neben der Fragestellung, wann eine Sequenz gut genug ist, um ein Fahrzeug einzusteuern, muss geklärt werden, was mit den Fahrzeugen geschieht, die zunächst von

diesem Test abgelehnt werden. Der einfachste Ansatz wäre, den gleichen Test durchzuführen, bis durch den nahenden Fälligkeitstermin ein zwingendes Eingliedern erfolgt. Zusätzlich könnte mit jedem Takt, den der Fälligkeitstermin näher rückt, der Test dahingehend verändert werden, dass der zu erreichende Zielfunktionswert etwas erhöht wird, damit es schneller zu einer Einsteuerung kommt und nicht im letzten möglichen Moment eine drastische Steigerung der Zielfunktion in Kauf genommen werden muss.

Ein alternativer Ansatz wird von dem sogenannten Sekretärinnenproblem abgeleitet. Ein Überblick hierzu findet sich z. B. bei Freeman [Fre83]. Grob formuliert geht es dabei darum, bei einer nacheinander vorgestellten Reihe von Möglichkeiten, von denen nur die aktuelle gewählt werden kann, die beste zu wählen, ohne die zukünftigen zu kennen. Übertragen auf das vorliegende Problem bedeutet dies, dass es das Ziel ist, die optimale Stoppzeit zu finden, wann eine Einsteuerung akzeptiert wird. Eine optimale Strategie für die Lösung der verschiedenen Facetten des theoretischen Problems ist es, eine bestimmte Zeit zu warten und die möglichen Ergebnisse zu betrachten. Ab einem geeigneten Moment weicht man von dieser Abwartestrategie ab und wählt die erste Variante, die besser ist, als alle zuvor beobachteten.

Im klassischen Problem ist es das Ziel, mit größter Wahrscheinlichkeit die optimale Lösung zu verwenden. Dies ist jedoch für das vorliegende Problem nicht sinnvoll. Hier sollte eher der Erwartungswert der gewählten Lösung optimiert werden. Von daher kann es weiterhin zu besseren Resultaten führen, auch in der ersten „Abwartephase“ schon zu stoppen, wenn eine hinreichend gute Einsteuerungsmöglichkeit gefunden wird.

Ein Problem bei der Verwendung dieser Strategie ist die im Vergleich zum klassischen Sekretärinnenproblem fehlende Unabhängigkeit der Werte in jedem einzelnen Takt. Dadurch, dass in jedem Takt mehrere aufeinanderfolgende Positionen überprüft werden, die sich – wenn sonst keine anderen Änderungen erfolgen – über mehrere Takte hinweg immer wiederholen, liegt eine hochgradige Abhängigkeit zwischen benachbarten Takten vor.

Bei dieser Strategie ist außerdem zu beachten, dass der Zielfunktionswert vermutlich kein geeignetes Maß ist, wenn über viele Takte hinweg verglichen wird. Es sollte hier eher die Veränderung des Zielfunktionswertes durch ein Einsteuern des betrach-

teten Fahrzeugs als Entscheidungsmaß herangezogen werden. Das Ziel dieser Methode könnte somit beschrieben werden als das Finden des Moments, in dem ein Auftrag die bestehende Sequenz möglichst wenig verschlechtert.

Eine zusätzliche Erweiterung der Entscheidungsfindung, ob ein Fahrzeug in die Plansequenz eingegliedert wird, könnte eine Verringerung der Schwelle der Zielfunktionsveränderung in den Takten sein, in denen schon ein Fahrzeug zum Eingliedern ausgewählt wurde. Denn nur wenn mehrere Fahrzeuge gleichzeitig in die Plansequenz integriert werden, können das volle Optimierungspotential ausgeschöpft und die Ergebnisse nach dem Eingliedern eines einzelnen Fahrzeugs weiter verbessert werden.

Außerdem muss der Test, ob ein Fahrzeug schon früh eingesteuert werden soll, sehr schnell durchgeführt werden können. Deshalb wird nur geprüft, was die bestmögliche Veränderung der Zielfunktion bei einer Einsteuerung dieses einzelnen Fahrzeugs ist. Die möglichen weiteren Verbesserungsmöglichkeiten bei gemeinsamer Betrachtung mehrerer Fahrzeuge, die gerade eingesteuert werden, und die Anwendung der heuristischen Verfahren werden dabei ignoriert. Eine zusätzliche Möglichkeit liegt darin, nachdem alle zunächst ausgewählten Fahrzeuge optimal positioniert worden sind, noch einmal zu testen, ob nun eine Einsteuerung weiterer Fahrzeuge die Zielfunktion in einem angestrebten Ausmaß verändert. Wenn dies der Fall ist und eine Eingliederung weiterer Fahrzeuge vorgenommen wird, kann allerdings keine erneute Optimierung durchgeführt werden. Diese muss dann auf den nächsten Takt verschoben werden. Von einer solchen Variante wird jedoch in dieser Arbeit abgesehen.

Ein allgemeines Problem beim optionalen Einsteuern von Fahrzeugen ist, dass stets die Gefahr besteht, dass Aufträge, die an viele Positionen gut passen, relativ schnell eingesteuert werden und Fahrzeuge, die aufgrund ihrer Bearbeitungszeiten schwer in eine Sequenz zu integrieren sind, relativ spät in die Sequenz eingegliedert werden. Die beschriebene Abwartestrategie setzt bei dem zweiten hier erwähnten Punkt an und kann dazu führen, auch die schweren Aufträge vor dem letzten möglichen Termin einzusteuern. Allerdings bleibt das Problem, dass Aufträge, die an den meisten Stationen Unterlast besitzen, sofort bei ihrer Freigabe einen geeigneten Platz finden. Folglich sollte vorab für jedes Fahrzeug analysiert werden, welche Auswirkung auf die Zielfunktion beim Einfügen in eine gegebene Sequenz zu erwarten ist. Daraus kann ein individueller Wert Δ_i abgeleitet werden, der angibt, welche Veränderung der Zielfunktion für

einen jeweiligen Auftrag akzeptabel ist, um eine Eingliederung in die Plansequenz zu veranlassen. Aus diesen Überlegungen lässt sich der WAIT Algorithmus 6.5 ableiten.

Algorithmus 6.5: Abwartestrategie (WAIT)

```
Initialisierung;
für alle Takte j tue
    π sei die aktuelle Plansequenz;
    für alle freigegebenen und nicht in Plansequenz eingegliederten Fahrzeuge
    i ∈ I^W tue
        update Δ_i;
        bestimme Sequenz π_i mit i an der besten Position;
        wenn f(π_i) − f(π) ≤ Δ_i dann
            π := π_i;
        Ende
    Ende
Ende
```

Bei diesem Algorithmus ist anzumerken, dass im Schritt Update Δ_i viele Faktoren hereinspielen. Diese sind zum einen ein vorab berechneter Richtwert, dazu - in Abhängigkeit von der vergangenen Zeit bis zum Fälligkeitstermin - die Testergebnisse der vorangegangenen Takte und zum anderen die Resultate der vorausgegangenen Iterationen im selben Takt.

Als Verbesserung dieses Algorithmus kann auch hier auf den VTS Algorithmus zurückgegriffen werden. Immer wenn mehrere Fahrzeuge gleichzeitig in die Plansequenz eingegliedert werden, kann durch diese zusätzliche Prozedur das Ergebnis weiter verbessert werden. Diese Erweiterung wird WAIT+VTS genannt und in Algorithmus 6.6 formal beschrieben.

Dieser Ansatz kann außerdem mit einer REPLAN-Variante ergänzt werden. Dabei können Fahrzeuge, die schon in die Plansequenz eingegliedert worden sind und die ausreichend Abstand zu ihrem Fälligkeitstermin besitzen, erneut verschoben werden. D. h. in Takten ohne Eingliederung von Fahrzeugen in die Plansequenz wird die zuvor beschriebene REPLAN-Prozedur aufgerufen. Diese Variante ist in Algorithmus 6.7 dargestellt.

Sämtliche Algorithmen verwenden eine zentrale Annahme über die Sperrungen. Es

Algorithmus 6.6: WAIT+VTS

```
Initialisierung;
für alle Takte j tue
    π sei die aktuelle Plansequenz;
    für alle freigegebenen und nicht in Plansequenz eingegliederten Fahrzeuge
    i ∈ I^W tue
        update Δ_i;
        bestimme Sequenz π_i mit i an der besten Position;
        wenn f(π_i) − f(π) ≤ Δ_i dann
            π := π_i;
        Ende
    Ende
    wenn mehrere Fahrzeuge P^W in Takt j eingegliedert worden sind dann
        rufe Algorithmus VTS+ mit P^W ohne die Startprozedur SE auf;
    Ende
Ende
```

wird pessimistisch angenommen, dass Sperrungen auf unbestimmte Zeit aufrecht bleiben und die entsprechenden Aufträge erst in dem Moment der Freigabe wieder berücksichtigt werden. Diese Ansicht impliziert, dass gesperrte Aufträge, die noch nicht zurückgestellt wurden, zwar formal in der Plansequenz bleiben, aber für die Optimierung innerhalb der Plansequenz nicht berücksichtigt werden. Die Plansequenz wird also für den Fall optimiert, dass alle in ihr enthaltenen gesperrten Aufträge zurückgestellt werden müssen.

Der Vorteil dieser Sichtweise ist, dass nur in den Momenten, in denen Sperrungen aufgehoben werden, die Plansequenz neu optimiert werden sollte. Bei einer optimistischen Sichtweise, die gesperrte Aufträge in der Plansequenz weiter berücksichtigt, muss für jeden einzelnen Auftrag im Moment seiner Zurückstellung die Plansequenz - insbesondere an den vorderen Positionen - überprüft werden, was ein erhebliches Risiko birgt. Eine alternative Betrachtung würde nur sinnvoll sein, wenn eine gute Prognose über voraussichtliche Sperrdauern möglich ist. Dies erscheint in der Praxis jedoch zweifelhaft und wird daher nicht weiter verfolgt.

Zur verwendeten pessimistischen Sichtweise ist anzumerken, dass die Zahl der möglichen Positionen, auf die freigegebene Fahrzeuge gesetzt werden können, um die Zahl

Algorithmus 6.7: WAIT+REPLAN

```
Initialisierung;
für alle Takte j tue
    π sei die aktuelle Plansequenz;
    für alle freigegebenen und nicht in Plansequenz eingegliederten Fahrzeuge
    i ∈ I^W tue
        update Δ_i;
        bestimme Sequenz π_i mit i an der besten Position;
        wenn f(π_i) − f(π) ≤ Δ_i dann
            π := π_i;
        Ende
    Ende
    wenn mehrere Fahrzeuge P^W in Takt j eingegliedert worden sind dann
        rufe Algorithmus VTS+ mit P^W ohne die Startprozedur SE auf;
    sonst
        finde wiedereingesteuerte Fahrzeuge P^W mit ausreichendem Abstand zu
        ihrem Fälligkeitstermin in der Plansequenz;
        rufe Algorithmus VTS+ mit P^W ohne die Startprozedur SE auf;
    Ende
Ende
```

der im betrachteten Intervall gesperrten Fahrzeuge sinkt. Wiedereingesteuerte Fahrzeuge werden bei einer Platzierung auf eine Position, die an ein gesperrtes Fahrzeug angrenzt, immer hinter dieses gesetzt.

Zum Abschuss dieses Abschnitts wird gezeigt, dass alle vorgestellten Algorithmen die geforderte Robustheit besitzen, also stets zulässige Sequenzen erzeugen.

Satz 4. *Alle in diesem Abschnitt vorgestellten Algorithmen erzeugen für jede Probleminstanz, die die Annahmen* (6.2)*,* (6.5) *und* (6.6) *erfüllt, eine zulässige Lösung.*

Beweis. Der erste Teil des Beweises verläuft analog zum Beweis von Satz 3. Die Bedingungen (6.8d), (6.8e), (6.8f), (6.8h), (6.8j) und (6.8k) sind der Definition nach erfüllt. Es ist einfach zu sehen, dass von einem beliebigen Algorithmus erzeugte Sequenzen ebenfalls die Bedingungen (6.8b), (6.8c), (6.8g) und (6.8m) einhalten. Von sämtlichen Algorithmen werden nur geblockte Aufträge verschoben, folglich wird auch Bedingung (6.8i) erfüllt.

Wie zuvor bleibt zu zeigen, dass auch Bedingung (6.8l) unabhängig von den auftretenden Sperrungen erfüllt bleibt. Dazu sei i ein beliebiger Auftrag und t_i der früheste Takt, für den $\alpha_{is} = 1 \quad \forall s = t_i, \ldots, t_i + A_{t_i} + k - 1$ und $t_i \geq due_i - k$ gelte. Nun gibt es zwei mögliche Fälle.

Im ersten Fall ist i bereits zu Beginn von Takt t_i in der Plansequenz. Die Position von i kann dann höchstens $t_i + k + A^*_{t_i}$ sein, wobei $A^*_{t_i}$ die Anzahl der zuvor zurückgestellten Aufträge, die freigegeben und an eine Position vor i in die Plansequenz gesetzt worden sind, bezeichnet.

Im zweiten Fall ist i zu Beginn von Takt t_i noch nicht in der Plansequenz. Dann würde jeder Algorithmus in diesem Takt den Auftrag i auf eine der nächsten $k + A^*_{t_i}$ Positionen, also spätestens auf Position $t_i + k + A^*_{t_i}$ setzen. Die Position von i ist also in beiden Fällen am Ende von Takt t_i durch $t_i + k + A^*_{t_i}$ begrenzt.

Diese Position kann sich nun nur noch nach hinten verschieben, wenn ein zurückgestellter Auftrag vor i eingefügt wird oder wenn ein Auftrag, der hinter i in der Plansequenz ist, vorgezogen wird. Das Vorziehen eines Auftrags geschieht in den Algorithmen nur, wenn dieser vorher einmal zurückgestellt war. Also können höchstens $A_{t_i} - 1$ zurückgestellte Fahrzeuge vor i in die Plansequenz platziert werden. Davon sind noch die $A^*_{t_i}$ Aufträge abzuziehen, die bereits vor i in der Plansequenz sind. Somit kann der Auftrag i durch das Einsetzen und Versetzen anderer Aufträge höchstens auf Position $t_i + k + A^*_{t_i} + A_{t_i} - 1 - A^*_{t_i} = t_i + k + A_{t_i} - 1$ zurückgeschoben werden. Eine explizite Versetzung von i selber in der REPLAN Prozedur geschieht aufgrund der Bedingung an den Fälligkeitstermin ebenfalls nicht. Also ist auch die letzte Bedingung (6.8l) stets erfüllt.

□

6.4 Einstellungen

Die im vorherigen Abschnitt beschriebenen Algorithmen besitzen verschiedene Einstellungsmöglichkeiten. In diesem Abschnitt wird darauf eingegangen, welche Einstellungen sich in Konfigurationstests als sinnvoll herauskristallisiert haben. Mit den so

gewählten Einstellungen wird danach eine Testserie durchgeführt, um die verschiedenen Algorithmen zu vergleichen.

6.4.1 Zielfunktionen

Schon im Kapitel über das statische Problem wurde im Abschnitt der Anomalien erwähnt, dass eine Abwandlung der Zielfunktion in Betracht kommt, damit die Anzahl der Driftüberschreitungen auch indirekt minimiert wird. Noch wichtiger wird diese Möglichkeit im dynamischen Fall. Neben der Minimierung der aktuellen Zielfunktion sollte ein weiteres Ziel angestrebt werden. Die Produktionssequenz sollte so gebildet werden, dass auch Veränderungen, die in der Zukunft eintreten können, die Zielfunktion nicht zu negativ beeinflussen. Dieser Aspekt wird insbesondere relevant, wenn auf eine Verwendung der REPLAN Algorithmen verzichtet werden soll, da möglicherweise angestrebt wird, die Plansequenz so selten wie möglich zu verändern. In diesem Fall ist die Erweiterung der Zielfunktion um eine weitere Komponente eine vielversprechende Variante.

Entscheidend für die Größe der Wahrscheinlichkeit, dass zusätzliche Driftüberschreitungen entstehen, ist die Position der Werker. Wenn Werker permanent bis kurz vor ihrer Driftgrenze arbeiten, kann nur ein einzelner Auftrag, der eingefügt wird oder ein Auftrag, der entfernt wird, zu einer Driftüberschreitung führen. Deshalb wird nun vorgeschlagen, die Summe der Werkerpositionen nach Abzug des Stationsbereichs (TWO = total work overload) nach jedem Takt als zusätzlichen Wert in die Zielfunktion aufzunehmen. Dabei muss entschieden werden, wie dieser Wert gewichtet werden soll. Doch zunächst soll anhand eines Beispiels gezeigt werden, dass zwei Sequenzen mit der gleichen Anzahl an Driftüberschreitungen durch Verwendung der TWO sinnvoll bewertet werden können.

Beispiel 4. *Es werden ein einziger Arbeitsplatz und eine Sequenz von fünf Fahrzeugen betrachtet. Die Taktzeit beträgt genau wie die zusätzlich erlaubte Drift 10 s. Die Bearbeitungszeit aller fünf Planfahrzeuge beträgt 12 s. Daraus ergibt sich, wie Abbildung 6.1 zu entnehmen ist, eine TWO von 30 s.*

In diese Sequenz soll nun ein Fahrzeug mit einer Bearbeitungszeit von 5 s eingegliedert

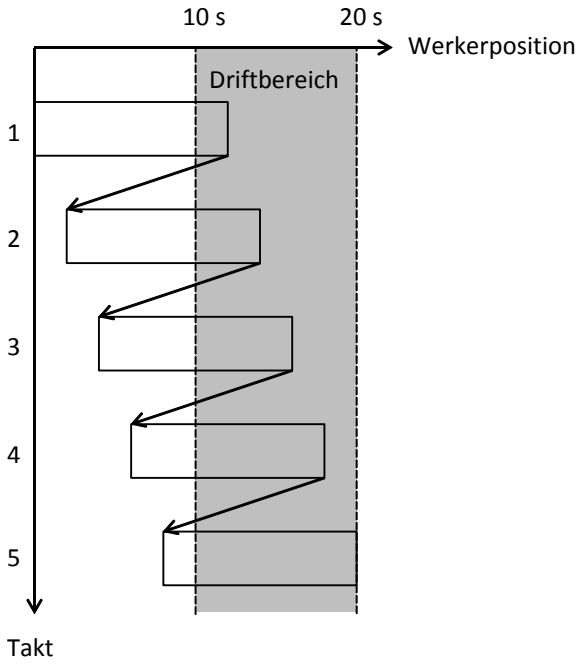


Abbildung 6.1: Werkerbewegung bei der Plansequenz

werden. Dies führt unabhängig von der Position zu keiner Driftüberschreitung. Jedoch sind die Auswirkungen im Hinblick auf weitere Veränderungen in der Zukunft nicht unerheblich, was auch durch die TWO ausgedrückt wird. In Abbildung 6.2 sind die Folgen für die Werker bei einer Eingliederung nach dem dritten und nach dem fünften Auftrag der Plansequenz gegenübergestellt.

Für die modellierte Zielfunktion sind beide Sequenzen gleich gut. Ebenso enden beide Sequenzen im gleichen Zustand (der Werker ist 5 s gedriftet). Jedoch würde jeder erfahrene Planer die rechte Variante bevorzugen. Ein relevanter Aspekt ist, dass der Werker im Durchschnitt weiter von der Driftgrenze entfernt arbeiten kann, was aus arbeitspsychologischer Sicht vorteilhaft ist. Aus rein mathematischer Sicht liegt der Vorteil darin, dass weitere Fahrzeuge mit geringer Überlast in die Sequenz integriert werden können, ohne direkt eine Driftüberschreitung zu verursachen. Somit stellt die Variante mit einer kleineren TWO die robustere Lösung dar.

In dem beschriebenen Beispiel wird eine weitere wichtige Größe deutlich, die Einfluss auf die Qualität einer Sequenz hat, nämlich der Endzustand des Systems. Damit ist die Position der Werker nach dem letzten betrachteten Auftrag gemeint. Während die

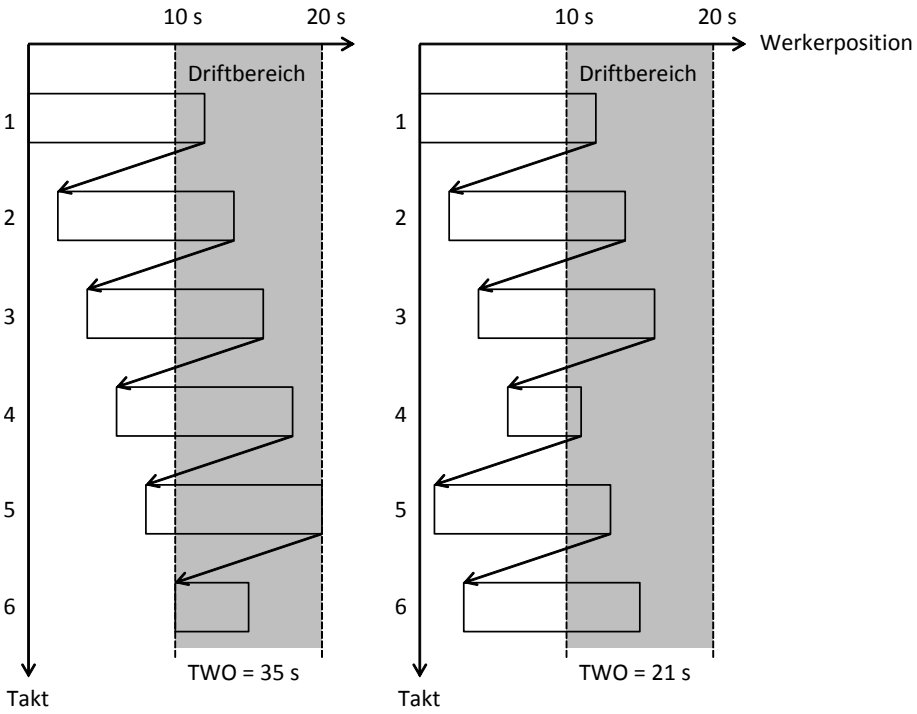


Abbildung 6.2: Werkerbewegung nach Einfügen des sechsten Auftrags

TWO ein Indikator für das Potential der Driftüberschreitungen bei Änderungen der Sequenz ist, ist die Endposition der Werker der Indikator für weitere Driftüberschreitungen im zukünftigen Produktionsverlauf hinter dem betrachteten Horizont. Dieser Wert, der auch ein Summand der TWO ist, nimmt also eine besondere Rolle ein.

Ausgehend von diesen Überlegungen muss für den Algorithmus festgelegt werden, welche dieser Werte wie stark in die Berechnung einer neuen Zielfunktion eingehen sollen.

Außer der allgemeinen Gewichtung ist weiterhin jede einzelne Station zu betrachten. Eine Ergänzung der Zielfunktion um die TWO muss nicht an allen Stationen sinnvoll sein. Dieser Wert ist darauf ausgelegt langfristig Driftüberschreitungen zu verhindern. Deshalb kann an Stationen mit großer Varianz der Bearbeitungszeiten, an denen wenige Fahrzeuge schon Driftüberschreitungen auslösen, auf die Verwendung der TWO verzichtet werden.

Unabhängig von der Entscheidung, ob an einer Station die TWO relevant ist, stellt sich

die Frage nach der Skalierung je Station. Einerseits kann losgelöst von den Spezifikationen einer Station die Werkerposition in Zeiteinheiten gemessen werden. Andererseits kann dieser Wert ins Verhältnis zur Kapazität einer Station gesetzt werden, da z. B. an Stationen mit mehreren Werkern eine Driftüberschreitung von einem ganzen Takt nur einem Bruchteil der Stationslänge entspricht. Da das eigentliche Ziel darin besteht, potentielle Driftüberschreitungen zu prognostizieren, wird hier vorgeschlagen, die Überschreitung der Stationsgrenze in Relation zum erlaubten Driftbereich zu setzen.

Verschiedene Varianten der Gewichtung und Skalierung wurden in einer Kalibrierungsphase getestet. Daraus abgeleitet wird auf eine besondere Beachtung der Endpositionen der Werker verzichtet. Die TWO wird - wie beschrieben - nur an Arbeitsplätzen mit einer begrenzten Varianz betrachtet. Außerdem wird die Werkerposition relativ zur jeweiligen Kapazität an den Arbeitsplätzen ermittelt. Als eine sinnvolle Gewichtung stellte sich der Faktor $\frac{1}{10}$ heraus. D. h. bei einer Vermeidung von Arbeiten im Driftbereich, die dem Umfang von zehn Driftbereichen entsprechen, wird eine zusätzliche Driftüberschreitung akzeptiert.

Diese Variante wird alternativ zur bekannten Zielfunktion, die die gewichtete oder ungewichtete Anzahl von Driftüberschreitungen minimiert, eingesetzt. Bei dieser wird die TWO in den Tests als Entscheidungsgröße zwischen zwei Sequenzen mit gleich vielen Driftüberschreitungen eingesetzt.

6.4.2 Intervalllänge

Eine gute Wahl des Bereichs, in den Fahrzeuge eingegliedert werden dürfen, ist ein wichtiger Schritt bei der Suche nach einem optimalen Algorithmus. Wenn dieses Intervall zu klein gewählt wird, kann das Optimierungspotential nicht ausgeschöpft werden. Außerdem können viele erneute Umsortierungen in weiteren Takten die Folge sein. Wenn das Intervall zu groß gewählt wird, können die Teilprobleme in den einzelnen Takten eventuell nur noch suboptimal gelöst werden. Zudem ist eine Betrachtung eines sehr großen Intervalls problematisch, wenn neue Sperrungen vorgenommen werden müssen. Je weiter nach hinten ein Auftrag in die Plansequenz eingefügt wird, umso

wahrscheinlicher ist es, dass sich die Sequenz in diesem Bereich nochmal verändert, bevor der Auftrag in die Montage eingeleitet wird. Danach ist die zuvor zugewiesene Position möglicherweise nicht mehr optimal. Eine Intervalllänge von 40 hat sich als guter Mittelweg erwiesen und liefert die Grundlage für die in dieser Arbeit präsentierten Tests.

6.4.3 Restsequenz

Bei der Entwicklung von approximativen Algorithmen muss immer zwischen den Aspekten Rechenzeit und Genauigkeit abgewogen werden. Diese Abwägung kann sich auch in einzelnen Details der Algorithmen wiederfinden. So gibt es bei dem hier beschriebenen Verfahren die Festlegung der Länge der *Restsequenz*. Dies ist die Bezeichnung für die Anzahl an Fahrzeugen, die bei der Berechnung der Zielfunktion zusätzlich zu den Fahrzeugen des Intervalls, in das die wiedereinzusteuernden Fahrzeuge integriert werden können, herangezogen werden. Je länger die Restsequenz festgelegt wird, desto länger dauert die Bewertung jeder möglichen Sequenz. Auf der anderen Seite steigt die Genauigkeit der Bewertung.

Zur Bestimmung der optimalen Größe der Restsequenz wird durch Simulationen ermittelt, welcher Wert zu den besten Endergebnissen bei realen Probleminstanzen führt. Dazu wird der beschriebene DVTS Algorithmus mit verschiedenen Restsequenzlängen zwischen 0 und 100 auf identische Testszenarien angewandt und die Zahl der über das gesamte Produktionsintervall entstandenen Driftüberschreitungen verglichen. Die Testgrundlage und die Testumgebung ist ähnlich zu den Vergleichstests, die im folgenden Abschnitt 6.5 ausführlich beschrieben werden.

Die durchschnittlichen Ergebnisse sind in Abbildung 6.3 dargestellt. Bei Gewichtung der Arbeitsplätze kann die Auswirkung auf wenige kritische Arbeitsplätze analysiert werden, was in Abbildung 6.4 zu sehen ist.

Das offensichtliche Ergebnis ist, dass unbedingt eine Restsequenz betrachtet werden soll. Die genaue Festlegung der Länge ist ausgehend von den Ergebnissen nicht eindeutig abzuleiten.

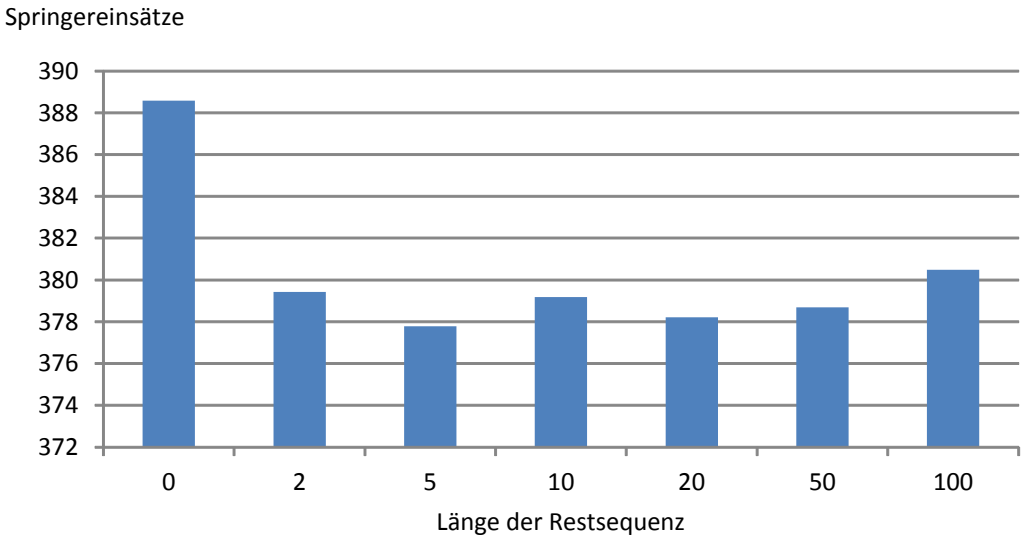


Abbildung 6.3: Auswirkungen der Restsequenz auf die Zahl der Springereinsätze

Im Hinblick auf die kritischen Driftüberschreitungen kann festgehalten werden, dass eine sehr weite Betrachtung der Restsequenz in manchen Fällen helfen kann, einzelne kritische Driftüberschreitungen zu verhindern. Je länger die betrachtete Restsequenz ist, desto besser werden die Ergebnisse. Allerdings sind die Differenzen sehr klein, sodass die Veränderung der Anzahl der normalen Driftüberschreitungen immer mitbetrachtet werden sollte. Hier zeigt sich ein anderes Bild. Die besten Ergebnisse liefern die Algorithmen, die mit Restsequenzlängen von 5 bis 50 arbeiten.

Auffallend hierbei ist das relativ schlechte Ergebnis bei einer Restsequenzlänge von zehn. Um dies näher zu analysieren, wird das Szenario mit der größten Differenz zur Restsequenzlänge von fünf betrachtet. Hier unterscheidet sich das durchschnittliche Ergebnis um über 3 %. Es ist jedoch unwahrscheinlich, dass diese Verschlechterung von der längeren Rechenzeit verursacht wurde, da in den zehn Testdurchläufen mit einer Restsequenzlänge von zehn nur zwei verschiedene Lösungen vom Algorithmus produziert wurden, obwohl dieser zufällige Aspekte verwendet.

Eine mögliche Erklärung liefert die Tatsache, dass die Zielfunktion, die optimiert wird, am Ende des betrachteten Produktionszeitraums über das für das Ergebnis relevante Intervall hinausgeht. Das bedeutet, dass die Unterschiede in den Ergebnissen lediglich kurzfristige Einflüsse widerspiegeln könnten. Ansonsten liegt die Vermutung nahe, dass

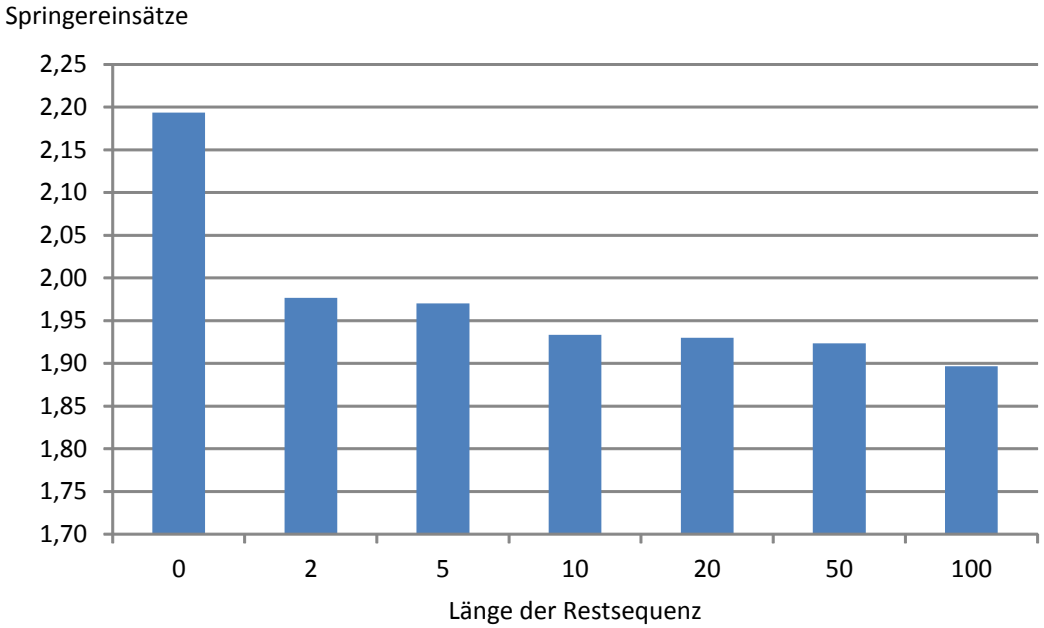


Abbildung 6.4: Auswirkungen der Restsequenz auf die Zahl der Springereinsätze an kritischen Arbeitsplätzen

die neu auftretenden Sperrungen und Freigaben für eine Restsequenzlänge von zehn zufällig negative Folgen haben. Die Resultate können bei einer kleinen Veränderung der Restsequenzlänge aufgrund einzelner Gegebenheiten in Testszenarien zwar deutlich positiv oder negativ ausfallen, diese Schwankungen sind jedoch nicht steuerbar.

Letztlich kann davon ausgegangen werden, dass die Auswirkungen der Restsequenzlänge auf die Ergebnisse in einem gewissen Rahmen ähnlich sind. Die Wahl der Restsequenzlänge ist folglich keine sensible Entscheidung. Zur weiteren Analyse der anderen Einstellungen wird an dieser Stelle die Länge der Restsequenz auf 20 festgesetzt.

6.4.4 Abwartestrategie

Bei der Anwendung eines WAIT Algorithmus wird folgende Strategie verwendet. Ein freigegebenes Fahrzeug wird nur dann in die Plansequenz integriert, wenn sich dadurch die Zielfunktion höchstens entsprechend eines vorgegebenen Wertes verändert. Dieser Wert wird vorab bestimmt. Dazu wird jeder Auftrag testweise an alle Positio-

nen einer hinreichend langen Sequenz gesetzt und die Veränderung der Zielfunktion betrachtet. Aus dem Mittelwert und der Standardabweichung der Veränderung lässt sich ein Wert ableiten, der nur bei Platzierung auf den besten Positionen erreicht wird. Unter Annahme einer Normalverteilung wird vom Mittelwert die zweieinhalbfache Standardabweichung abgezogen, sodass weniger als ein Prozent der Positionen - nur die am besten passenden - akzeptiert werden.

Der so ermittelte Richtwert dient in den ersten Takten nach Freigabe eines Auftrags als Entscheidungsgrundlage dafür, ob der jeweilige Auftrag in die Plansequenz eingegliedert wird. Vom Fälligkeitstermin leitet sich ein Takt ab, in dem ein Auftrag spätestens eine Position in der Plansequenz zugewiesen bekommen soll. Als sinnvoll hat sich die folgende Vorgehensweise herausgestellt. Während der ersten Hälfte der zur Verfügung stehenden Takte wird mit dem zuvor berechneten Richtwert verglichen. Wenn dieser Wert nie erreicht wird und der Auftrag also noch nicht in der Plansequenz ist, wird der Richtwert durch den besten Wert, der in den ersten Takten möglich gewesen wäre, ersetzt. Es wird also aus der Vergangenheit der realen Problemstellung gelernt.

Des Weiteren wird der Richtwert - egal ob vor oder während des laufenden Betriebs berechnet - temporär um eins herabgesetzt, wenn in dem aktuellen Takt schon ein anderer Auftrag in die Plansequenz integriert wurde. Dies dient dazu, das volle Optimierungspotential beim gleichzeitigen Eingliedern mehrerer Fahrzeuge auszunutzen.

6.5 Bewertung der Ansätze

In diesem Abschnitt wird analysiert, wie gut die von den verschiedenen Algorithmen produzierten Lösungen sind. Dazu wird zunächst beschrieben, welche Tests durchgeführt wurden. Danach wird darauf eingegangen, welche Möglichkeiten bestehen, um die Lösungsqualität zu bewerten. Die Ergebnisse werden dann im Vergleich zueinander dargestellt und schließlich Resultate abgeleitet.

6.5.1 Testbeschreibung

Die Basis für die durchgeführten Tests der vorgestellten Verfahren bilden optimierte Plansequenzen für fünf verschiedene Produktionstage. Dabei wurde in den Tests jeweils ein Zeitraum, der ungefähr einer Schicht entspricht, als Berechnungsgrundlage gewählt. Dieser Idee folgend wurde in jedem Test die Montage von 350 Aufträgen simuliert.

Für alle Testtage wurden sechs verschiedene Sperrszenarien generiert, die im Folgenden näher beschrieben werden. Also gibt es insgesamt 30 Testinstanzen. Alle Algorithmen, die zufällige Verfahren verwenden, wurden zehnfach mit allen Daten gestartet und die in dieser Arbeit dargestellten Ergebnisse repräsentieren immer den Durchschnitt über alle Durchläufe und Testtage.

Es wurden insgesamt sechs verschiedene Sperr- bzw. Freigabeszenarien generiert. Zunächst gibt es drei Szenarien, in denen zufällig ausgewählte Aufträge gesperrt werden. Dabei werden im Laufe der Schicht je drei Sperrungen aktiv, die insgesamt Sperrungen von knapp 5, 10 bzw. 20 % der Plansequenz darstellen sollen. Im Detail sehen die Sperrungen folgendermaßen aus:

- 5 %:
  - Sperrung von fünf Aufträgen aus den Positionen 21 bis 71 in Takt 20 und Freigabe in Takt 80
  - Sperrung von fünf Aufträgen aus den Positionen 101 bis 151 in Takt 80 und Freigabe in Takt 160
  - Sperrung von fünf Aufträgen aus den Positionen 161 bis 211 in Takt 120 und Freigabe in Takt 220
- 10 %:
  - Sperrung von elf Aufträgen aus den Positionen 21 bis 71 in Takt 20 und Freigabe in Takt 80
  - Sperrung von elf Aufträgen aus den Positionen 101 bis 151 in Takt 80 und

Freigabe in Takt 160

- Sperrung von elf Aufträgen aus den Positionen 161 bis 211 in Takt 120 und Freigabe in Takt 220

- 20 %:
  - Sperrung von 23 Aufträgen aus den Positionen 21 bis 71 in Takt 20 und Freigabe in Takt 80
  - Sperrung von 23 Aufträgen aus den Positionen 101 bis 151 in Takt 75 und Freigabe in Takt 160
  - Sperrung von 23 Aufträgen aus den Positionen 161 bis 211 in Takt 110 und Freigabe in Takt 220

Diese Wahl garantiert, dass bis zum Takt 350 wieder alle Fahrzeuge in die Sequenz integriert worden sind und somit immer die gleichen Fahrzeuge der betrachteten Schicht zugeordnet sind, sodass Vergleiche zwischen verschiedenen Algorithmen und insbesondere die Berechnung von Schranken möglich sind. Außerdem wurden die Zeitpunkte der Sperrungen und Freigaben so gewählt, dass immer alle Sperrungen unabhängig von der Produktionssequenz tatsächlich aktiv werden. Diese Überlegung führte bei dem dritten Testszenario zu etwas früheren Sperrungen. Wenn hier die gleichen Sperrtakte wie in den vorherigen Szenarien gewählt worden wären, wäre es möglich, dass bei später Einsteuerung der ersten gesperrten Fahrzeuge, die später zu sperrenden Fahrzeuge so weit vorgezogen werden, dass diese schon in die Montage eingeleitet werden, bevor die Sperrung aktiv wird. Dies könnte zu Verzerrungen der Ergebnisse führen und wurde durch die hier beschriebenen Sperrszenarien ausgeschlossen.

Des Weiteren wurden drei Sperrszenarien generiert, die das reale Geschehen besser widerspiegeln sollen. Dazu werden Fahrzeuge gesperrt, die einen gleichen Code, d. h. ein gleiches Ausstattungsmerkmal besitzen. Eine solche Art der Sperrung entspricht dem realen Fall, wenn eine Karosserievariante oder ein bestimmtes Teil vorübergehend nicht verfügbar ist. Für die Tests wurden die folgenden drei Sperrungen simuliert:

- Code A:
  - Sperrung von allen Aufträgen mit diesem Code in Takt 20 und Freigabe in Takt 120
- Code B:
  - Sperrung von allen Aufträgen mit diesem Code in Takt 100 und Freigabe in Takt 200
- Codes A und B:
  - Sperrung von allen Aufträgen mit Code A in Takt 20 und Freigabe in Takt 120
  - Sperrung von allen Aufträgen mit Code B in Takt 100 und Freigabe in Takt 200

Für die ersten Tests wurde das Mehrtaktermodell (6.8) verwendet. Danach wurden in Anlehnung an die realistischen Bedingungen einige Stationen als kritisch identifiziert und mit der entsprechenden Gewichtung der Zielfunktion (6.9) die Tests wiederholt. Die Testsimulation wurde in C# implementiert und die Tests wurden auf einem Intel® Xeon® Prozessor mit 2,66 GHz pro Kern, 22 GB RAM unter dem Betriebssystem Microsoft Windows Server 2003 durchgeführt. Mehrere Instanzen wurden dabei parallel auf je einem Kern berechnet.

Die verfügbare Rechenzeit der Kernprozedur VTS bzw. VNS wurde dabei auf 5 s reduziert. Dies hat im Wesentlichen zwei Gründe. Zum einen kann auf diese Weise eine umfassende Anzahl an Tests in einer akzeptablen Zeit durchgeführt werden, sodass aussagekräftige Resultate entstehen. Zum anderen ermöglicht dies auch eine Anwendung der Verfahren auf langsameren Rechnersystemen, die teilweise in der Praxis zum Einsatz kommen. Eine Erhöhung dieser Zeit auf 30 s wurde beim DVTS Algorithmus an einem Teil der Szenarien getestet. Dadurch wurde eine leichte Verbesserung der Ergebnisse um etwa 1 % erreicht. Diese Arbeit beschränkt sich im Folgenden jedoch auf die limitierte Rechenzeit.

Eine explizite Zeitbegrenzung der gesamten Rechnungen pro Takt kann nicht in allen Algorithmen verwendet werden, da die Dauer bestimmter Abschnitte, die komplett durchlaufen sollen, nicht vorab bekannt ist. Die Gesamtrechenzeiten der kompletten Simulationen einer Schicht sind stark abhängig vom Zeitpunkt der Einsteuerungen einzelner Aufträge und daher stark schwankend und nicht aussagekräftig.

6.5.2 Bewertungsmöglichkeit

Zunächst soll allgemein betrachtet werden, wie die Ergebnisse bewertet werden. Anders als im statischen Fall, bei dem zumindest für kleine Instanzen eine Optimallösung berechnet werden konnte, ist die Situation nun zu komplex, um die Ergebnisse mit optimalen Lösungen vergleichen zu können.

Deshalb wird versucht geeignete Vergleichswerte zu finden, um die Lösungsqualität abschätzen zu können. Aus wissenschaftlicher Sicht wäre eine Schranke, die den Zielfunktionswert nach unten begrenzt, erstrebenswert. Jedoch sind selbst einfache Relaxierungen des vorliegenden Problems nicht in angemessener Zeit zu lösen. Lediglich eine mehrfache Teilung des Problems mit zusätzlichen Relaxierungen und Abrundungen könnte zu einer sicheren Schranke führen, die dann jedoch weit von realistischen Werten entfernt ist, sodass dieser Ansatz für das vorliegende Praxisproblem nicht verfolgt wird.

Vielmehr wird versucht eine Schranke zu berechnen, die das unter realistischen Bedingungen zu erreichende Ergebnis begrenzt. Dieser Idee folgend kann aus praktischer Sicht eine untere Schranke für jede beliebige Zusammensetzung von Sperrungen herangezogen werden. So sollte der Zielfunktionswert der ursprünglichen Sequenz, der im Rahmen einer Planung, für die deutlich mehr Zeit als für die Einsteuerung zur Verfügung steht, berechnet wurde, nicht durch die nach Sperrungen und Freigaben veränderte Sequenz unterboten werden können.

Um diese Schranke zu verbessern, kann zudem der Bereich einer Sequenz, der fixiert ist, für sich bewertet werden und damit die Schranke eventuell verändert werden. Dies ist z. B. möglich, wenn die erste Freigabe von Aufträgen in einem Szenario erst nach k Takten erfolgt. Gerade in diesen k Takten kann jedoch durch Sperrungen

schon eine große Anzahl an Driftüberschreitungen verursacht worden sein. Deshalb sollte die ursprüngliche Schranke um die Differenz der Zielfunktion bezogen auf die ersten k Takte korrigiert werden. Problematisch dabei ist, dass für die restliche Sequenz andere Fahrzeuge als in der geplanten Sequenz zur Verfügung stehen. Somit wird der restliche Anteil der Schranke ungenau.

Die Schranke könnte zu hoch angesetzt werden, wenn „einfache“ Fahrzeuge gesperrt werden. Dies führt wahrscheinlich zu sehr vielen Driftüberschreitungen im ersten Teil der Sequenz, die auch in die Schranke übernommen werden. Im restlichen Teil, in den diese „einfachen“ Fahrzeuge wieder eingesteuert werden können, kann die Zielfunktion durch Entlastungen dank der wiedereingesteuerten Fahrzeuge gesenkt werden, was zu einem Zielfunktionswert unter der Schranke führen könnte.

Um diese Möglichkeit zu verhindern, können genau die Fahrzeuge ausgewählt werden, die ab dem ersten Freigabezeitpunkt für die Einsteuerung zur Verfügung stehen und diese können ohne Beachtung der Vorrangbedingungen sequenziert werden. Bei guter Lösung für diese Relaxierung wird allerdings in vielen Fällen ein Ergebnis vorliegen, das weit von der optimalen Lösung des eigentlichen Problems entfernt ist.

Dieser Effekt wird dadurch begünstigt, dass bei der Sequenzierung des restlichen Teils der Produktionssequenz das Ende dieser bekannt ist und somit auch ausgenutzt werden kann. So können die Werker am Ende sehr weit in den Driftbereich hereingedriftet sein, ohne dass es Auswirkungen auf die Zielfunktion hat. Die entwickelten Algorithmen jedoch operieren unabhängig von Produktionsintervallen und sind darauf ausgelegt, dass auch folgende Fahrzeuge mit wenigen Überlastungen produziert werden können.

Aus theoretischer Sicht ist der beschriebene Wert problematisch, da er keine garantierte Schranke darstellt. Aus praktischer Sicht stellt er eine Schranke dar, ist aber möglicherweise bedeutungslos, da er vermutlich in vielen Fällen deutlich unter dem tatsächlichen Minimum liegt. Trotz dieser Bedenken und mangels besserer Alternative wird das beschriebene Verfahren dennoch verwendet. Auf diese Weise steht zumindest ein Vergleichswert zur Verfügung, der im Folgenden als untere Schranke bezeichnet wird, auch wenn dies rein mathematisch nicht korrekt ist.

6.5.3 Testergebnisse

Nun werden die Ergebnisse der beschriebenen Tests vorgestellt. Dabei wird zunächst auf das Szenario ohne Gewichtungen der Arbeitsplätze eingegangen, in dem auch eine alternative Zielfunktion getestet wird. Danach werden die Resultate bei besonderer Beachtung von kritischen Arbeitsplätzen erläutert.

6.5.3.1 Ungewichtete Arbeitsplätze

Bevor die unterschiedlichen Algorithmen verglichen werden, wird die Wahl der Zielfunktion bei Verwendung des einfachen DSE Algorithmus analysiert. Dazu werden in Abbildung 6.5 die Testergebnisse der beiden zuvor beschriebenen Zielfunktionen gegenübergestellt.

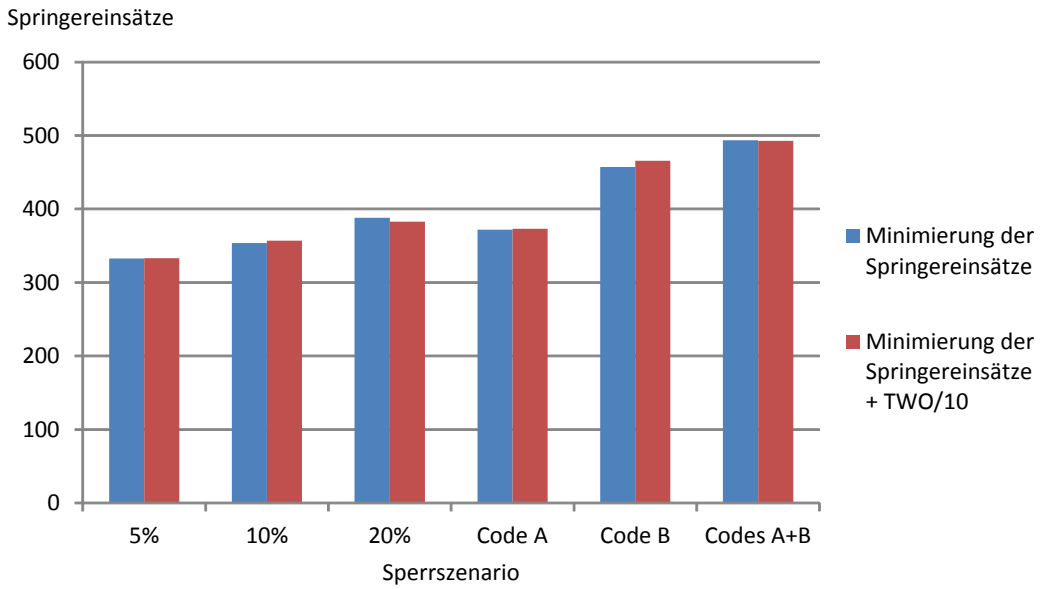


Abbildung 6.5: Auswirkungen der Zielfunktion auf die Anzahl der Springereinsätze

Die Wahl der Zielfunktion hatte keine sehr großen Auswirkungen auf das Ergebnis. Auffallend ist jedoch, dass die Verwendung der TWO lediglich bei den Sperrszenarien mit vielen Sperrungen (20 % oder 2 Codes) besser abschnitt als die Standardzielfunktion. Dies entspricht den Erwartungen, da die TWO langfristig ausgerichtet ist und

erst bei neuen Veränderungen der Sequenz der Vorteil zum Tragen kommt, dass die Sequenz robuster konstruiert wurde. Bei Verwendung der anderen komplexeren Algorithmen verschwand jedoch auch in diesen Szenarien der Nutzen der TWO. Deshalb wird im Folgenden lediglich die Standardzielfunktion verwendet.

Für den Vergleich der verschiedenen Algorithmen werden zunächst die zufälligen Sperrungen betrachtet. Als Vergleichswert dient hier der Zielfunktionswert der Originalsequenz, d. h. der geplanten Sequenz ohne Sperrungen. Mit der im vorherigen Abschnitt beschriebenen Methode konnte eine Sequenz bestimmt werden, die eine untere Schranke für die Zielfunktion impliziert. Jedoch lag bei diesen Tests der zugehörige Zielfunktionswert unter dem der Originalsequenz. Dies war dadurch möglich, dass zur Berechnung der unteren Schranke das betrachtete Intervall als bekannt angenommen wurde.

Bei der ursprünglichen Planung sowie bei den hier vorgestellten Algorithmen sind die Berechnungen jedoch - wie für die Praxis sinnvoll - nicht auf dieses begrenzte Intervall ausgerichtet. Es soll dadurch vermieden werden, in den folgenden Takten eine Häufung von Driftüberschreitungen herbeizuführen, zu Gunsten einer Verbesserung der hier betrachteten Zielfunktion bezogen auf das Untersuchungsintervall. Aufgrund dieser Überlegungen stellt die Originalsequenz bei diesen Sperrszenarien einen geeigneteren Vergleichswert dar.

Neben diesem werden in Abbildung 6.6 die Ergebnisse von den beschriebenen Algorithmen grafisch dargestellt.

Das erste offensichtliche Ergebnis ist, dass mit zunehmender Zahl von Sperrungen die resultierenden Sequenzen eine größere Anzahl an Springereinsätzen benötigten. Außerdem zeigt sich, dass die Algorithmen, die aufeinander aufbauen, entsprechend ihrer Ausbaustufe zu bewerten sind, d. h. die zusätzliche Verwendung von DVTS stellte eine Verbesserung dar und der ergänzende Einsatz der REPLAN Prozedur ebenso.

Bei der Bewertung der Abwartestrategie ist zu unterscheiden, welche Ausbaustufe verwendet wurde. Ein Vergleich der einfachen Methoden WAIT und DSE zeigt, dass die Abwartestrategie deutlich besser war. Dieses Ergebnis blieb bei der zusätzlichen Verwendung von VTS+ erhalten. In den Testszenarien mit 20 % Sperrungen schlug

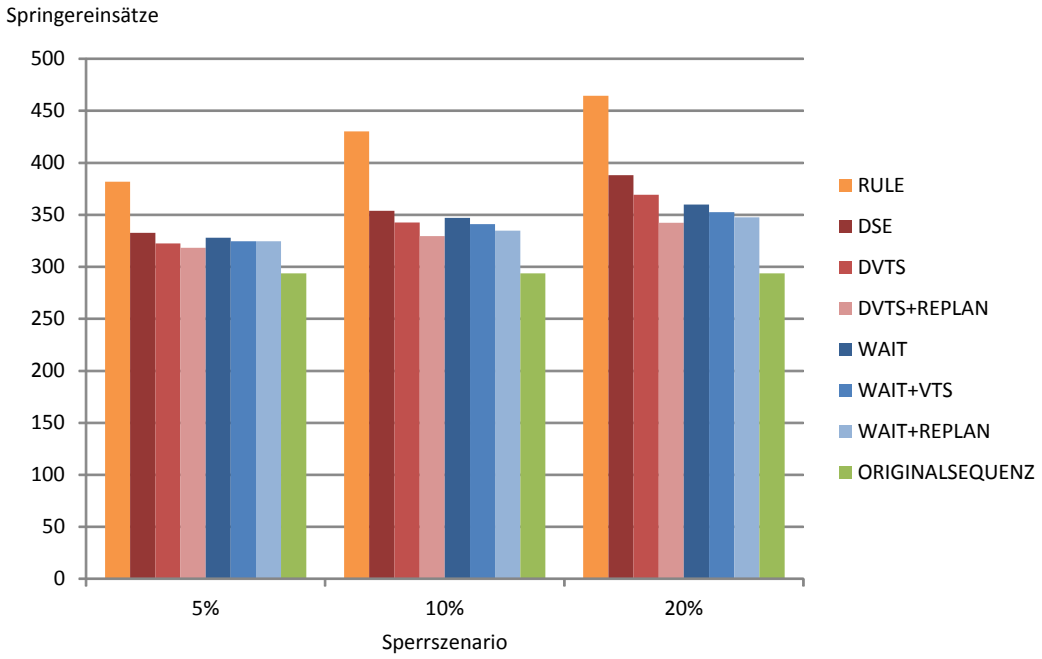


Abbildung 6.6: Vergleich der Algorithmen bei zufälligen Sperrungen

die einfache Abwartestrategie ohne VTS+ sogar die direkt einsteuernde Alternative mit VTS+. Bei der Verwendung der REPLAN Prozedur änderte sich jedoch diese Rangfolge, sodass insgesamt DVTS+REPLAN die besten Ergebnisse erzielte.

Dieser Algorithmus reduzierte die Anzahl der bei diesen Sperrszenarien benötigten Springereinsätze im Durchschnitt um 22 %. Das entspricht einer Verringerung des Abstandes zu den Zielfunktionswerten bei der Originalsequenz um 72 %.

Als nächstes werden die Testergebnisse mit simulierten Codesperrungen präsentiert (Abbildung 6.7). Dabei kann festgehalten werden, dass die Anzahl der benötigten Springereinsätze im Vergleich zu den zufälligen Sperrungen deutlich anstieg. Dieser Anstieg kann nicht durch die Anzahl der Sperrungen begründet werden, da die Sperrung von Code A ca. 5 % der Fahrzeuge entsprach, Code B ca. 11 % und die Kombination beider Codes ca. 15 %. Bei diesen Szenarien kann auch die zuvor beschriebene untere Schranke als Benchmark verwendet werden. Diese lag hier klar über dem Zielfunktionswert der Originalsequenz ohne Sperrungen.

Beim Vergleich der Algorithmen ergibt sich ein ähnliches Bild wie bei den zufälligen

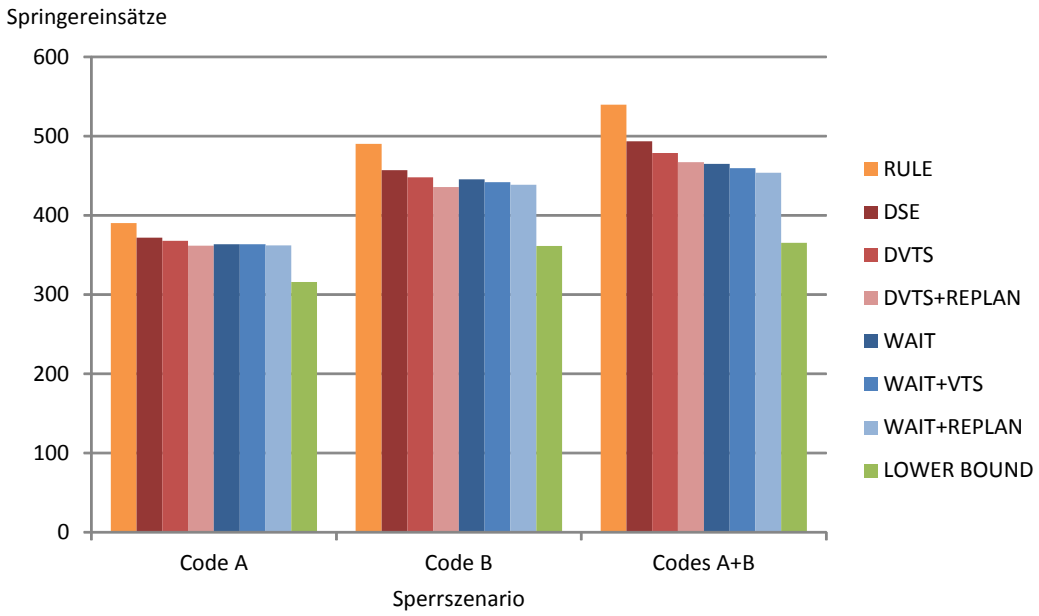


Abbildung 6.7: Vergleich der Algorithmen bei gezielten Codesperrungen

Sperrungen. Lediglich bei den komplexesten Algorithmen, d. h. denen mit REPLAN Prozedur, lag WAIT+REPLAN im Falle einer Sperrung eines einzelnen Codes näher an der DVTS+REPLAN Lösung als zuvor und in den Instanzen mit der Sperrung zweier Code war dieser Algorithmus sogar besser.

Die durchschnittlichen Verbesserungen von WAIT+REPLAN gegenüber RULE lagen nur noch bei 12 % bzw. die Lücke zur unteren Schranke wurde um 44 % verringert.

Als ergänzende Bewertungsgröße der Algorithmen kann zudem die Varianz betrachtet werden. Die Varianz bezieht sich in diesem Zusammenhang auf die Zielfunktionswerte der Ergebnisse von den jeweils zehn Testdurchläufen derselben Probleminstanzen. Während die einfachen Algorithmen (RULE, DSE, WAIT) vollkommen deterministisch ablaufen und somit keine Varianz erzeugen, ist diese bei der Verwendung von DVTS nicht unerheblich. Abbildung 6.8 stellt die durchschnittlichen Standardabweichungen von DVTS und WAIT+VTS dar. Auf eine zusätzliche Betrachtung der REPLAN Algorithmen wird verzichtet, da die Standardabweichungen dieser Algorithmen die gleichen Auffälligkeiten zeigen.

Zunächst ist festzuhalten, dass unabhängig vom verwendeten Algorithmus die Vari-

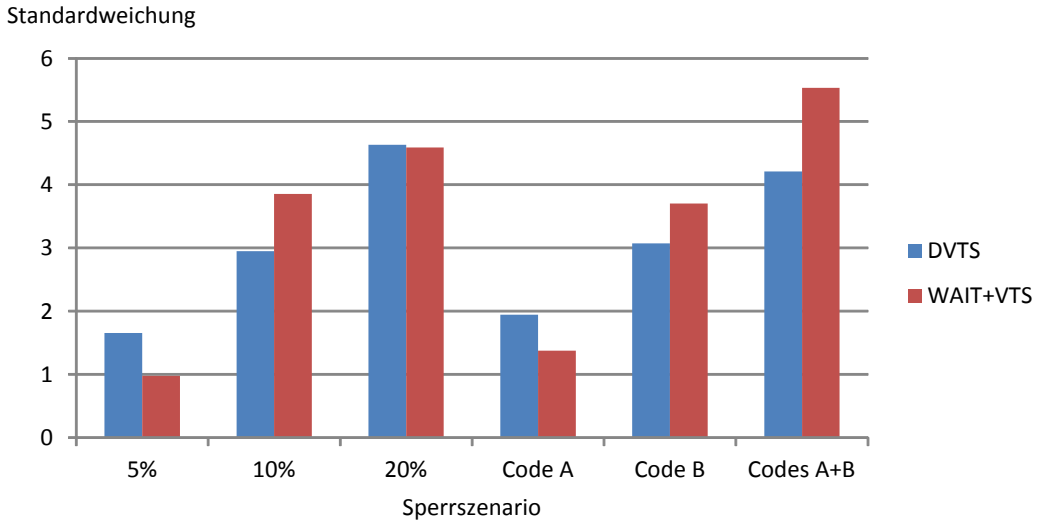


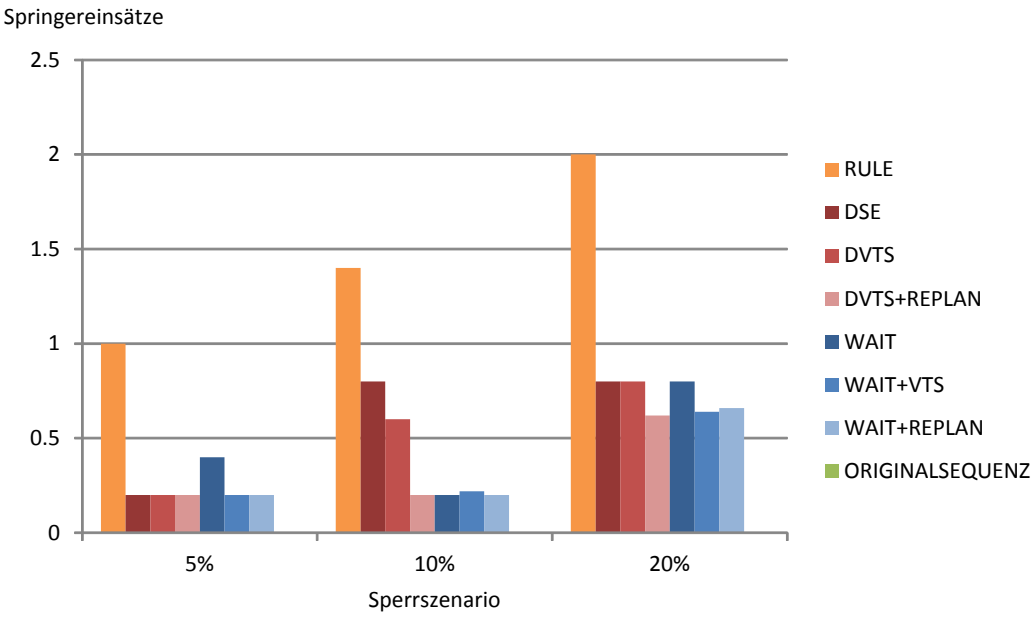
Abbildung 6.8: Durchschnittliche Standardabweichungen der Zielfunktionswerte

anz mit zunehmender Anzahl an Sperrungen anstieg. Die Unterschiede zwischen den Testszenarien waren abhängig vom Algorithmus. Der WAIT+VTS Algorithmus hatte in den Szenarien mit wenigen Sperrungen eine geringere Varianz als DVTS. Allerdings stieg die Varianz mit zunehmender Anzahl an Sperrungen stärker an, sodass bei der Sperrung zweier Codes eine hohe Standardabweichung zu beobachten war.

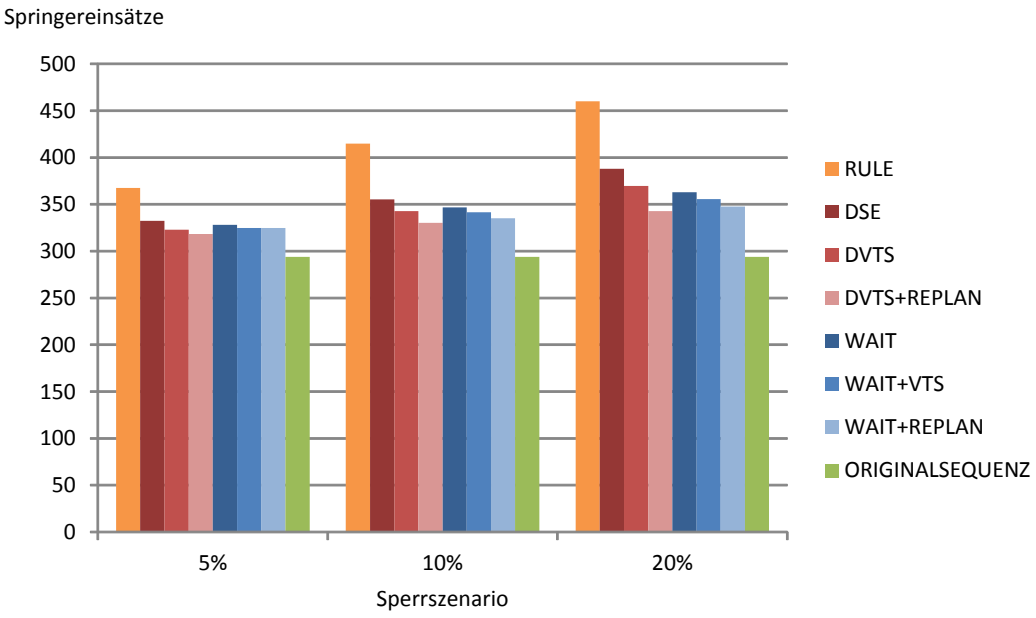
6.5.3.2 Gewichtete Arbeitsplätze

Um zusätzlichen Anforderungen der Praxis gerecht zu werden, wurden die Tests mit einer Änderung in den Einstellungen wiederholt. Es wurde der Gewichtungsfaktor w_l für die Arbeitsplätze eingeführt. Dabei wurde dieser auf zwei Werte $w_l \in \{1, 1000000\}$ eingegrenzt. Ein kleiner Teil der Arbeitsplätze wurde stark gewichtet, sodass eine hierarchische Zielfunktion vorliegt. Zunächst sollen die stark gewichteten Arbeitsplätze minimiert werden und dann untergeordnet die anderen Arbeitsplätze.

Nun werden analog zum vorangegangenen Abschnitt die Testergebnisse für den Fall mit gewichteten Arbeitsplätzen vorgestellt. In Abbildung 6.9 sind die Resultate für zufällige Sperrungen dargestellt.



(a) kritische Arbeitsplätze



(b) alle Arbeitsplätze

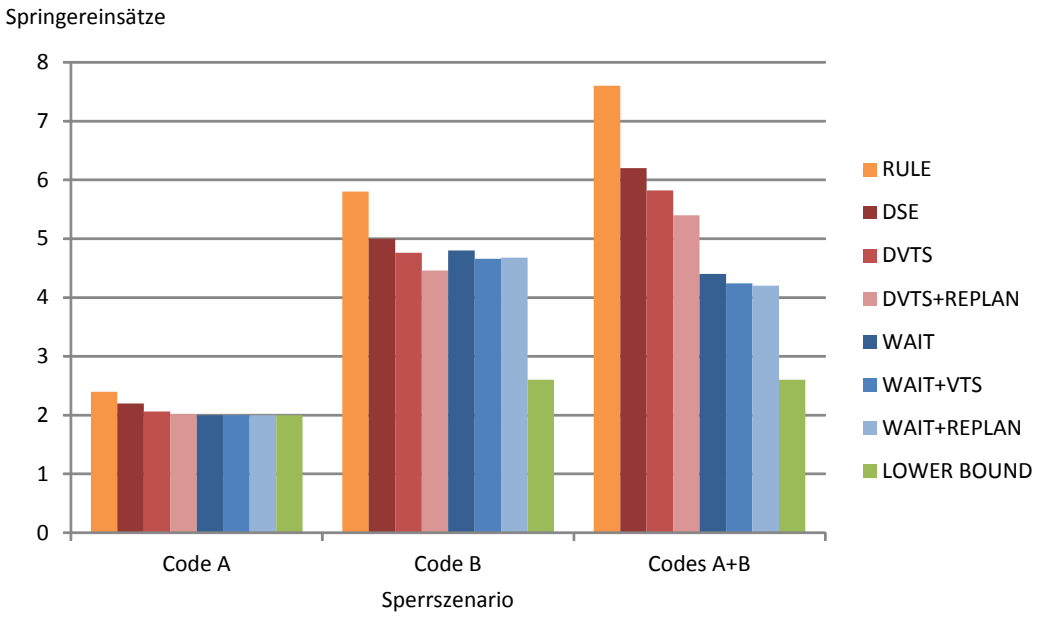
Abbildung 6.9: Vergleich der Algorithmen bei zufälligen Sperrungen

Bezüglich der gesamten Driftüberschreitungen können sämtliche Aussagen von der Version ohne Gewichtungen übernommen werden. Interessanter sind die Ergebnisse bei den kritischen Driftüberschreitungen. Hier zeigten sämtliche vorgestellte Algorithmen eine deutliche Verbesserung gegenüber dem RULE Ansatz. Die Methoden mit REPLAN Prozedur konnten die Anzahl der kritischen Driftüberschreitungen um über 75 % reduzieren. Dabei entspricht diese Verbesserung auch dem Teil der Lücke zur Originalsequenz, der geschlossen wurde, da in dieser keine kritischen Driftüberschreitungen vorlagen.

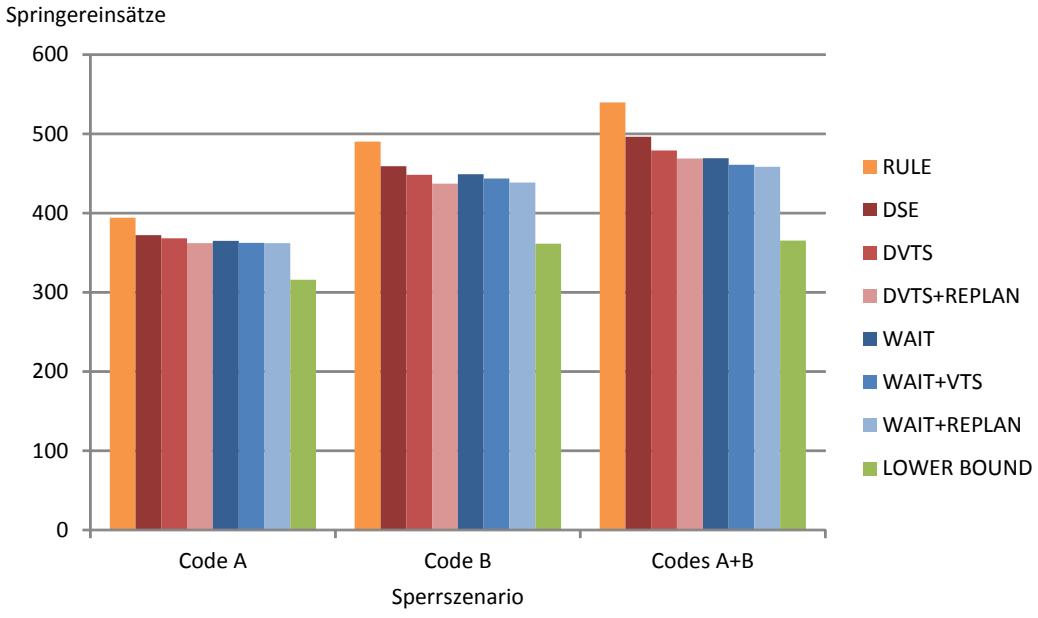
Es bleibt jedoch anzumerken, dass diese relativen Zahlen nicht überbewertet werden dürfen. Im Hinblick auf die absoluten Größen waren die Verbesserungen im kleinen einstelligen Bereich. Die maximale Verbesserung zum RULE Ansatz betrug bei einem Szenario mit 20 % Sperrungen vier Driftüberschreitungen.

Zuletzt werden auch die Ergebnisse der Codesperrungen mit Betrachtung von gewichteten Arbeitsplätzen in Abbildung 6.10 dargestellt.

Erneut wird der Fokus auf die kritischen Driftüberschreitungen gelegt, da bei den gesamten Driftüberschreitungen keine signifikanten Veränderungen im Vergleich zu den Szenarien ohne Gewichtung der Arbeitsplätze vorlagen. Bei den kritischen Driftüberschreitungen ist jedoch eine individuelle Betrachtung der Sperrszenarien nötig. Bei der Sperrung des Codes A waren die Verbesserungen aller Algorithmen relativ gering. Allerdings erreichten die vier besten Algorithmen stets das Niveau der unteren Schranke. Diese wurde bei Sperrung von Code B nicht mehr erreicht, dafür waren die relativen Verbesserungen größer. Die Rangordnung der Algorithmen entsprach dabei der von der Betrachtung aller Driftüberschreitungen. Im Fall mit Sperrungen beider Codes konnten dagegen alle WAIT Algorithmen wesentlich bessere Sequenzen erzeugen als die anderen Algorithmen. Insgesamt ergab sich für den WAIT+REPLAN Algorithmus eine durchschnittliche Verbesserung der RULE Lösungen um 31 %, was einer Verringerung des Abstandes zur unteren Schranke um 57 % entspricht.



(a) kritische Arbeitsplätze



(b) alle Arbeitsplätze

Abbildung 6.10: Vergleich der Algorithmen bei gezielten Codesperrungen

6.5.4 Folgerungen

Zunächst wird geklärt, ob die Integration einer Hilfsgröße in die Zielfunktion eine sinnvolle Maßnahme darstellt. Die Ergebnisse legen nahe, dass dies in den meisten Szenarien nicht der Fall ist. Lediglich unter den Voraussetzungen, dass anspruchsvolle Algorithmen aufgrund von begrenzter Rechenkapazität nicht verwendet werden können und gleichzeitig eine hohe Anzahl an Sperrungen und damit Veränderungen der Plansequenz erwartet wird, kann über eine Ergänzung der Zielfunktion um eine langfristig ausgerichtete Komponente nachgedacht werden. In diesem Fall scheint die Summe der Werkerpositionen eine plausible Orientierungsgröße zu sein.

Generell müssen die Resultate so beurteilt werden, dass ein auf die Anforderungen der Einsteuerung zugeschnittenes Verfahren verwendet werden sollte. Alle in dieser Arbeit vorgestellten Algorithmen bieten ein klares Verbesserungspotential gegenüber dem regelbasierten Ansatz. Eine allgemeine Empfehlung ist die Nutzung des DVTS+REPLAN Algorithmus, der in den meisten Fällen die besten Ergebnisse liefert. Unter bestimmten Bedingungen mit vielen zu erwartenden Sperrungen und bei besonderer Beachtung von kritischen Arbeitsplätzen sollte auch über die Verwendung des WAIT Ansatzes ebenfalls in Verbindung mit der VTS+ Heuristik und einer REPLAN Prozedur nachgedacht werden.

Auch die Analyse der Varianz liefert keine zusätzliche Entscheidungshilfe zur Bewertung der Algorithmen. Die Verwendung der Abwartestrategie führt bei wenigen Sperrungen dazu, dass die Ergebnisse relativ stabil sind. Allerdings führt bei steigender Zahl der Sperrungen häufig eine etwas unterschiedliche Zwischenlösung nach einem Takt dazu, dass in den Folgetakten unterschiedliche Fahrzeuge in die Plansequenz integriert werden, sodass die Varianz im Vergleich zum DVTS Algorithmus höher ist.

Unter der Annahme, dass die verwendeten Testszenarien repräsentativ für die Realität sind, kann eine Reduzierung der benötigten Springereinsätze um über 15 % erwartet werden. In Anbetracht der gegebenen Originalsequenz bzw. der berechneten unteren Schranke bedeutet dies, dass das Optimierungspotential um über 50 % ausgenutzt wird. Dabei ist zu bemerken, dass die Schranke sehr pessimistisch berechnet wurde und in den Szenarien mit zufälligen Sperrungen eine Sequenz auf Niveau der Origi-

nalsequenz sehr unrealistisch ist. Dies legt die Vermutung nahe, dass die vorgestellten Algorithmen weit mehr als nur die Hälfte des vorhandenen Potentials erreichen können. Allerdings ist eine fundierte Aussage hierzu aufgrund fehlender enger Schranken nicht möglich.

Bei Konzentration auf wenige wichtige Arbeitsplätze können noch größere Verbesserungen erzielt werden. Die Anzahl der kritischen Springereinsätze ließ sich im Durchschnitt um über 50 % reduzieren. Dies ist gleichbedeutend mit einer Ausnutzung des abgeschätzten Verbesserungspotentials um über 70 %. Eine wichtige Ergänzung hierzu ist, dass auch bei dieser Gewichtung die normal gewichteten Arbeitsplätze ähnlich optimiert werden wie zuvor. Durch die Fokussierung auf wenige kritische Arbeitsplätze ergeben sich keine signifikanten negativen Auswirkungen auf die anderen Arbeitsplätze.

Zuletzt muss jedoch angemerkt werden, dass sämtliche vorgestellten Ansätze nicht die exakte Anzahl an Springereinsätzen in einer realen Schicht bestimmen können. Gesucht werden stets Richtwerte, die die geschätzte Anzahl der Springereinsätze beschreiben. Die Werker sind Menschen, die die vorgegebenen Zeiten nur ungefähr einhalten können. Allerdings legen praktische Untersuchungen nahe, dass die berechneten Springereinsätze eine sehr gute Abschätzung für die Überlastungsgefahr einer Station in der Realität widerspiegeln, was die Verwendung dieses Ansatzes in der Praxis rechtfertigt.

7 Betrachtung der Springeranzahl

In diesem Kapitel wird untersucht, wie die in einer Schicht benötigte Anzahl an Springern minimiert werden kann. Dazu werden zunächst Grundlagen erläutert, die dabei beachtet werden müssen. Danach wird das im vorangegangenen Kapitel entwickelte Modell (6.8) erweitert. Es wird ein anwendungsorientierter Ansatz vorgestellt, in dem der Fokus auf die Anzahl der Springer gerichtet wird. Abschließend wird dieser Ansatz getestet und die Ergebnisse werden analysiert.

7.1 Grundlagen

Die Berechnung der Anzahl der benötigten Springer ist komplexer als die der zuvor betrachteten Zielfunktionen. Es kann nicht ohne zusätzliche Informationen von den berechneten Springereinsätzen auf die benötigte Anzahl an Springern geschlossen werden. Wie schon bei der Beschreibung der Problemstellung erläutert wurde, organisieren sich die Springer in Gruppen. Jede Gruppe von Springern kann an einer bestimmten Menge von Arbeitsplätzen aushelfen. Die Informationen darüber, wie diese Aufteilung aussieht, werden in diesem Kapitel benötigt. Die relevante Zielgröße lässt sich als Summe der Anzahlen der Springer über alle diese Gruppen ermitteln.

Für eine Gruppe entspricht die Anzahl der benötigten Springer der maximalen Anzahl an gleichzeitigen Springereinsätzen. Da folglich die zeitliche Verteilung der Springereinsätze relevant ist, müssen auch die räumlichen - und damit ermittelbaren zeitlichen - Abstände zwischen den einzelnen Arbeitsplätzen herangezogen werden. Nur so kann in Abhängigkeit von der Position innerhalb der Produktionssequenz der genaue Zeitpunkt bestimmt werden, zu dem ein Springer eingesetzt werden muss.

Ein weiterer Aspekt ist, dass für eine praktische Anwendung die Aufteilung des Produktionsablaufes in einzelne Schichten essentiell ist. Bereits im letzten Kapitel wurden die Tests auf Zeiträume, die etwa eine Schicht umfassen, begrenzt. Allerdings darf dies nicht als Optimierung einer Montageschicht missverstanden werden. Abbildung 7.1 verdeutlicht, welche Arbeitsvorgänge betrachtet werden, wenn die Einsteuerung über eine Schicht optimiert wird. Der zeitliche Verlauf zwischen den Arbeitsplätzen, der schematisch als linear dargestellt ist, kann in der Realität äußerst unregelmäßig sein. Es gibt Arbeitsplätze, die sich an der gleichen Station am Band befinden und somit gleichzeitige Arbeit erfordern. Aber es existieren auch längere Lücken zwischen Arbeitsplätzen, insbesondere beim Wechsel zwischen einzelnen Montagebändern.

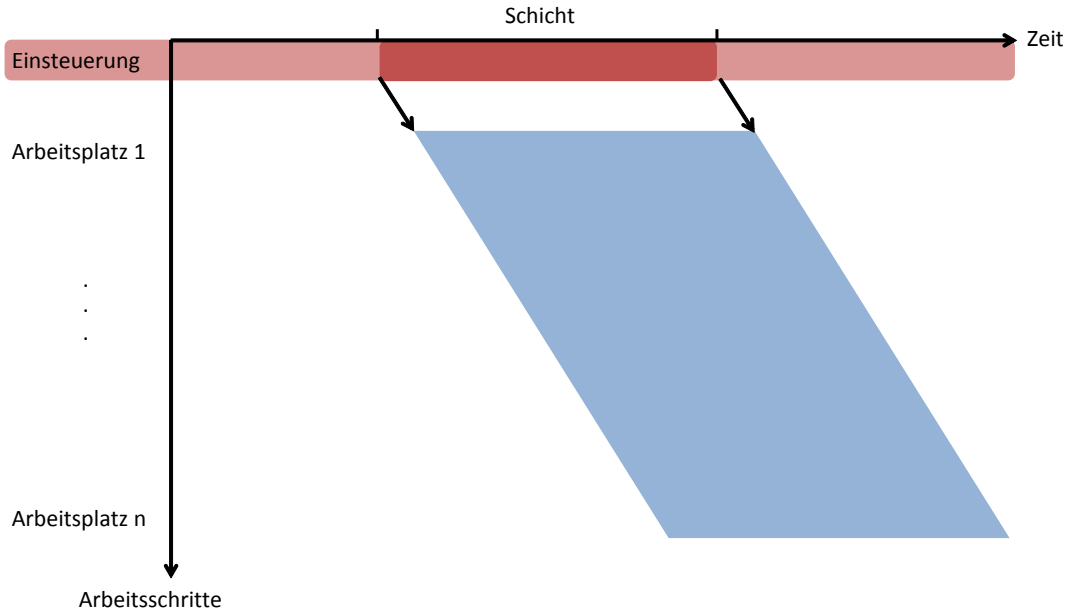


Abbildung 7.1: Schematische Darstellung des Montageintervalls von in einer Schicht eingesteuerten Fahrzeugen

Ein großer Teil der an den eingesteuerten Fahrzeugen durchgeführten Arbeitsvorgänge findet erst in späteren Schichten statt. Die Aufträge, die erst spät in einer Schicht eingesteuert werden, werden komplett in späteren Schichten montiert. Im letzten Kapitel wurde dieser Aspekt bereits erwähnt und verdeutlicht, dass dies keinen Einfluss auf die Aussagekraft der Ergebnisse hatte.

In diesem Kapitel ist es jedoch entscheidend, wann welche Springereinsätze nötig werden. Insbesondere sollen alle in einer Schicht durchgeführten Montageschritte erfasst

werden. Deshalb muss das bezogen auf die Einsteuerung relevante Produktionsintervall genau bestimmt werden. Wie in Abbildung 7.2 deutlich wird, muss zur Bewertung der Anzahl der in einer Schicht benötigten Springer eine Sequenz in der Einsteuerung betrachtet werden, die deutlich länger als eine Schicht ist und weit vor Beginn der analysierten Schicht beginnt.

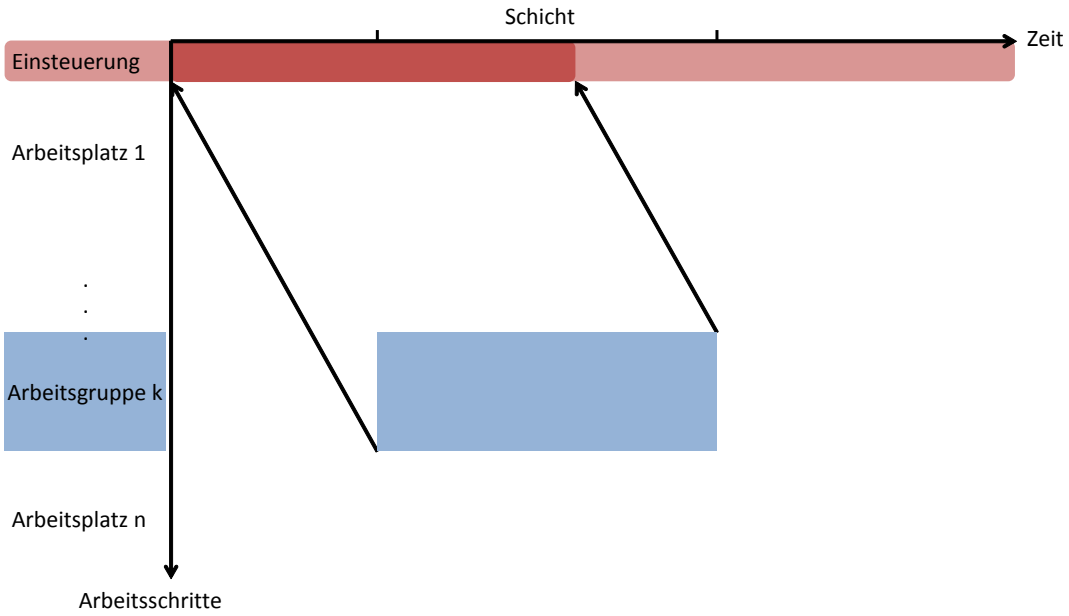


Abbildung 7.2: Schematische Darstellung des Einflussbereichs der Einsteuerung auf eine Springergruppe

In dieser Grafik wird eine beispielhafte Menge von Arbeitsplätzen, die von einer Gruppe von Springern unterstützt wird, und deren Arbeiten in einer Schicht dargestellt. Jede Gruppe mit mindestens zwei Arbeitsplätzen, an denen nicht zeitgleich am gleichen Auftrag gearbeitet wird, bearbeitet im Verlauf einer Schicht mehr Fahrzeuge, als während einer Schicht eingesteuert werden. Darüber hinaus wurde ein Großteil, der für diese Gruppe relevanten Fahrzeuge, bereits im letzten Takt oder sogar noch früher eingesteuert.

Damit zusammenhängend ist ebenfalls die Tatsache zu berücksichtigen, dass jeder Auftrag über mehrere Schichten montiert wird. Folglich müssen bei der Einsteuerung stets der Einfluss auf mindestens zwei Schichten und damit unterschiedliche Anteile an der Zielfunktion beachtet werden.

7.2 Modellierung

In diesem Abschnitt werden Modelle entwickelt, in denen die Anzahl der verwendeten oder verfügbaren Springer als zentrale Größe enthalten ist.

7.2.1 Springeranzahl in der Zielfunktion

Zunächst wird die Anzahl der benötigten Springer direkt als Element in die Zielfunktion übernommen. Um diesen Wert berechnen zu können, sind einige Änderungen im Vergleich zum Modell (6.8) aus dem vorangegangenen Kapitel notwendig.

Der Produktionszeitraum muss in Intervalle unterteilt werden. Entsprechend der Realität werden einzelne Schichten betrachtet, die den Zeitraum beschreiben, während dessen Werker arbeiten können. Die Anzahl der Springer muss nun für jede einzelne Schicht individuell berechnet werden. Diese Schichten werden benannt durch:

$$\sigma_1, \sigma_2, \sigma_3, ... \in \Sigma$$

Bei den in dieser Arbeit verwendeten Testdaten benötigt ein Auftrag maximal die Zeit zweier Schichten, um alle Arbeitsplätze der zugrunde liegenden Montagelinie zu durchlaufen. Daraus folgt, dass bei der Einsteuerung eines Fahrzeugs maximal drei Schichten berücksichtigt werden müssen, um den Weg dieses Fahrzeugs durch die komplette Montage zu erfassen. In den folgenden Tests wird die Sequenz auch in drei Abschnitte eingeteilt, von denen jedoch nur der mittlere einer tatsächlichen Schicht entspricht.

Für die Zuordnung von den einzelnen Arbeiten zu einer Schicht muss der zeitliche Verlauf genau beachtet werden. Nur so kann das exakte zeitliche Auftreten von Drifterschreitungen und die Erkennung von gleichzeitigen Springereinsätzen, die die Zahl der Springer bestimmen, ermittelt werden. Dazu wird eine Verzögerungskonstante $\tau_l,\ l \in A$ eingeführt, die angibt, wie lange es ab dem Zeitpunkt der Einsteuerung eines Fahrzeugs dauert, bis es den jeweiligen Arbeitsbereich erreicht.

Außerdem muss die Organisation der Springer modelliert werden. Diese umfasst die Einteilung der Arbeitsplätze in Gruppen und die Zuteilung der Springer zu einer be-

stimmten Gruppe, innerhalb der sie an jedem Arbeitsplatz einspringen können. Die Menge der Gruppen wird mit Λ bezeichnet, sodass die Arbeitsplätze wie folgt partitioniert werden können:

$$A = \cup_{\lambda \in \Lambda} A_\lambda$$

Für die Springereinsätze wird die vereinfachende Annahme getroffen, dass Springer immer am Anfang einer jeweiligen Station mit der Arbeit beginnen. Dies ermöglicht eine diskrete Betrachtung der Bearbeitungszeiten, indem diese auf die benötigte Anzahl an Takten aufgerundet werden. Die Aufrundung verursacht keine Einschränkungen, da ein weiterer Einsatz des Springers erst zu Beginn des nächsten Takts erfolgen würde.

Gleichzeitig wird der zeitliche Aufwand des Bewegens von einem Arbeitsplatz zu einem anderen vernachlässigt. Springer können sich also zwischen ihren Einsätzen mit unendlicher Geschwindigkeit bewegen. Bei zwei direkt aufeinanderfolgenden Springereinsätzen kann diese Annahme problematisch werden. Da jedoch vorher zur vollen Taktzeit aufgerundet wurde, steht im Durchschnitt dennoch die halbe Taktzeit zur Verfügung.

Ohne diese beiden Annahmen müssten zum einen zusätzliche Informationen über die Wegezeiten bekannt sein, die von den Konstanten τ_l eventuell abgeleitet werden könnten. Andererseits würde ein zusätzliches Schedulingproblem für jede mögliche Produktionssequenz entstehen. Die Springer müssten auf die Springereinsätze verteilt werden, wobei für jeden Einsatz ein Intervall zur Verfügung stünde, innerhalb dessen der Einsatz angeordnet werden müsste. Diese zusätzliche Komplexität soll vermieden werden, da jede Erhöhung der Rechenzeit bei der Bestimmung des Zielfunktionswerts für eine gegebene Sequenz die Anwendbarkeit der entwickelten Verfahren verringert. Außerdem ist eine sekundengenaue Zuordnung von Springern zu ihren Einsätzen in der Praxis schwer umsetzbar. Daher erscheinen die getroffenen Annahmen sinnvoll.

In einem ersten Modell wird das neue Ziel der Reduktion der Anzahl der benötigten Springer direkt in die Zielfunktion integriert. Dazu wird folgende Entscheidungsvariable verwendet:

$u_{\sigma\lambda}$: Anzahl der benötigten Springer in Schicht σ in Arbeitsgruppe λ

Der Gewichtungsfaktor w_j wird nun angepasst, sodass zum einen die Schichten gewichtet werden können und zum anderen alle Arbeitsplätze in einer Gruppe denselben Faktor erhalten. Er ist also nicht mehr abhängig vom Arbeitsplatz, sondern von der Gruppe λ von Arbeitsplätzen, an denen Springer aushelfen können, und der Schicht σ. Dazu ändert sich folglich die Indizierung zu $w_{\sigma\lambda}$. Als Spezialfall könnte auch eine Gewichtung w_λ gewählt werden, wenn zwischen den Schichten nicht unterschieden werden soll.

Zuletzt wird z als Zählvariable verwendet, die in die Vergangenheit zählt. Damit kann überprüft werden, ob an früher eingesteuerten Fahrzeugen Springereinsätze nötig sind, deren Bearbeitung mehrere Takte dauert. Die längste Bearbeitungszeit eines Auftrags, der ausgeführt werden muss, bestimmt dabei z_{max}, die maximale Anzahl der Takte, die ein Arbeitseinsatz dauern kann. Entsprechend viele Aufträge müssen auf einen möglichen Springereinsatz bis in den aktuellen Takt hinein überprüft werden.

$$\min \sum_{\sigma\in\Sigma}\sum_{\lambda\in\Lambda} w_{\sigma\lambda}\cdot u_{\sigma\lambda} \tag{7.1a}$$

$$\text{s. t.} \sum_{j\in J} x_{ij} = 1 \qquad \forall i \in I \tag{7.1b}$$

$$\sum_{i\in I} x_{ij} = 1 \qquad \forall j \in J \tag{7.1c}$$

$$t_{jl} = \sum_{i\in I} x_{ij}\cdot b_{il} \qquad \forall j\in J,\ l\in A \tag{7.1d}$$

$$s_{jl} = \begin{cases} 1, \text{ wenn } p_{j-nw_l,l} - nw_l\cdot c + t_{jl} > d_l \\ 0, \text{ sonst} \end{cases} \qquad \forall j\in J,\ l\in A \tag{7.1e}$$

$$p_{jl} = \max\left(p_{j-nw_l,l} - nw_l\cdot c + (1-s_{jl})\cdot t_{jl}, nw_l\cdot c\right) \qquad \forall j\in J,\ l\in A \tag{7.1f}$$

$$x_{ij} \le \beta_{ij} \qquad \forall i\in I,\ j\in J \tag{7.1g}$$

$$\alpha_{ij} = \begin{cases} 1, \text{ wenn } \exists k < i : \sum_{t=1}^{j} x_{kt} + (1-\alpha_{kt}) = 0 \\ \beta_{ij}, \text{ sonst} \end{cases} \qquad \forall i\in I,\ j\in J \tag{7.1h}$$

$$\sum_{j=1}^{t} x_{ij} - x_{i'j} \geq 0 \tag{7.1i}$$

$$\forall i, i' \in I \text{ mit } i < i' \text{ und} \sum_{j=1}^{t} 2 - \alpha_{ij} - \alpha_{i'j} = 0, t \in J$$

$$a_{it} = \begin{cases} 0, \text{ wenn } \sum_{j=1}^{t} \alpha_{ij} = t \\ 1, \text{ sonst} \end{cases} \qquad \forall i \in I,\ t \in J \tag{7.1j}$$

$$A_t = \sum_{i \in I} \max\left(a_{it} - \sum_{j=1}^{t-1} x_{ij}, 0\right) \qquad \forall t \in J \tag{7.1k}$$

$$\sum_{j=1}^{t+A_t-1+k} x_{ij} \geq \sum_{s=t}^{t+A_t-1+k} \alpha_{is} - A_t - k + 1 \tag{7.1l}$$

$$\forall i \in I, t = due_i, \ldots, n \text{ und } A_t \geq 1$$

$$\sum_{l \in A_\lambda} \sum_{z=0}^{z_{max}} s_{(j+\tau_l - z),l} \cdot \mathbb{1}_{\{t_{j+\tau_l - z}, l/c > z\}} \leq u_{\sigma\lambda} \qquad \forall \lambda \in \Lambda, \sigma \in \Sigma, j \in \sigma \tag{7.1m}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{7.1n}$$

Die Zielfunktion (7.1a) enthält nun die gewichtete Anzahl der benötigten Springer, summiert über alle Gruppen und betrachteten Schichten und soll minimiert werden. Als neue Nebenbedingung ergänzt (7.1m) das Modell. Dabei wird die Indikatorfunktion $\mathbb{1}_{\{.\}}$ verwendet, die wahre Ereignisse auf 1 und falsche Ereignisse auf 0 abbildet. In diesen Ungleichungen werden für jede Gruppe von Springern und jeden Takt die Springereinsätze gezählt. Die Anzahl der Springer muss mindestens so groß sein wie die Anzahl der Springereinsätze in jedem Takt.

In diesem Modell scheint auf den ersten Blick der Kostentreiber direkt minimiert zu werden. Die Lohnkosten, die von der Anzahl der benötigten Werker abhängen, können bei einer Verwendung dieser Zielfunktion reduziert werden. Die einfache Optimierung in diese Richtung ignoriert jedoch wesentliche Aspekte der Problemstruktur. Ein entscheidender Aspekt ist, dass der Zielfunktionswert letztlich nur von einer kleinen Anzahl an Aufträgen abhängt. Jede Arbeitsgruppe hat im Verlauf einer Schicht (mindestens) einen Moment, in dem die benötigte Anzahl an Springern gleichzeitig eingesetzt wird. Dies stellt die Belastungsspitze für die Gruppen dar. Dieser Wert sagt jedoch nichts über die weitere Verteilung der Springereinsätze aus.

Der Zeitpunkt des Auftretens dieser Belastungsspitzen beeinflusst die weitere Optimierung enorm. Eine Belastungsspitze zu Beginn einer Schicht stellt kein allzu großes Problem dar. Für den Rest der Schicht können die Fahrzeuge dann relativ flexibel angeordnet werden, ohne dass dies die Zielfunktion beeinflussen kann. Ein Nachteil dabei ist, dass die vorhandenen Springer auf diese Weise wahrscheinlich häufiger als nötig eingesetzt werden, da ihre Anwesenheit notwendig ist und ihr Einsatz nicht zusätzlich bewertet wird. An dieser Stelle wäre es jedoch aus praktischer Perspektive sinnvoll, weitere Einsätze eines Springers zu vermeiden, sodass dieser sich in einer entspannten Produktionsphase komplett auf eine andere Tätigkeit konzentrieren kann und dabei nicht von Unterstützungsgesuchen unterbrochen wird.

Die Konstruktion einer sinnvollen Produktionssequenz wird vor allem dann schwierig, wenn im Laufe einer Schicht immer höhere Belastungsspitzen auftreten. Dies kann dadurch passieren, dass der Einsatz eines zusätzlichen Springers aufgrund der Zielfunktion so lange wie möglich herausgezögert wird. Es wird dabei versucht mit wenigen Springern alle Überlastungen abzufangen. Dennoch wird am Ende ein weiterer Springer notwendig, der bei früheren Einsätzen in der Schicht schon eine bessere Verteilung ermöglicht hätte.

7.2.2 Springeranzahl in den Nebenbedingungen

Aus einer praktischen Perspektive betrachtet, besitzt das beschriebene Modell einen wesentlichen Nachteil. Die verfügbare Anzahl der Springer muss in der Realität schon vor jeder Schicht bekannt sein. Deshalb ist die direkte Verbindung zwischen der Minimierung der Anzahl der benötigten Werker zu der Reduzierung der Kosten nicht gegeben. Vielmehr sollte das Ziel sein, eine Produktionssequenz zu erzeugen, die optimal an die gegebenen Ressourcen angepasst ist.

Zusätzliche praxisrelevante Aspekte wie eine systematische Einteilung von mehr Werkern als nötig, um z. B. Krankheitsfälle ausgleichen zu können, werden an dieser Stelle ignoriert. Die genaue Personalorganisation liegt außerhalb des Rahmens dieser Arbeit. Es wird außerdem angenommen, dass die Vorlaufzeit der Personaleinteilung größer ist, als die Zeit, die zwischen Auftragseinsteuerung und letzter Montagestation liegt. Also

ist die verfügbare Anzahl der Springer für alle relevanten Schichten zum Zeitpunkt der Einsteuerung gegeben.

Aus den beschriebenen Gründen wird in dieser Arbeit, die sich auf die Einsteuerung konzentriert, ein alternativer Ansatz gewählt. Die Anzahl der Springer soll nicht direkt minimiert werden, sondern geht als begrenzender Wert in Form einer Nebenbedingung in das Modell ein. Diese Variante entspricht genau der praktischen Realisierbarkeit der Optimierung. Die zur Verfügung stehenden Springer sind zu Beginn einer Schicht bekannt und sollen möglichst selten eingesetzt werden.

Im Gegensatz zum ersten Modell (7.1) werden die variablen Anzahlen von Springern $u_{\sigma\lambda}$ durch Konstanten $cap_{\sigma\lambda}$, die vorab festgelegt werden müssen, ersetzt. Dadurch verändert sich die entscheidende Nebenbedingung.

$$\sum_{l \in A_\lambda} \sum_{z=0}^{z_{max}} s_{(j+\tau_l-z),l} \cdot \mathbb{1}_{\{t_{j+\tau_l-z,l/c>z}\}} \leq cap_{\sigma\lambda} \qquad \forall \lambda \in \Lambda, \sigma \in \Sigma, j \in \sigma \quad (7.2\text{m})$$

Sämtliche anderen Nebenbedingungen bleiben unverändert im Modell. Als Zielfunktion kann bei dieser Variante wieder die gewichtete Anzahl der Springereinsätze wie im vorherigen Kapitel verwendet werden. Die Zielfunktion für das beschriebene Modell ist formal:

$$\min \sum_{\sigma \in \Sigma} \sum_{j \in \sigma} \sum_{l \in A} w_l \cdot s_{jl} \qquad (7.2\text{a})$$

Dieses Modell (7.2) besitzt ein großes Problem. Wenn die verfügbare Anzahl an Springern zu gering ist, kann eine unzulässige Probleminstanz entstehen. Um die Existenz einer Lösung zu garantieren, ist daher eine weitere Änderung am Modell notwendig. Dazu werden nun zwei Möglichkeiten vorgestellt, die beide in verschiedenen Bereichen des im nächsten Abschnitt beschriebenen Lösungsansatzes ihre Anwendung finden.

Die zunächst eingeführten Relaxierungen garantieren die Zulässigkeit zwar nicht, reichen jedoch für alle praxisrelevanten Probleminstanzen mit sinnvollen Randbedingungen aus. Die Nebenbedingung (7.1i) muss im folgenden Modell nicht mehr beachtet werden. Allerdings wird jedes Vertauschen zweier nicht zurückgestellter Aufträge i und i' mit einem Strafterm $F_{ii'} \in \{0, 1\}$ in der Zielfunktion bewertet. Dieser wird

in der Zielfunktion zusätzlich mit einem ausreichend großen w_F gewichtet, sodass die Einhaltung der Reihenfolgebedingungen wichtiger ist, als die Minimierung der ursprünglichen Zielfunktion.

$$\sum_{j=1}^{t} x_{ij} - x_{i'j} \geq -F_{ii'} \tag{7.3i}$$

$$\forall i, i' \in I \text{ mit } i < i' \text{ und} \sum_{j=1}^{t} 2 - \alpha_{ij} - \alpha_{i'j} = 0, t \in J$$

Außerdem dürfen die Zulässigkeitstermine überschritten werden. Dazu wird ein Verspätungsindikator $V_i \in \{0, 1\}$ eingeführt, der die Aufträge markiert, die ihren Termin nach Bedingung (7.1l) nicht einhalten können. Dieser kann ebenfalls mit einem Gewichtungsfaktor w_V in die Zielfunktion eingebunden werden.

$$\sum_{j=1}^{t+A_t-1+k} x_{ij} + V_i \geq \sum_{s=t}^{t+A_t-1+k} \alpha_{is} - A_t - k + 1 \tag{7.3l}$$

$$\forall i \in I, t = due_i, \ldots, n \text{ und } A_t \geq 1$$

Die Nebenbedingung zur Begrenzung der Springeranzahl wird aus dem zweiten Modell übernommen (7.2m), alle anderen Bedingungen bleiben wie im ersten Modell (7.1) erhalten. Als angepasste Zielfunktion wird in diesem dritten Modell die folgende verwendet:

$$\min \sum_{\sigma \in \Sigma} \sum_{j \in \sigma} \sum_{l \in A} w_l \cdot s_{jl} + w_F \cdot \sum_{i \in I} \sum_{i' \in I : i' > i} F_{ii'} + w_V \cdot \sum_{i \in I} V_i \tag{7.3a}$$

Eine alternative Relaxierung führt wieder näher an das Ausgangsmodell (7.1) heran. Allerdings wird nun nicht versucht die Anzahl der Springer zu minimieren, sondern die Anzahl der fehlenden Springer. Dadurch, dass aber grundsätzlich wieder zusätzliche Springer erlaubt werden, existieren wieder Lösungen für alle Probleminstanzen.

Im folgenden Modell bezeichnet $u_{\sigma\lambda}$ die Springer, die zusätzlich zu denen, die durch die Kapazität $cap_{\sigma\lambda}$ schon gegeben sind, benötigt werden. Diese werden mit einem großen Faktor w_u gewichtet, sodass der Einsatz von eigentlich nicht verfügbaren Springern

das wichtigste Element der Zielfunktion wird.

$$\min \sum_{\sigma\in\Sigma}\sum_{j\in\sigma}\sum_{l\in A} w_l \cdot s_{jl} + \sum_{\sigma\in\Sigma}\sum_{\lambda\in\Lambda} w_u \cdot u_{\sigma\lambda} \tag{7.4a}$$

$$\text{s. t.} \sum_{j\in J} x_{ij} = 1 \qquad \forall i \in I \tag{7.4b}$$

$$\sum_{i\in I} x_{ij} = 1 \qquad \forall j \in J \tag{7.4c}$$

$$t_{jl} = \sum_{i\in I} x_{ij} \cdot b_{il} \qquad \forall j \in J,\ l \in A \tag{7.4d}$$

$$s_{jl} = \begin{cases} 1, \text{ wenn } p_{j-nw_l,l} - nw_l \cdot c + t_{jl} > d_l \\ 0, \text{ sonst} \end{cases} \qquad \forall j \in J,\ l \in A \tag{7.4e}$$

$$p_{jl} = \max\left(p_{j-nw_l,l} - nw_l \cdot c + (1 - s_{jl}) \cdot t_{jl}, nw_l \cdot c\right) \qquad \forall j \in J,\ l \in A \tag{7.4f}$$

$$x_{ij} \leq \beta_{ij} \qquad \forall i \in I,\ j \in J \tag{7.4g}$$

$$\alpha_{ij} = \begin{cases} 1, \text{ wenn } \exists k < i : \sum_{t=1}^{j} x_{kt} + (1 - \alpha_{kt}) = 0 \\ \beta_{ij}, \text{ sonst} \end{cases} \qquad \forall i \in I,\ j \in J \tag{7.4h}$$

$$\sum_{j=1}^{t} x_{ij} - x_{i'j} \geq 0 \tag{7.4i}$$

$$\forall i, i' \in I \text{ mit } i < i' \text{ und } \textstyle\sum_{j=1}^{t} 2 - \alpha_{ij} - \alpha_{i'j} = 0, t \in J$$

$$a_{it} = \begin{cases} 0, \text{ wenn } \sum_{j=1}^{t} \alpha_{ij} = t \\ 1, \text{ sonst} \end{cases} \qquad \forall i \in I,\ t \in J \tag{7.4j}$$

$$A_t = \sum_{i\in I} \max\left(a_{it} - \sum_{j=1}^{t-1} x_{ij}, 0\right) \qquad \forall t \in J \tag{7.4k}$$

$$\sum_{j=1}^{t+A_t-1+k} x_{ij} \geq \sum_{s=t}^{t+A_t-1+k} \alpha_{is} - A_t - k + 1 \tag{7.4l}$$

$$\forall i \in I, t = due_i, \ldots, n \text{ und } A_t \geq 1$$

$$\sum_{l\in A_\lambda} \sum_{z=0}^{z_{max}} s_{(j+\tau_l-z),l} \cdot \mathbb{1}_{\left\{t_{j+\tau_l-z,l/c>z}\right\}} \leq cap_{\sigma\lambda} + u_{\sigma\lambda} \qquad \forall \lambda \in \Lambda, \sigma \in \Sigma, j \in \sigma \tag{7.4m}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in I, j \in J \tag{7.4n}$$

Dieses Modell wird im Kern des Optimierungsansatzes verwendet. Der wesentliche Unterschied zu den vorherigen Modellen liegt in der Nebendingung (7.4m), bei der nun sowohl die verfügbaren als auch die zusätzlichen Springer beachtet werden. Da in der tatsächlichen Sequenz allerdings keine Springer verwendet werden sollen, die über die gegebene Kapazität hinausgehen, wird in der Folge auch auf das vorherige Modell (7.3) zurückgegriffen.

7.3 Anwendungsansatz

Wie die Erläuterungen im vorangegangenen Abschnitt nahe legen, sollte nicht die Minimierung der Anzahl der Springer als direktes Optimierungsziel gewählt werden. Es wird im Folgenden davon ausgegangen, dass die Anzahlen der verfügbaren Springer für die jeweiligen Gruppen und Schichten als Eingangsgröße gegeben sind. Getestet werden soll nun in erster Linie, ob alle vorgegebenen Bedingungen überhaupt eingehalten werden können.

In der Anwendung muss eine Lösung produziert werden, die gegebenenfalls auch gegen Bedingungen verstoßen darf, wenn es anders nicht möglich ist. Deshalb muss geprüft werden, welche Bedingung verletzt werden kann, ohne einen Bandstillstand zu verursachen.

Die Verwendung von nicht bereitgestellten Springern wird in temporären Lösungen zugelassen, damit die Heuristiken die Möglichkeit haben durch Verbesserungen eine zulässige Lösung zu finden. Dabei werden die Unzulässigkeiten über einen Strafterm in der Zielfunktion minimiert. Innerhalb der Optimierungsverfahren wird also das Modell (7.4) verwendet. Allerdings könnten die Auswirkungen von einer solchen Sequenz gravierend sein und unter Umständen einen Bandstopp erzwingen. Deshalb darf in der abschließenden Produktionssequenz nicht mehr gegen diese neue Bedingung verstoßen werden.

Dafür dürfen zwei andere Nebenbedingungen wie im Modell (7.3), auf das das Gesamtproblem ausgelegt ist, im Notfall ignoriert werden. Es dürfen auch Aufträge, die nie zurückgestellt waren, auf eine neue Position in der Produktionssequenz gesetzt

werden. Die Engpässe werden also mit Mehraufwand durch den Logistikbereich abgefangen. Daraus leitet sich eine wichtige Untersuchungsgröße ab. Die Anzahl dieser nicht zurückgestellten und dennoch verschobenen Fahrzeuge muss überprüft werden. Diese Fahrzeuge werden *Folgerücksteller* genannt. Für den Einsatz in der Realität muss abgewogen werden, wie viele Folgerückstellungen akzeptabel sind und sich durch die Reduzierung der Springeranzahl rechtfertigen lassen. Neben dem zusätzlichen logistischen Aufwand muss in der Praxis ebenfalls beachtet werden, ob die verfügbare Fläche - insbesondere zum Vertauschen von großen Bauteilen - ausreicht.

Außerdem können sich Aufträge wegen des Fehlens von Springern verspäten. Die vorgegebenen Fälligkeitstermine müssen nicht mehr eingehalten werden, wenn dies gegen die neue Nebenbedingung verstoßen würde. Deshalb wird bei den Tests überprüft, ob durch diese Variante viele neue Terminüberschreitungen verursacht werden.

Zur Festsetzung der Anzahl der verfügbaren Springer in der Praxis wird empfohlen, diese Zahl in Absprache mit Experten sukzessive zu verringern und die Auswirkungen zu analysieren. So kann mit einem relativ geringen Risiko nach einiger Zeit das volle Optimierungspotential ausgeschöpft werden.

Nun wird auf die algorithmische Umsetzung dieses Ansatzes eingegangen. Die entscheidende Erweiterung zu den Algorithmen in den bisherigen Kapiteln ist eine Prüfprozedur, die Aufträge aus der Plansequenz entfernt, wenn bei diesen ein nicht verfügbarer Springer verwendet werden müsste. Diese wird nötig, da wie beschrieben während des Einsatzes der lokalen Suchverfahren diese Nebenbedingung entfällt und durch einen Strafterm in der Zielfunktion ersetzt wird.

Neben dieser Prozedur wird im Wesentlichen auf den DVTS+REPLAN Algorithmus zurückgegriffen. Auf eine Verwendung der Abwartestrategie wird verzichtet, damit eine bessere Kontrolle über die Laufzeit erhalten bleibt und diese nicht den gegebenen Rahmen von 30 s überschreitet.

Algorithmus 7.1 beschreibt das Verfahren, das zur Einhaltung der verfügbaren Springeranzahl in den folgenden Tests eingesetzt wird. In jedem Takt wird zunächst der bekannte DVTS+REPLAN Algorithmus verwendet, um freigegebene Aufträge in die Plansequenz zu integrieren oder die Positionen bereits eingegliederter Aufträge zu

Algorithmus 7.1: Springeranzahl

```
Initialisierung;
für alle Takte j tue
    rufe Algorithmus DVTS+REPLAN auf;
    solange ein zusätzlicher Springer eingesetzt werden muss tue
        entferne den Auftrag, der den zusätzlichen Springer verursacht aus der
        Plansequenz;
    Ende
    rufe Algorithmus DVTS mit gerade entfernten Aufträgen auf;
    solange ein zusätzlicher Springer am ersten Auftrag eingesetzt werden muss
    tue
        entferne den ersten Auftrag aus der Plansequenz;
    Ende
Ende
```

optimieren. Für diesen Teil steht jedoch nur knapp die Hälfte der Rechenzeit zur Verfügung. Wenn die resultierende Sequenz im Sinne des Gesamtproblems unzulässig ist, d. h. Springer benötigt werden, die nicht vorhanden sind, wird diese Sequenz angepasst. Es wird der Auftrag entfernt, der den ersten Springereinsatz verursacht, für den kein Springer zur Verfügung steht. Auf diese Weise können iterativ auch mehrere Fahrzeuge in einem Takt aus der Plansequenz genommen werden. Die Anzahl dieser Fahrzeuge wird auf zehn pro Takt begrenzt. Diese Grenze wurde in den Tests jedoch nie erreicht.

Falls ein Auftrag entfernt werden muss, wiederholen sich die Methoden. Die freien Aufträge werden erneut eingefügt und die Plansequenz optimiert. Danach wird erneut der Zulässigkeitstest wiederholt. Allerdings wird dieses Mal nur der Auftrag an der ersten Position der Plansequenz getestet, da nur dieses Fahrzeug nach Ende der Iteration fixiert wird. Durch dieses Verfahren wird garantiert, dass der Auftrag, der fixiert wird, keinen Springereinsatz verursacht, der nicht von den vorhandenen Springern ausgeübt werden kann.

7.4 Bewertung des Ansatzes

In diesem Abschnitt werden zunächst die durchgeführten Tests beschrieben. Danach werden die Ergebnisse dargestellt und schließlich analysiert.

7.4.1 Testbeschreibung

Das Ziel der Tests ist es, eine komplette Schicht bewerten zu können. Dazu ist - wie zuvor beschrieben - eine lange Produktionssequenz nötig. Die Testsequenzen sind in drei Abschnitte gegliedert. Dabei liegt der Fokus auf dem mittleren, der einer einzelnen Schicht entspricht. Der erste Abschnitt muss lang genug sein, damit auch am letzten Arbeitsplatz ein Fahrzeug während dieses Abschnitts bearbeitet wird. Auf diese Weise kann in der Folge der laufende Betrieb simuliert werden. Die Konsequenz ist, dass dieser Teil länger als eine eigentliche Schicht ist. Dennoch werden zu Testzwecken die während der Zeit des Einsteuerns dieser Fahrzeuge benötigten Springer ebenfalls berechnet.

Der folgende Abschnitt enthält so viele Aufträge, wie in einer Schicht eingesteuert werden. Er wird mit sämtlichen Montagevorgängen und allen Einsteuerungsentscheidungen betrachtet und die in dieser Schicht benötigte Anzahl an Springern stellt eine zentrale Größe der Tests dar. Darüber hinaus werden einige weitere Fahrzeuge in einem dritten Teil angehangen. Hier sind hauptsächlich die Auswirkungen der Einsteuerungen im zweiten Abschnitt interessant, die in einige Bewertungsgrößen einfließen.

Als relevante Größen werden die Anzahl der verwendeten Springer (bei mindestens einer Testinstanz), die Anzahl der durchschnittlich verwendeten Springer (bei einer kompletten Testsequenz) sowie die Anzahl der durchschnittlich in einer Schicht verwendeten Springer (im mittleren Abschnitt) analysiert. Darüber hinaus wird die Zielfunktion vom vorherigen Kapitel, die Anzahl der Springereinsätze, für jede Testsequenz überprüft. Zusätzlich wird die durchschnittliche Zahl der nötigen Folgerückstellungen und der verspätet eingesteuerten Fahrzeuge untersucht.

Bei diesen Tests wurden dieselben Daten wie im vorangegangenen Kapitel verwendet.

Dabei wurden die Daten von einem Tag zur Kalibrierung herangezogen, um eine sinnvolle Verteilung der verfügbaren Springer zu erzeugen. Die restlichen vier Tage wurden im Rahmen der folgenden Analyse ausgewertet.

Die Sperrszenarien wurden ähnlich wie im vorangegangenen Kapitel gewählt. Insbesondere bei den Codesperrungen wurde beachtet, dass sich die Konsequenzen der Wiedereinsteuerung der freigegebenen Fahrzeuge im Wesentlichen in der mittleren Schicht auswirken. Die zufälligen Sperrungen wurden dagegen über die gesamte Sequenz verteilt.

Um die Verteilung von verfügbaren Springern sinnvoll festlegen zu können, wurden die Testdaten mit dem DVTS Algorithmus aus dem letzten Kapitel ausgewertet. Die Anzahl der dabei maximal benötigten Springer wurde als größter sinnvoller Wert angenommen. Dieser wurde dann schrittweise reduziert. Dabei wurden in jedem Schritt solche Springer entfernt, die nur wenige Einsätze im Vergleichstest hatten. Außerdem wurden verschiedene Aufteilungen der gesamten Menge von Springern auf die Gruppen an einem Datensatz getestet und die beste Verteilung für die abschließenden Tests verwendet.

Die Testumgebung war identisch zur Umgebung der Tests im vorherigen Kapitel. Die verfügbare Rechenzeit der Kernprozeduren VTS bzw. VNS wurde nun bei beiden Aufrufen im Verlauf eines Takts auf 5 s gesetzt.

Bevor die Ergebnisse beschrieben werden, muss angemerkt werden, dass eine relevante Größe nicht realistisch bestimmt wurde. Die Verzögerungskonstanten τ_l sind nur innerhalb von Montagebändern und damit auch innerhalb aller Springergruppen konsistent. Die Zeiten zum Einlauf auf dem ersten Montageband und die Übergangszeiten zwischen den Bändern wurden vereinfacht auf einen Takt gesetzt. Damit entspricht der zeitliche Ablauf nicht exakt den realistischen Bedingungen. Da dieser aber nur innerhalb der Springergruppen relevant wird, sind die Testresultate komplett aussagekräftig.

7.4.2 Testergebnisse

Das erste Ergebnis ist, dass der DVTS Algorithmus Sequenzen erzeugte, für die 60 Springer notwendig sind, damit für alle Szenarien ausreichend Springer vorhanden sind. Ausgehend davon wurden Verteilungen von 50 bis 25 Springern auf die Gruppen ermittelt. Die Resultate des DVTS Algorithmus sind in den Grafiken in diesem Kapitel mit der Bezeichnung *unbegrenzt* dargestellt, da keine Limitierung der Anzahl der Springer vorliegt.

Zunächst wird die in den verschiedenen Varianten benötigte Anzahl an Springern betrachtet. Dazu wurden drei relevante Werte identifiziert. Als erstes ist die Anzahl der Springer dargestellt, die in mindestens einem Testszenario eingesetzt wurden. Zusätzlich wird die durchschnittliche Anzahl der Springer genannt, die in mindestens einem der drei Abschnitte der jeweiligen Tests benötigt wurden. Außerdem ist die durchschnittliche Anzahl der tatsächlich eingesetzten Springer in der mittleren Schicht, auf der der Fokus liegt, angegeben. Diese Ergebnisse sind in Abbildung 7.3 dargestellt.

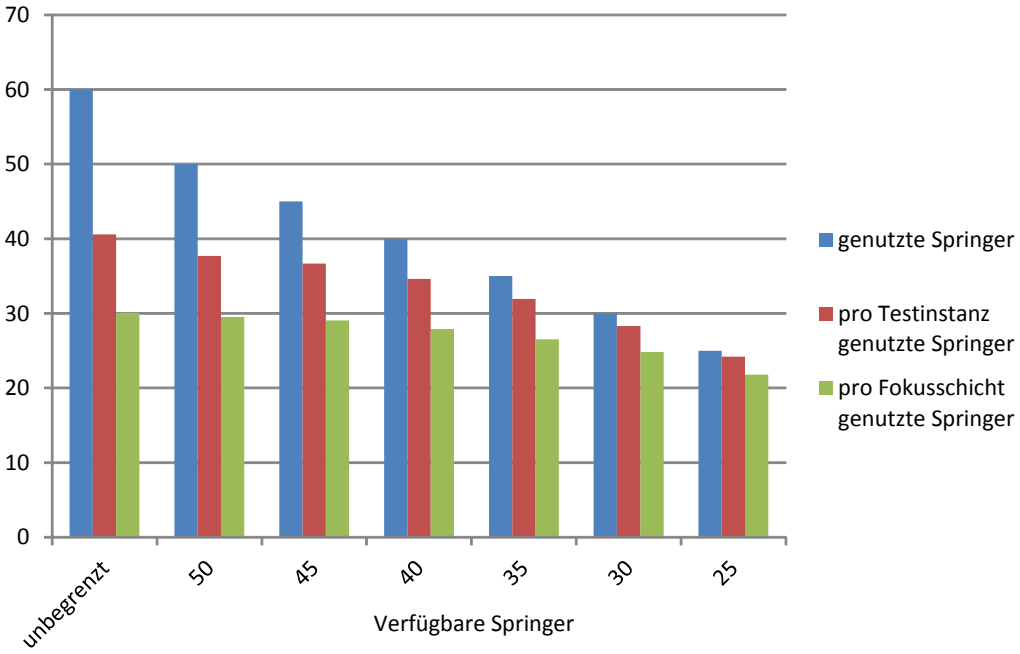


Abbildung 7.3: Darstellung der verwendeten Springer

Die Ergebnisse zeigen, dass jeder in den neuen Verfahren verfügbare Springer in min-

destens einem Testszenario eingesetzt wurde. Auch die anderen Ergebnisse überraschen nicht. Je weniger Springer zur Verfügung standen, desto weniger Springer wurden auch eingesetzt. Dies gilt sowohl für eine komplette Testinstanz als auch für die isoliert betrachteten mittleren Abschnitte. Die neuen Nebenbedingungen wurden in allen Tests tatsächlich eingehalten. Interessanter ist es nun, die Auswirkungen dieser Bedingungen auf andere Werte zu analysieren.

Deshalb wird als nächstes die eigentliche Zielfunktion, die schon im letzten Kapitel verwendet wurde, untersucht. Die durchschnittliche Anzahl der Springereinsätze über die gesamte Sequenz ist in Abbildung 7.4 dargestellt.

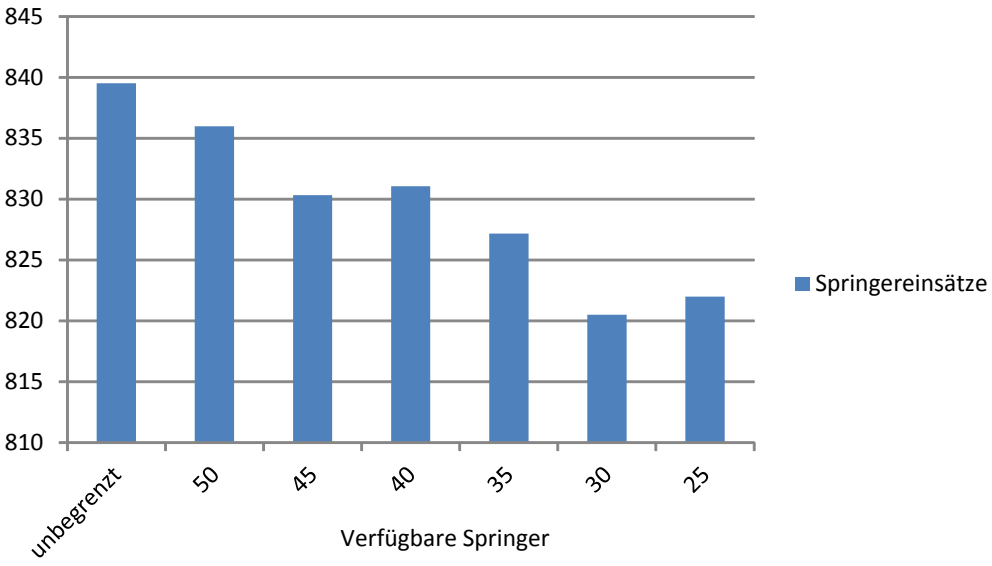


Abbildung 7.4: Darstellung der nötigen Springereinsätze

Die Ergebnisse ähneln den vorherigen. Mit sinkender Zahl von verfügbaren Springern sank tendenziell auch die Anzahl von Springereinsätzen. Dieses Ergebnis konnte jedoch nicht ohne weiteres antizipiert werden. Dadurch, dass gewisse Springer nicht verfügbar sind, sind bestimmte Sequenzen nicht mehr zulässig. Daher könnte der eingeschränkte Lösungsraum zu einer Verschlechterung dieser Zielfunktion führen. Dass dies nicht passiert, ist eine wichtige Erkenntnis, die für die Verwendung dieses Verfahrens spricht. Möglich wird dies jedoch nur dadurch, dass andere Nebenbedingungen in gewissen Fällen ignoriert werden dürfen. Deshalb müssen diese genauer analysiert werden.

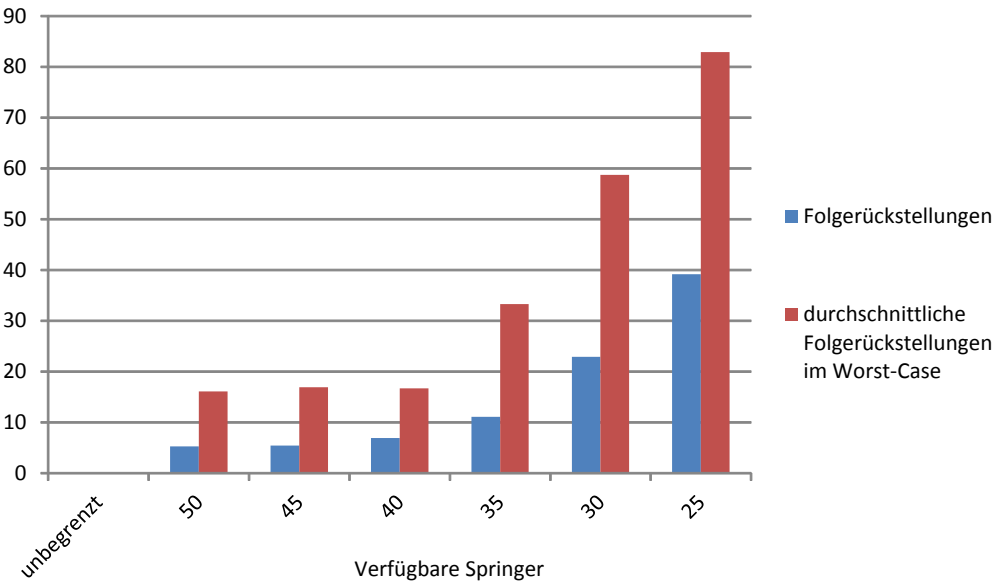


Abbildung 7.5: Darstellung der Folgerückstellungen

In Abbildung 7.5 ist die durchschnittliche Anzahl der Folgerückstellungen im schlimmsten Szenario und über alle Szenarien gemittelt zu sehen. Hier zeigt sich, dass die Reduzierung der Anzahl der Springer und Springereinsätze nur auf Kosten von Folgerückstellungen möglich ist. Dabei konnte die Anzahl der Springer in den Tests bis auf 40 reduziert werden und dennoch wurden im Durchschnitt deutlich unter zehn Folgerückstellungen notwendig. Selbst in den schwersten Testinstanzen reichten 20 Folgerückstellungen aus, um die Engpässe der Springer zu überstehen. Die durchschnittliche Anzahl der Folgerückstellungen entsprach bei diesem Test etwa 10 % der Rückstellungen, die von Sperrungen verursacht wurden.

Bei einer weiteren Reduzierung der Springeranzahl wurde jedoch ein starkes Ansteigen der Anzahl der Folgerückstellungen deutlich. Bei 30 verfügbaren Springern wurden durchschnittlich über 20 Folgerückstellungen notwendig, bei nur noch 25 verfügbaren Springern fast 40 und im schwierigsten Szenario sogar über 80. Dies ist der Grund dafür, dass eine Wahl von weniger als 25 Springern nicht mehr untersucht wurde.

Die Anzahl der verspäteten Aufträge wird in Abbildung 7.6 dargestellt. Bereits in der ursprünglichen Methode ohne Beschränkung der Springeranzahl wurden allein durch die Sperrungen einige Termine überschritten. Diese Zahl stieg bei Limitierung der

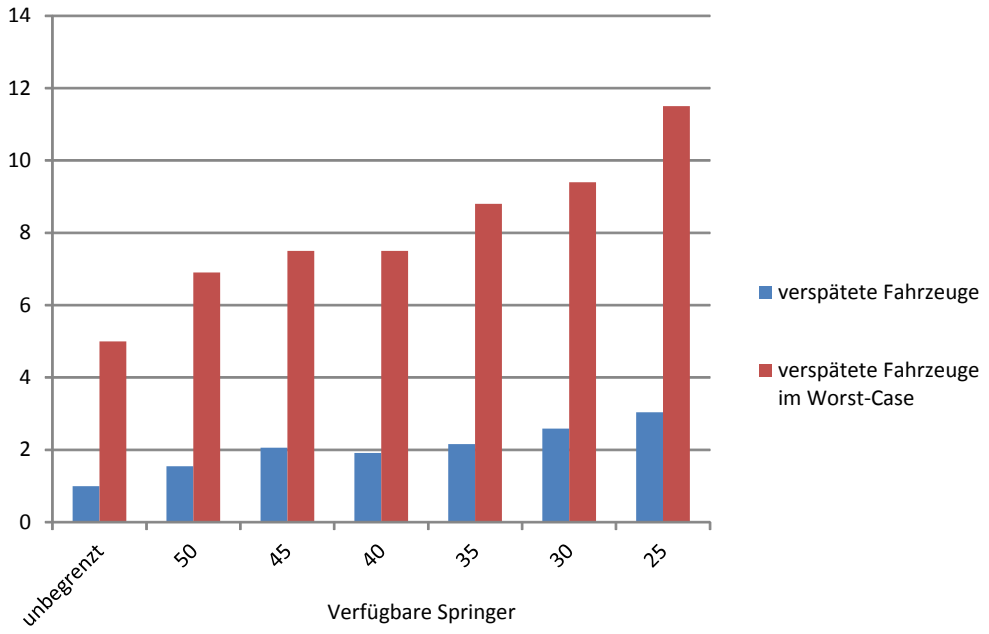


Abbildung 7.6: Darstellung der Terminverstöße

verfügbaren Springer leicht an. Insgesamt gab es jedoch selbst bei einer sehr starken Reduzierung der Springeranzahl, trotz einer beachtlichen Anzahl an Folgerückstellungen, nur wenige Überschreitungen der Fälligkeitstermine.

7.4.3 Folgerungen

Das wichtigste Resultat der Tests ist, dass es unter den gewählten Bedingungen möglich ist, die Anzahl der verwendeten Springer deutlich zu reduzieren. Sogar eine Reduzierung der Anzahl der verfügbaren Springer um mehr als die Hälfte wurde getestet und könnte umsetzbar sein. Letztlich muss zwischen einer Reduzierung dieser Anzahl und einem gesteigerten Logistikaufwand, der durch Folgerückstellungen verursacht wird, abgewogen werden. Die Ergebnisse legen nahe, dass mit einem relativ geringen Mehraufwand in der Logistik ein Drittel der Springer eingespart werden kann, wenn die entwickelte Methode eingesetzt wird. Dabei wurden die konkreten Auswirkungen der Folgerückstellungen nicht näher betrachtet.

Bei Verwendung einer Variante, in der viele Folgerückstellungen nötig sind, steigt

auch die Anzahl der Terminüberschreitungen. Allerdings bleibt dieser Wert - absolut betrachtet - auf einem niedrigen Niveau. Dieses Resultat belegt, dass es für fast alle Aufträge zulässige Positionen in der Produktionssequenz gibt. Alle Folgerückstellungen entstehen durch Unzulässigkeiten, dennoch findet sich meistens innerhalb eines kurzen Intervalls eine alternative zulässige Position.

Die Anwendbarkeit in der Praxis lässt sich letztlich nicht mehr analytisch bestimmen. Die Testergebnisse unterstreichen, dass ein gewisses Optimierungspotential existiert, das mit einer genauen Betrachtung aller Tätigkeiten von allen Arbeitern an einer Montagelinie ausgeschöpft werden kann. Inwieweit sich die Resultate auf die Praxis übertragen lassen und wie groß die Abweichungen der realen Springereinsätze und -verteilungen sind, kann nur durch Versuche im realen Betrieb untersucht werden.

Genau dafür wurde allerdings durch diese Analyse eine Grundlage geschaffen. Das Potential kann schrittwiese erschlossen werden. Selten eingesetzte Springer können nach und nach erst aus der Datengrundlage und bei Erfolg aus der realen Fabrik entfernt werden und mit der entwickelten Methodik kann versucht werden, eine Produktionssequenz zu erzeugen, sodass diese tatsächlich nicht mehr genutzt werden. Auf diese Weise kann die in diesem Kapitel entwickelte Idee ohne Risiko in die Praxis eingeführt werden.

Dabei kann unterschieden werden, ob die gleiche Anzahl an Werkern und Springern in jeder Schicht verwendet werden soll, oder ob - bei einer guten Prognose der Arbeitsbelastungen einer Schicht - insbesondere die Zahl der zur Verfügung stehenden Springer variieren darf. Diese Entscheidungen würden bei Einsatz des Verfahrens zunächst von Experten getroffen werden müssen. Als Verbesserung könnte hier eine vertiefende Analyse zur Entwicklung automatischer Methoden, die zur Bestimmung der benötigten Anzahl an Springern entscheidungsunterstützend eingesetzt werden können, durchgeführt werden.

Allerdings sollen an dieser Stelle auch mögliche Schwachpunkte nicht verschwiegen werden. Eine wesentliche Grundlage für die Berechnungen ist eine konstante oder zumindest vorab bekannte Produktionsmenge pro Schicht. Da diese in der Realität jedoch Schwankungen unterworfen ist, kann es zu Beginn und zum Ende von Schichten zu Verschiebungen kommen, sodass Springer fehlen könnten, wenn sich die Anzahl

zwischen den Schichten unterscheidet. Außerdem müssten sämtliche Arbeitszeiten exakt eingehalten werden, damit das zeitliche Auftreten von Springereinsätzen korrekt vorhergesagt werden kann. Da dies offensichtlich bei von Menschen ausgeführten Tätigkeiten nicht der Fall ist, bleibt die Frage bestehen, wie gravierend Auswirkungen von Abweichungen der Arbeitszeiten sind.

Eine mögliche Erweiterung des beschriebenen Ansatzes ist, die Anzahl der benötigten Folgerückstellungen direkt in die Zielfunktion zu integrieren. Dabei müsste außerdem eine Methode entwickelt werden, wie ein optimales Auswählen von Folgerückstellern erfolgen kann und wie diese dann verschoben werden können. Die Entscheidung darüber, welches Fahrzeug aus der Sequenz entfernt werden soll, um einen nicht abgedeckten Springereinsatz zu verhindern, stellt eine große Herausforderung dar.

8 Zusammenfassung und Ausblick

In dieser Arbeit wurde das Problem der Einsteuerung von Aufträgen auf Variantenfließlinien in verschiedenen Ausprägungen analysiert. Dazu wurde der Mixed-Model-Sequencing Ansatz auf die Resequenzierung angewandt. In den ersten beiden Hauptkapiteln wurde die Minimierung der Anzahl der Springereinsätze modelliert und untersucht.

Zunächst wurde dabei das statische Optimierungsproblem der Wiedereinsteuerung beschrieben. Dazu wurden aus der Literatur bekannte Ansätze um einige in der Praxis relevante Aspekte erweitert. Neue Modelle enthalten nun auch mehrtaktige Arbeitsplätze. Außerdem kann durch eine geeignete Modellierung ein freiwilliges Vorziehen von Springereinsätzen vermieden werden. Es konnte gezeigt werden, unter welchen Bedingungen diese Einschränkungen auch bei Mehrtaktern nicht nötig sind.

Zur Lösung des Problems wurden verschiedene Algorithmen entwickelt. Die besten Ergebnisse erzielte eine Methode, die Ansätze der Variable Neighborhood Search und der Tabu Search kombiniert. Dabei konnte gezeigt werden, dass bis zu einer Problemgröße, für die in akzeptabler Zeit eine optimale Lösung durch vollständige Enumeration bestimmt werden kann, die Ergebnisse optimal sind oder zumindest sehr nah am Optimum liegen.

Der entscheidende Beitrag dieser Arbeit zur Forschung liegt in der konkreten Betrachtung der Dynamik und der Analyse längerer Produktionszeiträume. Hierzu wurden Modelle formuliert, die das Problem mathematisch greifbar darstellen. Auf Basis der Algorithmen für das statische Problem wurden Methoden entwickelt, die die gesamte Einsteuerung lenken können. Für diese wurde bewiesen, dass unter realistischen Annahmen jede Sperrung verarbeitet werden kann und eine zulässige Produktionsse-

quenz erzeugt wird. Die entwickelten Methoden können im Rahmen einer robusten Produktionssteuerung eingesetzt werden.

Die Testergebnisse zeigen, dass eine signifikante Reduzierung der nötigen Springereinsätze möglich ist, wenn diese neuen Optimierungsverfahren anstatt einfacher, aus der Praxis bekannter Verfahren eingesetzt werden. Dabei kann eine rollierende Anwendung der statischen Methoden mit zusätzlicher Optimierung in Takten ohne Freigaben neuer Aufträge sehr gute Ergebnisse erzielen. Eine alternative Strategie, die ein Zurückhalten freigegebener Fahrzeuge über mehrere Takte erlaubt, kann in bestimmten Szenarien bessere Sequenzen erzeugen. Beide Verfahren konnten bei der Betrachtung kritischer Stationen die Anzahl der Springereinsätze an diesen halbieren und gleichzeitig die Springereinsätze an den restlichen Stationen um über 15 % senken.

Ein Resultat ist, dass in Abhängigkeit von der Testinstanz verschiedene Algorithmen zu bevorzugen sind. Ebenso wurde auch bei der Beschreibung der Einstellungsmöglichkeiten eines einzelnen Ansatzes deutlich, dass viele Varianten möglich sind. Letztlich muss der Algorithmus nach Analyse der jeweiligen praktischen Gegebenheiten an der zu optimierenden Montagelinie gewählt werden. Außerdem muss dieser dann nach Betrachtung der gegebenen strategischen Entscheidungen und linienabhängigen Eigenschaften kalibriert werden.

Zuletzt wurde das Potential untersucht, das sich bei Analyse der Anzahl der benötigten Springer ergibt. Dabei wurde gezeigt, dass bei genauer Betrachtung der Springer deren benötigte Anzahl deutlich reduziert werden kann. Daraus resultierende Konsequenzen im Hinblick auf Logistikbereiche und Termineinhaltungen wurden aufgezeigt und können bei der Praxisanwendung mit den Einsparungen bei den Springern abgewogen werden.

Aus wissenschaftlicher Sicht wäre die Berechnung besserer Schranken für das dynamische Problem eine große Hilfe zur Bewertung der entwickelten Methoden. Aufgrund der Ergebnisse in dieser Arbeit kann festgehalten werden, dass ein großer Teil des Optimierungspotentials durch eine Einsteuerung mit den beschriebenen Methoden genutzt werden kann. Wie groß das verbleibende Potential ist, konnte leider nicht bestimmt werden.

Unter bestimmten Voraussetzungen können die entwickelten Verfahren verbessert und gezielter eingesetzt werden. Informationen über die Dauer von Sperrungen wären sehr nützlich für die Konstruktion der Plansequenz. Wenn bei dem Auftreten von Sperrungen deren Dauer schon bekannt wäre, sollte diese Information unbedingt in die Verfahren integriert werden.

In einigen Passagen dieser Arbeit wurde schon auf die Auswirkungen für den Logistikbereich eingegangen. Diese sollten vor einem Praxiseinsatz genau überprüft werden und gegebenenfalls muss eine zusätzliche Nebenbedingung verwendet werden. Aufgrund des beschränkten Platzes zur Zwischenlagerung von Teilen, kann es nötig sein, die Zahl der zurückgestellten Aufträge zu begrenzen. Das bedeutet, dass bei vielen vorhandenen Sperrungen die Fahrzeuge nach Freigabe eventuell schneller in die Produktionssequenz integriert werden sollten, als es bei bloßer Beachtung der Fälligkeitstermine vorgesehen ist.

Die Analyse der Mehrtakter in dieser Arbeit kann auch losgelöst vom Resequenzierungskontext weitere Erkenntnisse bringen. Eine detaillierte Betrachtung der möglichen Organisationen von Mehrtaktern ist auch bei allgemeinen Sequenzierungsansätzen ein sinnvoller Forschungsansatz.

Diese Arbeit zeigt also einige Ansätze auf, die in der Praxis die Effizienz steigern können. Dabei sollten gerade die einschränkenden Elemente, die sich als potentialmindernd in den theoretischen Analysen gezeigt haben, regelmäßig überdacht werden. So können neben dem Einsatz verbesserter Algorithmen vor allem organisatorische Veränderungen erst Möglichkeiten zur Weiterentwicklung der Einsteuerung schaffen.

Literaturverzeichnis

[AHKD10] Altemeier, S. ; Helmdach, M. ; Koberstein, A. ; Dangelmaier, W.: Reconfiguration of assembly lines under the influence of high product variety in the automotive industry – A decision support system. In: *International Journal of Production Research* 48 (2010), Nr. 21, S. 6235–6256

[AHME+13] Artigues, C. ; Hebrard, E. ; Mayer-Eichberger, V. ; Siala, M. ; Walsh, T.: SAT and hybrid models of the car-sequencing problem. In: *Third International Workshop on the Cross-Fertilization Between CSP and SAT, in conjunction with CP 2013*, 2013

[Alt09] Altemeier, S.: *Kostenoptimale Kapazitätsabstimmung in einer getakteten Variantenfließlinie unter expliziter Berücksichtigung des Unterstützereinsatzes und unterschiedlicher Planungszeiträume*, Fakultät für Wirtschaftswissenschaften, Universität Paderborn, Dissertation, 2009

[BBC11] Bertsimas, D. ; Brown, D. B. ; Caramanis, C.: Theory and applications of robust optimization. In: *SIAM review* 53 (2011), Nr. 3, S. 464–501

[BC11] Bautista, J. ; Cano, A.: Solving mixed model sequencing problem in assembly lines with serial workstations with work overload minimisation and interruption rules. In: *European Journal of Operational Research* 210 (2011), Nr. 3, S. 495 – 513

[BDGG06] Bianchi, L. ; Dorigo, M. ; Gambardella, L. M. ; Gutjahr, W. J.: Metaheuristics in stochastic combinatorial optimization: a survey / Dalle

Molle Institute for Artificial Intelligence. 2006. – Forschungsbericht

[BEY98] Borodin, A. ; El-Yaniv, R.: *Online computation and competitive analysis*. Cambridge University Press Cambridge, 1998

[BFS07] Boysen, N. ; Fliedner, M. ; Scholl, A.: Level-Scheduling bei Variantenfließfertigung: Klassifikation, Literaturüberblick und Modellkritik. In: *Journal für Betriebswirtschaft* 57 (2007), Nr. 1, S. 37–66

[BFS09] Boysen, N. ; Fliedner, M. ; Scholl, A.: Sequencing mixed-model assembly lines: Survey, classification and model critique. In: *European Journal of Operational Research* 192 (2009), Nr. 2, S. 349 – 373

[BGR11] Boysen, N. ; Golle, U. ; Rothlauf, F.: The car resequencing problem with pull-off tables. In: *BuR - Business Research* 4 (2011), Nr. 2, S. 276–292

[BKS11] Boysen, N. ; Kiel, M. ; Scholl, A.: Sequencing mixed-model assembly lines to minimise the number of work overload situations. In: *International Journal of Production Research* 49 (2011), Nr. 16, S. 4735–4760

[Boy05] Boysen, N.: *Variantenfließfertigung*. Deutscher Universitätsverlag, 2005 (Betriebswirtschaftliche Forschung zur Unternehmensführung)

[BR03] Blum, C. ; Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. In: *ACM Computing Surveys* 35 (2003), Nr. 3, S. 268–308

[BS09] Bautista, J. ; Suarez, R.: Mixed-model sequencing problem with overload minimization considering workstations dependencies. In: *IEEE International Symposium on Assembly and Manufacturing, ISAM 2009*, 2009, S. 351 –357

[BSJ94] Bard, J. F. ; Shtub, A. ; Joshi, S. B.: Sequencing mixed-model assembly lines to level parts usage and minimize line length. In: *International Journal of Production Research* 32 (1994), Nr. 10, S. 2431–2454

[BSW12] Boysen, N. ; Scholl, A. ; Wopperer, N.: Resequencing of mixed-model assembly lines: Survey and research agenda. In: *European Journal of Operational Research* 216 (2012), Nr. 3, S. 594 – 604

[BZ13] Boysen, N. ; Zenker, M.: A decomposition approach for the car resequencing problem with selectivity banks. In: *Computers & Operations Research* 40 (2013), Nr. 1, S. 98–108

[CLP08] Cordeau, J.-F. ; Laporte, G. ; Pasin, F.: Iterated tabu search for the car sequencing problem. In: *European Journal of Operational Research* 191 (2008), Nr. 3, S. 945 – 956

[CS97] Choi, W. ; Shin, H.: A real-time sequence control system for the level production of the automobile assembly line. In: *Computers & Industrial Engineering* 33 (1997), Nr. 3–4, S. 769 – 772

[DSVH88] Dincbas, M. ; Simonis, H. ; Van Hentenryck, P.: Solving the car sequencing problem in constraint logic programming. In: *European Conference on Artificial Intelligence (ECAI-88)*, 1988

[Če85] Černý, V.: Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. In: *Journal of Optimization Theory and Applications* 45 (1985), S. 41–51

[EBS10] Emde, S. ; Boysen, N. ; Scholl, A.: Balancing mixed-model assembly lines: A computational evaluation of objectives to smoothen workload. In: *International Journal of Production Research* 48 (2010), Nr. 11, S. 3173–3191

[Fre83] Freeman, P. R.: The secretary problem and its extensions: A review. In: *International Statistical Review* 51 (1983), Nr. 2, S. 189–206

[Gen03] Gendreau, M.: An introduction to tabu search. In: Glover, F. (Hrsg.) ; Kochenberger, G. (Hrsg.): *Handbook of Metaheuristics*. Boston: Kluwer, 2003, S. 37–54

[GG09a] Gujjula, R. ; Günther, H.-O.: Rescheduling blocked workpieces at

mixed-model assembly lines with just-in-sequence supply. In: *Proceedings of Asia Pacific Industrial Engineering and Management Systems Conference (APIEMS 2009)*, 2009, S. 2758–2763

[GG09b] Gujjula, R. ; Günther, H.-O.: Resequencing mixed-model assembly lines under just-in-sequence constraints. In: *International Conference on Computers and Industrial Engineering (CIE 2009)*, 2009, S. 668–673

[GG09c] Gujjula, R. ; Günther, H.-O.: Scheduling utility workers at mixed-model assembly lines. In: *Proceedings of the 2009 IEEE International Conference on Industrial Engineering and Engineering Management*, 2009, S. 1092–1096

[GG10] Gujjula, R. ; Günther, H.-O.: An efficient heuristic to sequence mixed-model assembly lines. In: *Proceedings of the 2010 IEEE International Conference on Industrial Engineering and Engineering Management*, 2010, S. 1209–1213

[GKR+01] Grötschel, M. ; Krumke, S. O. ; Rambau, J. ; Winter, T. ; Zimmermann, U. T.: Combinatorial online optimization in real time / Konrad-Zuse-Zentrum für Informationstechnik Berlin. 2001. – Forschungsbericht

[Glo86] Glover, F.: Future paths for integer programming and links to artificial intelligence. In: *Computers & Operations Research* 13 (1986), Nr. 5, S. 533–549

[Gra69] Graham, R. L.: Bounds on multiprocessing timing anomalies. In: *SIAM Journal on Applied Mathematics* 17 (1969), Nr. 2, S. 416–429

[GRB11] Golle, U. ; Rothlauf, F. ; Boysen, N.: Car sequencing versus mixed-model sequencing: A computational study. In: *On the Car Sequencing Problem: Analysis and Solution Methods* (2011), S. 22–52

[IS03] Inman, R. R. ; Schmeling, D. M.: Algorithm for agile assembling-to-order in the automotive industry. In: *International Journal of Production*

Research 41 (2003), Nr. 16, S. 3832–3848

[KGV83] Kirkpatrick, S. ; Gelatt, C. D. ; Vecchi, M. P.: Optimization by simulated annealing. In: *Science* 220 (1983), S. 671–680

[KMRS88] Karlin, A. ; Manasse, M. ; Rudolph, L. ; Sleator, D.: Competitive snoopy caching. In: *Algorithmica* 3 (1988), S. 79–119

[LA87] Laarhoven, P. J. M. ; Aarts, E. H. L.: *Simulated annealing: theory and applications*. Springer Netherlands, 1987

[LLMS09] Liebchen, C. ; Lübbecke, M. E. ; Möhring, R. H. ; Stiller, S.: The concept of recoverable robustness, linear programming recovery, and railway applications. In: Ahuja, R. K. (Hrsg.) ; Möhring, R. H. (Hrsg.) ; Zaroliagis, C. D. (Hrsg.): *Robust and Online Large-Scale Optimization*. Springer Berlin Heidelberg, 2009, S. 1–27

[Mas03] Mastrolilli, M.: Scheduling to minimize max flow time: Offline and online algorithms. In: Lingas, A. (Hrsg.) ; Nilsson, B. (Hrsg.): *Fundamentals of Computation Theory*. Springer Berlin Heidelberg, 2003, S. 989–997

[Mey04] Meyr, H.: Supply chain planning in the German automotive industry. In: *OR Spectrum* 26 (2004), S. 447–470

[MF00] McMullen, P. R. ; Frazier, G. V.: A simulated annealing approach to mixed-model sequencing with multiple objectives on a just-in-time line. In: *IIE Transactions* 32 (2000), Nr. 8, S. 679–686

[MH97] Mladenović, N. ; Hansen, P.: Variable neighborhood search. In: *Computers & Operations Research* 24 (1997), Nr. 11, S. 1097 – 1100

[OY79] Okamura, K. ; Yamashina, H.: A heuristic algorithm for the assembly line model-mix sequencing problem to minimize the risk of stopping the conveyor. In: *International Journal of Production Research* 17 (1979), Nr. 3, S. 233–247

[PKW86] Parrello, B. D. ; Kabat, W. C. ; Wos, L.: Job-shop scheduling using automated reasoning: A case study of the car-sequencing problem. In: *Journal of Automated Reasoning* 2 (1986), S. 1–42

[PS82] Papadimitriou, C. H. ; Steiglitz, K..: *Combinatorial optimization: Algorithms and complexity.* Dover Publications, Inc., New York, 1982

[Rot11] Rothlauf, F.: *Design of Modern Heuristics.* Springer-Verlag Berlin Heidelberg, 2011

[SCNA08] Solnon, C. ; Cung, V. D. ; Nguyen, A. ; Artigues, C.: The car sequencing problem: Overview of state-of-the-art methods and industrial case-study of the ROADEF'2005 challenge problem. In: *European Journal of Operational Research* 191 (2008), Nr. 3, S. 912 – 927

[SDR09] Shapiro, A. ; Dentcheva, D. ; Ruszczyński, A. P.: *Lectures on Stochastic Programming: Modeling and Theory.* SIAM, 2009

[SJJN99] Smed, J. ; Johnsson, M. ; Johtela, T. ; Nevalainen, O.: Techniques and applications of production planning in electronics manufaturing systems / Turku Centre for Computer Science. 1999. – Forschungsbericht

[SKD98] Scholl, A. ; Klein, R. ; Domschke, W.: Pattern Based Vocabulary Building for Effectively Sequencing Mixed-Model Assembly Lines. In: *Journal of Heuristics* 4 (1998), Nr. 4, S. 359–381

[SPG+97] Séguin, R. ; Potvin, J-Y. ; Gendreau, M. ; Crainic, T. G. ; Marcotte, P.: Real-time decision problems: An operational research perspective. In: *The Journal of the Operational Research Society* 48 (1997), Nr. 2, S. 162–174

[Tsa95] Tsai, L.-H.: Mixed-model sequencing to minimize utility work and the risk of conveyor stoppage. In: *Management Science* 41 (1995), S. 485 – 495

[VHL03] Vieira, G. E. ; Herrmann, J. W. ; Lin, E.: Rescheduling manufacturing systems: A framework of strategies, policies, and methods. In: *Journal*

of Scheduling 6 (2003), Nr. 1, S. 39–62

[WK64] Wester, L. ; Kilbridge, M.: The assembly line model-mix sequencing problem. In: *Proceedings of the Third International Conference on Operations Research*, 1964, S. 247–260

[Yav13] Yavuz, M.: Iterated beam search for the combined car sequencing and level scheduling problem. In: *International Journal of Production Research* 51 (2013), Nr. 12, S. 3698–3718

[YB89] Yano, C. A. ; Bolat, A.: Survey, development, and application of algorithms for sequencing paced assembly lines. In: *Journal of Manufacturing and Operations Management* 2 (1989), S. 172–198

[YR91] Yano, C. A. ; Rachamadugu, R.: Sequencing to minimize work overload in assembly lines with product options. In: *Management Science* 37 (1991), Nr. 5, S. 572–586