EZOLAT

Softwarekomponente für die echtzeitfähige Optimierung innerbetrieblicher Lager- und Transportvorgänge

Schlussbericht

Denise Holfeld Axel Simroth

Fraunhofer-Institut für Verkehrs- und Infrastruktursysteme (IVI), Dresden

13. Dezember 2013

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS10024B gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Bearbeitungszeitraum:
01.07.2010 – 31.12.2012
mit Verlängerung bis 31.05.2013

Bundesministerium
für Bildung
und Forschung

Inhaltsverzeichnis

I. Übersicht

I.1. Kurze Darstellung zur Aufgabenstellung

Primäre Zielsetzung des geplanten Vorhabens war die Entwicklung einer neuartigen Softwarekomponente für die echtzeitfähige Optimierung von innerbetrieblichen Lager- und Transportvorgängen unter Ungewissheit. Die Entwicklung dieser Komponente sollte einhergehen mit der Erarbeitung eines allgemeingültigen Konzepts, in dem die Planung von Material- und Produktflüssen, Transport- und Lagersystemen unter Nutzung von RFID-Technologien zu einer Optimierungsaufgabe zusammengefasst ist. Die Softwarekomponente sollte prototypisch realisiert und in ein bestehendes Lagerverwaltungssystem integriert werden, sowie im Rahmen von Pilotanwendungen seine Praxistauglichkeit demonstrieren.
Die zentrale Aufgabe, der sich dieses Vorhaben gestellt wurde, ist die optimale Nutzung der in einem Fertigungsbetrieb vorhandenen Transportmittel zur Bedienung der laufend anfallenden ungewissen Transportaufträge. Die Kriterien, die bei dieser Planungsaufgabe eine Rolle spielen, sind die folgenden:

- Bearbeitungszeiten der Transportaufträge: Ein zügiger Material- und Produktfluss gewährleistet einen reibungslosen Ablauf der Fertigung. Zu transportierende Aufträge werden an Arbeitsplätzen zur Bearbeitung benötigt oder müssen von Arbeitsplätzen weggebracht werden, um diese freizugeben.

- Auslastung der Kapazitäten der Transportsysteme: Leerfahrten und lange Fahrtstrecken sind zu vermeiden, um mit den gegebenen Kapazitäten den anfallenden Transportaufwand zu bewältigen.

Zusätzlich zu der Planung der Transportmittel muss die Belegung der Lagerplätze definiert werden. Die meisten Transportaufträge sind mit Ein- oder Auslagervorgängen in den die Fertigungsbereiche entkoppelnden Lagern verbunden, die Auswahl der richtigen Lagerplätze hat entscheidenden Einfluss auf die erreichbare Güte der Transportplanung. Besonders zu beachten ist, dass es sich bei der betrachteten Aufgabenstellung um ein sogenanntes Problem unter Ungewissheit handelt, welches durch folgende Eigenschaften gekennzeichnet ist:

- Die Informationen, unter denen das Problem (algorithmisch) gelöst wird, sind zu keinem Zeitpunkt vollständig oder mit Sicherheit bekannt. Vielmehr gehen neue Informationen permanent während der Ausführung bereits berechneter Transportpläne ein. Diese Pläne sind deshalb ständig den neuen Gegebenheiten anzupassen und zu erweitern.

- Die Neu-Berechnung oder Anpassung von Lösungen bei Bekanntwerden neuer Informationen muss sehr schnell, nahezu in Echtzeit erfolgen. Da sich die Gegebenheiten bei vielen zu transportierenden Aufträgen sehr oft ändern können ist es wichtig, die Transport und Lagervorgänge zeitnah zu steuern.

I.2. Voraussetzungen, unter denen das Vorhaben durchgeführt wurde

Dresden Informatik GmbH

Die Dresden Informatik GmbH ist ein mittelständisches IT-Unternehmen, das sich erfolgreich als Entwickler von branchenunabhängiger Software am Markt etabliert hat. Schwerpunkte

liegen auf den strategischen Geschäftsfeldern Optimierung der operativen Logistik und Prozessoptimierung, Steuerung von Fördertechnik sowie Prozessvisualisierung. Bei namhaften weltweit agierenden Konzernen sind die innovativen Softwarelösungen ebenso im Einsatz wie bei vielen kleinen und mittelständischen Unternehmen. Dabei reichen die Leistungen der Dresden Informatik von der Analyse und Optimierung von Geschäftsprozessen und technischen Prozessen, Planung und Lieferung von Soft- und Hardware, Programmierung, Customizing, Systemintegration, Anwenderschulung und Wartung bis zur Leitung der übernommenen Kundenprojekte. Dieses Projekt baute hauptsächlich auf die Kernsysteme zur Lagerverwaltung (DiLVS) und zur RFID Kopplung (DiRFID-Link RFID-Middleware) und auf den positiven Ergebnissen des abgeschlossenen Entwicklungsprojektes MESOS (Manufacturing Execution Simulation and Optimization System) auf, welches vom BMWi gefördert wurde.

Fraunhofer-Institut für Verkehrs- und Infrastruktursysteme

Die Forschungsgebiete des Fraunhofer IVI sind technologieorientiert und umfassen

- Intermodale Verkehrsinformations- und Managementsysteme,
- Verkehr, Energie und Umwelt sowie
- Fahrzeug- und Verkehrssystemtechnik.

Interdisziplinäre Forschungs- und Entwicklungsleistungen werden auf den Gebieten der Verkehrstelematik, Verkehrsökologie, Verkehrswirtschaft und Verkehrs-, Transport- und Logistikplanung sowie für die Automatisierung von Ver- und Entsorgungssystemen der Infrastruktur angeboten. Die Planung und Optimierung von Verkehrs-, Transport- und Logistiksystemen ist thematisch ein fester Bestandteil im Aufgabenprofil des Institutes. Es konnten seit langem Erfahrungen im Entwurf von Optimierungsmodellen und -verfahren gesammelt werden, die z.B. in Projekten zur statischen Tourenplanung eingesetzt worden sind.
Ein Forschungsschwerpunkt in der Gruppe Operations Research umfasst die Entwicklung einer algorithmischen Methodik für die Optimierung unter Ungewissheit, die speziell im Rahmen verschiedener Projekte auf dynamische und stochastische Tourenplanungsprobleme angewandt wird. Der algorithmische Lösungsansatz besteht in einer Adaption von Ideen aus der Spielbaumsuche unter Einbeziehung von Monte-Carlo-Verfahren. Ziel ist es, Entscheidungen für die aktuelle Situation abzuleiten, die auch in der Zukunft zu guten Lösungen führen werden.
Unsere Erfahrungen, die wir durch die Bearbeitung von dynamischen und stochastischen VRPs gesammelt haben und die zu der Entwicklung einer allgemeinen Methodik geführt haben, wurden bei der Bearbeitung dieses Projekts ausgenutzt. Da das hier betrachtete Problem zum einen in Grundzügen Ähnlichkeit zu VRPs aufweist, zum anderen ebenso als Problem unter Ungewissheit auftritt, wurde angestrebt unsere bewährte algorithmische Methodik zu übertragen und weiterzuentwickeln.

Liebherr Hausgeräte Ochsenhausen GmbH

Die Liebherr Hausgeräte Ochsenhausen GmbH fungiert als assoziierter Partner im Vorhabensverbund. Sie hat sich bereit erklärt, die Wirkungsweise der zu entwickelnden Softwarekomponente anhand ihrer Fertigungsprozesse am Standort Ochsenhausen zu demonstrieren.

Die dortige Fertigung von Kühlgeräten ist gekennzeichnet durch eine Aufteilung in autonome Fertigungsbereiche wie Gehäusevorbereitung, Gehäuseschäumerei, Türenschäumerei, Endmontage, durch eine Entkopplung der Bereiche durch zahlreiche Zwischenlager und Puffer, und durch einen hohen Einsatz an automatischen Transportsystemen, hauptsächlich bestehend aus horizontalen Transportbändern. Die Fertigungsbereiche umfassen Insel-, Linien- und Werkstattfertigung und werden unter Einsatz verschiedenster Planungsmethoden feingeplant und -gesteuert. Durch diese Charakteristik ist der Fertigungsprozess und das gesamte Umfeld nahezu prädestiniert für eine Anwendung der im Vorhaben zu entwickelnden Software.

Heinrichsthaler Milchwerke GmbH

Die Heinrichsthaler Milchwerke GmbH hat sich bereiterklärt als zweiter assoziierter Partner im Vorhabensverbund die zu entwickelnde Softwarekomponente im Halbfertigwarenlager ihres Distributionszentrums in Radeberg zu validieren. Im Halbfertigwarenlager wird Rohkäse in Form von Großblöcken in Kisten bis zur weiteren Verarbeitung zwischengelagert. Dabei durchläuft der Käse in Abhängigkeit von der konkreten Sorte über bestimmte Zeiträume verschiedene Temperaturstufen. Im Lager gibt es keine Regalanlage, d.h. die Kisten werden in mehreren Reihen hintereinander und bis zu zehn Kisten übereinander gestapelt. Durch Optimierung der Stapelreihenfolge unter Berücksichtigung der Transportkapazität der eingesetzten Transportmittel ergibt sich ein hohes Einsparungspotenzial und damit einen erheblichen Wettbewerbsvorteil für den Anwender der zu entwickelnden Softwarekomponente. Der Einsatz von RFID-Technik in Kombination mit einer automatischen Lokalisierung der Transportmittel bietet die Möglichkeit, den genauen Standort der einzelnen Kisten exakt zu lokalisieren und damit die softwaregestützte Transportoptimierung umzusetzen.

I.3. Planung und Ablauf des Vorhabens

Das Forschungsprojekt sollte in sechs Arbeitspaketen bearbeitet werden, zu dem die beteiligten Partner Dresden Informatik (DI), Fraunhofer IVI, Liebherr Hausgeräte und Heinrichsthaler Milchwerke jeweils folgende Beiträge leisteten.

A Konzeption

Es wurden vom Fraunhofer IVI und DI typische Anwendungsszenarien für die zu entwickelnde Softwarekomponente klassifiziert und deren Eigenschaften hinsichtlich der Themenstellung festgelegt. Unter starker Einbeziehung der assoziierten Partner Liebherr Hausgeräte und Heinrichsthaler Milchwerke wurde der Einsatz des Planungsinstruments bestimmt. Die Spezifikation der systemtechnischen Anforderungen an die zu entwickelnde Softwarekomponente umfasste den Ablauf der dynamischen Planungsprozesse im Echtbetrieb, sowie die Anforderungen an das Echtzeitverhalten. Daraus resultierten die Anforderungen an die Planungsalgorithmen. Besonderheiten der Optimierung unter Ungewissheit und der damit verbundenen Eigenschaften eines dynamischen Planungsprozesses wurden vom Fraunhofer IVI spezifiziert. DI spezifizierte die Integration der zu entwickelnden Softwarekomponente in bestehende IT-System, sowie die technischen und organisatorischen Rahmenbedingungen für den Einsatz von RFID-Technologie.

B Weiterentwicklungen am DiLVS

Dieser Abschnitt wurde von DI bearbeitet. Die Softwarekomponente zur Optimierung wurde in das Lagerverwaltungssystem DiLVS integriert. Dazu notwendige Erweiterungen des bestehenden Systems wurden realisiert. Das Lagerverwaltungssystem DiLVS wurde für die Integration der zu entwickelnden Echtzeit- Optimierungskomponente umfassend umstrukturiert. Das Systems wurde in verschiedene Module, die auf bestimmte Dienste spezialisiert sind, aufgeteilt. Die Schnittstellen wurden verallgemeinert und für die RFID-Integration wurden Funktionen zu Speicherung von Objektinformationen auf dem Transponder realisiert.

C Entwicklung der Optimierungskomponente

Das Hauptergebnis liegt in einer Softwarekomponente zur Optimierung unter Ungewissheit und wurde vom Fraunhofer IVI realisiert. Diese wurde auf dem bestehenden algorithmischen Rahmen, einer Meta-Heuristik, aufgesetzt und in den Einzelschritten an die realen Gegebenheiten der assoziierten Partner angepasst. Diese innerhalb des Meta-Algorithmus arbeitenden Subroutinen für problemspezifische algorithmische Detailfragen wurden implementiert. Der theoretische Entwurf der verwendeten Algorithmen wurde vom Fraunhofer IVI erarbeitet und implementiert. Bei der effizienten Implementierung wurde das Fraunhofer IVI durch DI unterstützt.

D Erstellung von Simulationsmodellen

Dieser Abschnitt wurde vom Fraunhofer IVI bearbeitet. Die Demonstration der Prinzipien von Transport- und Lageroptimierung, ihrer Realisierbarkeit und deren Effekte erfolgt mit Hilfe einer prototypischen Realisierung der Softwarekomponente anhand von Simulationsmodellen. Zur Vorstudie der zu lösenden Problematik bei den Heinrichsthaler Milchwerken und als akademisches Beispiel wurde die Softwarekomponente anhand des 2-Server-Problems getestet und analysiert. Als Vorstudie der Anwendung bei Liebherr Hausgeräte wurde ein einfaches Sackgassenlager simuliert. Ein komplexes Simulationsmodell des Pilotanwenders Liebherr Hausgeräte wurde in enger Zusammenarbeit mit diesem Partner, anhand von realen Produktionsdaten erstellt. Dabei wurden die Rahmenbedingungen für die Pilotanwendung definiert.

E Pilotanwendung

Für die Pilotanwendung der realisierten Prototypen bei dem assoziierten Partner Liebherr Hausgeräte wurden die abzubildenden Fertigungsbereiche, die dazwischenliegenden Lager und Transportmittel, sowie der funktionelle Umfang der Pilotanwendungen festgelegt. Die für den Pilotbetrieb notwendigen Anpassungsprogrammierungen am DiLVS wurden durchgeführt. Es erfolgten, mit Hilfe von Durchführungen, Auswertungen und Evaluierungen von Testläufen der Simulationsmodelle aus AP D.2, mehrere Vor- und Zwischenuntersuchungen. Dabei wurden alle notwendigen Daten für die Pilotanwendung festgelegt. Die Pilotanwendung wurde auf diesem Weg vorbereitet ohne in den laufenden Betrieb einzugreifen. Die eigentliche Pilotanwendung vor Ort wurde kurzfristig von Liebherr Hausgeräte abgesagt. Die Verbundpartner entschieden sich stattdessen, für den Aufbau eines Demonstrators, der die realen Logistikszenarien bei Liebherr Hausgeräte abbildet und eine Simulation der Logistikabläufe mit realen operativen Daten durchführt. Damit soll die hier entwickelte Softwarekomponente getestet werden.

F Projektmanagement

Federführend im Projektverbund ist DI. Bei den Aufgaben des Projektmanagement wie Koordinierung und Leitung der Projektaktivitäten, Überwachung des Projektfortschritts, Kommunikation mit dem Projektträger, Management der Projektmittel und Berichtswesen wurde DI vom Fraunhofer IVI unterstützt.

I.4. Wissenschaftlicher und technischer Stand, an dem angeknüpft wurde

Der aktuelle Stand wird aus Sicht der IT-Planungssysteme und aus Sicht der mathematischen Optimierungsmodelle und -algorithmen beschrieben.

Stand der Technik - Planungssysteme

Softwaresysteme für betriebswirtschaftliche Aufgaben wie die Produktionsplanung und -steuerung oder die Lagerverwaltung sind bei Fertigungsbetrieben in der Regel im Einsatz. Auch automatische und teilautomatische Lager- und Transportsysteme zwischen Fertigungsbereichen sind sehr häufig anzutreffen. Die Steuerung der Regalbediengeräte, Förderbänder, automatischen Staplern und ähnlichen erfolgt zumeist über das Lagerverwaltungssystem. Aus algorithmischer Sicht sind dort regelbasierte Steuerungen implementiert, die zum Beispiel nach Prioritäten, kürzesten Wegen oder anderen einfachen Kriterien Transportvorgänge anstoßen, aber nicht aktiv planen im Sinne von Optimierungsproblemen. Lagervorgänge werden zudem separat in völlig eigenständigen Modulen nach festen Ein- und Auslagerstrategien unabhängig von der umgebenden Fertigung mit ihren aktuellen Transportbedarfen gesteuert. Oftmals sind diese Lagerstrategien in dem vom Anbieter der Lagertechnik mitgelieferten Komponenten gekapselt. Zur Bewertung des Verhaltens von Logistiksystemen in der Produktion, Distribution und Versand und zu deren Optimierung werden in der industriellen Praxis verbreitet Optimierungsmethoden und -werkzeuge eingesetzt, mit deren Hilfe Ergebnisse für ein bestimmtes Auftragsszenario für Fertigungs- oder Transportaufträge berechnet werden. Die im Einsatz befindlichen Softwarepakete für die Fertigungslogistik und Lagerlogistik unterstützen vor allem die logistischen Anforderungen zu Menge und Zeit aus Sicht der Kundenbelieferung. Es wird dabei nach Qualität und Pünktlichkeit der Lieferung bzw. hohem Durchsatz optimiert. Die Grundlage für die daraus resultierenden Umstrukturierungen und Prozessänderungen beruht auf einfachen statischen Modellen oder auch einfach nur der Erfahrung des entsprechenden Logistikplaners. Auf kurzfristige dynamische Einflüsse auf den Bearbeitungsprozess (z.B. der Auslager- oder Bereitstellungsauftrag) von außen kann nur schwerfällig oder überhaupt nicht mehr reagiert werden. RFID-Technologie wird mittlerweile für eine Vielzahl von Anwendungsfällen erfolgreich eingesetzt, vor allem da, wo Kennzeichnung, Codierung, Identifizierung, Sicherung und Authentifizierung von Objekten gefragt ist. Mit der Weiterentwicklung der technischen Möglichkeiten werden auch ständig weitere Einsatzgebiete erschlossen. Von der technischen Seite her sind alle Voraussetzungen gegeben, um auf Basis von RFID die im Vorhaben gestellte Aufgabenstellung der Optimierung von Transport- und Lagervorgängen zu erfüllen.

Stand der Wissenschaft - Modelle und Algorithmen

In der wissenschaftlichen Literatur zum Operations Research zählt das Vehicle Routing Problem (VRP) zu einem der bekanntesten und meistbehandelten anwendungsorientierten Pro-

blemen. Vor allem zu den zahlreichen Varianten des statischen Problemfalls - also unter Voraussetzung vollständiger und unveränderlicher Information - gibt es eine Fülle von Publikationen, in denen Modellaspekte und Lösungsansätze diskutiert werden, zu Übersichten siehe [1,2,3,4]. Die bekannten Arbeiten zu dynamischen und stochastischen VRPs sind zumeist eher der Grundlagenforschung zuzuordnen und beschäftigen sich z.B. mit worst- oder average-case-Analysen von Heuristiken und Näherungsverfahren für sehr eingeschränkten akademischen Problemfälle [5,6,7,8,9] oder mit exakten Lösungsverfahren durch Dynamische Programme [10,11], die für praxisrelevante Problemdimensionen jedoch nicht geeignet sind. Erst seit kurzer Zeit wird versucht, echtzeitfähige Algorithmen für echte Praxisanwendungen zu entwerfen. Erfolgversprechend sind dabei auch Ansätze, bei denen Ideen aus der Spielbaumsuche und Monte-Carlo-Verfahren auf Optimierungsprobleme unter Ungewissheit übertragen werden. Beispiele hierzu finden sich für Anwendungsgebiete wie der Flugzeugeinsatzplanung [12] oder für verschiedene Problemvarianten in der Produktionsplanung [13,14]. Auf dem Gebiet der Anwendung von Monte-Carlo-Verfahren zur Lösung praxisnaher VRPs und verwandter Problemstellungen ist das Fraunhofer IVI sehr aktiv [15,16,17]. Insbesondere die Entwicklung und Untersuchung sogenannter Offline- Monte-Carlo-Verfahren, die bei starken Echtzeitanforderungen ihren Einsatz finden, wird am Institut vorangetrieben [18]. Soweit es uns bekannt ist, sind im Umfeld der VRP-Literatur aus Informatik und Mathematik keine speziellen Modelle für innerbetriebliche Transporte mit dortigen speziellen Restriktionen sowie einer Integration der Lagerplatzbelegung betrachtet worden. Dementsprechend fehlen auch noch spezielle Algorithmen, die Problem als mathematisches Optimierungsproblem - noch dazu unter Ungewissheit - auffassen.

I.5. Zusammenarbeit mit anderen Stellen

Im Zuge der Abwicklung des Vorhabens erfolgte eine Zusammenarbeit überwiegend mit den beteiligten Projektpartnern. Die Möglichkeit des Erfahrungsaustausches mit anderen Forschungsstellen und Softwareunternehmen wurde auf fachspezifischen Veranstaltungen wahrgenommen. Das Fraunhofer IVI stellte die Ergebnisse des Projekts auf der „8th Scientific Conference Economy and Efficiency – contemporary solutions in logistics and production OIE 2013“ in Poznan, Polen vor.

II. Detaillierte Darstellung

II.1. Eingehende Darstellung des erzielten Ergebnisses

In diesem Abschnitt werden die wichtigsten Ergebnisse des Vorhabens aus Sicht des Fraunhofer IVI dargestellt. Es erfolgt eine Gliederung der Ergebnisse nach den bearbeiteten Arbeitspaketen gemäß Antrag.

— A Konzeption —

Zu Beginn wurde in drei Teilschritten ein Konzept für die Bearbeitung des Projekts erstellt.

A.1 Modellspezifikation

Dresden Informatik hat die Anwendungsszenarien spezifiziert. Dabei wurde in Zusammenarbeit mit dem Fraunhofer IVI auf den besonderen Schwerpunkt der Optimierung unter Ungewissheit und der damit verbundenen Eigenschaften eines dynamischen Planungsprozesses geachtet.

Beim ersten Anwendungsfall, Liebherr Hausgeräte, handelt es sich um die Steuerung eines Zwischenlagers für Türenwagen. In der Türen-Schäumerei werden Türen für Kühlgeräte gefertigt und für die Weiterverarbeitung in der Endmontage bereitgestellt. Auch wenn sich das Produktionsprogramm der Türen-Schäumerei an dem der Endmontage orientiert, sind beide Fertigungsbereiche doch zeitlich voneinander entkoppelt, was durch ein Zwischenlager realisiert wird. Aus Platzgründen handelt es sich dabei um ein Blocklager, was zwangsläufig zu Umlagerungen bei einer Auslagerung führen kann. Da die Reihenfolge der Fertigung in der Endmontage unbekannt ist und sich auch kurzfristig ändern kann, enthält die Reihenfolge der Auslagerungen aus dem Zwischenlager den Aspekt der Ungewissheit. Um die Türen auf einem Türen-Wagen identifizieren zu können, wurde auf RFID Technologie zurückgegriffen.

Die Heinrichsthaler Milchwerke stellen den zweiten Anwendungsfall dar. Während der Käseherstellung werden die Schnittkäseblöcke zur Reifung in Reifekisten gepackt und in einem Blocklager im LIFO-Prinzip organisiert. Die Reifungsdauer ist je nach Artikel und Qualität unterschiedlich lang und liegt zwischen 2 bis 8 Wochen. Zielsetzung war eine chaotische, nicht sortenreine Einlagerung der Kisten um die Anzahl der Umlagervorgänge möglichst zu minimieren. Dazu ist eine genaue Lokalisierung der Kisten und das Wissen in welcher Kiste welches Produkt gelagert ist notwendig. Um diese Informationen bereitzustellen, wurde auf die RFID Technologie zurückgegriffen. Zusätzlich sollte eine Wegeoptimierung bei den Ein- und Auslagerung unter Berücksichtigung der zukünftigen Transporte gemäß der Vorschau auf die Kundenaufträge und der Vorschau auf die Fertigungsaufträge erfolgen. Die Lösung dieser Aufgaben zur Lagerplatz- und Wegeoptimierung einschließlich der Bestandsoptimierung war mit bisherigen Lösungsansätzen nicht realisierbar.

Eine detaillierte mathematische Modellbeschreibung, die sich aus der Problem-Spezifikation ableitet, sowie eine erste Spezifikation der Ungewissheit kann im Anhang in den Kapiteln 1 und 2 nachgelesen werden.

A.2 Spezifikation von Systemeigenschaften

In diesem schwerpunktmäßig durch DI bearbeiteten Paket wurden die systemtechnischen Anforderungen an die zu entwickelnde Softwarekomponente spezifiziert und die sicherzustel-

lenden Funktionalitäten festgelegt. Zu diesen spezifizierenden Funktionalitäten zählen:

- Zugriffskontrolle - sicherheitstechnische Lösungen
- Notwendige Programmkonzeptänderungen im LVS
- Wiederverwendbarkeit des Datenmodells - Modulkonzept für die LVS-Architektur
- Technologie - Umstellung auf aktuelle Entwicklungsframeworks
- Intelligenz Bestandssuche und Platzsuche - Wegeoptimierung durch Bestandsreservierung

Bei der Bearbeitung des letzten Punktes wurde das Fraunhofer IVI mit einbezogen. Die algorithmische Realisierbarkeit der Spezifikation wurde geprüft. Des Weiteren wurden die Anforderungen an das Echtzeitverhalten festgelegt. Dies führte zu Einschränkungen im algorithmischen Rahmen der Optimierungskomponente, aber auch bei der Datenübertragung zwischen dem LVS und der Optimierungskomponente vom Fraunhofer IVI, um die Echtzeitfähigkeit zu garantieren.

A.3 Einsatz und Nutzung von RFID

Die technischen und organisatorischen Rahmenbedingungen für den Einsatz von RFID Technologie wurden von DI erarbeitet. Das Fraunhofer IVI wurde zur Absicherung der algorithmischen Nebenbedingungen mit einbezogen. Es wurde festgelegt welche Informationen die Optimierungskomponente mit Hilfe der RFID Technik erhält.

Bei den Heinrichsthaler Milchwerken (HMW) wurden Transponder eingesetzt, um zwischen eigenen Reifungskisten, Reifungskisten aus Zukauf und Paletten mit zugekaufter Ware unterscheiden zu können. Der normale Eingang der Kisten ins Lager wird im Allgemeinen sicher abgebildet. Bei der Auslagerung und Übergabe der Reifungskisten an die Abpackung (Produktion) wird jede Reifungskiste sicher aus dem Lagerbestand ausgebucht. Dazu gibt es an der Abpackung einen „Briefkasten“ mit einem fest installierten UHF-Reader, in den die Transponder-Pads der ausgelagerten Kisten oder Paletten einzuwerfen sind. Der Reader liest dabei die Pad-Nr. aus dem EPC-Code des Transponders aus und sammelt die Daten. Eine Kopplung mit dem Lagerverwaltungssystem erfolgte.

Daten die innerhalb der Optimierungskomponente benötigt werden:

- Kistennummer
- Kistenart
- Artikelanzahl
- Zeitpunkt der Einlagerung

Bei Liebherr Haushaltsgeräte wurde jeder Türenwagen mit einem Transponder ausgestattet. Die Türenwagen aus der Schäumerei werden an einem RFID-Gate erfasst und mit der Artikelnummer und der Anzahl der Türen auf dem Wagen verknüpft. Für Ein- und Auslagerungen ist eine einzige Person zuständig, die alle relevanten Informationen an einem Terminal

der Bereitstellungsfläche erhält. Durch scannen eines neuen Wagens wird ein entsprechender Platz im Lager angegeben, sowie die Information übermittelt welcher Wagen auf diesem Weg mit ausgelagert werden soll. Es wurde von DI die Erkennung der Türenwagen bei der Durchfahrt mittels RFID Technik getestet. Es wurden mögliche Befestigungspositionen für die Transponder ermittelt, sodass die Wagen hundertprozentig bei der Durchfahrt erkannt werden und gleichzeitig problemlos mittels Handheld gelesen werden können. Mit einer Antenne die mittig über dem Fahrweg angebracht wurde, werden die Türenwagen mittels RFID Technik sicher erkannt.

Daten die innerhalb der Optimierungskomponente benötigt werden:

- Türenwagennummer
- Artikelnummern
- Artikelanzahl
- Zeitpunkt der Durchfahrt am RFID Gate

A.4 Algorithmische Methodik

Angesichts der Komplexität der Probleme können nur heuristische Lösungsverfahren zum Einsatz kommen. Es soll dabei auf eine einfache Basisheuristik zurückgegriffen werden, die anschließend mit Hilfe einer Meta-Heuristik verbessert werden soll. Als Meta-Heuristik wird ein Monte-Carlo-Rollout Ansatz (Mc-Ro) verfolgt. Dies ist ein Lösungsverfahren für Planungsprozesse oder allgemein für kombinatorische Optimierungsprobleme sequentieller Art, bei denen wiederholend die folgenden beiden Schritte abwechselnd durchgeführt werden: Einem Entscheider (Planer/Disponent) liegt eine aktuelle Instanz des Planungsproblems vor, es ist eine Entscheidung zu treffen, die zu einer neuen Lösung für das Problem führt. Beim Abarbeiten der vorliegenden Lösung werden durch die Umwelt Abweichungen vom geplanten Ablauf erzeugt (in Form von Störungen, Bekanntwerden neuer Informationen usw.). Diese machen eine erneute Reaktion des Entscheiders notwendig. Die Lösung muss entsprechend angepasst werden. Das Grundprinzip des Monte-Carlo-Rollout-Verfahrens ist wie folgt: Für eine Entscheidung existiert eine einfache Heuristik H, die zunächst nur „kurzsichtig“ nach Lösungen für die aktuelle Planungsinstanz mit aktuellen Informationen sucht. Mit Hilfe des Monte-Carlo-Rollout-Verfahren soll diese Entscheidung auf „längere Sicht“ geprüft werden. Unter Einbeziehung von Vorausschau auf zukünftig mögliche Konsequenzen einer Entscheidung wird jede mögliche Entscheidung getestet. Die dabei am besten bewertete Entscheidung wird ausgewählt. Die Vorausschau erfolgt durch Zufallsspiele in denen der Entscheider immer die Heuristik H verwendet. Dem Zufallsspieler liegt ein stochastisches Modell zugrunde, welches die gegebenen Unsicherheiten wiederspiegelt. Eine ausführliche Beschreibung des Verfahrens befindet sich im Anhang in Kapitel 3.

— B Weiterentwicklungen am DiLVS —

Dieses Arbeitspaket wurde ausschließlich durch DI bearbeitet.

— C Entwicklung der Optimierungskomponente —

Das Arbeitspaket C bildete im Vorhaben den Schwerpunkt der Tätigkeit des Fraunhofer IVI. Der theoretische Entwurf der verwendeten Algorithmen war Schwerpunkt der Arbeit des Fraunhofer IVI. Die effiziente Implementierung erfolgte in Zusammenarbeit mit DI.

C.1 Entwurf des Optimierungsalgorithmus

Für kombinatorische Optimierungsprobleme sequentieller Art ist der Wechsel zwischen Zufallsspieler und Entscheider zu simulieren. Dazu dient folgende Zeitschrittschleife:

1. *Zug-Generator für die Umwelt*:
   Gemäß eines Wahrscheinlichkeitsmodells mit Maß $\mathcal{P}$ wird die aktuelle Instanz Π zu Π' verändert. Das Wahrscheinlichkeitsmodell soll dabei die gegebene Ungewissheit möglichst genau wiederspiegeln.

2. *Zug-Generator für den Entscheider*:
   Mit Hilfe des Monte-Carlo-Rollout-Verfahrens $mcr.solver(\Pi, H)$ wird eine Lösung für die Instanz Π' ermittelt. Diesem wird die aktuelle Instanz Π' und die zu benutzende Heuristik H übergeben.

3. Die Lösung wird abgearbeitet bis eine erneute Änderung durch die Umwelt erfolgt. Dabei wird eine geeignete Kostenfunktion aufdatiert.

Im Zug-Generator für den Entscheider wird das Monte-Carlo-Rollout-Verfahren verwendet, welches im Detail wie folgt aussieht:

1. *Generation der Alternativen*:
   Erzeugen einer endliche Menge von Kandidatenlösungen $S = \{S_1, S_2, \ldots\}$ für Π'.

2. *für jede Kanidatenlösung* S_i:
   a) *Erzeugung der Folgeinstanz* Π'' *die durch die Lösung* S_i *entsteht*
   b) *width-malige Durchführung eines MCR*:
      Es werden ausgehend von der Instanz Π'' *depth* Schritte analog zur Zeitschrittschleife simuliert. Nur der Zug-Generator für den Entscheider unterscheidet sich von obiger Schleife. Anstelle von $mcr.solver(\Pi, H)$ wird hier nur die Heuristik H angewendet.
   c) *Bewertung:*
      Die in den einzelnen MCR generierten Bewertungen werden gemittelt und ergeben die Lösung $S_i^{(N)}$ zur Kanidatenlösung S_i.

3. *Entscheidung:*
   Es wird für die aktuelle Entscheidung die Stellfläche ausgewählt, die zum besten Ergebnis $S^* = \min\{S_1^{(N)}, \ldots, S_k^{(N)}\}$ führt.

Dieses Grundmodell wird mit Hilfe von Subroutinen an die einzelnen Problemstellungen angepasst:

- *Basisheuristik*
  Eine einfache, nicht zu komplexe Heuristik die zur Lösung einer Instanz Π benutzt wird.
- *Zug-Generator für die Umwelt*
  Wahrscheinlichkeitsmodell welches für die Instanz Π ein zukünftiges Szenario generiert.
- *Kostenberechnung*
  Eine geeignete Kostenfunktion bewertet eine Instanz Π oder einen Schritt bei der Abarbeitung der Lösung.
- *Generation der Alternativen*
  Für Π muss einer endliche Menge von Kandidatenlösungen S ermittelt werden.
- *Daten Aktualisierung*
  Nach jedem Schritt innerhalb des Monte-Carlo-Rollout-Verfahrens müssen alle dazugehörigen Daten aktualisiert werden.

Diese grobe Zusammenfassung der Methodik für Optimierungsprobleme unter Ungewissheit wird im Anhang in Kapitel 4 detailliert diskutiert und dargestellt. Die entsprechenden Prototypen wurden vom Fraunhofer IVI erstellt.

C.2 Implementierung des Meta-Algorithmus

Das Fraunhofer IVI hat den Quellcode für den Meta-Algorithmus in der Programmiersprache Java aufgearbeitet. Geeignete Datenstrukturen und die Schnittstellen zum LVS wurden in Zusammenarbeit mit DI festgelegt. Da beide Komponenten auf Java basieren, wurde eine gemeinsame Datenstruktur eingeführt und die Übergabeparameter festgelegt.

Die notwendigen Klassen wurden wie folgt strukturiert:

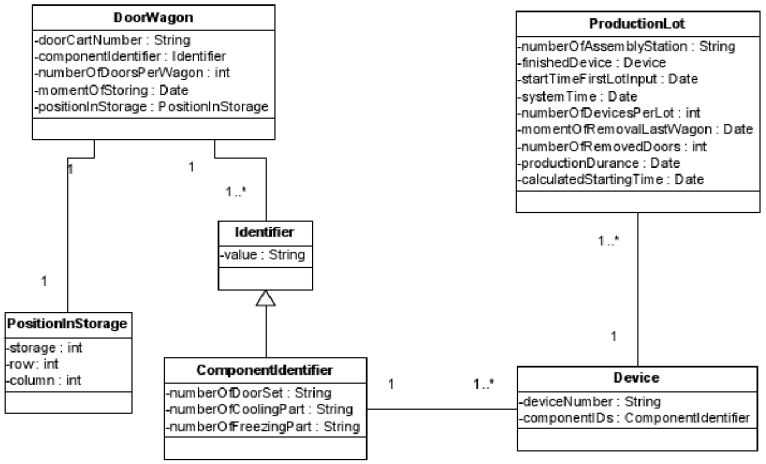

Abbildung 1: Gemeinsam definierte Schnittstellenklassen.

Das LVS ruft die Optimierungskomponente durch eine Funktion zur Ein- oder Auslagerung auf. Dabei übergibt das LVS bei jedem Aufruf:

- Eine Liste die das aktuelle Lager abbildet:
  ArrayList<DoorWagon> stock;
- Eine Liste der nächsten Lagerausgänge, die Reihenfolge ist dabei ungewiss:
  ArrayList<ProductionLot> removals;
- Eine Objekt das neu eingelagert werden soll:
  DoorWagon storage;

Innerhalb der Optimierungskomponente werden alle relevanten Daten und zugehörigen Funktionen in einer Klasse *Instance* gespeichert und verarbeitet. Dazu gehören auch alle problemspezifischen Subroutinen aus Paket C.1. Es gibt zwei Löser: *Solver* der die Heuristik anwendet und *MC-Solver* der das Monte-Carlo-Rollout-Verfahren steuert, welches wiederum den *Solver* benutzt. An die Löser wird nur die *Instance* übergeben. Rückgabewert von der Optimierungskomponente an das LVS ist eine exakte Position im Lager bzw. zwei Positionen, wenn eine Ein- und Auslagerung in Kombination erfolgt.

C.3 Vorbereitung der Subroutinen

Die innerhalb des Meta-Algorithmus arbeitenden Subroutinen für problemspezifische algorithmische Detailfragen wurden zunächst so strukturiert, dass die insgesamt entstehende Softwarekomponente auf spezifische Anwendungen hin anpassungsfähig bleibt. Struktur der einzelnen Subroutinen:

- *Daten Aktualisierung*
  Alle notwendigen Daten und Methoden befinden sich in der Klasse *Instance*. Die Daten, z.B. der aktuelle Lagerbestand, müssen nach jedem Schritt innerhalb eines MC-Ro aktualisiert werden.
- *Basisheuristik*
  Dieser Funktion wird die Instanz übergeben, die berechnete Lösung wird als eine Variable des *Solver* gespeichert.
- *Zug-Generator für die Umwelt*
  Dieser Funktion wird die Liste der Daten übergeben die einer Ungewissheit unterliegen. Die Liste wird dem Wahrscheinlichkeitsmodell entsprechend gegebenenfalls verändert und zurückgegeben.
- *Kostenberechnung*
  Die Kosten für eine Lösung werden vom *Solver* mitübergeben und aufdatiert.
- *Generation der Alternativen*
  Dieser Funktion wird die Instanz, die Heuristik und eine Anzahl von gewünschten Kanidatenlösungen übergeben. Die Heuristik wird mehrfach, der übergebenen Anzahl entsprechend oft, auf die Instanz angewandt. Eine dabei erzeugte Lösung wird für den nächsten Aufruf der Heuristik gesperrt. Zurückgegeben wird eine endliche Menge von Lösungen die weiter untersucht werden soll.

— D Erstellung von Simulationsmodellen —

Die Demonstration der Prinzipien von Transport- und Lageroptimierung, ihrer Realisierbarkeit und deren Effekte erfolgte mit Hilfe einer prototypischen Realisierung der Softwarekomponente anhand von Simulationsmodellen. Hierzu wurden zunächst zwei kleinere Simulationsmodelle erarbeitet. Anschließend wurde ein komplexes Simulationsmodell für die Pilotanwendung bei Liebherr Hausgeräte erstellt. Dieses wurde für Voruntersuchungen der Pilotanwendung verwendet. In der ursprünglichen Projektplanung war eine Pilotanwendung auch bei den Heinrichsthaler Milchwerken vorgesehen. Im Laufe der Feinplanung zur Spezifikation zeigte sich allerdings, dass für eine erfolgreiche Pilotanwendung die fachlichen Voraussetzungen in der Logistikaufgabe nicht ausreichend waren. Daher wurde von den Verbundpartnern der Schwerpunkt der Pilotanwendung auf Liebherr Haushaltgeräte gelegt. Auf das komplexe Simulationsmodell für den Anwendungsfall der Heinrichsthaler Milchwerken wurde daher verzichtet.

D.1 Simulationsmodelle für Test- und Demonstrationszwecke

In einem ersten Modell wurde das 2-Server-Problem behandelt. Dieses Modell diente der Vorstudie für den Anwendungsfall der Heinrichsthaler Milchwerke. Hierbei muss die Bewegung von 2 Servern in einem abstrakten metrischen Raum gesteuert werden. Dabei müssen die Server Anfragen bearbeiten die an jedem Punkt des metrischen Raumes auftauchen können und im Voraus unbekannt sind. Um dieses Problem an das eigentliche Problem anzupassen, soll hier nicht davon ausgegangen werden, dass alle Anfragen komplett unbekannt sind. Vielmehr soll eine Startinstanz von bekannten Anfragen gegeben sein, die während der Abarbeitung gestört wird. Aufgrund der eingeschränkten Komplexität kann eine große Anzahl an Berechnungen durchführt werden. Diese ist notwendig für eine Analyse der Heuristiken und deren Verbesserung durch die Metaheuristik. Ein weiterer Vorteil der Betrachtung eines akademischen Beispiels ergibt sich aus der Kenntnis der optimalen online Lösung, welche aus der offline Lösung hergeleitet werden kann. Diese optimale online Lösung bildet die Grundlage für die Analyse. Alle theoretischen Aspekte werden in der Anlage Kapitel 4.2 betrachtet. Die Ergebnisse und deren Analyse befindet sich in Kapitel 5.1.

Als zweites Anwendungsszenario wurde ein Sackgassenlager mit 25 Reihen und jeweils 5 Stellplätzen betrachtet. Die Ein- und Auslagerungen wurden zufällig erzeugt. Über 100 Zeitschritte erfolgt jeweils eine Auslagerung und eine Einlagerung. Die Artikel die ein- bzw. ausgelagert werden sollen, wurden anfangs gleichmäßig verteilt angenommen. In Anlehnung an die Gegebenheiten bei Liebherr Hausgeräte wurde auch eine Verteilung bei der 40% der Artikel 80% des Tagesumsatzes ausmachen benutzt. In Anlehnung an die Pilotanwendung wurde die Ungewissheit durch Permutationen in der Liste der Auslagerungen umgesetzt. Es wurden verschiedene Heuristiken getestet und die Anzahl der notwendigen Umlagerungen (UL) verglichen. Die detaillierte Beschreibung befindet sich im Anhang Kapitel 5.2. Als Ergebnis dieser Untersuchung erfolgte die Festlegung der Heuristiken für das Simulationsmodell für Liebherr Hausgeräte. Es soll eine Heuristik verwendet werden die das Lager gleichmäßig auffüllt. Diese ist schnell und liefert mit Hilfe des Monte-Carlo-Rollout-Verfahrens gute Ergebnisse. In einer zweiten Heuristik sollen Informationen über die Zukunft mit einbezogen werden. Hier wird bei einer Einlagerung darauf geachtet, welche Reihen bei den nächsten Auslagerungen benötigt werden. Diese werden für die Einlagerung ausgeschlossen. Die Rechenzeiten sind erheblich höher im Vergleich zur ersten Heuristik. Eine Anwendbarkeit im Zusammenhang mit

dem Monte-Carlo-Rollout-Verfahren, bei welchen diese Heuristik sehr oft aufgerufen wird, soll geprüft werden.

D.2 Simulationsmodell für Pilotanwendung

Entsprechend der Modellspezifikation in Arbeitspaket A.1 wurde ein Simulationsmodell für den Anwendungsfall Liebherr entwickelt. Dazu wurden die Subroutinen für die Optimierungskomponente entsprechend Arbeitspaket C angepasst. Der detaillierte Ablauf von Einlagerungen und Auslagerungen wird im Anhang in Kapitel 6.2 beschrieben. Die Subroutinen sehen wie folgt aus:

- *Basisheuristik* $planned_{in}(x)$
  Bei der Bestimmung der Position für eine Einlagerung sollen die nächsten x zukünftig geplanten Auslagerungen berücksichtigt werden und ggf. nicht blockiert werden. Für $x = 0$ wird das Lager unabhängig von kommenden Auslagerungen gleichmäßig gefüllt.

- *Zug-Generator für die Umwelt*
  Die geplante Reihenfolge der Auslagerungen ergibt sich aus den Produktionszeiten der einzelnen Montageinseln. Eine Abweichung von den durchschnittlichen Produktionszeiten wird für jede Montageinsel m und jede Schicht s

  $$\alpha_{ms} \in (1 - \beta, 1 + \beta)$$

  zufällig erzeugt. β ist der Prozentsatz um den die realen Produktionszeiten von den durchschnittlichen Produktionszeiten maximal abweichen kann. Detaillierte Beschreibung der Ungewissheit befindet sich im Anhang in Kapitel 6.4.

- *Kostenberechnung*
  Die notwendiger Umlagerungen bei Auslagerungen stellen die Kosten dar.

- *Daten Aktualisierung*
  Lagerabbild, Auslagerliste, Einlagerliste werden aktualisiert.

- *Generation der Alternativen*
  Alle zur Verfügung stehenden Plätze im Lager sollen betrachtet werden.

In der Produktion soll die Lagerfläche für Einbau- und Standgeräte getrennt werden. Dadurch ergeben sich für die Simulation der gesamten Produktion zwei Teilsimulationen die unabhängig voneinander sind. Für die Standgeräte soll Lagerfläche 1 verwendet werden. Diese umfasst 33 Reihen mit 3 Türenwagen und 31 Reihen mit 2 Wagen hintereinander. Für Türen der Einbaugeräte ist Lagerfläche 2 vorgesehen: 14 Reihen mit 4 Wagen und jeweils 7 Reihen mit 5,6 und 8 Wagen.

Für die Simulation wurden reale Daten der Produktion vom 4.5.2011-9.5.2011 verwendet. Die Einlagerungen ergeben sich aus dem Produktionsplan der Türenschäumerei, die Auslagerungen aus der den Montagebändern zugeordneten Losproduktionen. Der Simulationszeitraum umfasst 3 Tage. Eine detaillierte Beschreibung der verwendeten Daten befindet sich im Anhang Kapitel 6.

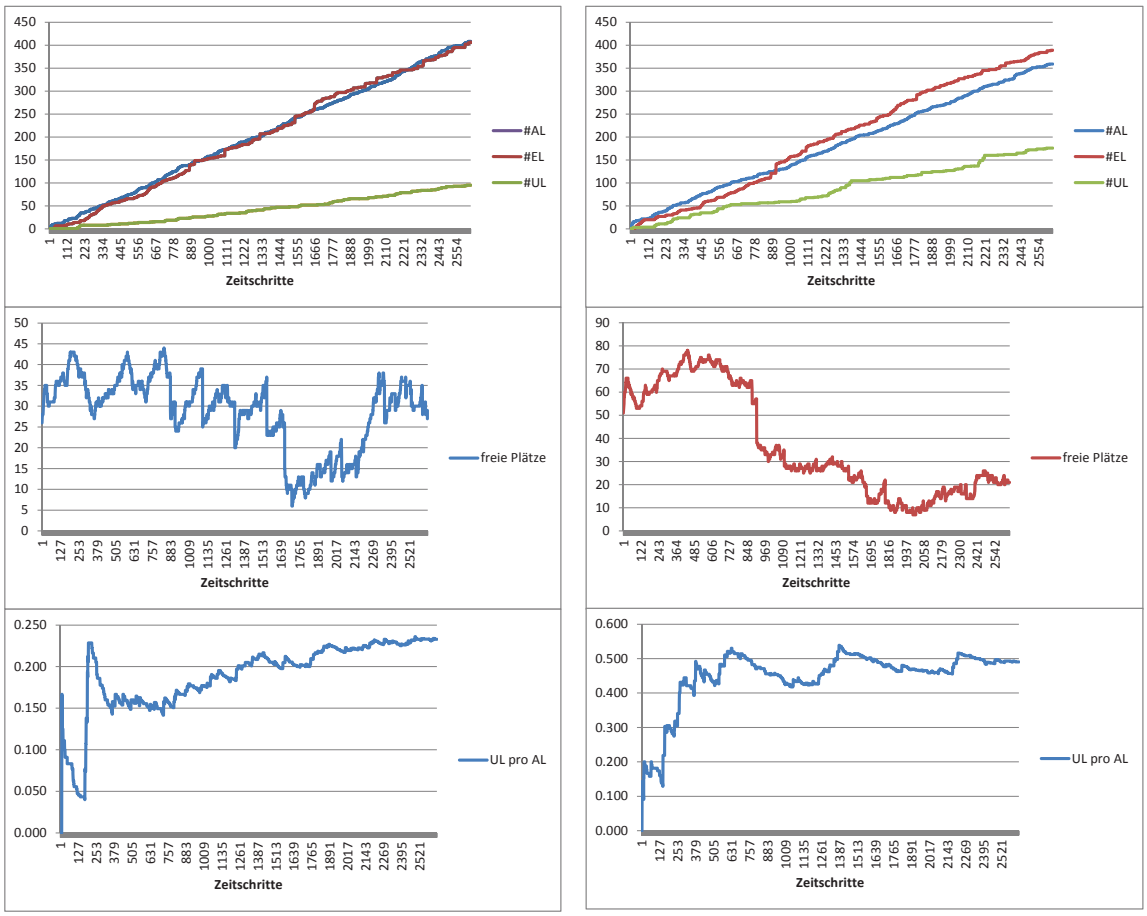

Abbildung 2: Lager im zeitlichen Verlauf - Lager 1 (links), Lager 2 (rechts)

In Abbildung 2 sind die Ergebnisse der Simulation dargestellt. Auf der linken Seite sind die Daten von Lager 1 (Lager für alle Standgeräte), auf der rechten Seite die Daten von Lager 2 (Lager für alle Einbaugräte) abgebildet. Im oberen Diagramm wird im zeitlichen Verlauf die Anzahl der Auslagerungen (AL), Einlagerungen (EL) und Umlagerungen (UL) dargestellt. Die Anzahl der freien Plätze im jeweiligen Lager ist im mittleren Diagramm verdeutlicht. Das untere Diagramm spiegelt das Ergebnis wieder wie viele Umlagerungen pro Auslagerung notwendig sind. Dabei ist zu berücksichtigen, dass zur Vereinfachung der Simulation für jedes Lager 30 zusätzliche Plätze angenommen wurden. Diese sollen Wagen fassen, die in der Realität noch in einer Baugruppe zur Montage stehen oder auf der Bereitstellungsfläche stehen bleiben können, wenn das Lager voll ist.

Die Reduktion der Anzahl von Umlagerungen die mit Hilfe der Meta-Heuristik erzielt wurde, wird in Abbildung (29) dargestellt. Die maximale Abweichung von den durchschnittlichen Produktionszeiten wurde dabei variiert und die Anzahl der notwendigen Umlagerungen unter Benutzung zweier Basisheuristiken wurde über viele verschiedene Szenarien gemittelt. Die Heuristik $planned_{in}(5)$, die mit Informationen über die nächsten 5 zukünftige Auslagerungen arbeitet, erzielt dabei deutlich bessere Ergebnisse als die Heuristik $short_{in} = planned_{in}(0)$. Als Ergebnis wird deutlich, dass mit $planned_{in}(5)$ in Kombination mit dem Monte-Carlo-Rollout-Verfahren die Anzahl der Umlagerungen pro Auslagerung am geringsten ist, unabhängig davon

welche Abweichung in der Realität vorliegt. Die dabei notwendige höhere Rechenzeit liegt in dem Maß, dass eine echtzeitfähige Anwendung möglich ist.

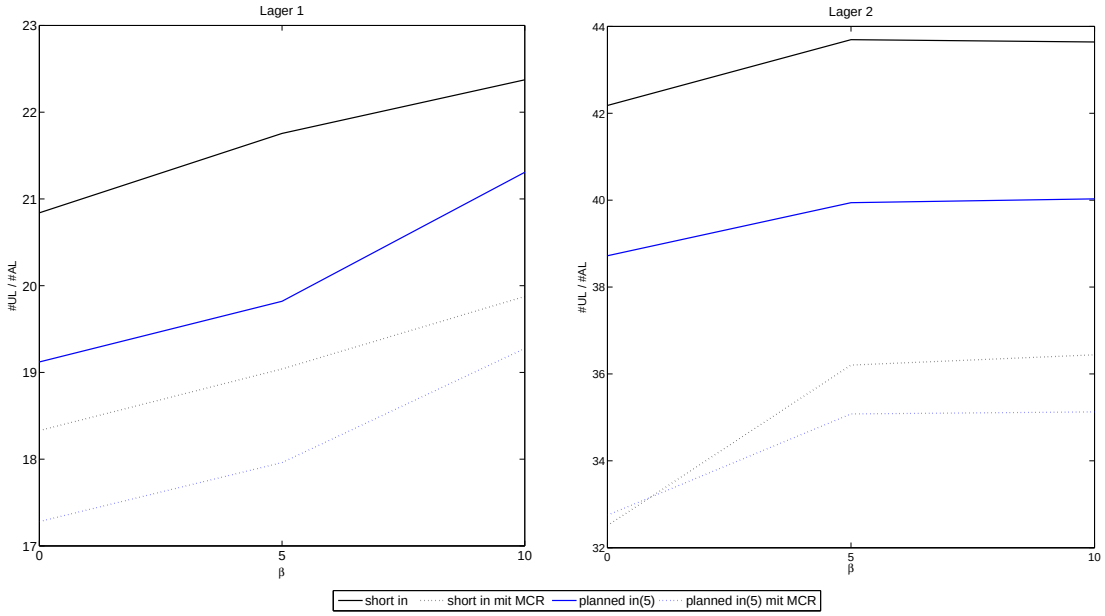

Abbildung 3: UL/AL unter Verwendung der Basisheuristiken und der Meta-Heuristik für verschieden β.

Abbildung (4) zeigt die Ergebnisse der anschließenden Untersuchung, wie viele zukünftige Auslagerungen x in der Heuristik $planned_{in}(x)$ berücksichtigt werden sollten. Es wird deutlich, dass mit zunehmender Anzahl x die Anzahl der notwendigen Umlagerungen pro Auslagerung immer geringer wird. Die Verbesserung ab $x = 12$ aber immer kleiner wird.

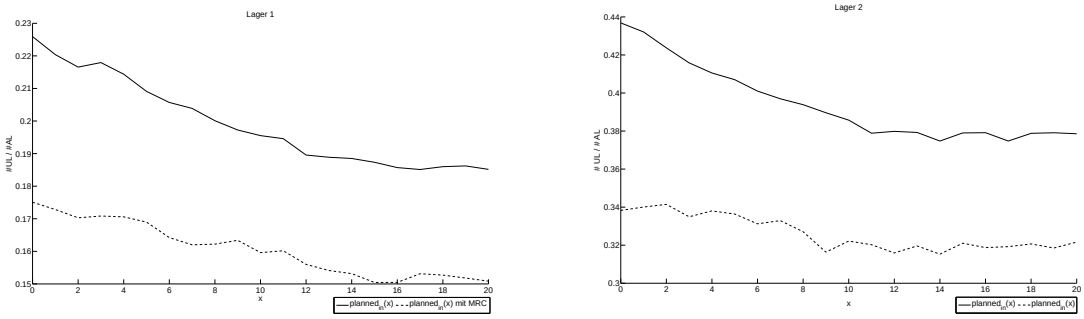

Abbildung 4: Basisheuristik $planned_{in}(x)$ unter der Berücksichtigung verschiedener Anzahl von zukünftigen AL x.

E Pilotanwendung

In der Endphase des Projektes war durch die operative Belastung durch Kundenprojekte beim Pilotanwender Liebherr ein personeller Engpass aufgetreten, der eine Verschiebung der Pilotanwendung und somit des Projektendes notwendig machte. Es wurde eine Pilotanwendung im Frühjahr 2013 geplant.

E.1 Spezifikation

Für die Pilotanwendung des realisierten Prototypen bei dem assoziierten Partner Liebherr Hausgeräte wurden die im Pilotbetrieb abzubildenden Fertigungsbereiche, den dazwischenliegenden Lagern und Transportmittel ebenso wie der funktionelle Umfang der Pilotanwendungen festgelegt. Eine erste Festlegung erfolgte zu Beginn des Projekts und wurde im Laufe der Zeit weiter angepasst. Vor- und Zwischenuntersuchungen anhand der Simulationsmodelle aus AP D.2 wurden Liebherr vorgestellt und die Umsetzbarkeit diskutiert. Um den Aufwand einzuschränken, wurde sich darauf verständigt, nur das Lager 1 und damit die Türen für Standgeräte zu betrachten. Türenwagen für Einbaugeräte sollen weiter wie bisher auf freien Flächen gelagert werden. Daher werden von dem hier entwickelten System nur die Türenwagen für Standgeräte gebucht, ein- und ausgelagert. Es wurden mehrere Arbeitstreffen mit dem Pilotanwender durchgeführt. Dabei wurden auch folgende Punkte diskutiert

- Auf welche Weise werden die Türenwagen mit Wagennummern versehen. Diese sollen gut lesbar sein, auch wenn der Wagen weiter hinten in einer Lagerreihe steht. Laminierte Schilder sollen oben an den Türenwagen befestigt werden.
- Welche Befestigung der Transponder kommt in Frage. Diese soll für den Piloten nicht zu aufwendig sein (ohne Bohrungen), aber die Reinigung der Türenwagen unbeschadet überstehen. Dabei wurde sich auf Panzertape geeinigt.
- Genaue Maße und Umsetzung des Lagers wurden definiert. Die Umsetzung wurde von Liebherr geplant.
- Dem Lagerverwalter soll an einer Computerstation mitgeteilt werden, wohin ein Wagen eingelagert werden soll und welchen Wagen er auf diesem Weg auslagern soll. Beide Daten sollen ausgedruckt werden.
- Aufbau des RFID-Gate und der Standort des dazugehörigen Computers mit Handlesegerät wurde definiert und getestet.
- Die geplante Kopplung zwischen der Planung der Schäumerei und der Optimierungskomponente konnte durch die Gegebenheiten vor Ort von Liebherr nicht umgesetzt werden. Damit geht die Information über zukünftige Einlagerungen verloren. Der Algorithmus musste entsprechend angepasst werden.
- Mit Hilfe eines Tools werden die geplanten Produktionslose der einzelnen Montageinseln an die Optimierungskomponente weitere gegeben. Dieses Tool wird von DI bereitgestellt.

E.2 Anpassungen

Die für den Pilotbetrieb notwendigen Anpassungsprogrammierungen am DiLVS wurden von DI durchgeführt. Die Implementierung der Subroutinen für algorithmische Detailfragen im

Optimierungskonzept wurden vom Fraunhofer IVI ausgeführt und DI bereit gestellt. An die Tatsache das zukünftige Einlagerungen nicht mehr bekannt sind musste die Optimierungskomponente angepasst werden. Innerhalb eines Monte-Carlo-Rollouts werden nur noch zukünftige Auslagerungen und deren Reihenfolge betrachtet. Einlagerungen erfolgen innerhalb eines Monte-Carlo-Rollouts nicht. Beide Komponenten wurden miteinander gekoppelt und getestet.

Während dieser abschließenden Arbeiten hat die Firma Liebherr uns mitgeteilt, dass sie keine freien Kapazitäten für die Durchführung des Pilotbetriebs aufbringen können. Auch hat sich im Verlauf der Modelleinführung bei Liebherr herausgestellt, dass die geplante Lagerfläche vorerst nicht zur Verfügung steht. Die Verbundpartner haben sich stattdessen für den Aufbau eines Demonstrators entschieden, der die realen Logistikszenarien bei Liebherr Hausgeräte abbildet und eine ausführlichere Simulation der Logistikabläufe mit realen operativen Daten durchführt. Die Schnittstelle zum LVS entspricht dem Echtzeitbetrieb und somit der ursprünglichen Aufgabenstellung und Planung.

E.3 Durchführung, Auswertung und Evaluierung von Testläufen

Im Rahmen der weiterführenden Simulation hat das Fraunhofer IVI die Produktion über einen Zeitraum von 7 Tagen ausgewertet. Für die Simulation wurden reale Daten aus der Produktion verwendet. Die produzierten Produktionslose wurden in Teillose zerlegt und auf die Montageinseln aufgeteilt. Die Lagereingänge ergeben sich ebenfalls aus dem Produktionsplan. Türen für die zu produzierenden Lose werden einen Tag im Voraus eingelagert. In einer ersten Untersuchung wurde gezeigt, welche Lagervariante mit den bisher angenommenen Lagerflächen für die Türenwagen den geringsten Aufwand für den Lagerverwalter bedeutet. In den folgenden Abbildungen wird die Anzahl von Auslagerungen, Einlagerungen und Umlagerungen, sowie die Anzahl an freien bzw. fehlenden Plätzen in den einzelnen Lagern gezeigt.

Abbild (5) zeigt die Variante in der Türen für Einbaugeräte in Lager 1 und Türen für Standgeräte in Lager 2 gelagert werden. Abbild (6) zeigt die Variante in der Türen für Standgeräte in Lager 1 und Türen für Einbaugeräte in Lager 2 gelagert werden. Diese Kombination war für die Pilotanwendung geplant, um den Aufwand diese Vorhabens zu verringern. Abbild (7) zeigt die Ergebnisse für eine gemischte Einlagerung von Türen für Stand- und Einbaugeräte in Lager 1 und 2. Die Abbildungen zeigen deutlich, dass im Gegensatz zu den anderen beiden Varianten, bei der gemischten Einlagerstrategie immer ausreichend Lagerplatz zur Verfügung steht.

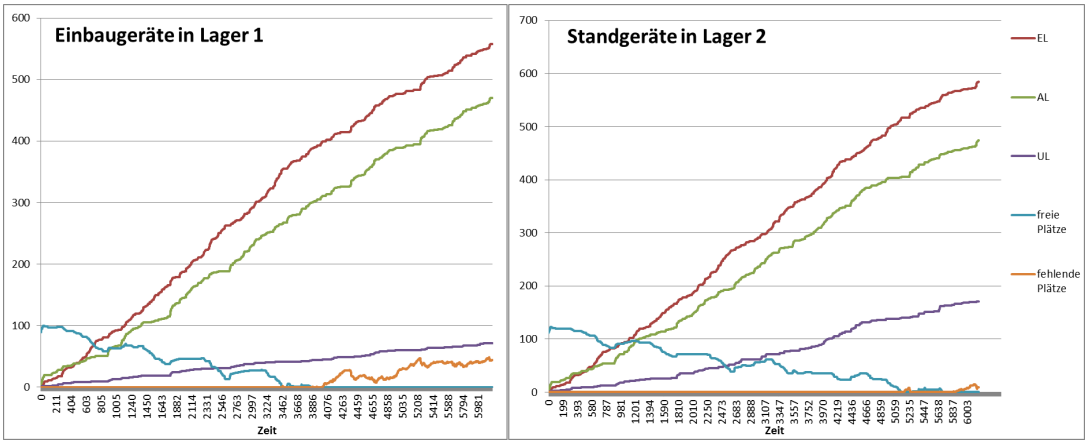

Abbildung 5: Türen für Einbaugeräte in Lager 1, Türen für Standgeräte in Lager 2.

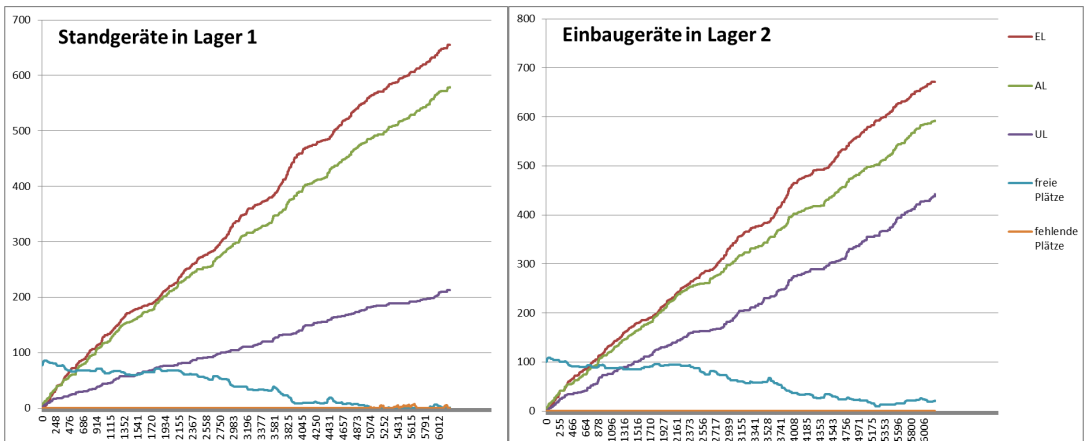

Abbildung 6: Türen für Standgeräte in Lager 1, Türen für Einbaugeräte in Lager 2.

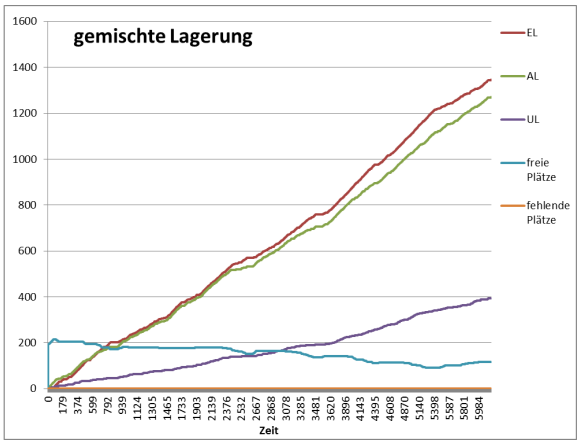

Abbildung 7: Türen werden unabhängig vom Typ in Lager 1 und 2 gelagert.

Auch im Vergleich des Aufwands, also wie viele Umlagerungen pro Auslagerung notwendig sind, gewinnt die gemischte Einlagerstrategie deutlich. Die Vergleichswerte für die 3 verschiedenen Varianten, unter Benutzung der Basisheuristiken ohne MCR, sind in Abbildung (8) dargestellt.

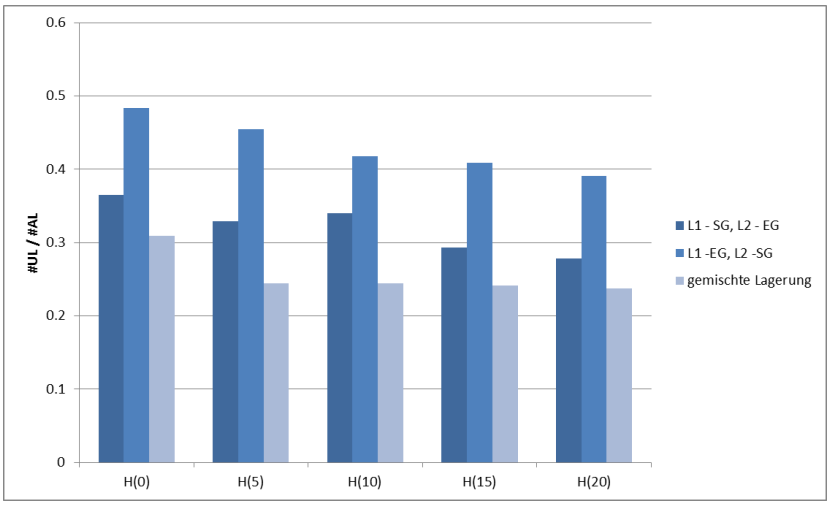

Abbildung 8: Anzahl der UL pro AL bei den verschiedenen Lagervarianten.

Die Verbesserung die mit dem Monte-Carlo-Rollout-Verfahren erreicht werden kann, wird exemplarisch an dem Fall der gemischten Einlagerstrategie bei einer maximalen Abweichung von den durchschnittlichen Produktionszeiten von 4% in Abbildung (9) gezeigt. Wieder wird deutlich, unter Benutzung der Heuristik $H(x)$ werden die Ergebnisse mit steigender Vorausschau besser. Benutzt man im Vergleich dazu das Monte-Carlo-Rollout-Verfahren, reicht die einfachste Heuristik $H(0)$ ohne Vorausschau, um die gleichen guten Ergebnisse zu erzielen. Im Ergebnis ist nur bei 20% der Auslagerungen eine Umlagerung erforderlich.

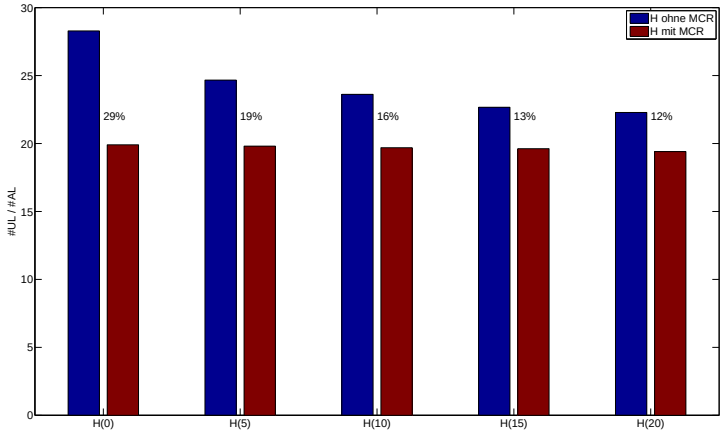

Abbildung 9: Anzahl der UL pro AL mit und ohne MCR bei $\beta = 4\%$.

F Projektmanagement

Das Vorhaben begleitend wurde durch das Fraunhofer IVI die Projektkoordination und -dokumentation übernommen und dabei u.a. folgende Aufgaben wahrgenommen:

- Koordinierung der Projektaktivitäten
- Überwachung des Projektfortschritts
- Management der Projektmittel
- Kommunikation mit dem Projektträger
- Organisation von Projekttreffen
- Berichtswesen und Dokumentation der Projektergebnisse

II.2. Die wichtigsten Positionen des zahlenmäßigen Nachweises

Tabelle 1: Positionen des zahlenmäßigen Nachweises

Kostenposition	Plan lt. Antrag	Ist
Personal	304.263,00	306.532,02
Reisen	3.252,00	1.064,58
Gesamt	307.515,00	307.596,60

II.3. Notwendigkeit und Angemessenheit der geleisteten Arbeit

Im Rahmen des Vorhabens wurden neue Lösungswege für die Modellierung und Simulation von intralogistischen Vorgängen erarbeitet. Dies betrifft die Simulation und Optimierung von Transportaufträgen unter Ungewissheit in der Fertigungs- und Lagerlogistik. Ohne Förderung durch öffentliche Mittel hätte der Entwicklungspartner Dresden Informatik GmbH diese Thematik nicht bearbeiten können. Die vom Fraunhofer IVI bearbeiteten Aufgaben waren notwendig, um die im Projekt definierten Zielstellungen vollständig zu erreichen. Da die im Projekt geleisteten Arbeiten seitens des Fraunhofer IVI nicht aus Mitteln der Grundfinanzierung erfolgen konnten, war auch die Mitarbeit des Fraunhofer IVI nicht ohne Förderung möglich.

II.4. Nutzen und Verwertbarkeit

Durch die gute Zusammenarbeit von Dresden Informatik, Liebherr und dem Fraunhofer IVI während des Projektes konnte für jeden der beteiligten Partner ein Nutzen erzielt werden. Auf Seiten des Fraunhofer IVI konnten durch die praxisorientierte Zusammenarbeit weitere Erkenntnisse im Bereich der logistischen Anforderungen und Bedürfnisse für innerbetriebliche Logistikprobleme gesammelt werden. Auf Grundlage der Systemspezifikation und der Simulation konnte der algorithmische Rahmen getestet und weiter entwickelt werden. Dabei konnte zum einen die grundlegende algorithmische Entwicklung am Fraunhofer IVI weitergeführt werden. Bei der Arbeit an methodischen Grundlagen und ihrer Umsetzung innerhalb

des Vorhabens konnte zum anderen neues problemspezifisches Know-how aufgebaut werden, dass besonders bei ähnlichen Fragestellungen bzw. verwandten Problemfeldern nützlich war. Die Kopplung zwischen der hier entwickelten Softwarekomponente und der Lagerverwaltungssoftware DiLVS von DI bildet eine gute Grundlage für Folgeanwendungen. Die Arbeitsgruppe „Operations Research“ am Fraunhofer IVI konnte sich durch dieses Projekt noch stärker als kompetenter Forschungsdienstleister für die industriellen Partner aus dem Logistiksektor profilieren. Abschließend wurden durch die Projektergebnisse die Grundlagen für wissenschaftliche Publikationen gelegt, die im Anschluss vorbereitet werden.

II.5. Fortschritt an anderer Stelle

Es sind von dritter Seite keine Ergebnisse bekannt geworden, die für die Durchführung des Vorhabens relevant sind.

II.6. Veröffentlichung der Ergebnisse

Das Fraunhofer IVI stellte die Ergebnisse des Projekts auf der „8th Scientific Conference Economy and Efficiency – contemporary solutions in logistics and production OIE 2013“ in Poznan, Polen vor.

III. Anhang

III.1. Problem-Spezifikation

Ein Anwendungsfall findet sich in der Kühlgeräteproduktion der Firma Liebherr. In der Türen-Schäumerei werden Türen für Kühlgeräte gefertigt und für die Weiterverarbeitung in der Endmontage bereitgestellt. Auch wenn sich das Produktionsprogramm der Türen-Schäumerei an dem der Endmontage orientiert, sind beide Fertigungsbereiche zeitlich voneinander entkoppelt, was durch ein Zwischenlager realisiert wird. Türenwagen aus der Türen-Schäumerei stellen somit Einlagerungen dar und Türenwagen die in der Endmontage gebraucht werden entsprechen den Auslagerungen. Dieses Zwischenlager soll im Rahmen dieses Projekts neu organisiert werden.

Die Türen werden typenrein auf Türenwagen gelagert und transportiert. Bei Kühl-Gefrier-Kombinationen werden entsprechend die Türensätze bestehend aus zwei Türen (Kühlteil- und Gefrierteil-Tür) auf dem Türenwagen gelagert. Türenwagen nehmen eine Fläche von $0,85m \times 2,4m$ ein und fassen je nach Türen- und Wagentyp 16, 18 oder 27 Türensätze. Es gibt in der aktuellen Produktion 219 Türentypen für insgesamt ca. 650 Endgerättypen. Pro Tag werden ca. 4500 Geräte produziert. Im Lager befinden sich daher etwa 125 Türenwagen.

Abbildung 10 zeigt einen Ausschnitt der Produktionshalle. Die relevanten Bereiche sind farbig markiert. Die Türen-Schäumerei befindet sich im blauen Bereich, die Endmontage ist rot markiert und das dazwischen liegende Lager grün. Es gibt Türentypen die eine zusätzliche Montage von Griffen oder Scharnieren erfordern. Dazu muss der Türenwagen vor der Einlagerung in die entsprechenden, hier gelb markierten, Bereiche transportiert werden. Ebenfalls eingezeichnet sind geplante RFID Schnittstellen auf die später eingegangen wird.

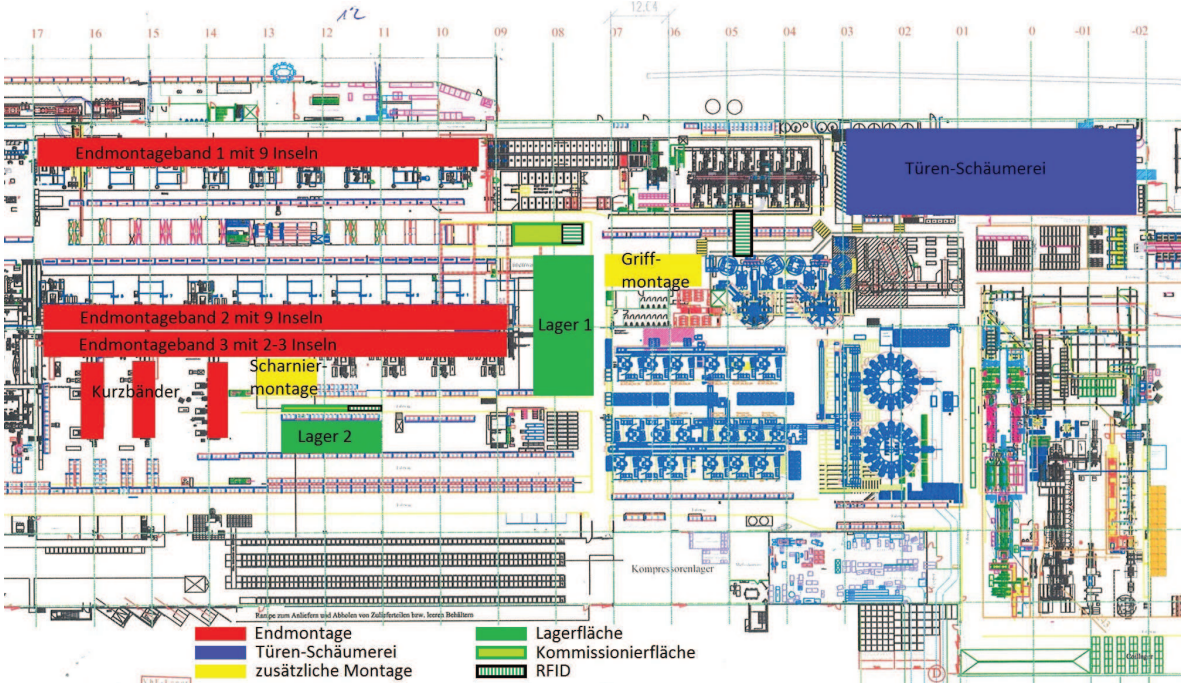

Abbildung 10: Ausschnitt der Produktionshalle

Lager-Layout

Für das Türenzwischenlager stehen zwei Stellflächen zur Verfügung. Die Fläche von Lager 1 ist $10,6m \times 28,9m$ und soll vorrangig für Türenwagen verwendet werden die an den Endmontagebändern 1 und 2 gebraucht werden. Aus Platzgründen soll hier ein Blocklager mit evtl. einer Gasse entstehen. In unmittelbarer Nähe befindet sich eine weitere Fläche ($13,4m \times 4,3m$) die als Kommissionier- und Eingangsfläche genutzt werden soll. Die Türenwagen sollen in diesem Bereich zwei Reihen mit Einzelzugriff abgestellt werden. 20 Plätze stehen als Kommissionierfläche für die Bänder 1 und 2 zur Verfügung. 6 Plätze sollen für Neueingänge ins Lager reserviert werden. Einlagerungen können sowohl neue Türenwagen aus der Schäumerei sein, als auch angebrochene Wagen die von den Endmontageinseln zurück gebracht werden. Lager 2 befindet sich in der Nähe des Montagebands 3 und den 3 Kurzbändern. Die Fläche ($20,2m \times 6,2m$) soll daher vorrangig von Türenwagen für diese Inseln als Lagerfläche genutzt werden. Auch hier befindet sich eine zusätzliche Kommissionierfläche mit Einzelzugriff für 6 Türenwagen und eine Eingangsfläche für 2 Wagen. Eine Person soll für das Zwischenlager verantwortlich sein. Diese stellt von den Inseln angeforderte Wagen auf den Kommissionsflächen bereit und lagert neue Türenwagen ein.

Einlagerung

Eingänge im Türenwagenlager kommen hauptsächlich aus der Türen-Schäumerei. Es werden immer mehrere Typen gleichzeitig an insgesamt 3 Maschinen produziert. In der Regel stehen die Formenbelegungen der Schäumerei für 2-3 Tage fest und orientieren sich an den Produktionsplänen der Endmontage. Da die Zahl der Formen für die einzelnen Türentypen begrenzt ist, kann es auch vorkommen, dass ein Typ mit 2-3 Tagen Vorlauf produziert wird. Im Laufe der Früh- und Spätschicht produziert die Türen-Schäumerei pro Maschine 32 Wagen pro Schicht. Die Nachtschicht produziert pro Maschine 20 Wagen. Im Laufe der Frühschicht wird ein aktualisierter Tagesplan für die nächsten 24 Stunden festgelegt. Dieser gilt ab der Spätschicht. Hier werden notwendige Bedarfsabweichungen vom 3-Tages-Produktionsplan berücksichtigt und umgesetzt. Die Türen-Schäumerei produziert die Kühlgerätetüren mit ungefähr einer Schicht Vorlauf. Das heißt, am Ende der Frühschicht (11Uhr) stehen der Endmontage alle Türen für das Tagesprogramm zur Verfügung. Für die Steuerung des Lagers steht die Information der tatsächlichen Produktion nach jeder Tür bereit. Folglich ist bekannt, wann ein neues Los angefangen wird und damit auch wann ein neuer Türenwagen angefangen wird und dieser voraussichtlich im Lager eintrifft. Jede der 3 Maschinen ist mit 5 Formen besetzt, daher sind die nächsten 15 Türenwagen die im Lager eintreffen bekannt. Die Lose in der Türen-Schäumerei sind um 4-8 Türensätze größer, als die in der Endmontage. Auch wird der Bedarf der einzelnen Endmontageinseln in der Türen-Schäumerei zusammengefasst. Daher ergibt sich ein zweiter Zugang von Türenwagen im Lager. Türenwagen die von einer Montageinsel nicht komplett verbaut wurden, werden an das Lager zurück geliefert. Benötigt ein Türentyp Griffe und/oder Scharniere muss der Türenwagen vor der Einlagerung zu den entsprechenden Montagestellen gebracht werden. Erst nach der Montage wird der Türenwagen eingelagert.

Auslagerung

Die Auslagerungen erfolgen entsprechend der Endmontage. Hier gibt es 3 Montagebänder und 3 Kurzbänder. Montageband 1 und 2 umfassen jeweils 9 Montageinseln, Band 3 nur 2

oder 3. Die Kapazitäten an den 3 Montagebändern liegen zwischen 60 und 110 Endgeräten pro Insel und Schicht. Die Kurzbänder schaffen 180-250 Endgeräte pro Schicht, da hier i.d.R. große Lose mit einfachen Gerätetypen montiert werden. Für die Endmontage gibt es keine Inselfeine-Planung. Folgende zusammengefasste Tagespläne sind bekannt:

- 1 Plan für Montageband 1, d.h. 1 Liste für 9 Inseln
- 3 Pläne für Montageband 2, d.h. 3 Listen für jeweils 3 Inseln
- 1 Plan für Montageband 3, d.h. 1 Liste für 2-3 Inseln
- 1 Plan für Kurzbänder, d.h. 1 Liste für 3 Kurzbänder

Die genauen Lose und die Reihenfolge der Produktion einer Inseln sind ungewiss. Die Endmontage prüft den Bestand der Türen im Zwischenlager, bevor dort ein Auftrag gestartet wird. So wird sichergestellt, dass alle notwendigen Türenwagen rechtzeitig bereitgestellt werden können. Bekannt sind das aktuelle Los und der voraussichtliche Folgeauftrag jeder Insel. Ebenfalls ungewiss ist die Anzahl der Mitarbeiter an einer Insel und damit die genauen Taktzeiten für die verschiedenen Endgerätetypen. Die Anzahl der Lose die eine Insel pro Schicht schafft, schwankt zwischen 1 und 6 Losen. Der Durchschnitt liegt bei 2,4 Losen pro Schicht und Insel.

RFID

Die Lagerverwaltung soll mittels RFID unterstützt werden. Dazu werden alle Türenwagen mit RFID Transpondern ausgestattet. In Abbildung 10 sind die vorgesehene RFID Reader straffiert abgebildet. Jeder Türenwagen der aus der Türen-Schäumerei kommt, muss den oberen RFID Reader passieren. Dort werden auf dem Chip des Wagens die Artikelnummer des Türentyps und die Anzahl der Türen gespeichert. Die Eingangsflächen zum Lager, dargestellt als straffierter Bereich auf den Kommissionierungsflächen, werden ebenfalls mittels RFID überwacht. Für einen Wagen auf dieser Fläche wird anhand der RFID Informationen ein optimaler Stellplatz im Lager bestimmt und dem Lagerverwalter mitgeteilt. Eine erneute RFID Erkennung an den Lagereingängen ist zusätzlich möglich.

Pilot

Für die Pilot-Anwendung werden folgende Vereinfachungen festgelegt. Da die Türenwagen für das Montageband 3 und die Kurzbänder mit Einzelzugriff im Lager 2 zur Verfügung stehen, werden diese in der Pilotanwendung nicht betrachtet. Desweiteren sollen Einlagerungen ohne vorherige Umlagerungen anderer Türenwagen erfolgen. Notwendige Umlagerungen beim Auslagern werden nur vereinfacht betrachtet. Dazu werden die Wagen die für eine Auslagerung im Weg stehen aus der Bahn gefahren, der notwendige Türenwagen wird rausgeholt und die umzulagernden Wagen in gleicher Reihenfolge wieder in die Bahn geschoben. Zu Beginn der Pilot-Anwendung werden in den ersten Tagen die Türenwagen-Ströme aufgezeichnet. Mit diesen realen Daten sollen die entwickelten Algorithmen vor der Anwendung getestet werden.

Zusammenfassung der Zahlen

	pro Schicht	pro Tag
`# Produktion der Türen in Türenwagen`	32 (Nacht 20)	
`# Endprodukte pro Insel`	60 - 110 (Ø 85)	
`# Endprodukte pro Kurzband`	180 - 250 (Ø 215)	
`# versch. Typen`	40 - 45 (Ø 43)	Ø 56
`# versch. Endgerättypen`	45 - 50 (Ø 46)	Ø 68

III.2. Modell

Ausgangspunkt für einen algorithmischen Entwurf wird der Anwendungsfall der Lageroptimierung der Firma Liebherr sein. Dieser wird vorerst wie folgt vereinfacht. Es wird nur die Steuerung einer Lagerfläche betrachtet. Dies bietet sich an, da Lager 1 für die Montagebänder 1 und 2 vom Lager 2 für das Montageband 3 sowie für die Kurzbänder unabhängig voneinander sind. Weiterhin werden in Lager 2 die Türenwagen mit Einzelzugriff gelagert. Es werden keine Nebenlager und Inselpuffer betrachtet.

Lagerlayout

Als Lager soll ein Blocklager mit möglichen Gassen betrachtet werden. Das Lager wird durch folgende Parameter definiert:

- Anzahl der Bahnen m
- Kapazitäten der Bahnen k_i, $1 \leq i \leq m$
- Lagerzugänge in eine Bahn $z_i \subseteq \{1, \ldots, 2k_i\}^{Z_i}$ mit $1 \leq Z_i \leq 2k_i$

Die Menge $\{1, \ldots, 2k_i\}$ stellt dabei alle möglichen Zugänge in eine Bahn dar. Diese sind jeweils die linke und rechte Seite eines Platzes über alle Plätze von links nach rechts durchnummeriert. Zwischen zwei Stellplätzen sind also zwei Zugänge möglich, einer nach links und einer nach rechts in die entsprechende Bahn. Ein Zugangsnummer in z_i legt folglich eindeutig fest in welche Richtung dieser Zugang führt. Ist die Zugangsnummer gerade bzw. ungerade so führt der Zugang nach links bzw. rechts in eine Lagerbahn. Der Fall $Z_i = 1\ \forall i$ stellt ein Sackgassenlager mit nur einem Zugang dar. Im Gegensatz dazu stellt $Z_i = 2k_i\ \forall i$ ein Lager dar, bei dem jeder Platz von beiden Seiten zugänglich ist. Mit $z_i = \{1, 2k_i\}\ \forall i$ wird ein Blocklager mit zwei Zugängen dargestellt.

Der Inhalt einer Bahn B_i zum Zeitpunkt t ist

$$B_i(t) = (b_{i1}, \ldots, b_{ik_i})$$

wobei

$$b_{ij} = (a_{ij}, n_{ij}, t_{ij}) \in N \times N \times N$$

ein Türenwagen mit n_{ij} Türen des Types a_{ij}, der zum Zeitpunkt t_{ij} auf den Platz (i, j) eingelagert wurde. Für $b_{ij} = (0, 0, 0) = 0$ ist der Platz b_{ij} im Lager frei. Durch eine Zugang z_{ij} ist auch der erste zugängliche Platz von diesem Zugang aus definiert:

$$b_{i1} \text{ für } z_{ij} = 1 \quad \text{bzw.} \quad b_{i\lceil z_{ij}/2 \rceil} \text{ für } 1 < z_{ij} \leq 2k_i.$$

Im Bahnenlager ist eine maximale Lagerzeit τ wie folgt einzuhalten: Bei jeder Auslagerung zu Zeitpunkt t eines Artikels a ist zu prüfen, ob es auf irgendeinen Platz b_{ij} einen Inhalt mit $a = a_{ij}$ und $t > t_{ij} + \tau$ gibt. Ist dies der Fall, so muss von demjenigen Platz ausgelagert werden, an dem t_{ij} minimal ist.

Lageroperationen

Ein Ein- bzw. Auslagerprogramme P besteht aus einer Folge von Lagervorgängen

$$P = (p_1, ..., p_N) \text{ mit } t_i \leq t_j \text{ für } i < j,$$

wobei N die Anzahl aller Ein- bzw. Auslagerungen ist. Ein Lagervorgang ist ein Tripel

$$p = (a, n, t) \in N \times N \times N$$

wobei a den Artikel und n die Anzahl des Artikels auf dem Türenwagen angibt der zum Zeitpunkt t ein- bzw. ausgelagert werden soll. Nach einem Auslagervorgang wird p als Türenwagen betrachtet.

Mögliche Lageroperationen sind Ein- und Auslagervorgänge. Für Auslagerungen können eventuell zusätzliche Umlangerungen notwendig sein. Im Folgenden werden Operationen sowie deren Voraussetzungen definiert, die im Lager durchgeführt werden können. Vor der Bearbeitung eines Lagervorgangs $p = (a, n, t)$ sei der Inhalt des Lagers

$$B_i(t) = (b_{i1}, ..., b_{ik_i}) \text{ mit } b_{ij} = (a_{ij}, n_{ij}, t_{ij}) \text{ für } 1 \leq i \leq m.$$

Lageraufträge p aus einem Programm P müssen entsprechend der Reihenfolge ihres zeitlichen Auftretens zum Zeitpunkt t sofort ein- bzw. ausgelagert werden. Dazu wird eine Bahn i, ein Platz j und ein Zugang $z \in z_i$, durch den der Lagervorgang bearbeitet werden soll, festgelegt. Wir definieren

$$b_{iz} = b_{i\lceil z/2 \rceil} \text{ für } z \neq 1, \text{ sonst } b_{iz} = 1$$

als den ersten zugänglichen Platz von z aus. z' sei der nächste Zugang in die Bahn i auf der anderen Seite von b_{ij}.

Für die *Einlagerung* von p auf den Platz b_{ij} durch den Zugang z gelte für ungerade z

$$\begin{aligned} b_{iz},\ b_{i(z+1)},\ \ldots,\ b_{i(j-1)},\ b_{ij} = 0, \\ b_{i(j+1)} = 0 \Rightarrow b_{i(j+2)},\ \ldots,\ b_{iz'} = 0, \end{aligned}$$

und für gerade z

$$\begin{aligned} b_{ij},\ b_{i(j+1)},\ \ldots,\ b_{i(z-1)},\ b_{iz} = 0, \\ b_{i(j-1)} = 0 \Rightarrow b_{i(j-2)},\ \ldots,\ b_{iz'} = 0. \end{aligned}$$

Durchgeführt wird die Operation

$$in(p, i, j, z) : b_{ij} = p = (a, n, t)$$

d.h. es wird ein Wagen mit Artikel a in Bahn i an Platz j gestellt.

Für die *Auslagerung* von p vom Platz b_{ij} durch den Zugang z muss die maximale Lagerzeit τ eingehalten werden, d.h.

$$b_{ij} = \min_{t_{ij}}(b_{ij} \mid a_{ij} = a,\ t \geq t_{ij} + \tau).$$

Durchgeführt wird die Operation

$$out(p,i,j,z) = \begin{cases} b_{ij} = b_{i(j-1)},\ b_{i(j-1)} = b_{i(j-2)},\ \ldots,\ b_{i(z+1)} = b_{iz},\ b_{iz} = 0, & \text{falls } z \text{ ungerade} \\ b_{ij} = b_{i(j+1)},\ b_{i(j+1)} = b_{i(j+2)},\ \ldots,\ b_{i(z-1)} = b_{iz},\ b_{iz} = 0, & \text{falls } z \text{ gerade} \end{cases}$$

d.h. es wird der Wagen auf Platz b_{ij} entfernt, falls sich weitere Türenwagen auf den Plätzen zum Zugang befinden rutschen diese nach. Hier finden damit gleichzeitig notwendige Umlagerungen statt.

Kostenfunktion

Die Kosten setzten sich aus den Zeitaufwänden zusammen, die für die Ausführung einer Lageroperation notwendig sind. Dazu sind folgende Daten gegeben:

- $\Delta t_{z_i} \in R^{Z_i}$ Zeiten die man von der Kommisionierfläche zu den entsprechenden Zugangspunkten z_i einer Bahn braucht.
- $\Delta t_{AL} \in R^3$ Zeiten die man für eine Auslagerung bracht in der 0, 1 oder 2 Plätze vor dem Platz b_{ij} besetzt sind.

Damit ergeben sich folgende Kosten für die Operationen:

$$c(in(p,i,j,z)) = \Delta t_{z_i}(z)$$

$$c(out(p,i,j,z)) = \Delta t_{z_i}(z) + \Delta t_{AL}(s),\ \text{mit } s = \begin{cases} \#\, b_{ik} \neq 0,\ z \leq k \leq j-1, & \text{falls } z \text{ ungerade} \\ \#\, b_{ik} \neq 0,\ j+1 \leq k \leq z, & \text{falls } z \text{ gerade} \end{cases}$$

Schäumerei- und Endmontageprogramm

Das Programm der Auslagerungen P^- wird durch den Tagesplan der Endmontage bestimmt. Die Produktion der Endmontage erfolgt losweise an n_I Montageinseln. Ein Los ist ein Tripel

$$l = (A, a, N) \in N \times N \times N$$

wobei A den Endgerättyp, a den dazugehörigen Türentyp und N die zu produzierende Anzahl angibt. Im Tagesplan ist nicht festgelegt welche Insel, welches Los und in welcher Reihenfolge produziert. Der Tagesplan wird lediglich anhand von vier Mengen

$$T_k = \{l_1, \ldots, l_{n_k}\},\ \text{mit } k = 1, \ldots, 4$$

festgelegt. Jede Insel ist genau einer dieser Mengen zugeordnet. n_k ist die Anzahl der Lose die insgesamt von allen Monatageinseln, die der Menge T_k zugeordnet sind, im Laufe des Tages produziert werden müssen. Die Information zum Produktionsstartzeitpunkt und der produzierenden Insel eines Loses werden erst bekannt, wenn eine Insel das Los als Folgeauftrag benennt. Bekannte Lose werden mit

$$L = (A, a, N, t, I) \in N \times N \times N \times N \times N$$

bezeichnet. Die Losinformationen werden dabei durch eine Anfangszeit t und die produzierende Insel I ergänzt. Die Anzahl aller bekannten Lose sei n_L. Des Weiteren seien die Taktzeiten der Endmontage für alle Endgerättypen in $\Delta t_A \in R^{n_A}$ gegeben, wobei n_A die Anzahl aller Endgerättypen ist.

Es wird angenommen, dass zum Zeitpunkt $t = 0$ an jeder Insel I ein Los L begonnen wird. Für ein Los $L_k = (A, a_k, N_k, t_k, I)$ ergibt sich eine Folge von Lagervorgängen $p_k = (p_1, \ldots, p_{S-1})$ mit

$$\begin{aligned} p_1 &= (a_k, n_1, t_k) \\ p_i &= \left(a_k, n_i, t_{i-1} + \frac{n_{i-1}}{\Delta t_A(A)}\right),\ i = 2, \ldots, S-1 \text{ mit } S \in N, \text{ so dass } \sum_{i=1}^{S} n_i \geq N_k \end{aligned}$$

Diese Folge beinhaltet die Zeitpunkte zu denen ein Türenwagen für Insel I ausgelagert werden muss. Eine solche Folge von Lagervorgängen ergibt sich für alle bekannten Lose $L_k, k = 1, \ldots, n_L$. Folglich ergibt sich das Programm der Auslagerungen wie folgt

$$P^- = \bigcup_{1 \leq k \leq n_L} p_k \text{ mit } t_i \leq t_j \text{ für } i \leq j.$$

Als Kontrollvariable fungiert

$$t_{S-1} + \frac{n_{S-1}}{\Delta t_A(A)} \stackrel{!}{=} t_{k'}$$

wobei $t_{k'}$ die Startzeit des Folgeloses von Los L_k ist.

Das Programm der Einlagerungen P^+ ergibt sich aus der Formenbelegung und der Produktionszeiten der Schäumerei. Daher ist P^+ im Voraus bekannt. Jede produzierte Tür wird im System erfasst. Daher kann eine Abweichung vom Tagesplan frühzeitig erkannt werden, wenn ein anderer Türenwagen begonnen wird als geplant. Anhand der Produktionszeiten kann P^+ entsprechend abgeändert werden.

Ungewissheiten

Die Ausgangsprogramme $P^+(0)$ und $P^-(0)$ unterliegen gewissen Ungewissheiten. Zu beliebigen Zeitpunkten t können sich die Programme ändern und es gelten neue Programme $P^+(t)$ und $P^-(t)$. Diese stimmen bis zum Zeitpunkt t mit den alten Programmen überein. Der Inhalt des Bahnenlagers zum Zeitpunkt t

$$B(t) = (B_1(t), ..., B_m(t))$$

ergibt sich durch das Abarbeiten der Programme während des Zeitraums $[0, t)$.

Die zugrundeliegenden Ungewissheiten sollen jetzt genauer beschrieben werden. Vereinfachend wird angenommen, dass das Folgelos sofort bekannt wird, wenn ein neues Los gestartet wird. Daher ist über einen relativ großen Zeitraum bekannt, welche Insel welchen Türentyp benötigt. Die Ungewissheit entsteht durch die Verkettung der Auslagerungen für die verschiedenen Inseln und eventuellen Taktzeitenabweichungen. Das Programm ist die zeitlich

aufsteigend sortierte Vereinigung aller notwendigen Wagen p_k aller bereits bekannter zu produzierenden Lose L_k. Die Auslagerungen an die verschiedenen Inseln wechseln sich folglich je nach Taktzeiten ab. Verzögert sich die Produktion des Loses L_k an einer Insel, so muss die Folge der benötigten Türenwagen p_k korrigiert werden. Diese Korrektur führt zu einer Permutation des Auslagerprogramms P^-. Zur Modellierung dieser Abweichungen soll ein stochastisches Model als Grundlage für die tatsächlichen Taktzeiten an den Montageinseln dienen. Tritt eine Störung $\delta t \in R$ bei der Produktion des Loses L_k an einer Insel auf, ergeben sich folgende Korrekturen

$$\begin{aligned} \tilde{p}_k &= ((a_k, n_1, t_1 + \delta t),\ \ldots\ , (a_k, n_{S-1}, t_{S-1} + \delta t)) \\ \tilde{p}_{k'} &= ((a_{k'}, n_1, t_1 + \delta t),\ \ldots\ , (a_k, n_{S'-1}, t_{S'-1} + \delta t)) \end{aligned}$$

Eine weitere Ungewissheit entsteht durch die Tatsache, dass nicht alle Türenwagen mit der gleichen Anzahl an Türen bestückt sind. Angebrochene Wagen können sowohl aus der Schäumerei kommen, als auch von Montageinseln zurückgebracht werden. Verschiedenes Vorgehen ist möglich:

- Festlegung aller Türenwagen zu Beginn der Produktion des Loses. Dies stellt eine sehr große Einschränkung dar, da ganz bestimmte Türenwagen ausgelagert werden müssen und nicht nur der Türentyp festgelegt ist. Eine Berücksichtigung der aktuellen Lagersituation ist nicht möglich.
- Unter der Annahme, dass alle Türenwagen für einen Türentyp die gleichen Kapazitäten besitzen, kann mit dieser Standartgröße gerechnet werden, um die Anzahl der Türenwagen und entsprechende Bereistellungszeiten festzulegen. Falls ein Türenwagen nicht komplett bestückt sein sollte, müssen die folgenden Bereitstellungszeiten und evtl. die Gesamtzahl an Türenwagen für dieses Los angepasst werden.
- Lege nur die nächsten x Wagen für ein Los bzw. Folgelos fest. Damit ergibt sich ein verkürztes Programm P^-, aber bei der Auswahl kann die aktuelle Lagersituation berücksichtigt werden.

Lagerstrategie

Die gegebenen Programme $P^+(0)$ und $P^-(0)$ werden mit Hilfe einer Lagerstrategie abgearbeitet. Diese legt sowohl fest an welchem Platz (i, j) ein Wagen eingelagert bzw. von welchem Platz ein Wagen ausgelagert werden soll. Auch der dazu zu benutzende Zugang z wird durch die Lagerstrategie bestimmt. Das Ziel der Lagerstrategie ist die Kosten aller Ein- und Auslagerungen zu minimieren

$$\sum_{p \in P^+} c(in(p, i, j, z)) + \sum_{p \in P^-} c(out(p, i, j, z)) \to \min .$$

Die Wahl einer Lagerstrategie Π basiert auf der Kenntnis der Ausgangsprogramme $P^+(0)$ und $P^-(0)$. Durch Abweichungen kann es notwendig werden, eine neue Lagerstrategie Π' zu ermitteln, die ab Zeitpunkt t auf $B(t)$, $P^+(t)$ und $P^-(t)$ angewandt wird. Durch die Ungewissheit der Programme sind die Lagerstrategien dynamisch in dem Sinne, dass sie selbst über die Zeit angepasst werden. Die Anpassung oder Neuberechnung der Strategie wiederholt sich nach jeder Änderung an einem der Programme. Schlimmstenfalls sind Programme so ungewiss, dass zu jedem Lagervorgang die Strategie berechnet werden muss.

III.3. Algorithmische Methodik

Mit Hilfe von komplexen Algorithmen, z.B. dem branch&bound Algorithmus, kann eine optimale Strategie für Planungsprozesse berechnet werden. Solche Algorithmen sind in der Regel für große Probleme sehr zeitaufwendig. Zusätzlich wird aufgrund der Ungewissheit eine sehr häufige Berechnung der Lösung notwendig sein. Da hier für jeden Lagervorgang sofort eine Entscheidung getroffen werden muss, ist die Anwendung komplexer Algorithmen nicht geeignet. Es soll innerhalb kurzer Zeit und ohne großen Aufwand eine zulässige Lösung geliefert werden. Daher werden Heuristiken zur Bestimmung einer Lösung Anwendung finden. Diese so genannte Basislösung soll anschließend mit Hilfe einer Metaheuristik verbessert werden. Dazu soll nicht nur die aktuelle Konfiguration und deren Kosten betrachtet werden. Vielmehr sollen zukünftige Entwicklungen die aus der Entscheidung resultieren mit einbezogen werden. Da nicht alle möglichen zukünftigen Verläufe simuliert werden können, wird der Monte-Carlo-Ansatz verfolgt. Ein Verfahren aus der Stochastik, bei dem sehr häufig durchgeführte Zufallsexperimente die Basis darstellen. Als Grundlage ist vor allem das Gesetz der großen Zahlen zu sehen. Es wird versucht, das Problem mit Hilfe der Wahrscheinlichkeitstheorie numerisch zu lösen. Das Monte-Carlo-Rollout-Verfahren als Metaheuristik wird im Folgenden detailiert beschrieben.

Monte-Carlo-Rollout (Mc-Ro) dient als Lösungsverfahren für Planungsprozesse oder allgemein für kombinatorische Optimierungsprobleme sequentieller Art, bei denen wiederholend die folgenden beiden Schritte abwechselnd durchgeführt werden:

- Einem *Entscheider* (Planer/Disponent) liegt eine aktuelle Instanz des Planungsproblems vor, es ist eine Entscheidung zu treffen, die zu einer neuen Lösung für das Problem führt.
- Beim Abarbeiten der vorliegenden Lösung werden durch die *Umwelt* Abweichungen vom geplanten Ablauf erzeugt (in Form von Störungen, Bekanntwerden neuer Informationen usw.). Diese machen eine erneute Reaktion des Entscheiders notwendig.

Das Grundprinzip des Monte-Carlo-Rollout-Verfahrens zur Anwendung auf solche sequentiellen Planungs- oder Lösungsprozesse ist wie folgt: Dem Entscheider liegt zur Auswahl seiner Entscheidungen oft eine ëinfache"Heuristik $\mathcal{H}$ vor, die zunächst nur "kurzsichtig"nach Lösungen für die aktuelle Planungsinstanz mit aktuellen Informationen sucht. Das Monte-Carlo-Rollout-Verfahren soll dazu dienen, die Entscheidungen der Heuristik $\mathcal{H}$ unter Einbeziehung von Vorausschau auf zukünftig mögliche Konsequenzen zu verbessern.

Voraussetzungen

Es werden zunächst die Voraussetzungen an den sequentiellen Lösungs-prozess genannt, unter denen das Monte-Carlo-Rollout-Verfahren angewandt werden kann, und dabei im Weiteren verwendete Begriffe eingeführt. Der sequentielle Prozess wird dabei im Kontext der Spielbaumsuche betrachtet.

Als *Basisproblem* wird die statisch-deterministische Variante des Planungsproblems bezeichnet, für die der Entscheider in jedem Schritt eine Entscheidung treffen muss. Dieses wird beschrieben durch alle zum Zeitpunkt der Entscheidung bekannten Informationen, auch über das Vorhandensein der Ungewissheiten im Lösungsprozesses und damit zusammenhängenden Informationen. Sei im Schritt t eine Instanz Π des Basisproblems zu lösen. Dazu liegt ein Verfahren vor, das eine endliche Menge möglicher Kandidatenlösungen $\mathbf{S} = \{S_1, S_2, \ldots\}$ für

Π vorgibt, unter denen der Entscheider seine Auswahl trifft. In Anlehnung an die Formulierung mittels Spielbäumen wird dieses Verfahren *Zug-Generator für den Entscheider* genannt. Ebenso liegt ein Verfahren vor, welches zu jeder Lösung S einer Instanz Π eine endliche Menge möglicher Folge-Instanzen $\{\Pi'_1, \Pi'_2, \ldots\}$ vorgibt. Diesem Verfahren, *Zug-Generator für die Umwelt* genannt, liegt ein Wahrscheinlichkeitsmodell mit Maß $\mathcal{P}$ zugrunde, das Ereignisse bei der Abarbeitung von Lösungen im Planungsprozess definiert. Die Folge-Instanzen enthalten dann alle das Basisproblem beschreibenden Informationen im nächsten Schritt des Entscheiders. Zusammen mit jeder möglichen Folge-Instanz Π'_i wird durch den Zug-Generator eine Wahrscheinlichkeit p_i ermittelt, mit welcher Π'_i eintritt. Sowohl die möglichen Lösungen $\mathbf{S}$ als auch die daraus resultierenden Folge-Instanzen bilden die Knoten des Spielbaumes. Situationen am Ende des Planungsprozesses, die *Blätter*, werden durch die (reellwertige) *Bewertungsfunktion* f exakt bewertet. Ziel ist es diese Bewertungsfunktion zu minimieren.

Mit diesen Elementen – Zug-Generatoren und Blattbewertung – lässt sich der Planungsprozess als Spielbaum darstellen. Die Darstellung kann als 1-Personen-Spiel mit Zufall oder als 2-Personen-Spiel mit einem Min-Spieler und einem Spieler der Zufallsentscheidungen trifft interpretiert werden. Situationen, in denen der Entscheider eine Kandidatenlösung wählt, werden *E-Knoten* genannt. Situationen, in denen die Umwelt Ereignisse für eine Folge-Instanz erzeugt, heißen *U-Knoten*. In Abbildung 11 ist ein Ausschnitt aus einem solchen Spielbaum mit den eingeführten Begriffen dargestellt.

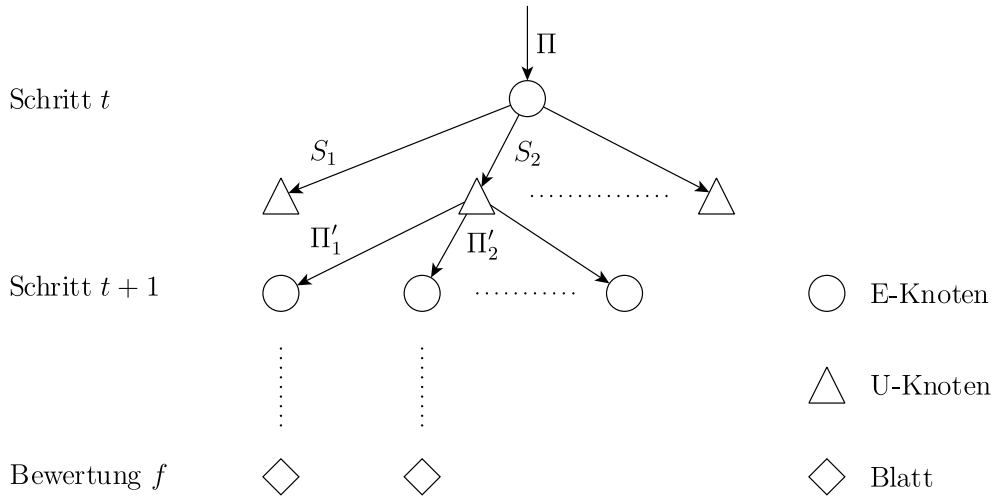

Abbildung 11: Sequentieller Planungsprozess dargestellt als Spielbaum

Für die Anwendung des Monte-Carlo-Rollout-Verfahrens wird weiterhin vorausgesetzt, dass für die Lösung einer Instanz Π des Basisproblems ein (heuristisches) Lösungsverfahren $\mathcal{H}$ gegeben ist, die der Entscheider zur Auswahl der Entscheidung aus der Menge seiner gegebenen Kandidatenlösung $\mathbf{S}$ anwendet. $\mathcal{H}$ wird als *Basisheuristik* bezeichnet, die gewählte Lösung $S^* = \mathcal{H}(\Pi, \mathbf{S})$ ist der *Zug* des Entscheiders. Grundlage der Basisheuristik $\mathcal{H}$ ist eine (reellwertige) Funktion h, welche die Lösungen S im Planungsprozess bewertet. Es wird vorausgesetzt, dass es zu Basisheuristik $\mathcal{H}$ immer eine Bewertungsfunktion h gibt, so dass $\mathcal{H}$ jeweils die unter allen $S \in \mathbf{S}$ nach h bestbewertetste Lösung S^* liefert:

$$S^* = \mathcal{H}(\Pi, \mathbf{S}) =_{S \in \mathbf{S}} h(S).$$

h ist im Gegensatz zur exakten Bewertung der Blätter f lediglich eine *heuristische Bewertung* der aktuellen Situation. Es kann aber angenommen werden, dass die Bewertung der Situation in Blättern nach f und der zugehörigen Lösungen nach h übereinstimmen.

Die Entscheidungen der Heuristik $\mathcal{H}$ sind auf das Basisproblem bezogen und enthalten somit Fehler im Vergleich zur theoretisch exakten Bewertung. Diese Fehler verstärken sich bei fortgesetzter Anwendung im sequentiellen Lösungsprozess, unabhängig davon, ob und wie stark die Basisheuristik die Ungewissheiten des Planungsproblems bei der Bewertung nach h berücksichtigt. Beim MC-RO wird versucht, die fehlerhaften Entscheidungen der Basisheuristik zu vermeiden.

Das Monte-Carlo-Rollout-Verfahren kombiniert die Grundideen zweier unterschiedlicher Ansätze aus verschiedenen Gebieten – der Spielbaumsuche und der kombinatorischen Optimierung – die zunächst kurz erläutert werden, bevor dann auf den Ablauf des MC-RO eingegangen wird.

Monte-Carlo Tree Search (MCTS)

Betrachtet wird ein Mehr-Personen-Spiel, welches in Form seines Spielbaums vorliegt. Beispielhaft wird hier ein Zwei-Personen-Spiel mit perfekter Information (also mit vollständiger Information und ohne Zufall) angenommen. Die üblichen Lösungsverfahren für diese Spiele basieren auf folgendem Grundprinzip einer Baumsuche: Im Spielbaum wird beginnend in der aktuellen Situation iterativ bis zu einer gewissen (nicht notwendig gleichförmigen) Tiefe verzweigt. In den Blättern des erhaltenen Suchbaums wird auf die entstandene Spielsituation eine heuristische Bewertung angewandt, die mit Hilfe des Minimax-Algorithmus an den Startknoten der Suche hochgereicht wird. Basierend darauf wird der Zug mit der besten Bewertung ausgewählt.

In der praktischen Anwendung zeigt sich, dass die Güte dieser Baumsuche stark von der Verfügbarkeit einer geeigneten heuristischen Bewertungsfunktion für Spielsituationen abhängt. Insbesondere gibt es Spiele, für die eine solche Bewertung nicht gefunden werden kann, und für die der herkömmliche Ansatz zu keinen befriedigenden Algorithmen geführt hat. Ein prominentes Beispiel hierfür ist das Brettspiel Go.

Der folgende Ausweg führt zu einem Verfahren, welches unabhängig von einer heuristischen Bewertungsfunktion ist: Nach der Verzweigung werden beginnend in den Blättern des Suchbaums m zufällige Zugfolgen als Partien, sogenannte *Monte-Carlo-Playouts*, durchgeführt. Bei diesen wird nacheinander für den jeweils am Zug befindlichen Spieler ein Zufallszug ausgewählt, bis ein Blatt im Spielbaum erreicht wird. Für dieses Blatt ist wiederum die exakte Bewertung in Form des Ergebnisses der Zufallspartie bekannt. Die Partieergebnisse aus den m Playouts werden gemittelt als Bewertung des Suchbaumblatts verwendet, von dem aus der übliche Minimax-Algorithmus angewandt wird. In Abbildung 12 ist ein Ausschnitt des bei der Monte-Carlo Tree Search entstehenden Suchbaums schematisch dargestellt.

Von der hier beschriebenen Grundvariante der Monte-Carlo Tree Search gibt es eine Reihe spezialisierter Versionen, die aber alle auf dem gleichen Prinzip der Monte-Carlo-Playouts von Partien zur Bewertung einzelner Kandidatenzüge bestehen. MCTS wurde in den letzten Jahren erfolgreich angewandt bei Zwei-Personen-Spielen mit perfekter Information wie Go [**?**, **?**] oder Othello [**?**], aber auch bei Spielen mit Zufallselementen oder mit unvollständiger Information, wie z.B. bei Skat [**?**], Backgammon [**?**], Bridge [**?**] oder Poker [**?**].

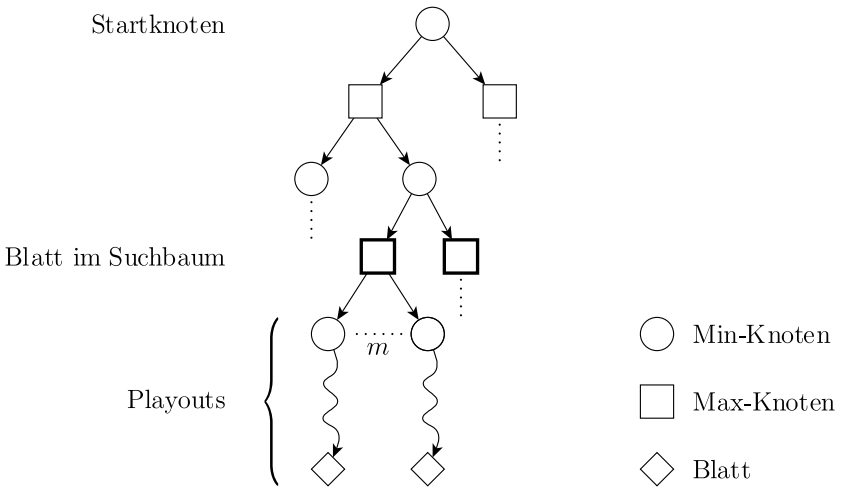

Abbildung 12: Ausschnitt aus einem Suchbaum der MCTS

Rollout-Algorithmen

Betrachtet werden jetzt kombinatorische Optimierungsprobleme mit folgender sequentieller Struktur: Eine Lösung ist ein Tupel $(u_1, \ldots, u_n)$, welches schrittweise aus den einzelnen Komponenten u_t zusammengesetzt wird. In jedem Schritt wird die Entscheidung über die Erweiterung der bisherigen Teillösung $(u_1, \ldots, u_t)$ um eine neue Komponente u_{t+1} getroffen.

Bei der Anwendung der Rollout-Algorithmen wird davon ausgegangen, dass für das betrachtete Problem eine Heuristik $\mathcal{H}$ vorliegt, die eine Teillösung $(u_1, \ldots, u_t)$ eindeutig zu einer vollständigen Lösung

$$(u_1, \ldots, u_t, u_{t+1}, \ldots, u_n) = \mathcal{H}(u_1, \ldots, u_t)$$

erweitert. Ein Rollout-Algorithmus besteht aus der iterativen Anwendung von $\mathcal{H}$ auf alle möglichen Erweiterungen $(u_1, \ldots, u_t, u')$ einer Teillösung $(u_1, \ldots, u_t)$. Diese Anwendungen von $\mathcal{H}$ werden als *Rollouts* bezeichnet, da hierbei die Teillösungen mittels der Heuristik zu vollständigen Lösungen ausgerollt werden, um bewertet zu werden. In Schritt $t+1$ wird dann um diejenige Komponente u^*_{t+1} erweitert, für die

$$u^*_{t+1} =_{u' \neq u_1, \ldots, u_t} f(\mathcal{H}(u_1, \ldots, u_t, u'))$$

gilt. Im Vergleich zur einfachen Anwendung der Heuristik $\mathcal{H}$ verzweigt der Rollout-Algorithmus den Suchbaum in jedem Schritt und untersucht dabei zusätzliche Varianten und deren vollständige Lösung wenn in den nacfolgenden Schritten $\mathcal{H}$ Anwendung findet.

In Abbildung 13 ist schematisch der Suchbaum nach t Schritten dargestellt, beginnend bei der bisher erreichten Teillösung $S^{(t)}$. Der Rollout-Algorithmus untersucht alle möglichen Erweiterungen, $S_1^{(t+1)}, \ldots, S_k^{(t+1)}$, der aktuellen Teillösung und wendet dort jeweils $\mathcal{H}$ bis zu vollständigen Lösung an. Die Menge der Lösungen $\mathbf{S}^{(n)} = \{S_1^{(n)}, \ldots, S_k^{(n)}\}$ wird verglichen, und es wird im Schritt $t+1$ diejenige Komponente ausgewählt, die zur besten Lösung $S^* \in \mathbf{S}^{(n)}$ führt. Im Vergleich dazu würde die einfache Anwendung der Heuristik $\mathcal{H}$ von $S^{(t)}$ aus über den dick gezeichneten Pfad direkt zur Lösung $S_1^{(n)}$ führen, ohne $S_2^{(n)}, \ldots, S_k^{(n)}$ zu betrachten.

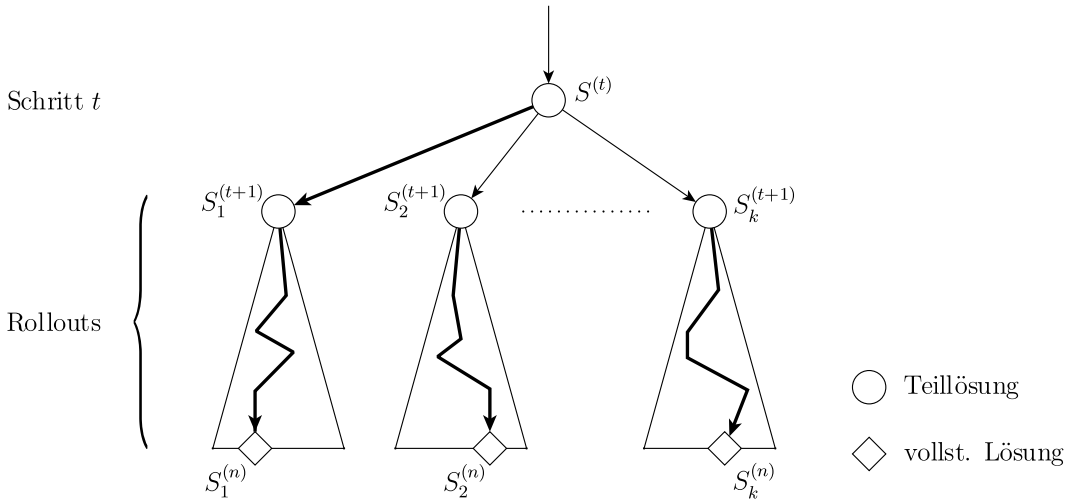

Abbildung 13: Suchbaum in einem Schritt des Rollout-Algorithmus

Als theoretisches Ergebnis ist bekannnt, dass unter bestimmten (relativ einfachen) Voraussetzungen an die Heuristik $\mathcal{H}$ deren Ergebnis durch den Rollout-Algorithmus höchstens verbessert werden kann [**?**]. Praktisch wurden Rollout-Algorithmen erfolgreich angewandt auf z.B. ein General Graph Search-Problem [**?**], eine Klasse stochastischer Scheduling-Probleme, dem sogenannten Quiz-Problem [**?**], das Sequential Ordering Problem [**?**] sowie das Resource-constrained Project Scheduling [**?**].

In Abbildung 14 ist anhand eines einfachen Suchbaums der Vergleich zwischen der Anwendung der Heuristik $\mathcal{H}$ und dem auf $\mathcal{H}$ basierenden Rollout-Algorithmus dargestellt. In den Blättern steht die entsprechende Bewertung der Lösung. Die an den Knoten angegebenen Werte sind jeweils die Ergebnisse bei Anwendung der Heuristik auf die Teillösung am Beginn der Kante. Der dick eingezeichnete Pfad führt zu der von der Heuristik direkt gefundenen Lösung mit $f = 2$. Bei Anwendung des Rollout-Algorithmus im ersten Schritt werden von den drei alternativen Teillösungen a, b und c ausgehend die über die gestrichelten Pfade dick umrandeten Blätter besucht. Es wird dann zu Teillösung c verzweigt, weil dort das Rollout von $\mathcal{H}$ die beste Bewertung ($f = 1$) liefert.

Ablauf des Monte-Carlo Rollout

Es folgt die Beschreibung des Ablaufs eines Schritts im Monte-Carlo-Rollout-Verfahren, beginnend in einem E-Knoten in der oben beschriebenen Spielbaum-Repräsentation eines sequentiellen Planungsproblems. In diesem Knoten ist durch den Entscheider eine Auswahl der Lösung S^* für Instanz Π aus der Menge der Kandidatenlösungen $\mathbf{S} = \{S_1, S_2, \ldots\}$ zu treffen. Jede der Entscheidungen $S_1, S_2, \ldots$ wird nacheinander betrachtet und wie folgt bewertet.

Ausgehend von dem U-Knoten, der nach Auswahl der Lösung $S_i \in \mathbf{S}$ entsteht, wird eine Anzahl m sogenannter *Monte-Carlo-Rollouts* durchgeführt. Ein Rollout ist dabei ein Pfad im Spielbaum von S_i bis zu einem Blatt. Dabei werden abwechselnd in E-Knoten und U-Knoten Züge unter den Kandidaten ausgewählt, die von den jeweiligen Zug-Generatoren vorgegeben werden. In U-Knoten werden Züge zufällig mit den Wahrscheinlichkeiten für Folge-

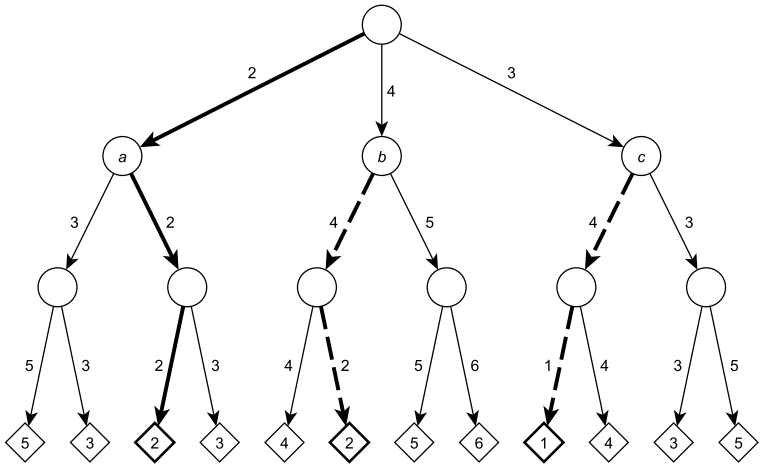

Abbildung 14: Beispiel eines Suchbaums des Rollout-Algorithmus

Instanzen Π' entsprechend des Maßes $\mathcal{P}$ gewählt, in E-Knoten wird zur Auswahl der Züge die Basisheuristik $\mathcal{H}$ angewandt. Das so erreichte Blatt wird mittels f bewertet, die Bewertungen werden über die m Rollouts zur Kandidaten-Entscheidung S_i gemittelt.

Abschliessend wird die bestbewertetste Entscheidung S^* ausgewählt und als aktuelle Lösung verwendet. Der Planungsprozess setzt mit der Abarbeitung der Lösung S^* in einem U-Knoten fort, bis dem Entscheider eine neue Folge-Instanz Π' vorgelegt wird. Hier wiederholt sich der Schritt zur Bewertung der Kandidatenlösungen und Auswahl.

In Abbildung 15 ist ein einzelner Bewertungsschritt dargestellt. Die Monte-Carlo-Rollouts beginnen in den U-Knoten nach Auswahl der Kandidatenlösungen S_i mit Anwendung von $\mathcal{P}$ zur Erzeugung von Folge-Instanzen, zur Lösung dieser wird im nachfolgenden E-Knoten die Basisheuristik $\mathcal{H}$ angewandt und so weiter.

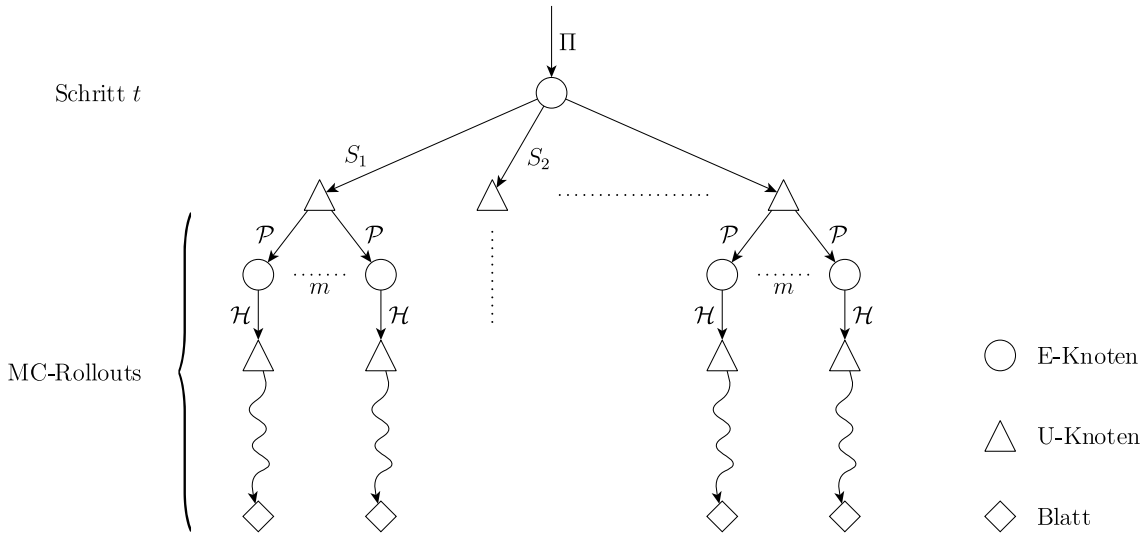

Abbildung 15: Schritt im Monte-Carlo-Rollout-Verfahren

Der Monte-Carlo-Aspekt bei diesem Verfahren liegt darin, dass wie bei der MCTS innerhalb des Suchbaums nicht die Bewertungsfunktion f zur Auswahl verwendet, sondern über in Monte-Carlo-Läufen bewertete Spielbaumblätter gemittelt wird. Der Rollout-Aspekt findet sich in der Erzeugung der Pfade wieder, die durch Ausrollen der Basisheuristik $\mathcal{H}$ entstehen.

III.4. Algorithmenentwurf

In diesem Abschnitt soll der Algorithmus, wie im vorangegangenen Abschnitt allgemein beschrieben, am Fallbeispiel Liebherr und dem 2-Server-Problem detailliert dargestellt werden.

III.4.1. Anwendungsfall Liebherr

Instanz Π. Simuliert wird ein Blocklager mit zwei Zugängen. Analog zum Anschnitt 2 ergibt sich eine Startinstanz wie folgt. Die Anzahl der Bahnen m ist flexibel, um verschiedene Füllgrade g des Lagers simulieren zu können. Am Beispiel des hier betrachteten Produktionsplans erhält man bei $m = 40$ einen Füllgrad von 80%. Entsprechend der realen Räumlichkeiten sind die Kapazitäten der Bahnen auf $k_i = 5,\ 1 \leq i \leq m$ und die Zugänge in eine Bahn mit $z_i = \{1, 10\},\ 1 \leq i \leq m$ festgelegt. Der Lagerbestand der einzelnen Türentypen zum Zeitpunkt $t = 0$ wurde dem von Liebherr zur Verfügung gestellten Produktionsplan der Schäumerei von einem Tag entnommen und zufällig in die Lagerplätze

$$B_i(t) = (b_{i1}, \ldots, b_{ik_i}) \text{ mit den Positionen}$$

$$b_{ij} = (a_{ij}, n_{ij}, t_{ij}) \in N \times N \times N$$

eingeordnet. Dazu wurden für jeden Türentyp a nach Möglichkeit nur volle Türenwagen $w = (a, n, t)$ mit $n = 18$ erzeugt und einer Position b_{ij} zugeordnet. Folglich existiert maximal ein Türenwagen pro Türentyp im Lager mit weniger als 18 Türen. Das zum Zeitpunkt $t = 0$ bekannte Einlagerprogramm P^+ ergibt sich aus alle Türenwagen die laut Produktionsplan in der Türenschäumerei im Laufe des Tages gefertigt werden und in das Lager eingelagert werden müssen. Dazu werden entsprechend des Produktionsplans alle Türenwagen $w = (a, n, t)$ mit dem jeweiligen Fertigstellungszeitpunkt t erzeugt die die Schäumerei im Laufe des Tages verlassen. Eine geordnete Liste dieser Wagen w mit $t_i \leq t_j,\ i < j$ stellt das Einlagerprogramm P^+ dar. Eine Liste *todo* der von den Endmontageinseln zu produzierende Lose wurde ebenfalls aus dem Produktionsplan der Türenschäumerei abgeleitet. Hierbei wurde eine maximale Losgröße mit 3 Türenwagen festgelegt. Aus dieser Liste wählen die 18 Montageinseln das nächste von ihnen zu produzierende Los $l = (a, n)$ zufällig aus. Wird ein Los gewählt, wird überprüft, ob alle notwendigen Türenwagen sich im Lager befinden und noch nicht für ein anderes Los reserviert worden sind. Bei positivem Test wird die entsprechende Anzahl von Türenwagen mit diesem Typ reserviert. Dabei erfolgt keine konkrete Zuweisungen zwischen Los und Türenwagennummern. Eine Bestellung $b = (a, n, t, I)$ für den ersten Türenwagen für dieses Los wird erzeugt. Die zeitlich sortierten Bestellungen aller Montageinseln bilden das Auslagerprogramm P^-. Eine Instanz Π beinhaltet damit folgende Informationen:

- Lagerbelegung: $B_i(t) = (b_{i1}, \ldots, b_{i5})\ 1 \leq i \leq m$
- Einlagerprogramm: P^+ als Liste von Wagen w
- Auslagerliste: P^- als Liste von Bestellungen b
- bis zu diesem Schritt anfallende Kosten: *cost*

Zeitschrittschleife. Ausgehend von dieser Startinstanz erfolgt die Simulation eines Tages über eine Zeitschrittschleife. Ein Tag umfasst 880 Minuten. Folgende Schritte umfassen einen Zeitschritt *time*:

1. Alle Auslagervorgänge $b = (a, n, t, I)$ mit $t = time$ aus P^- werden abgearbeitet.
   a) *Zug-Generator für die Umwelt*: Gemäß eines Wahrscheinlichkeitsmodells mit Maß $\mathcal{P}$ ist es möglich, dass ein anderer als in P^- geplanter Wagen ausgelagert werden muss.
   b) *Auslagerung und Kostenberechnung*: Die bei der Auslagerung notwendigen Umlagerungen (UL) werden aufsummiert, $cost+ = \#UL$.
   c) *Aktualisierung von P^+ und P^-*
2. Alle Wagen $w = (a, n, t)$ mit $t = time$ aus P^+ werden eingelagert.
   a) *Zug-Generator für den Entscheider*: Für jede Entscheidung welcher Lagerplatz für eine Wagen w genutzt werden soll, kommt das Monte-Carlo-Rollout-Verfahren $mcr.solver(\Pi, H)$ zur Anwendung. Diesem wird die aktuelle Instanz Π und die zu benutzende Heuristik H übergeben.
   b) *Einlagerung* Der Wagen w wird entsprechend der Lösung von (ii)(a) eingelagert.

Monte-Carlo-Rollout-Verfahren als Meta-Heuristik. Mit Hilfe des Monte-Carlo-Rollout-Verfahrens werden die Auswirkung verschiedener Entscheidungsmöglichkeiten über den Lagerplatz eines Wagens in Monte-Carlo-Rollouts durchgespielt und dadurch bewertet. Ausgangspunkt des Mc-Ro-Verfahrens in (ii)(a) ist ein E-Knoten mit der aktuellen Instanz Π. Hier stellt sich für den Entscheider die Frage an welche Position b_{ij} der nächste Türenwagen w aus P^+ eingelagert werden soll. Für diese Entscheidung stehen alle möglichen freien Plätze des Lagers zur Verfügung und bilden die Menge der Kanidaten $\mathbf{S} = \{S_1, S_2, \ldots\}$. Für jede dieser Positionen wird innerhalb des Mc-Ro-Verfahrens eine Kanidatenlösung erzeugt. Aus der Menge von Kanidatenlösungen $\mathbf{S}^{(N)} = \{S_1^{(N)}, S_2^{(N)}, \ldots\}$ wird vom Entscheider die beste Lösung ausgewählt. Es folgt die Beschreibung der Monte-Carlo-Rollouts zur Erzeugung der Kanidatenlösung $S_i^{(N)}$ im Detail.

Ein Mote-Carlo-Rollout (MCR). Der Wagen w wird an Position S_i eingelagert. Dieser so entstandene U-Knoten bildet den Ausgangspunkt für die Monte-Carlo-Rollouts. In einem MCR werden die Programme P^+ und P^- weiter abgearbeitet. Die Entscheidungen der Umwelt an den U-Knoten erfolgen nach dem gleichen Wahrscheinlichkeitsmodell wie auch in der Zeitschrittschleife. Der Entscheider wendet für seine Entscheidungen an den E-Knoten entsprechend des Rollout-Verfahrens die Heuristik H an. Das heißt, obige Zeitschrittschleife unterscheidet sich in einem MCR nur im Schritt (ii)(a). In einem MCR soll der gesammte Tag simuliert werden, d.h. es soll wie bei der MCTS die volle Tiefe *depth* betrachtet werden. Somit erhält man am Ende eines MCR als Bewertung die Anzahl der Umlagerungen die vom Startpunkt mit der Entscheidung S_i notwendig waren um alle Auslagerungen zu realisieren. Aufgrund des Wahrscheinlichkeitsmodells ist es notwendig einen solchen MCR für den Kanidaten S_i *width*-mal zu wiederholen um eine aussagekräftige Bewertung dieser Position zu erhalten. Das Mittel der Bewertungen aller Monte-Carlo-Rollouts ergibt die Kanidatenlösung $S_i^{(N)}$.

Zusammenfassend setzt sich das Monte-Carlo-Rollout-Verfahren in $mcr.solver(\Pi, H)$ aus folgenden Schritten zusammen:

1. *Generation der Alternativen*: Erzeugen einer endliche Menge von Kandidatenlösungen $\mathbf{S} = \{S_1, S_2, \ldots\}$ für Π.
2. *für jede Kanidatenlösung*:
   a) *Erzeugung der Instanz für S_i*: Der Wagen w der aktuellen Entscheidung wird auf dem Platz b_i der Alternative S_i eingelagert.
   b) *width-malige Durchführung eines MCR*:
   Es werden die Zeitschritte *time* bis *time* + *depth* simuliert. Nur der Zug-Generator für den Entscheider unterscheidet sich von obiger Zeitschrittschleife. Anstelle von $mcr.solver(\Pi, H)$ wird hier nur die Heuristik H angewendet.
   c) *Bewertung:* Die in den einzelnen MCR generierten Bewertungen werden gemittelt und ergeben die Lösung $S_i^{(N)}$ zur Kanidatenlösung S_i.
3. *Entscheidung:* Es wird für die aktuelle Entscheidung die Stellfläche b^* ausgewählt, die zur besten Lösung $S^* = \min\{S_1^{(N)}, \ldots, S_k^{(N)}\}$ führt.

Basisheuristiken.

- **short_in**
  Die Heuristik verwendet für die Entscheidung über einen Lagerplatz nur das Wissen über die aktuelle Lagersituation. Die Bahnen des Lagers werden gleichmäßig befüllt, d.h. ein Wagen wird immer in die leerste Bahn gestellt.

- **planned_in(x)**
  Die Heuristik verwendet neben dem Wissen über die aktuelle Lagersituation zusätzlich die Information über geplante Auslagerungen, d.h. Entscheidungen werden unter dem Wissen von P^- getroffen. Dazu wird für die nächsten x Türentypen in P^- die Lagerposition des vollsten Wagens der die wenigsten Kosten verursacht ermittelt. Die entsprechenden Bahnen dieser Positionen werden für die nächste Einlagerung blockiert. Aus den übrigen Bahnen wird die leerste Bahn gewählt und eine passende Position ermittelt.

Zug-Generator für die Umwelt. Der Grad der Dynamik gibt an, wie häufig eine aktuelle Auslagerung von der in P^- geplanten Auslagerung abweichen kann. Dafür gibt es zwei Ursachen. Entweder ein später geplanter Auslagervorgang muss sofort abgearbeitet werden oder der zu diesem Zeitpunkt geplante Auslagervorgang verschiebt sich nach hinten. Die mögliche zeitliche Verschiebung ist auf 5 Minuten festgelgt. Dabei nimmt die Wahrscheinlichkeit mit der Größe der Verschiebung ab.

Kostenberechnung. Um eine Bestellung $b = (a, n, t, I)$ zu erfüllen wird jeweils der Türenwagen mit dem Typ a ausgelagert der den geringsten Aufwand erfordert. Sind mehrere Türenwagen mit gleicher Anzahl an UL verfügbar, wird der Türenwagen mit der größeren Anzahl n an Türen gewählt. Als Kostenfunktion wird die Anzahl der notwendigen Umlagerungen aufsummiert.

Aktualisierung von P^+ und P^-. Nach Auslagerung einer Bestellung $b = (a, n, t, I)$ für die Montageinsel I wird für diese Insel eine neue Bestellung b' zum erwarteten Zeitpunkt t' erzeugt und der Auslagerliste P^- hinzugefügt. Diese Bestellung ordert entweder weitere Türen für das aktuelle Los oder bei Beendigung eines Loses mit dem gelieferten Türenwagen werden die Türen für ein zufällig gewähltes Folgelos bestellt. Das Auslagerprogramm P^- verändert sich folglich mit jeder abgearbeiteten Bestellung. Auch das Einlagerprogramm P^+ kann sich mit der Abarbeitung einer Bestellung verändern. Werden mehr Türen geliefert als für das aktuelle Los notwendig, werden die restlichen Türen nach Beendigung des Loses zurück in das Lager gebracht. Dazu wird ein Türenwagen $w = (a, n, t')$ zum erwarteten Zeitpunkt t' erzeugt und in das Einlagerprogramm einsortiert.

Generation der Alternativen. Neben dem Lagerplatz $S_1 = b_1$ der durch die Heuritik H bestimmt wurde, werden alle weiteren leeren Plätze b_i, $i = 2, 3, \ldots$ im Lager bestimmt, auf die ein Wagen gestellt werden kann ohne einen anderen leeren Platz zu blockieren. Damit erhält man einer endliche Menge von Kandidatenlösungen. $\mathbf{S} = \{S_1, S_2, \ldots\}$ für Π.

III.4.2. 2-Server-Problem

Um den Lösungsansatz besser analysieren zu können, soll als akademisches Beispiel das 2-Server Problem betrachtet werden. Hierbei muss die Bewegung von 2 Servern in einem abstrakten metrischen Raum gesteuert werden. Dabei müssen die Server Anfragen bearbeiten die an jedem Punkt des metrischen Raumes auftauchen können und im Voraus unbekannt sind. Um dieses Problem an das eigentliche Problem anzupassen, soll hier nicht davon ausgegangen werden, dass alle Anfragen komplett unbekannt sind. Vielmehr soll eine Startinstanz von bekannten Anfragen gegeben sein, die während der Abarbeitung gestört wird. Aufgrund der eingeschränkten Komplexität kann eine große Anzahl an Berechnungen durchführt werden. Diese ist notwendig für eine Analyse der Heuristiken und deren Verbesserung durch die Metaheuristik. Ein weiterer Vorteil der Betrachtung des akademischen Beispiels ergibt sich aus der Kenntnis der optimalen online Lösung, welche aus der offline Lösung hergeleitet werden kann. Diese optimale online Lösung bildet die Grundlage für die Analyse.

Offline Lösung. Die offline Lösung des 2-Server Problems ergibt sich wie folgt. Es seinen u_0 und v_0 die Startpositionen von Server 1 bzw. 2 und $A = A_1 = (a_1, \ldots, a_n)$ eine Anfragesequenz der Länge n. A_i bezeichne die aktuelle noch abzuarbeitende Anfragensequenz nach der $(i-1)$-sten Anfrage, d.h. $A_i = \{a_i, \ldots, a_n\}$. Gesucht sind zwei Folgen

$$U = (u_i)_{i=0}^{n_u},\ u_i \in \{a_1, \ldots, a_n\} \quad \text{und} \quad V = (v_i)_{i=0}^{n_v},\ v_i \in \{a_1, \ldots, a_n\}$$

die alle Anfragen aus A disjunkt abdecken mit

$$f(U, V) = \sum_{i=0}^{n_u - 1} d(u_i, u_{i+1}) + \sum_{i=0}^{n_v - 1} d(v_i, v_{i+1}) \to \min!$$

Dabei sei $d(.,.)$ die Manhattan-Metrik, bei welcher die Distanz zwischen zwei Punkten als die Summe der absoluten Differenzen ihrer Einzelkoordinaten definiert wird, d.h.

$$d(a, b) = \sum_i |a_i - b_i|.$$

Mit $e(u, v, A_i)$ werden die minimalen Kosten bezeichnet wenn Server 1 auf u steht, Server 2 auf v steht und die Folge A_i von Anfragen noch zu bedienen ist. Rekursiv ergibt sich damit

$$\begin{aligned} e(u, v, \emptyset) &= 0 \\ e(u, v, A_n) &= \min\{d(u, a_n), d(v, a_n)\} \\ e(u, v, A_i) &= \min\{d(u, a_i) + e(a_i, v, A_{i+1}), d(v, a_i) + e(u, a_i, A_{i+1})\},\ i = n-1, \ldots, 1 \end{aligned}$$

Dabei müssen nur für bestimmte Konfigurationen (u, v) die Werte entsprechend (**??**) berechnet werden. Einer der beiden Server muss auf der zuletzt bedienten Position stehen, der andere kann auf irgendeiner bereits bedienten Positionen stehen. Genauer:

$$e(u, v, A_i) = \begin{cases} e(u, v, A_i) & \text{für } u = a_{i-1}, v \in \{a_1, \ldots, a_{i-2}\} \cup \{v_0\} \\ & \quad v = a_{i-1}, u \in \{a_1, \ldots, a_{i-2}\} \cup \{u_0\} \\ 0 & \text{sonst} \end{cases}$$

$$\text{für } i = n, \ldots, 2 \text{ und}$$

$$e(u_0, v_0, A) = \min\{d(u_0, a_1) + e(a_1, v_0, A_2), d(v_0, a_1) + e(u_0, a_1, A_2)\}.$$

Ausgehend von $e(u_0, v_0, A)$ sind die Folgen U und V zur optimalen Abarbeitung der Anfragensequenz A festgelegt. Die dabei entstehenden Kosten sind der Wert von $e(u_0, v_0, A)$.

Instanz Π. Die aktuelle Instanz Π umfasst Informationen über den Standort der beiden Server u und v und die abzuarbeitende Anfragensequenz $A = (a_1, \ldots, a_n)$. Simuliert wird das Abarbeiten der gesammten Anfragensequenz.

Zeitschrittschleife. Folgende Schritte umfassen einen Zeitschritt t:

1. *Zug-Generator für den Entscheider*: Mittels Monte-Carlo-Rollout-Verfahren $mcr.solver(\Pi, H)$ wird für die aktuelle Anfrage a_t entschieden, welcher Server s diese Anfrage bedient. Dazu wird die aktuelle Instanz Π und die zu benutzende Heuristik H übergeben.
2. *Zug-Generator für die Umwelt*: Gemäß eines Wahrscheinlichkeitsmodells mit Maß $\mathcal{P}$ ist es möglich, dass eine andere Anfrage als a_t bedient werden muss. Die Koordinaten von a_t werden entsprechend geändert.
3. *Bedienen und Kostenberechnung*: Der zum Bedienen der Anfrage a_t zurückgelegte Weg des Servers s wird aufsummiert, $cost+ = d(s, a_t)$. Server s erhält die Position der Anfrage a_t.

Monte-Carlo-Rollout-Verfahren als Meta-Heuristik. $mcr.solver(\Pi, H)$ setzt sich analog zum Liebherr-Anwendungsfall aus folgenden Schritten zusammen:

1. *Generation der Alternativen*: Es existieren nur zwei Alternativen, d.h. $S = \{u, v\}$.
2. *für jede Kanidatenlösung*:
   a) *Erzeugung der Instanz für S_i*: S_i bedient die nächste Anfrage, d.h. $s = S_i$.

b) *width-malige Durchführung eines MCR*:
Es werden die Zeitschritte t bis $t + depth$ simuliert. Nur der Zug-Generator für den Entscheider unterscheidet sich von obiger Zeitschrittschleife. Anstelle von $mcr.solver(\Pi, H)$ wird hier nur die Heuristik H angewendet.

c) *Bewertung:* Die in den einzelnen MCR generierten Bewertungen werden gemittelt und ergeben die Lösung $S_i^{(N)}$ zur Kanidatenlösung S_i.

3. *Entscheidung:* Es wird für die aktuelle Anfrage der Server s^* ausgewählt, der zur besten Lösung $S^* = \min\{S_1^{(N)}, S_2^{(N)}\}$ führt.

Basisheuristiken. Ein Vorteil der schnellen Rechenzeit für das akademischen Beispiels ist, dass das Verhalten der Basisheuristiken mit verschiedene Instanzen über allen möglichen Störungssequenzen getestet werden kann. Folgende Basisheuristiken wurden getestet. Dabei sind $c_i(t)$ die vom Server i und $c(t)$ die von allen Servern erzeugten Kosten bis zur Abarbeitung der Anfrage a_t. i_t sei die Position des Servers i und a_t die Position der Anfrage zum Zeitpunkt t.

– **greedy**
Der Server bedient die nächste Anfrage, der dabei die wenigsten Gesamtkosten erzeugt.

$$\min_i \{c(t-1) + d(i_{t-1}, a_t)\}$$

– **nextexp**
Der Server bedient die nächste Anfrage, der dabei die wenigsten zu erwartenden Kosten erzeugt. Dazu ist ein Wissen der Wahrscheinlichkeit einer Störung notwendig.

$$\min_i \{p\,(c(t-1) + d(i_{t-1}, a_t)) + (1-p)\,(c(t-1) + d(i_{t-1}, a_{t+1}))\}$$

– **offlike**
Für die nächsten 3 Anfragen wird die offline Lösung des 2-Server Problems berechnet und anhand dieser Lösung die Entscheidung getroffen, welcher Server die nächste Anfrage bedient.

– **bell**
Im Sinne der Bellman'schen Gleichung soll neben den Kosten für das Bedienen der nächsten Anfrage zusätzlich eine Bewertung der Position der Server einen Einfluss nehmen. Sei s der Server der nach der greedy Heuristik die Anfrage a_t bedient, $\bar{s}$ der andere Server,

$$\begin{aligned} rating(\bar{s}) &= \max_{1 \le i \le n-t} \mid \bar{s} - a_{t+i} \mid \text{ die Bewertung der aktuellen Position von } \bar{s} \text{ und} \\ rating(a_t) &= \max_{1 \le i \le n-t} \mid a_t - a_{t+i} \mid \text{ die Bewertung der Position } a_t. \end{aligned}$$

Dann bedient $\bar{s}$ statt s die nächste Anfrage a_t, wenn

$$\mid s - a_t \mid \ge \mid \bar{s} - a_t \mid -0.5(rating(\bar{s}) - rating(a_t)).$$

– **balance**
Der Server bedient die nächste Anfrage, der die geringsten Einzelkosten bis zu dieser Anfrage erzeugt.

$$\min_i \{c_i(t-1) + d(i_{t-1} - a_t)\}$$

Zug-Generator für die Umwelt. Das stochastische Modell zur Darstellung der Ungewissheiten wurde wie folgt gewählt. Mit einer Wahrscheinlichkeit p wird die nächste tatsächliche Anfrage der nächsten Anfrage der Sequenz entsprechen. Mit einer Wahrscheinlichkeit von $(1-p)$ werden die nächsten zwei Anfragen in der Sequenz vertauscht. Die Störungen werden erst bekannt nachdem die Entscheidung getroffen wurde, welcher Server die nächste Anfrage bedienen soll. Für eine Anfragensequenz der Länge 21 ergeben sich somit 20 mögliche Störungen und folglich 2^{20} verschiedene Störsequenzen.

Optimale online Lösung. Für dieses Modell kann in Analogie zur offline Lösung eine optimale online Lösung hergeleitet werden. Es gelten die gleichen Anfangsbedingungen wie bei der offline Lösung. A_i bezeichne wieder die aktuelle noch abzuarbeitende Anfragensequenz nach der $(i-1)$-sten Anfrage. Allerdings ist hier zusätzlich Aufgrund der Ungewissheit zu beachten, dass es zu Vertauschungen innerhalb der Anfragensequenz kommt. Für A_i ergibt sich daher $A_i = (q_i, a_{i+1}, \ldots, a_n)$ wobei $q_i \in \{a_1, \ldots, a_i\}$ und $e(u, v, A_i)$ stellt die erwarteten minimalen Kosten dar. Rekursiv ergibt sich damit

$$\begin{aligned}
e(u, v, \emptyset) &= 0 \\
e(u, v, A_n, q_n) &= \min\{d(u, q_n), d(v, q_n)\}, \\
&\quad \text{für } q_n \in \{a_1, \ldots, a_n\} \\
e(u, v, A_i, q_i) &= \min\Big\{p\big(d(u, q_i) + e(q_i, v, A_{i+1}, a_{i+1})\big) + (1-p)\big(d(u, a_{i+1}) + e(a_{i+1}, v, A_{i+1}, q_{i+1})\big), \\
&\qquad p\big(d(v, q_i) + e(u, q_i, A_{i+1}, a_{i+1})\big) + (1-p)\big(d(v, a_{i+1}) + e(u, a_{i+1}, A_{i+1}, q_{i+1})\big)\Big\}, \\
&\quad \text{für } q_i \in \{a_1, \ldots, a_i\}
\end{aligned} \tag{1}$$

Dabei müssen wieder nur für bestimmte Konfigurationen (u, v) die Werte $e(u, v, A_i, q_i)$ entsprechend (1) berechnet werden.

$$e(u, v, A_i, q_i) = \begin{cases}
e(u, v, A_i, a_i), & \text{falls } q_i = a_i \quad \text{für } u = a_{i-1}, v \in \{a_1, \ldots, a_{i-2}\} \cup \{v_0\} \\
& \qquad\qquad\quad\; v = a_{i-1}, u \in \{a_1, \ldots, a_{i-2}\} \cup \{u_0\} \\
e(u, v, A_i, q_i), & \text{falls } q_i \neq a_i \quad \text{für } q_i \in \{a_1, \ldots, a_i - 1\} : \\
& \qquad\qquad\quad\; u = a_i, v \in \{a_1, \ldots, a_{i-1}\} \cup \{v_0\} \setminus \{q_i\} \\
& \qquad\qquad\quad\; v = a_i, u \in \{a_1, \ldots, a_{i-1}\} \cup \{u_0\} \setminus \{q_i\} \\
0 & \text{sonst}
\end{cases}$$

$$\text{für } i = n, \ldots, 2 \text{ und}$$

$$\begin{aligned}
e(u_0, v_0, A) = \min\Big\{&p\big(d(u_0, a_1) + e(a_1, v_0, A_2, a_2)\big) + (1-p)\big(d(u_0, a_2) + e(a_2, v_0, A_2, a_1)\big), \\
&p\big(d(v_0, a_1) + e(u_0, a_1, A_2, a_2)\big) + (1-p)\big(d(v_0, a_2) + e(u_0, a_2, A_2, a_1)\big)\Big\}.
\end{aligned}$$

Analog liefert $e(u_0, v_0, A)$ die optimalen erwarteten Kosten die bei der Abarbeitung der Anfragensequenz A entstehen. Und die Folgen U und V mit den zu bedienenden Positionen für Server 1 bzw. 2 kann aus den Berechnungen ermittelt werden. Die optimale online Lösung soll in den Vergleich der Basisheuristiken mit einbezogen werden.

III.5. Anwendungsszenarien

III.5.1. 2-Server-Problem als akademisches Beispiel

Diese Heuristiken wurden anhand von 10 Instanzen der Länge 21 über alle 2^{20} möglichen Störfälle getestet. Alle Störfälle mit der gleichen Anzahl von Störungen, k, werden dabei zusammengefasst. Deren Summe aus Eintrittswahrscheinlichkeit mal verursachte Kosten über alle möglichen Störungen wird abgebildet.

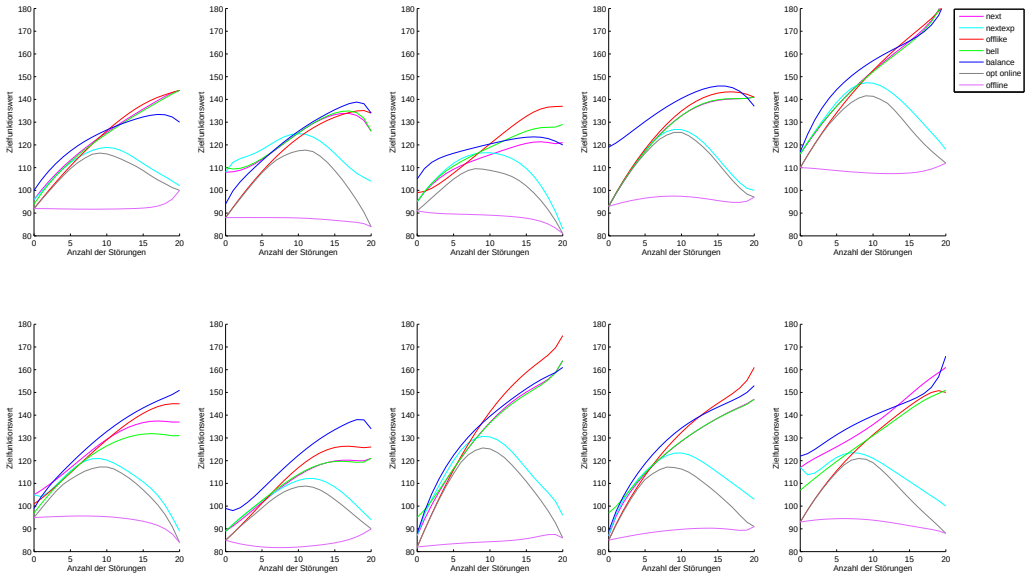

Abbildung 16: Simulation über alle Störfälle bei 10 Beispielinstanzen

Approximationsfehler bei Reduktion der Simulationszahl. Anschließend wurde untersucht mit welcher Anzahl von Simulationen der Verlauf der Ergebnisse einer Instanz ausreichend gut approximiert werden kann. Hierbei werden zu den Störanzahlen $5 \leq k \leq 15$ nicht alle $\binom{20}{k}$ Störsequenzen betrachtet, sondern nur ein zufällig ausgewählter Teil davon. Dazu wird das $(1-\alpha)$-Konfidenzintervalls (KI) für den Erwartungswert eines unbekannt verteilten Merkmals mit unbekannter Varianz betrachtet. Das 95%-Konfidenzintervall enthält bei durchschnittlich 95 von 100 gleichgroßen Zufallsstichproben den Erwartungswert. Falls der Stichprobenumfang n genügend groß ist, kann aufgrund des zentralen Grenzwertsatzes das Konfidenzintervall bestimmt werden. Dazu seinen $X_1, \ldots, X_n$ unabhängige und identisch verteilte Zufallsvariablen. Der Erwartungswert wird anhand der Stichprobe geschätzt, dazu wird die Summe aus der Eintrittswahrscheinlichkeit mal der verursachten Kosten c_k über alle möglichen Störungen

gebildet

$$\bar{X} = \sum_k w_p^k \quad \text{mit} \quad w_p^k = p^k(1-p)^{n-k}c_k.$$

Die Varianz der Grundgesamtheit durch die korrigierte Stichprobenvarianz geschätzt mit

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})^2.$$

Damit ergibt sich das $(1-\alpha)$-Konfidenzintervall als

$$\left[\bar{X} - z(1-\frac{\alpha}{2})\frac{s}{\sqrt{n}}, \bar{X} + z(1-\frac{\alpha}{2})\frac{s}{\sqrt{n}}\right],$$

dabei ist $z(1-\frac{\alpha}{2})$ das $(1-\frac{\alpha}{2})$-Quantil der Standardnormalverteilung. In folgender Tabelle werden die maximalen Konfidenzintervallängen für verschiedene Stichprobenumfänge n dargestellt.

n	1'000'000	100'000	10'000	5'000	4'000
max. KI-länge	0.06	0.17	0.53	0.75	0.84

Die Reduktion der Anzahl der Simulationen pro Störanzahl ist für den Fall $n = 4000$ in folgender Tabelle dargestellt. Der daraus resultierende Approximationsfehler für die 10 Testinstanzen ist in 17 abgebildet.

k	0,20	1,19	2,18	3,17	4,16	5,15	6,14	7,13	8,12	9,11	10
$\binom{20}{k}$	1	20	190	1140	4845	15504	38760	77520	125970	167960	184756
n	1	20	190	1140	4845	4000	4000	4000	4000	4000	4000

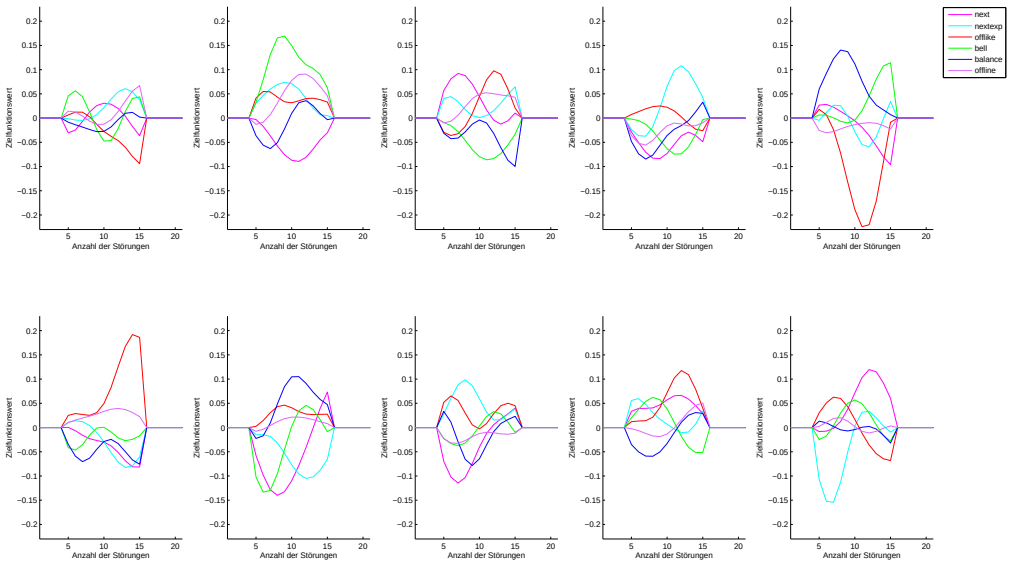

Abbildung 17: Approximationsfehlers aufgrund der Reduktion der Simulationszahl

In Abbildung 18 zeigt das 95%-Konfidenzintervall zu allen Basisheuristiken am Beispiel der zweiten Musterinstanz mit der reduzierten Anzahl an Simulationen wie oben beschrieben, d.h. $n = 4000$.

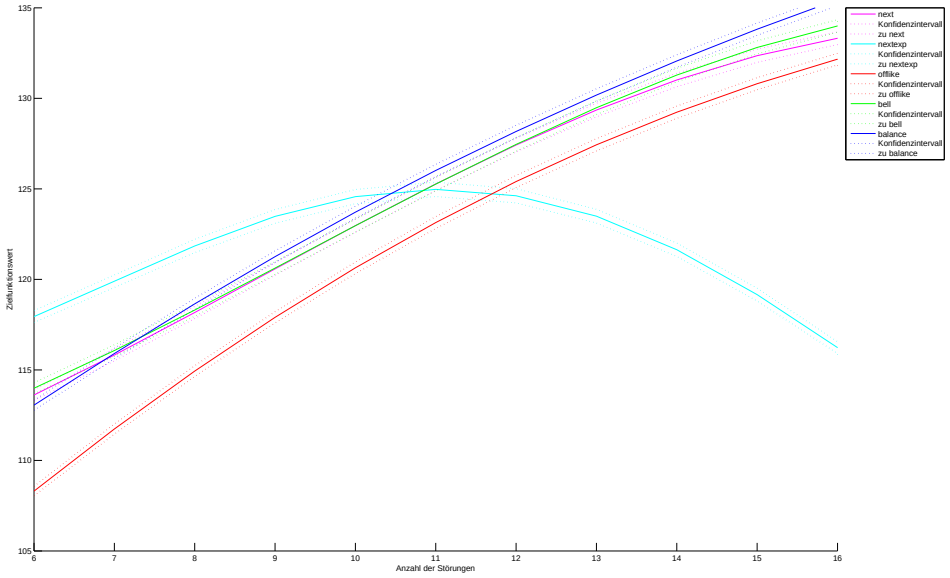

Abbildung 18: 95%-Konfidenzintervall am Beispiel 2 bei n=4000.

Bisher wurde die Anzahl der Simulationen betrachtet die notwendig sind um das Verhalten der Heuristik an einer Beispielinstanz ausreichend gut zu approximieren. Jetzt soll betrachtet werden, wieviele Ergebnisse von Beispielinstanzen gemittelt werden müssen, um eine allgemeine Aussage zur Qualität der einzelnen Heuristken treffen zu können. In Abbildung 16 ist deutlich zu sehen, dass die Ergebnisse der einzelnen Heuristiken bei den verschiedenen Beispielinstanzen deutlich voneinander abweichen können. Ein Trend ist aber deutlich zu sehen. Dies bestätigt sich auch in der Mittelung der Ergebnisse für diese 10 Instanzen. In der folgenden Graphik ist die Mittelung der Ergebnisse aus den Abbildungen 16 und 17 dargestellt.

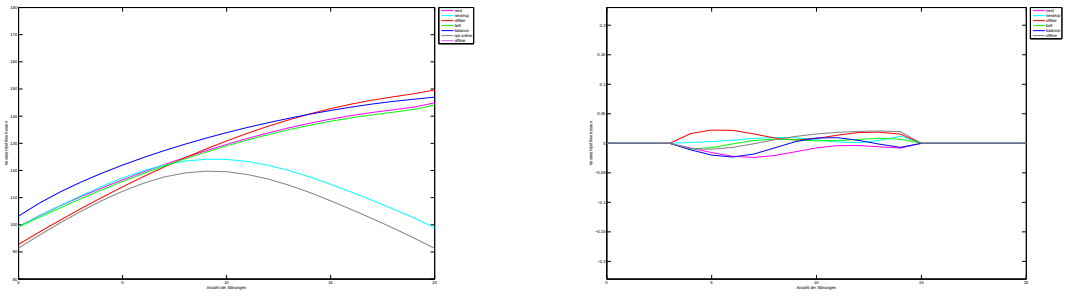

Abbildung 19: Mittelung der Ergebnissverläufe der Heuristiken (links), Mittelung des Approximationsfehlers aufgrund der Reduktion der Simulationszahl

Für die Untersuchung über die Anzahl der Instanzen stehen die Ergebnisse von 100 Beispielinstanzen zur Verfügung. Der Referenzverlauf ergibt sich aus der Mittelung der 100 Instanzverläufe. Abbildung 20 zeigen dieses Mittel links absolut und rechts relativ zur optimalen online Lösung.

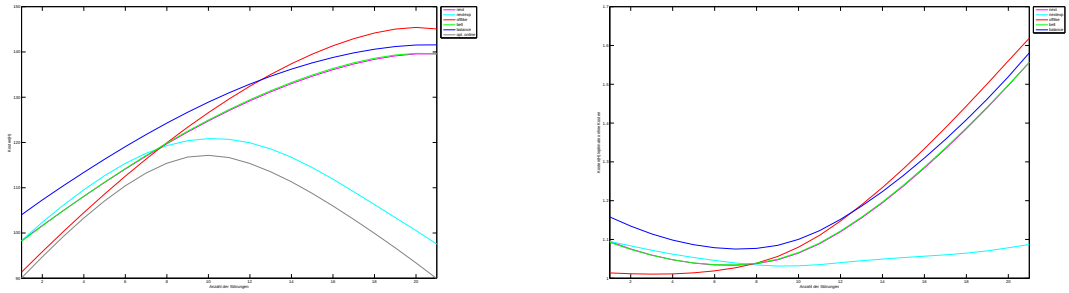

Abbildung 20: Mittelung von 100 Ergebnissverläufen der Heuristiken (links), Mittelung von 100 Ergebnissverläufen der Heuristiken relativ zur optimlane online Lösung

Diese Verläufe stellen den ëxaktenm̈ittleren Verlauf dar. In Abhängigkeit von der Anzahl an Beispielen entsteht ein Approximationsfehler welche nun analysiert werden soll.

Als Referenzinstanzen dienen ab hier: [4 5 11 13 18 19 23 29 30 35 41 43 50 53 59 65 71 73 77 79]

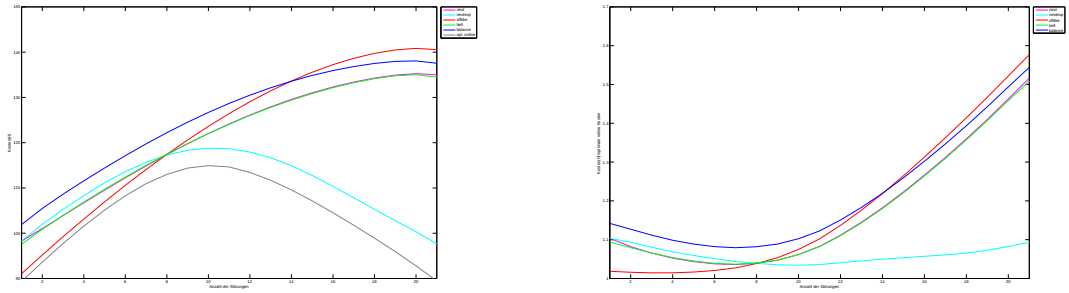

Abbildung 21: Mittelung von 20 Referenzverläufen (links), Mittelung von 20 Referenzverläufen relativ zur optimlane online Lösung

III.5.2. Sackgassenlager als Vorstudie

Als erstes Anwendungsszenario wurde ein Sackgassenlager mit 25 Reihen und jeweils 5 Stellplätzen betrachtet. Die Ein- (EL) und Auslagerungen (AL) wurden zufällig erzeugt. Über 100 Zeitschritte erfolgt jeweils eine AL und eine EL. Die Türentypen die ein- bzw. ausgelagert werden sollen wurden einmal gleichmäßig verteilt angenommen. In Anlehnung an die Gegebenheiten bei Liebherr wurde auch eine Verteilung bei der 40% der Türentypen 80% des Tagesumsatzes ausmachen benutzt. Es wurden verschiedene Heuristiken getestet und die Anzahl der notwendigen Umlagerungen (UL) verglichen.

- **random_in:** EL in eine zufällige Bahn.
- **short_in:** EL in eine Bahn mit den meisten freien Plätzen.
- **long_in:** EL in eine Bahn mit den wenigsten freien Plätzen.
- **small_frequ_in:** EL in eine Bahn in der der vorderste Wagen eine kleine Wahrscheinlichkeit hat ausgelagert zu werden.
- **short_multi_in:** EL in die kürzeste Bahn in der ein Türentyp an erster Stelle steht, der auch in einer anderen Reihe an erster Stelle steht.
- **bnb_in:** Anwendung eines Branch&Bound Verfahrens über einen bestimmten Horizont.
- **bnb_roll_in:** Anwendung eines Branch&Bound Verfahrens über einen rollenden Horizont.
- **planned_in(x):** Bei der EL wird berücksichtigt wo sich die nächsten x AL befinden. Entsprechende Bahnen werden nicht als EL Möglichkeit berücksichtigt.

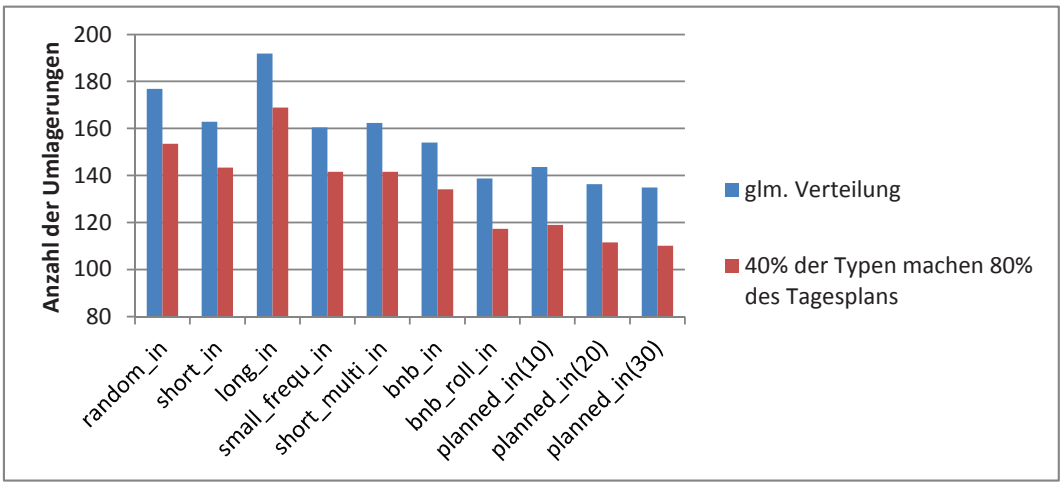

Abbildung 22: Anzahl der notwendigen UL bei Verwendung verschiedener Heuristiken. Jeweils für gleichverteilte Türentypen und 40%/80% verteilte Türentypen.

Die Heuristik planned_in liefert identische Ergebnisse wie die Heuristik die ein Branch&Bound Verfahren einsetzt. Allerdings ist der Rechenaufwand deutlich geringer. Für die Pilotanwendung wird daher planned_in als Basisheuristik verwendet.

Im nächsten Schritt wurde die Reihenfolge der geplanten Auslagerungen gestört. Im Falle einer Störung wird die nächste geplante AL nach hinten verschoben. Der Grad der Dynamik gibt an wie häufig eine solche Störung auftritt.

Heuristiken die ohne Wissen über zukünftige AL arbeiten liefern unabhängig vom Grad der Störung identische Ergebnisse. Dagegen werden Heuristiken die zukünftig geplante Ereignisse mit in ihre Entscheidung einbeziehen mit steigendem Grad der Störung immer schlechter. Zu

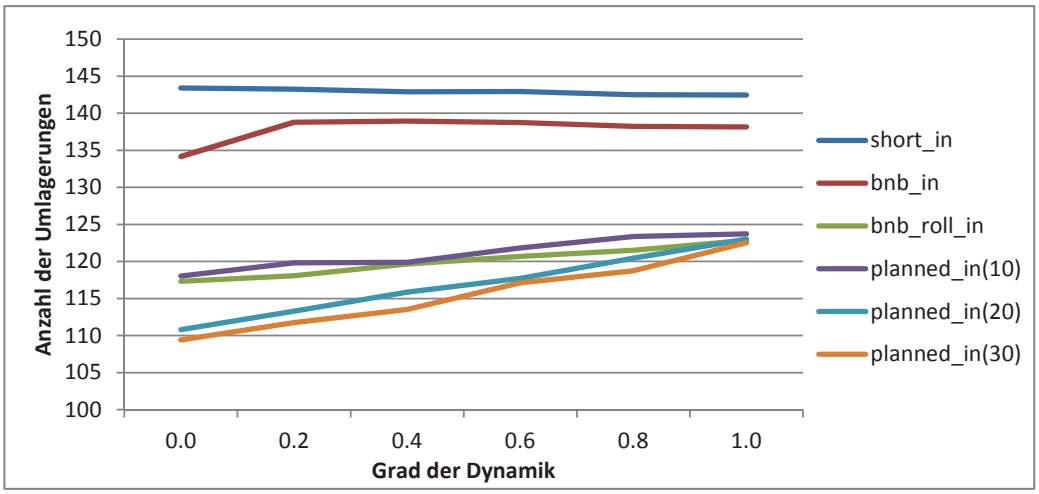

Abbildung 23: Anzahl der notwendigen UL bei Störung der AL Reihenfolge für verschiedene Heuristiken.

erkennen ist aber, dass diese trotzdem deutlich besser sind als erstgenannte Heuristiken. Die im Projekt entwickelte Meta-Heuristik soll dieser Verschlechterung entgegenwirken. Ergebnisse dazu werden im nächsten Abschnitt detailliert diskutiert.

III.6. Pilot

In enger Zusammenarbeit mit der Firma Liebherr wurden die Vorstudien zu Sackgassenlagern an die realen Gegebenheiten der Türenproduktion und deren Zwischenlagerung für die Kühlschrankproduktion angepasst.

Lagerlayout

Es stehen für die Türenwagenlagerung zwei Bereiche zur Verfügung. Aus Platzgründen soll auf den Flächen Blocklager mit nur einem Zugang organisiert werden. Lager 1 besteht aus 33 Reihen mit 3 Türenwagen und 31 Reihen mit 2 Türenwagen hintereinander. Lager 2 umfasst 14 Reihen mit 4 Wagen und jeweils 7 Reihen mit 5, 6 und 8 Wagen hintereinander. Die Bereitstellungsfläche dient auf der einen Seite zum Abstellen von Türenwagen die eingelagert werden sollen. Zum anderen stellt ein Lagerverwalter auf dieser Fläche alle Türenwagen bereit, die von den Montageinseln in nächster Zeit gebraucht werden. Die Bereitstellungsfläche umfasst insgesamt 34 Plätze mit Einzelzugriff.

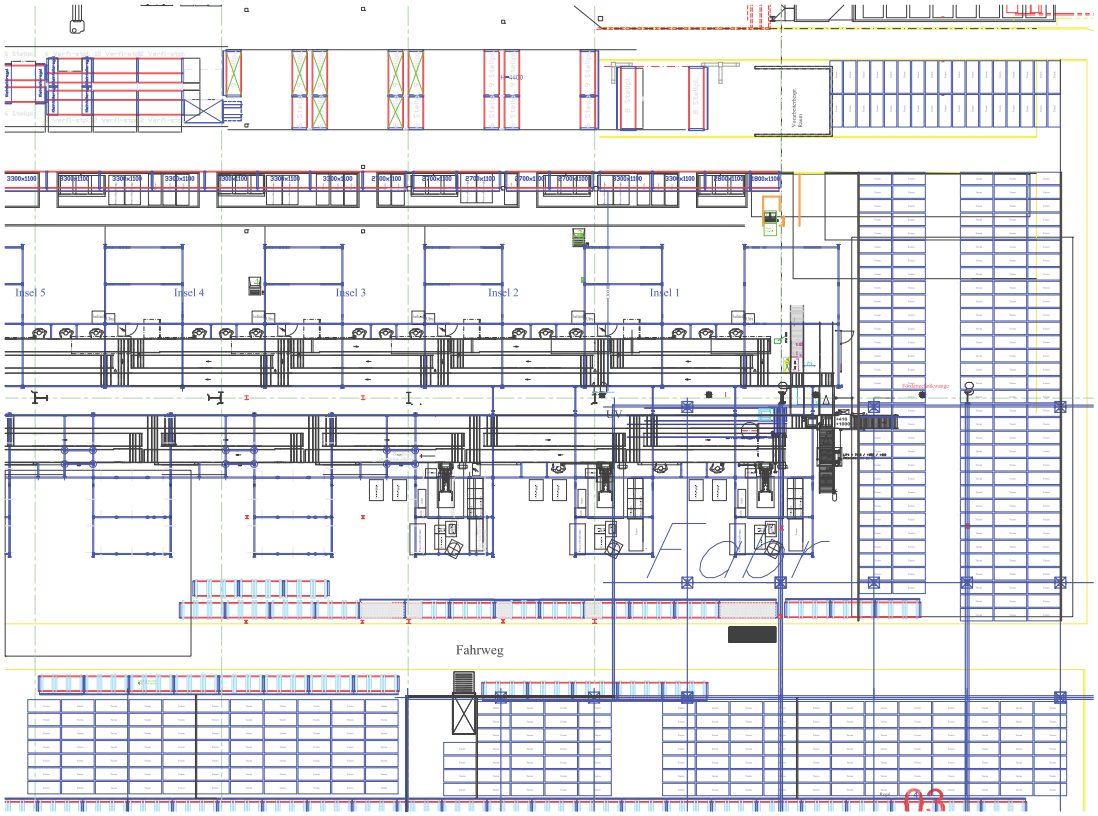

Abbildung 24: Geplante Lagerflächen in der Produktionshalle von Liebherr.

Zusammenhänge und Ablauf

Lagereingänge sind Türenwagen aus der Türenschäumerei. Alle Wagen aus der Türenschäumerei werden beim Durchfahren des RFID-Gates im System aufgenommen. Es wird erfasst welche

Türen sich auf dem Wagen befinden und wie viele. Dabei kann es sich um Kühlteiltüren, Gefrierteiltüren oder eine Kombination beider handeln. Anhand der Produktionsdaten wird dem Mitarbeiter mitgeteilt, ob der Wagen in Lager 1, 2 oder zu einer zusätzlichen Baugruppenmontage gefahren werden muss. Geht der Wagen direkt in eines der beiden Lager, so muss er vom Mitarbeiter auf eine entsprechende Bereitstellungsfläche geschoben werden. Ist eine zusätzliche Baugruppe notwendig wird der Wagen an den vorgesehenen Montageplatz gefahren. Nach der Montage an der Baugruppe erfolgt ein erneuter Scan des Wagens und dem Mitarbeiter wird mitgeteilt zu welcher Lagerbereitstellungsfläche dieser Wagen gebracht werden muss.

Lagerausgänge ergeben sich aus der Produktion der Fertigungsgeräte an den einzelnen Montageinseln. Für die einzelnen Montagebänder werden Listen mit zu produzierenden Losen vorgegeben. Die Mitarbeiter an einer Montageinsel wählen anhand dieser Listen welchen Fertigungsgerättyp sie als nächstes produzieren wollen und in welcher Stückzahl. Zu große Losgrößen werden von den Mitarbeitern eigenständig nochmal in kleinere Lose unterteilt. Zu jedem Schichtbeginn muss jede Montageinsel angeben, wann der erste Wagen benötigt wird. Anhand durchschnittlicher Produktionszeiten für die einzelnen Fertigungsprodukte errechnet der Algorithmus zu welchen Zeitpunkten welche Türentypen für welche Insel bereitgestellt werden muss. In diesen Berechnungen ist ein Zeitpuffer von 15 Minuten beinhaltet.

Ein Wagen aus der Türenschäumerei auf der Bereitstellungsfläche wird vom Lagerverwalter eingelagert. Dazu scannt er den Wagen und der Algorithmus liefert einen Platz im Lager an dem der Wagen eingelagert werden soll. Um eine effektive Lagerverwaltung zu gewährleisten wird vom Algorithmus gleichzeitig eine Wagennummer und die Position ausgegeben, der ausgelagert werden soll. Dieser Wagen wird nach erfolgreicher Auslagerung vom Lagerverwalter gescannt und auf der Bereitstellungsfläche abgestellt. Die Montageinseln holen sich die Türenwagen die sie brauchen von dieser Bereitstellungsfläche ab.

In folgender Abbildung wird der Ablauf einer Einlagerung noch einmal schematisch dargestellt

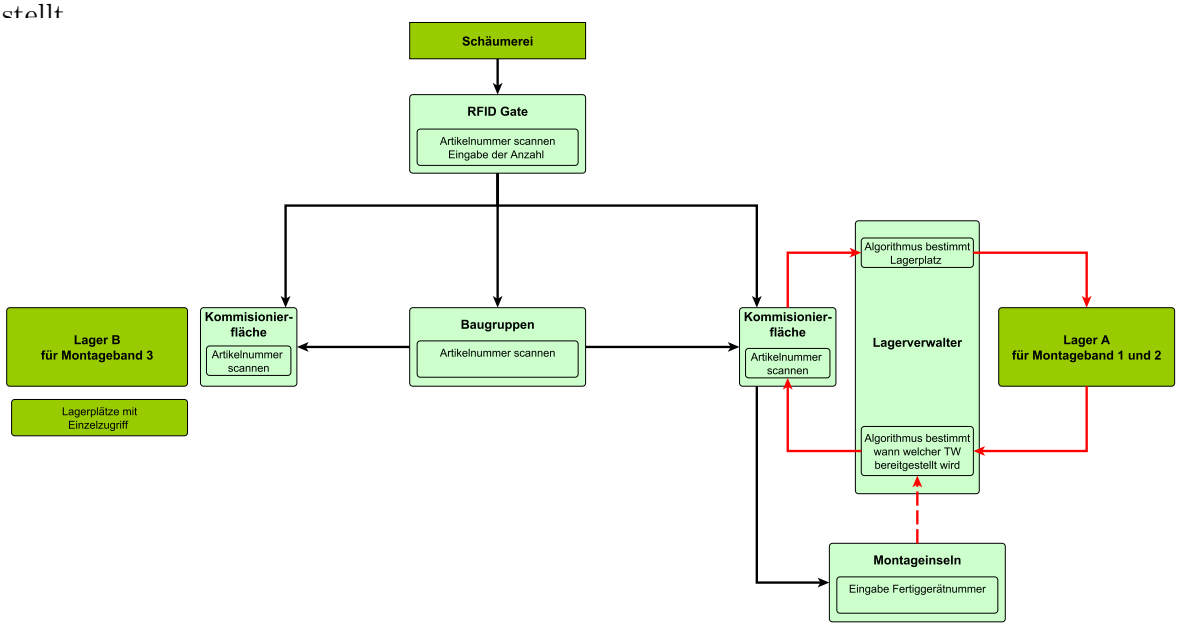

Abbildung 25: Ablaufplan einer Einlagerung.

Bleiben bei der Produktion an einer Montageinsel Türen übrig wird der Wagen wieder zur Bereitstellungsfläche gebracht und dort vom Lagerverwalter erneut erfasst. Sind weniger als 8 Türen übrig werden diese nicht wieder ins Lager eingelagert, sondern auf einem gemischten Sammelwagen gelagert. Werden im Gegenzug nur kleine Mengen von einer Montageinsel benötigt, wird geprüft ob die Menge sich auf einem solchen Sammelwagen befindet.

Um die Komplexität der Pilotanwendung einzuschränken, soll nur ein Teil der Türenlagerung über die Software gesteuert werden. Dazu wird die kleinere Lagerfläche 1 und deren Bereitstellungsfläche von Liebherr bereitgestellt. Um eine Trennung zwischen den Lagern auch für die Mitarbeiter eindeutig zu machen, wurde entschieden im Piloten nur Standgeräte zu betrachten. Einbaugeräte werden wie bisher auf der noch freien Lagerfläche gelagert und nicht vom System erfasst.

Simulation

In der dem Piloten vorangestellten Simulation zur Abschätzung von personellem und platzmäßigem Bedarf, sowie der Leistungsfähigkeit des Algorithmus wurden beide Lager betrachtet. Dabei erfolgte die Trennung ebenfalls zwischen Einbau- und Standgeräten. Der Aufbau der Simulation wurde bereits im Abschnitt III.4 detailliert beschrieben.

Der Aufrufbaum der einzelnen Funktionen entspricht dem Ablaufplan im vorangegangen Abschnitt und wird in Abbildung 26 dargestellt.

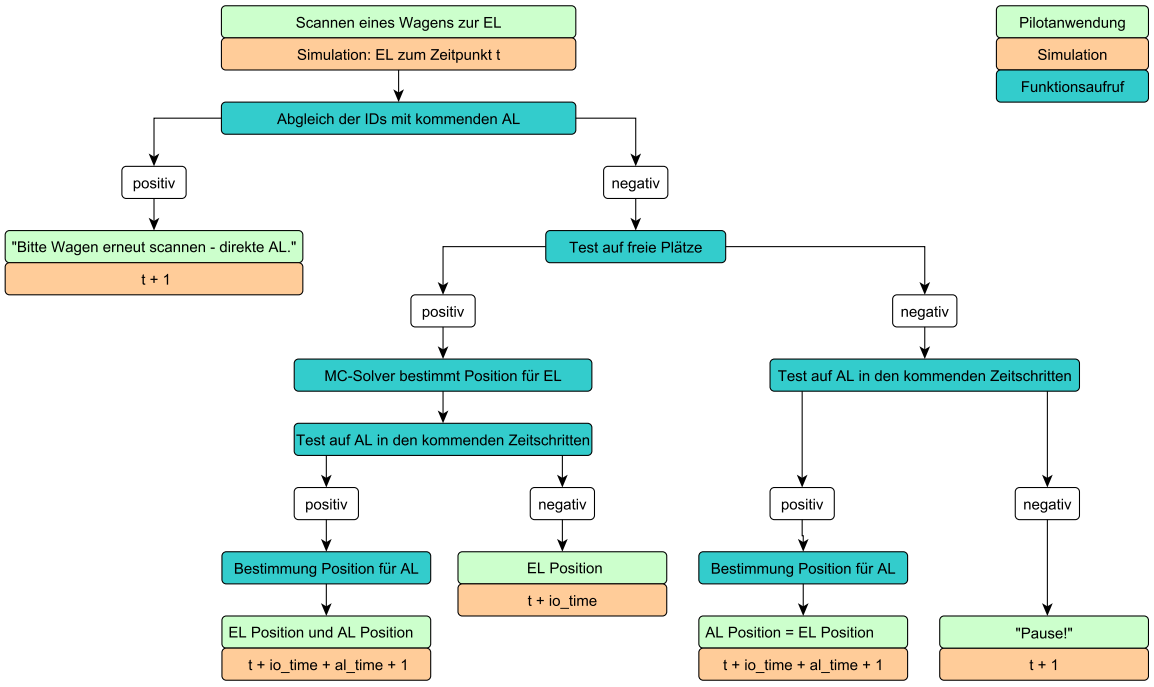

Abbildung 26: Funktionsaufrufe nach dem Scannen eines Türenwagens zur Einlagerung.

Für die Simulation wurden reale Daten der Produktion vom 4.5.2011-9.5.2011 verwendet. Eine Liste der in diesem Zeitraum gefertigten Lose mit einer Zuteilung auf die verschiedenen Montagebänder bildet die Grundlage der Simulation. In der Simulation wird für jede Montageinseln aus der jeweiligen Liste das nächste zu produzierende Los für welches bereits alle

Türen im Lager vorrätig sind gewählt. Auf diese Art und Weise werden alle Lagerausgänge zeitlich bestimmt.

Der Simulation der Lagereingänge liegt der Produktionsplan der Türenschäumerei zugrunde. Hier wurde anhand des schichtweisen Formenbelegungsplans und der Angabe der zu produzieren Türen berechnet, wann ein fertiger Türenwagen im Lager eintrifft.

Vereinfachend wurden Baugruppen nicht mit in die Simulation aufgenommen. Der dadurch entstehende zeitliche Unterschied einer Einlagerung wurde durch zusätzliche fiktive Reihen im Lager ausgeglichen.

Der Simulationszeitraum umfasst 3 Werktage. Die Türenproduktion erfolgt im 3-Schichtsystem, die Endmontage im 2- Schichtsystem. Türen die in der Nachtschicht gefertigt werden, werden erst im Laufe der Frühschicht ins Lager eingelagert. Für die Vereinfachung bzgl. der Baugruppen wurden in beiden Lagern 30 zusätzliche Lagerplätze betrachtet. In Abbildung 27 ist der simulierte Lagerzustand dargestellt. Auf der linken Seite sind die Daten von Lager 1 (Lager für alle Standgeräte), auf der rechten Seite die Daten von Lager 2 (Lager für alle Einbaugräte) abgebildet. Im oberen Diagramm stellt sich der zeitliche Verlauf der Anzahl der AL, EL und UL dar. Die Anzahl der freien Plätze im jeweiligen Lager über die Zeit ist im mittleren Diagramm verdeutlich. Das untere Diagramm spiegelt das Ergebnis wieder wie viele Umlagerungen pro Auslagerung notwendig sind.

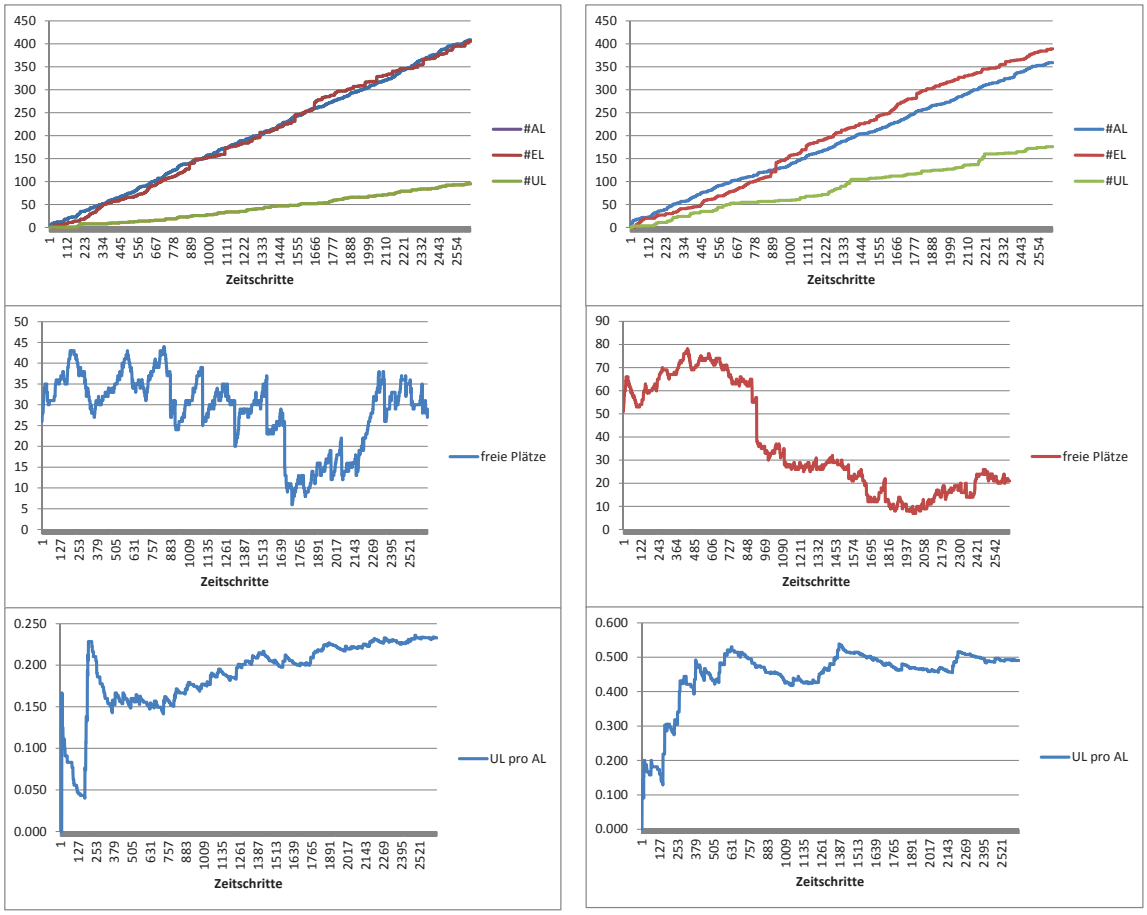

Abbildung 27: Lagerzustand im zeitlichen Verlauf - Lager 1 (links), Lager 2 (rechts)

Für die Basisheuristik planned_in(x) wurde getestet, wie sich das Einbeziehen verschiedener Anzahlen von AL x auf das Ergebniss auswirkt. An Abbildung 28 ist die Anzahl der UL pro AL für Lager 1 und 2, mit und ohne Rollout, für verschiedene x dargestellt.

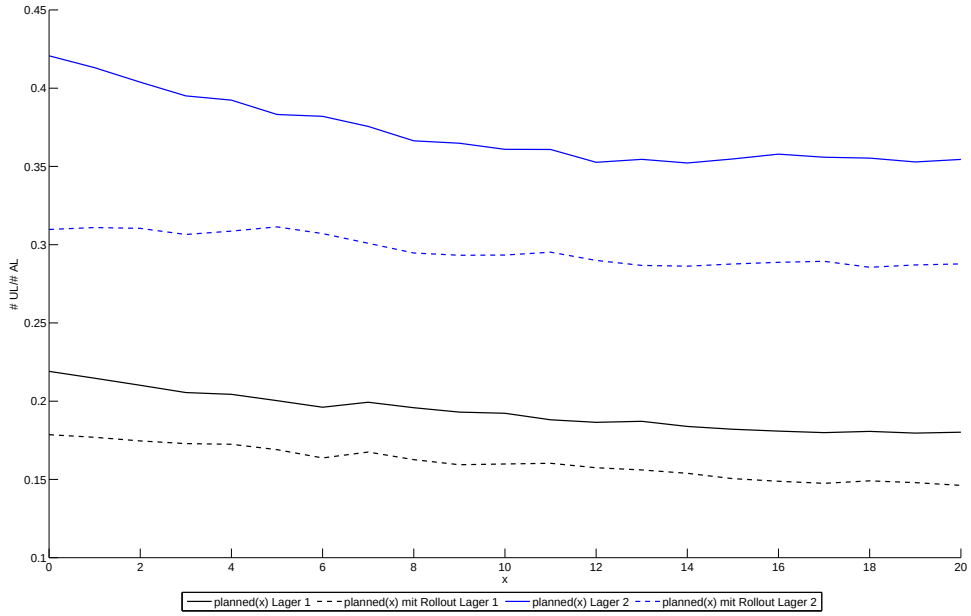

Abbildung 28: UL/AL unter Verwendung von planned_in(x) für verschieden x.

Ungewissheit

Die Ungewissheit die für unsere Betrachtungen von Bedeutung sind, ergibt sich aus der Abweichung der Produktionszeiten t_{FG} die eine Montageinsel für ein Endgerät braucht. Diese kann schwanken, je nachdem ob ein Team besonders schnell oder ehr langsam ist. Eine Montageinsel kann zum Beispiel auch langsamer als t_{FG} sein, weil an dieser Insel Personal fehlt. Die Ungewissheit über die Abweichung von der Produktionszeit t_{FG} soll über einen Prozentsatz β angegeben werden. In der Produktion können die Zeiten t_{FG} je nach Montageinsel und Schicht abweichen. Daher wird für jede Insel m und jede Schicht s ein für die Simulation der Ungewissheit notwendiger Parameter

$$\alpha_{ms} \in (1-\beta, 1+\beta)$$

zufällig erzeugt.

Der geplante Zeitpunkt t_{i+1} für den nächsten Wagen der für eine Montageinsel bereitgestellt werden muss, ergibt sich aus t_{FG}, dem Zeitpunkt t_i zu dem der letzte Wagen bereitgestellt wurde und der Anzahl der Türen n_i die sich auf diesem Wagen befanden. Der Zeitpunkt t_0 für den ersten Wagen der in einer Schicht von einer Montageinsel benötigt wird, wird zum Schichtbeginn angegeben. Dabei wird eine Pufferzeit P von 15 Minuten eingeplant.

$$t_1 = t_0 - P$$

$$t_{i+1} = t_i + t_{FG} \cdot n, \quad i = 1, 2, \ldots$$

Dieser geplante Zeitpunkt kann aufgrund oben genannter Unsicherheit vom Zeitpunkt $\tilde{t}$ zu dem die Montageinsel m den nächsten Wagen abholt abweichen:

$$\tilde{t}_{i+1} = \tilde{t}_i + \alpha_{ms} \cdot t_{FG} \cdot n, \quad i = 1, 2, \ldots.$$

Da ein ausgelagerter Wagen nur vom Lagerverwalter gescannt wird und nicht nochmal gescannt wird wenn die Montageinsel den Wagen abholt, ist die Differenz zwischen t_i und $\tilde{t}_i$ nicht sofort bekannt. Die Differenz kann wie folgt aussehen und entsprechende Wirkungen nach sich ziehen.

- **Montageinsel verspätet sich $\tilde{t}_i > t_i$**
  Diese zeitliche Verschiebung wird nicht erkannt, summiert sich aber über die Schicht hinweg auf. Erst zu Beginn der neuen Schicht wird die Verschiebung bekannt und kann damit ins System übernommen werden. Dies führt zu einer Reihenfolgepermutation der Auslagerliste.

- **Montageinsel ist schneller als errechnet $\tilde{t}_i < t_i$**
  Diese Verschiebung wird ebenfalls nicht sofort erkannt. Erst wenn die Verschiebung größer ist als der zeitliche Puffer P, d.h. $t_i - \tilde{t}_i > P$, wird für die Montageinsel zum Zeitpunkt $\tilde{t}_i$ eine sofortige Auslagerung notwendig sein. Die AL verschiebt sich daher zum Zeitpunkt $\tilde{t}_i$ in der Auslagerreihgenfolge an erste Position und muss sofort bedient werden.

Das so definierte stochastische Modell dient sowohl als Grundlage für die Simulation als auch für die Berechnungen innerhalb eines Monte-Carlo-Rollouts. Die Verbesserung die mit Hilfe der Meta-Heuristik erzielt wird, wird in Abbildung (29) dargestellt.

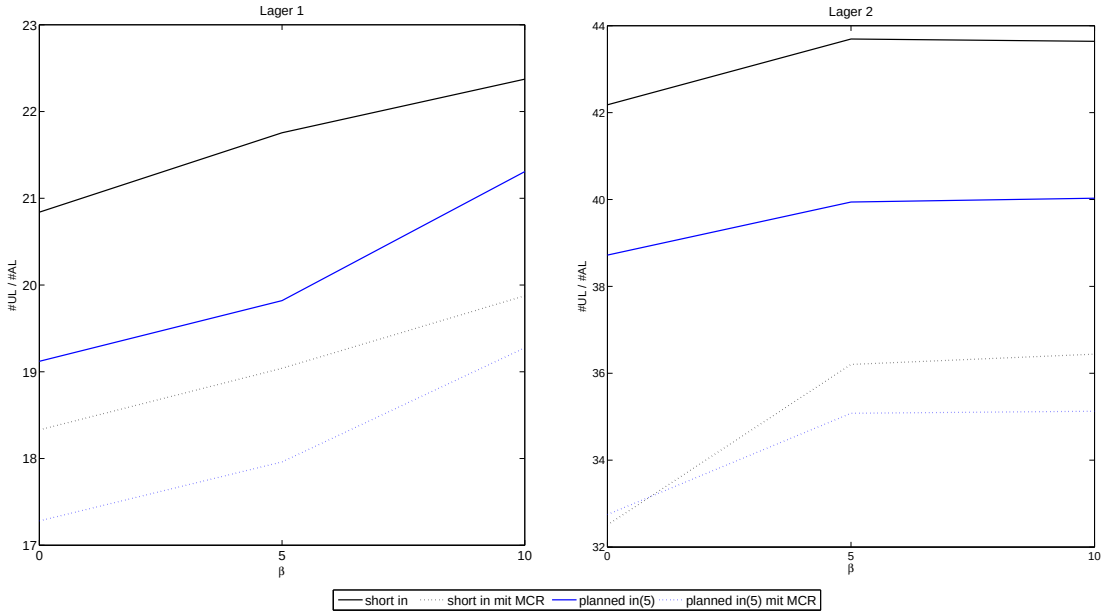

Abbildung 29: UL/AL unter Verwendung der Basisheuristiken und der Meta-Heuristik für verschieden β.