Lösung des Container Sequencing Problems unter Berücksichtigung von Ladeluken

Matthias Bauer

Veröffentlicht in:
Multikonferenz Wirtschaftsinformatik 2012
Tagungsband der MKWI 2012
Hrsg.: Dirk Christian Mattfeld; Susanne Robra-Bissantz

Braunschweig: Institut für Wirtschaftsinformatik, 2012

Lösung des Container Sequencing Problems unter Berücksichtigung von Ladeluken

Matthias Bauer

Universität Passau, Wirtschaftswissenschaftliche Fakultät, Produktion und Logistik, 94032 Passau, E-Mail: matthias.bauer@uni-passau.de

Abstract

Die Minimierung der Ship Turnaround Time gehört zu den wichtigsten Zielsetzungen der operativen Planung in Containerterminals. Einen erheblichen Einfluss auf die Ship Turnaround Time hat die Reihenfolge in der der Kai-Kran die umzuladenden Container eines Schiffes aus- und einlädt. In diesem Beitrag wird das Container Sequencing Problem unter Berücksichtigung von Ladeluken betrachtet. Bei der Ermittlung zulässiger Umladesequenzen werden Doppelspiele und interne Umladungen von Rehandle-Containern berücksichtigt. Die Problemstellung wird als gemischt-ganzzahliges Modell formuliert. Zur Lösung des Problems wird eine Multistart-Heuristik vorgeschlagen. Anhand der Ergebnisse numerischer Experimente wird der Lösungsansatz mit anderen Lösungsverfahren verglichen. Dabei zeigt sich, dass die Multistart-Heuristik die besten Ergebnisse erzielt.

1 Einleitung

Eine der bedeutendsten Kennzahlen im Hinblick auf die Performance und Wettbewerbsfähigkeit eines Containerterminals ist die „Ship Turnaround Time“, die durchschnittliche Dauer, die ein Schiff (unproduktiv) im Hafen verbringt (vgl. [6], S. 1). In den meisten Containerhäfen entfällt ein großer Teil dieser Zeit auf die Be- und Entladung der Containerschiffe (vgl. [3], S. 1). Kai-Kräne, welche den Transport der Container zwischen Schiff und Land durchführen, gehören zu den bedeutendsten Engpassressourcen eines Containerhafens und können aufgrund ihrer Größe nur in begrenzter Anzahl zeitgleich an einem Schiff operieren. Der Schlüssel zur Reduzierung der Ship Turnaround Time liegt daher häufig in der Verbesserung der Kai-Kran-Produktivität (vgl. [2], S. 473 und [4], S. 570). Einen erheblichen Einfluss auf die Kai-Kran-Produktivität hat die Reihenfolge, in welcher der Kai-Kran die umzuladenden Container aus- und einlädt. Das sich daraus ergebende Container Sequencing Problem (CSP) wurde erstmals von Meisel und Wichmann (2010) vorgestellt (siehe [4]) und kann den Quay Crane Scheduling Problemen zugeordnet werden (vgl. [1], S. 623f).[1] Die Aufgabe besteht darin, eine zulässige Umladesequenz für eine Ladebucht zu finden, so dass die für die Umladung der Ladebucht insgesamt benötigte Zeit minimal ist. In diesem Beitrag wird das CSP unter Berücksichtigung von Ladeluken

[1] Einen Literaturüberblick zu Quay Crane Scheduling Problemen geben [1].

(Container Sequencing Problem with Hatches = CSPH) betrachtet. Der Beitrag ist wie folgt aufgebaut. Kapitel 2 liefert zunächst eine Problembeschreibung für das CSPH. Anschließend wird die Problemstellung als gemischt-ganzzahliges Modell formuliert (Kapitel 3). In Kapitel 4 wird eine Multistart-Heuristik zur Lösung des CSPH vorgeschlagen. Kapitel 5 präsentiert die Ergebnisse numerischer Experimente auf Basis zufällig erzeugter Probleminstanzen. Der Beitrag schließt mit einer Zusammenfassung und einem Ausblick (Kapitel 6).

2 Problembeschreibung

Welche Container ein- und ausgeladen werden müssen, lässt sich aus dem Beladungsplan des einlaufenden Schiffes (Ankunftskonfiguration) und dem Beladungsplan des auslaufenden Schiffes (Abfahrtskonfiguration) der betrachteten Bucht ableiten. Beladungspläne werden zuvor im Rahmen der Beladungsplanung erstellt und können daher als gegeben angesehen werden (vgl. [4]). In Bild 1 ist die Ankunfts- und Abfahrtskonfiguration einer Ladebucht beispielhaft abgebildet.

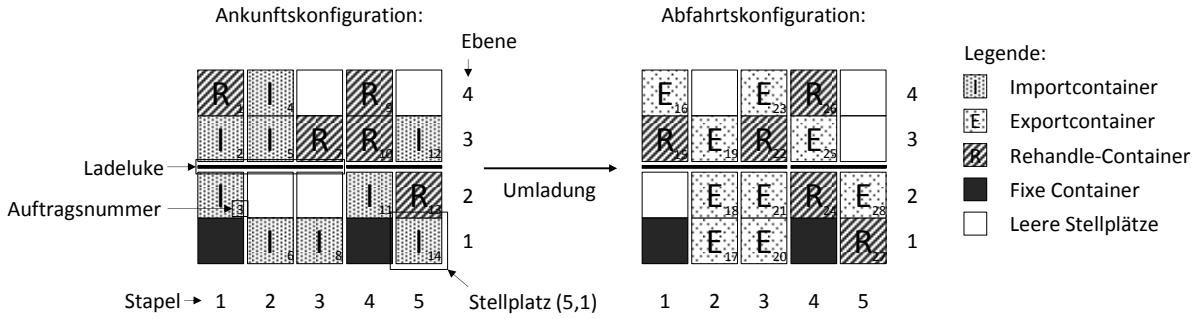

Bild 1: Ankunfts- und Abfahrtskonfigurationen einer Ladebucht (in Anlehnung an [4])

Die Ladebucht besteht aus 5 Stapeln, 4 Ebenen und 2 Ladeluken. Durch die Ladeluken werden die Ebenen oberhalb und unterhalb der Ladeluke voneinander getrennt. Ladeluken erstrecken sich über mehrere Stapel (hier: Ladeluke 1 über die Stapel 1-3 und Ladeluke 2 über die Stapel 4-5) und verursachen Vorrangbeziehungen, die bei der Umladung berücksichtigt werden müssen. So können Ein- und Ausladungen unterhalb der Ladeluke erst durchgeführt werden, wenn alle Container oberhalb der Ladeluke ausgeladen wurden. Zudem kann erst mit der Einladung oberhalb der Ladeluke begonnen werden, wenn die Ein- und Ausladungen unterhalb der Ladeluke abgeschlossen sind. Weitere Vorrangbeziehungen entstehen durch die stapelweise Anordnung der Container. Beispielsweise kann ein Container erst entladen werden, wenn der direkt über ihm stehende Container entladen wurde. Ein einzelner Container kann durch seinen Stellplatz und seine Containerklasse beschrieben werden. Hinsichtlich der Containerklassen wird zwischen Import-Containern, Export-Containern, Rehandle-Containern und fixen Container unterschieden. Import-Container sind nur in der Ankunftskonfiguration zu finden. Sie müssen entladen werden und verbleiben anschließend im Containerterminal. Export-Container sind nur in der Abfahrtskonfiguration zu finden und müssen eingeladen werden. Rehandle-Container müssen von ihrer aktuellen Position entfernt werden, um den Zugriff auf einen Import-Container zu ermöglichen, oder um einen Export-Container an seine entsprechende Position stellen zu können. Aus diesem Grund werden sie vorübergehend entladen und später wieder eingeladen. Sie sind deshalb in beiden Konfigurationen in gleicher Anzahl vorhanden. Fixe Container verbleiben an

ihrer aktuellen Position und werden nicht bewegt. Jeder Import-Container und jeder Rehandle-Container in der Ankunftskonfiguration entspricht somit einem Entladungsauftrag, der vom Kai-Kran auszuführen ist. Die Menge alle Entladungsaufträge wird im Folgenden mit M^A bezeichnet. Analog entspricht jeder Export-Container und jeder Rehandle-Container in der Abfahrtskonfiguration einem Einladungsauftrag. Die Menge aller Einladungsaufträge wird im Folgenden mit M^E bezeichnet. Das Ziel beim CSPH besteht darin, die Aufträge in eine zulässige Reihenfolge zu bringen, so dass die für die Abarbeitung der Aufträge benötigte Zeit minimal ist. Dabei sind die genannten Vorrangbeziehungen zu berücksichtigen.

3 Mathematische Formulierung des CSPH

3.1 Annahmen

Dem mathematischen Modell liegen die folgenden Annahmen zugrunde:

- Die Buchtkonfiguration des einlaufenden Schiffes, die Buchtkonfiguration des auslaufenden Schiffes sowie alle Buchtkonfigurationen, die während der Umladung entstehen können, erfüllen die Stabilitätsbedingungen des Schiffes.
- Ladelukendeckel werden ohne Zeitaufwand entfernt und eingesetzt. Der Kran verbleibt dabei an seiner aktuellen Position.
- An Land stehen jederzeit ausreichend Transportfahrzeuge zur Verfügung, um ausgeladene Container abzutransportieren und einzuladende Container bereitzustellen.
- Die Rehandle-Container sind untereinander austauschbar, so dass jeder Rehandle-Container in der Ankunftskonfiguration an jede Rehandle-Position in der Abfahrtskonfiguration gesetzt werden kann.
- Die Anzahl an Rehandle-Containern innerhalb eines Stapels ist in der Buchtkonfiguration des einlaufenden Schiffes genauso groß wie in der Buchtkonfiguration des auslaufenden Schiffes. Dadurch ist sichergestellt, dass stets genügend Rehandle-Container an Land für die Durchführung entsprechender Beladungsaufträge verfügbar sind.

3.2 Notation

Parameter:

M Menge an Aufträgen, wobei $M = \{0,1,2,...,m\}$. Auftrag 0 ist ein Dummyauftrag und dient der Initialisierung. Es gilt: $M = M^A \cup M^E \cup \{0\}$.

V Menge an Auftragspaaren (i, j) mit $i, j \in M$ für die gilt, dass Auftrag i vor Auftrag j bearbeitet werden muss (Vorrangbeziehungen). Es gilt: $(0, j) \in V$ für alle $j \in M \setminus \{0\}$.

c_{ij} anfallende Bearbeitungszeit, falls Auftrag j unmittelbar nach Auftrag i ausgeführt wird

d_{ij} Zeitersparnis, falls Auftrag j unmittelbar nach Auftrag i ausgeführt wird. Durch d_{ij} werden interne Umladungen zeitlich berücksichtigt. Es gilt:

$d_{ij} > 0$, falls Auftrag i eine Entladung eines Rehandle-Containers ist und Auftrag j eine Einladung eines Rehandle-Containers ist.

$d_{ij} = 0$, sonst.

Entscheidungsvariablen:

u_i Entscheidungsvariablen zur Verhinderung von Subtouren und zur Abbildung der Vorrangbeziehungen.

x_{ij} Binärvariable, wobei

$x_{ij} = 1$, wenn Auftrag j unmittelbar nach Auftrag i ausgeführt wird

$x_{ij} = 0$, sonst.

3.3 Zusammensetzung der Bearbeitungszeiten c_{ij} und der Zeitersparnis d_{ij}

Für die Beschreibung werden die folgenden Symbole benötigt:

O^A Menge der Aufträge, die die Entladung eines Rehandle-Containers vorsehen, wobei $O^A \subseteq M^A \subseteq M$

O^E Menge der Aufträge, die die Einladung eines Rehandle-Containers vorsehen, wobei $O^E \subseteq M^E \subseteq M$

$t_{i,j}^{LE}$ die für die Leerfahrt des Auftrags j benötigte Zeit, wenn dieser unmittelbar nach Auftrag i ausgeführt wird, wobei $i, j \in M$

t_j^{LA} die für die Lastfahrt des Auftrags j benötigte Zeit, wobei $j \in M$

$t_{i,j}^{LA,\text{int}}$ die für die Lastfahrt benötigte Zeit im Falle einer internen Umladung des Rehandle-Containers, wobei $i \in O^A$ und $j \in O^E$

Für die Bearbeitung eines Auftrags j werden jeweils eine Leerfahrt und eine Lastfahrt benötigt. Die Leerfahrt umfasst die Bewegung des Kai-Krans von seiner aktuellen Position zur Aufnahmeposition des zu bewegenden Containers sowie eventuell auftretende Wartezeiten für die Bereitstellung des Containers an der Aufnahmeposition. Da die aktuelle Position des Kai-Krans durch den unmittelbar zuvor bearbeiteten Auftrag i festgelegt wird, ist die für die Leerfahrt des Auftrags j benötigte Zeit $t_{i,j}^{LE}$ von der Auftragsfolge abhängig. Die Lastfahrt umfasst den eigentlichen Transport des zu bewegenden Containers von der Aufnahmeposition zu dessen Zielposition. Die für die Lastfahrt eines Auftrags j benötigte Zeit t_j^{LA} ist unabhängig von der Auftragsfolge. Es gilt:

$$c_{ij} = t_{i,j}^{LE} + t_j^{LA} \qquad \forall i, j \in M \tag{1}$$

Für die Umladung eines Rehandle-Containers sind insgesamt zwei Aufträge auszuführen, ein Entladungsauftrag j und ein Einladungsauftrag j'. Entsprechend fällt c_{ij} für die Entladung und $c_{i'j'}$ für die Einladung des Rehandle-Containers an. Folgt die Einladung j' jedoch unmittelbar auf die Entladung j, kann der Rehandle-Container intern umgeladen werden. Die durch c_{ij} berücksichtigte Zeit t_j^{LA} für den Transport des Rehandle-Containers an Land und die Bearbeitungszeit $c_{jj'}$ für die anschließende Einladung des Rehandle-Containers entfallen. Stattdessen findet eine schiffsinterne Lastfahrt $t_{j,j'}^{LA,\text{int}}$ von der Aufnahmeposition des Containers (bestimmt durch Auftrag j) zur Zielposition des Containers (bestimmt durch Auftrag j') statt. Die zeitliche Berücksichtigung interner Umladungen von Rehandle-Containern erfolgt in der Zielfunktion über die Zeitersparnis $d_{j,j'}$. Es gilt:

$$d_{j,j'} = \begin{cases} t_j^{LA} + t_{jj'}^{LE} + t_{j'}^{LA} - (t_{j,j'}^{LA,\text{int}}), \text{ falls } j \in O^A \text{ und } j' \in O^E \\ 0, \quad \text{sonst} \end{cases} \quad (2)$$

3.4 Mathematisches Modell

Das Problem lässt sich wie folgt formulieren:

$$\text{Minimiere Z} = \sum_{i,j \in M} \left(c_{ij} - d_{ij}\right) \cdot x_{ij} \quad (3)$$

$$\sum_{j \in M} x_{ij} = 1 \quad \forall i \in M \quad (4)$$

$$\sum_{i \in M} x_{ij} = 1 \quad \forall j \in M \quad (5)$$

$$u_i + 1 \leq u_j + m \cdot (1 - x_{ij}) \quad \forall i \in M,\ j \in 1,2,...,m \quad (6)$$

$$u_i + 1 \leq u_j \quad \forall (i,j) \in V \quad (7)$$

$$x_{ij} \in \{0;1\} \quad \forall i,j \in M \quad (8)$$

$$u_i \geq 0 \quad \forall i \in M \quad (9)$$

Die Zielfunktion (3) minimiert die vom Kai-Kran benötige Zeit zur Bearbeitung aller Aufträge. Die Nebenbedingungen (4) und (5) stellen sicher, dass jeder Auftrag genau einmal durchgeführt wird. Die Nebenbedingungen (6) stellen sicher, dass eine Auftragssequenz erzeugt wird. Es gilt: $u_j \geq u_i + 1$, falls Auftrag j nach Auftrag i ausgeführt wird. Die Einhaltung der Vorrangbeziehungen wird durch die Nebenbedingungen (7) sichergestellt. Die Nebenbedingungen (8) und (9) legen den Definitionsbereich der Entscheidungsvariablen fest.

4 Multistart-Heuristik

Das CSPH kann als 1-Maschinenbelegungsproblem mit reihenfolgeabhängigen Rüstkosten und Vorrangbeziehungen aufgefasst werden. Dieses Problem gehört zur Klasse der NP-schwierigen Probleme (vgl. [5], S. 84). Im Folgenden wird daher eine Multistart-Heuristik zur Lösung des CSPH vorgestellt. Das Vorgehen orientiert sich an dem von Meisel und Wichmann entwickelten GRASP-Ansatz zur Lösung des CSP (siehe [4]). Das Verfahren besteht aus zwei Phasen, welche iterativ durchlaufen werden. In der Konstruktionsphase wird eine zulässige Ausgangslösung generiert (Abschnitt 4.1). In der Verbesserungsphase (Abschnitt 4.2) wird versucht, die erzeugte Lösung durch ein lokales Suchverfahren zu verbessern. Um unterschiedliche Lösungen zu generieren, kommt in der Konstruktionsphase ein Zufallsmechanismus zum Einsatz. Das Verfahren endet nach einer vorgegebenen Anzahl an Iterationen und gibt die beste gefundene Lösung aus.

4.1 Konstruktionsphase

Für die Beschreibung der Konstruktionsphase werden die folgenden Symbole benötigt:

Ψ Kandidatenliste

Ω Menge der noch nicht eingeplanten Aufträge

Φ Auftragsfolge (Liste der bereits eingeteilten Aufträge)

$T(s)$ Menge der Aufträge des Stapels s. Es gilt: $T(0) = \{0\}$

$K(l)$ Menge der Aufträge der Luke l. Es gilt: $K(0) = \{0\}$

s_i Stapel des Auftrags i (durch die jeweilige Position des ein- bzw. auszuladenden Containers an Bord des Schiffes vorgegeben). $s_0 = 0$

l_i Ladeluke des Auftrags i (durch die jeweilige Position des ein- bzw. auszuladenden Containers an Bord des Schiffes vorgegeben). $l_0 = 0$

s^B Bearbeitungsstapel

l^B Bearbeitungsluke

Das Ziel der Konstruktionsphase besteht darin, eine zulässige Ausgangslösung für das Problem zu generieren. Das Verfahren beginnt mit einer leeren Auftragsfolge. In jeder Iteration wird die Auftragsfolge um einen Auftrag erweitert. Das Verfahren endet, sobald alle Aufträge in die Auftragsfolge aufgenommen wurden. Das Konstruktionsverfahren kann wie folgt beschrieben werden:

Initialisierung: Setze $s^B := 0$, $l^B := 0$, $\Phi := <0>$, $\Psi := \varnothing$ und $\Omega := M \setminus \{0\}$.

Schritt 1: Aus der Menge Ω werden zunächst alle Aufträge identifiziert, die in Φ aufgenommen werden können. Ein Auftrag j kann nur dann in Φ aufgenommen werden, wenn in Ω kein Auftrag i mit $(i, j) \in V$ existiert. Alle zulässigen Aufträge werden in Ψ abgelegt. Gehe zu Schritt 2.

Schritt 2: Enthält Ψ Aufträge der aktuellen Bearbeitungsladeluke ($K(l^B) \cap \Psi \neq \varnothing$), werden alle übrigen nicht zur Bearbeitungsladeluke gehörenden Aufträge aus der Kandidatenliste entfernt ($\Psi := K(l^B) \cap \Psi$). Gehe zu Schritt 3.

Schritt 3: Enthält Ψ einen Auftrag des aktuellen Bearbeitungsstapels ($T(s^B) \cap \Psi \neq \varnothing$), werden alle übrigen nicht zum Bearbeitungsstapel gehörenden Aufträge aus der Kandidatenliste entfernt ($\Psi := T(s^B) \cap \Psi$). Gehe zu Schritt 4.

Schritt 4: Aus der Kandidatenliste Ψ wird genau ein Auftrag j zufällig ausgewählt. Die Auftragsfolge Φ wird um den ausgewählten Auftrag j verlängert. Anschließend wird Auftrag j aus der Menge Ω entfernt ($\Omega := \Omega \setminus \{j\}$). Sind alle Aufträge eingeplant ($\Omega = \varnothing$), wird das Verfahren abgebrochen. Andernfalls setze $s^B := s_j$, $l^B := l_j$, $\Psi := \varnothing$ und gehe zu Schritt 1.

4.2 Verbesserungsphase

Im Folgenden werden drei Verschiebestrategien zur Verbesserung der Auftragsfolge vorgestellt. Für die Beschreibung der Strategien werden die folgenden Symbole benötigt:

$p, \bar{p}$ Position in der Auftragsfolge

P' Menge der zulässigen Verschiebepositionen

Q Anzahl der Positionen in der Auftragsfolge

4.2.1 Best-Shift-Left-Strategie

Ziel der Best-Shift-Left-Strategie ist es, Aufträge an frühere Positionen in der Auftragsfolge zu verschieben, um so eine bessere Lösung zu erhalten. Bei der Verschiebung der Aufträge innerhalb der Auftragsfolge sind die Vorrangbeziehungen zu berücksichtigen. Ein Auftrag j kann nur dann links von einem Auftrag i positioniert werden, wenn kein $(i, j) \in V$ existiert. Die Best-Shift-Left-Strategie kann wie folgt beschrieben werden:

Initialisierung: Setze $p := 2$.

Schritt 1: Ermittle für den an Position p der Auftragsfolge stehenden Auftrag j die Menge aller zulässigen Verschiebepositionen P' für die gilt: $p' < p \quad \forall p' \in P'$. Falls $P' = \varnothing$, gehe zu Schritt 4. Andernfalls gehe zu Schritt 2.

Schritt 2: Ermittle die Verschiebeposition $p^* \in P'$, die im Falle einer Verschiebung von Auftrag j nach p^* zur größten Verbesserung des Zielfunktionswertes führt. Gibt es mehr als ein p^*, dann wähle unter ihnen die kleinste Position.

Führt keine Verschiebung des Auftrags j zu einer Verbesserung des Zielfunktionswertes, gehe zu Schritt 4. Ansonsten gehe zu Schritt 3.

Schritt 3: Verschiebe den Auftrag j von seiner aktuellen Position p an die Position p^* der Auftragsfolge. Gehe zu Schritt 4.

Schritt 4: Falls $p = Q$, brich das Verfahren ab. Andernfalls setze $p := p + 1$ und gehe zu Schritt 1.

4.2.2 Best-Shift-Right-Strategie

Ziel der Best-Shift-Right-Strategie ist es, Aufträge an eine spätere Position in der Auftragsfolge zu verschieben, um dadurch eine bessere Lösung zu erhalten. Ein Auftrag j kann nur dann rechts von einem Auftrag i positioniert werden, wenn kein $(j, i) \in V$ existiert. Die Best-Shift-Right-Strategie kann wie folgt beschrieben werden:

Initialisierung: Setze $p := 1$

Schritt 1: Ermittle für den an Position p der Auftragsfolge stehenden Auftrag j die Menge aller zulässigen Verschiebepositionen P' für die gilt: $p' > p \quad \forall p' \in P'$. Falls $P' = \varnothing$, gehe zu Schritt 4. Andernfalls gehe zu Schritt 2.

Schritt 2: Ermittle die Verschiebeposition $p^* \in P'$, die im Falle einer Verschiebung von Auftrag j nach p^* zur größten Verbesserung des Zielfunktionswertes führt. Gibt es mehr als ein p^*, dann wähle unter ihnen die größte Position.

Führt keine Verschiebung des Auftrags j zu einer Verbesserung des Zielfunktionswertes, gehe zu Schritt 4. Ansonsten gehe zu Schritt 3.

Schritt 3: Verschiebe den Auftrag j von seiner aktuellen Position p an die Position p^* der Auftragsfolge. Falls $p = Q - 1$, brich das Verfahren ab. Andernfalls gehe zu Schritt 1.

Schritt 4: Falls $p = Q - 1$, brich das Verfahren ab. Andernfalls setze $p := p + 1$ und gehe zu Schritt 1.

4.2.3 Merging-Strategie

Ziel der Merging-Strategie ist es, die Anzahl interner Umladungen von Rehandle-Containern zu erhöhen, um so eine bessere Lösung zu erhalten. Ein Rehandle-Container kann intern umgeladen werden, wenn in der Auftragsfolge eine Rehandle-Container-Einladung direkt nach einer Rehandle-Container-Entladung folgt. Deshalb wird für jedes Auftragspaar (i, j), mit $i \in O^A$ und $j \in O^E$, geprüft, ob die Aufträge i und j innerhalb der Auftragsfolge so verschoben werden können, dass Auftrag i unmittelbar vor Auftrag j ausgeführt wird. Dabei sind die Vorrangbeziehungen zu berücksichtigen. Die Merging-Strategie kann wie folgt beschrieben werden:

Schritt 1: Setze $p := 1$.

Schritt 2: Sieht der an Position p stehende Auftrag i die Entladung eines Rehandle-Containers vor ($i \in O^A$), dann setze $\overline{p} := 2$ und gehe zu Schritt 3. Ansonsten gehe zu Schritt 6.

Schritt 3: Sieht der an Position $\overline{p}$ stehende Auftrag j die Einladung eines Rehandle-Containers vor ($j \in O^E$) und existiert kein $(j,i) \in V$, dann gehe zu Schritt 4. Ansonsten gehe zu Schritt 5.

Schritt 4: Ermittle die Menge aller zulässigen Verschiebepositionen für die Aufträge i und j. Können die Aufträge i und j so verschoben werden, dass Auftrag i unmittelbar vor Auftrag j ausgeführt wird und verbessert sich dadurch der Zielfunktionswert, dann verschiebe beide Aufträge entsprechend und gehe anschließend zu Schritt 1. Andernfalls gehe zu Schritt 5.

Schritt 5: Falls $\overline{p} = Q$, gehe zu Schritt 6. Ansonsten setze $\overline{p} := \overline{p} + 1$ und gehe zu Schritt 3.

Schritt 6: Falls $p = Q$, brich das Verfahren ab. Ansonsten setze $p := p + 1$ und gehe zu Schritt 2.

4.2.4 Ablauf und Abbruch der Verbesserungsphase

In der Verbesserungsphase werden die vorgestellten Verschiebestrategien nacheinander in einer Schleife ausgeführt. Führte in einer Iteration mindestens eine Verschiebestrategie zu einer Verbesserung der Lösung, wird eine weitere Iteration ausgeführt. Die Reihenfolge, in der die Verschiebestrategien ausgeführt werden, beeinflusst die Qualität der erzeugten Lösungen. In Experimenten wurden die durchschnittlich besten Ergebnisse erzielt, wenn erst die Best-Shift-Left-Strategie, dann die Best-Shift-Right-Strategie und zuletzt die Merging-Strategie ausgeführt wird.

5 Experimentelle Untersuchung

Anhand numerischer Experimente wurde die Lösungsgüte der Multistart-Heuristik für das CSPH untersucht.

5.1 Generierung der Probleminstanzen

Für die Experimente wurde eine große Anzahl an Probleminstanzen generiert. Die erzeugten Probleminstanzen unterscheiden sich hinsichtlich der Größe der betrachteten Bucht (3 Stufen), dem Anteil an Import-Containern (3 Stufen), dem Anteil an Export-Containern (3 Stufen) und

dem Anteil an Rehandle-Containern (5 Stufen). Bei der Größe der betrachteten Bucht wird zwischen kleinen, mittleren und großen Probleminstanzen unterschieden. Kleine Probleminstanzen bestehen aus 10 Stapeln, 10 Ebenen und 3 Ladeluken. Durch die Ladeluken werden die unteren fünf Ebenen von den oberen fünf Ebenen getrennt. Die erste Ladeluke umfasst die Stapel 1-3, Ladeluke zwei umfasst die Stapel 4-7 und Ladeluke drei umfasst die Stapel 8-10. Mittlere Probleminstanzen bestehen aus 15 Stapeln, 15 Ebenen und 5 Ladeluken. Durch die Ladeluken werden die unteren sieben Ebenen von den oberen acht Ebenen getrennt. Alle Ladeluken umfassen jeweils drei aneinander liegende Stapel. Große Probleminstanzen bestehen aus 20 Stapeln, 20 Ebenen und 6 Ladeluken. Die Ladeluken trennen die unteren zehn Ebenen von den oberen zehn Ebenen. Ladeluke eins umfasst die Stapel 1-3, Ladeluke zwei die Stapel 4-6, Ladeluke drei die Stapel 7-10, Ladeluke vier die Stapel 11-14, Ladeluke fünf die Stapel 15-17 und Ladeluke sechs die Stapel 18-20. Der Anteil an Import-Containern gibt an, wie viel Prozent der Stellplätze in der Buchtkonfiguration des einlaufenden Schiffes mit Import-Containern belegt sind. Es wird zwischen 30%, 50% und 70% unterschieden. Der Anteil an Export-Containern gibt an, wie viel Prozent der Stellplätze in der Buchtkonfiguration des auslaufenden Schiffes mit Export-Containern belegt sind. Auch hier wird zwischen 30%, 50% und 70% unterschieden. Der Anteil an Rehandle-Containern gibt an, wie viel Prozent der Stellplätze in der betrachteten Bucht mit Rehandle-Containern belegt sind. Es wird zwischen 0%, 5%, 10%, 15% und 20% unterschieden. Darüber hinaus sind 10% der Positionen an Bord des Schiffes mit fixen Containern belegt. Aus den 4 Faktoren und ihren Stufen ergeben sich insgesamt 3*3*3*5=135 Stufenkombinationen. Für jede Stufenkombination wurden 50 unterschiedliche Probleminstanzen generiert. Insgesamt wurden 135*50=6750 Probleme erzeugt. Bei der Problemerzeugung wurden die Import-Container, die Export-Container, die Rehandle-Container und die fixen Container zufällig auf die Stapel verteilt. Dabei wurde sichergestellt, dass die maximale Anzahl an Containern je Stapel nicht überschritten wird und einem Stapel höchstens so viele fixe Container zugeordnet werden, wie Ebenen unterhalb der Ladeluke zur Verfügung stehen. Fixe Container wurden stets unterhalb der Ladeluke positioniert. Import-Container, Rehandle-Container und Export-Container wurden hingegen zufällig auf den Bereich oberhalb und unterhalb der Ladeluke verteilt. Dabei wurde sichergestellt, dass in jedem Stapel die maximale Anzahl an Containern oberhalb und unterhalb der Ladeluke nicht überschritten wird. Rehandle-Container wurden zufällig zwischen den Import-Containern in der Buchtkonfiguration des einlaufenden Schiffes sowie zwischen den Export-Containern in der Buchtkonfiguration des auslaufenden Schiffes verteilt.

Den Experimenten liegen die folgenden Bearbeitungszeiten (s=Sekunden) zugrunde:

$$t_j^{LA} = 70 \text{ s} \quad \forall j \in M$$

$$t_{i,j}^{LA,\text{int}} = 70 \text{ s} \quad \forall i \in O^A \text{ und } \forall j \in O^E$$

$$t_{i,j}^{LE} = \begin{cases} 50 \text{ s, falls } i, j \in M^A \\ 50 \text{ s, falls } i, j \in M^E \end{cases} \} \text{Dauer der Leerfahrt bei einem Einzelspiel}$$

$$\begin{cases} 40 \text{ s, falls } i \in M^A \text{ und } j \in M^E \\ 40 \text{ s, falls } i \in M^E \text{und } j \in M^A \end{cases} \} \text{Dauer der Leerfahrt bei einem Doppelspiel}$$

5.2 Ergebnisse

Die Multistart-Heuristik wurde in C++ implementiert. Zur Lösung eines Problems wurden jeweils 1000 Iterationen durchlaufen und die beste gefundene Lösung ausgegeben. Für den Vergleich wurde die relative Abweichung (RE = relative error) des Zielfunktionswertes von der unteren Schranke[2] (LB) ermittelt. Es gilt:

$$RE = \frac{(Z - LB)}{LB} \cdot 100 \qquad (10)$$

Zudem wurden der stapelbasierte Ansatz von Zhang und Kim (2009) und der GRASP-Ansatz von Meisel und Wichmann (2010) in C++ implementiert und an die zugrundeliegende Problemstellung angepasst. Die Heuristik von Zhang und Kim ermittelt für jedes Problem die Reihenfolge, in der die Stapel ober- und unterhalb der Ladeluke abgearbeitet werden sollen. Um einen Vergleich zu ermöglichen, wurde die Stapelreihenfolge in eine zulässige Auftragsfolge überführt. Beim GRASP-Ansatz von Meisel und Wichmann werden Ladeluken nicht explizit berücksichtigt. Aufgrund der zusätzlichen Vorrangbeziehungen kann es in der Konstruktionsphase zu leeren Kandidatenlisten und damit zum Abbruch des Verfahrens kommen. Um dennoch zulässige Lösungen zu generieren, wurde das Konstruktionsverfahren wie folgt modifiziert: Kann kein Kandidat für die Kandidatenliste ermittelt werden, werden alle zulässigen Entladungsaufträge in die Kandidatenliste aufgenommen, die zu den Stapeln der Ladeluke des aktuellen Entladungsstapels gehören.

Beispiel: Die Ladeluke 1 umfasst die Stapel 1-3. Ist der aktuelle Entladungsstapel Stapel 1 und konnte kein Kandidat für die Kandidatenliste ermittelt werden, werden alle zulässigen Entladungsaufträge der Stapel 2 und 3 in die Kandidatenliste aufgenommen.

Tabelle 1 stellt die von den drei Verfahren erzielten Ergebnisse gegenüber. Die Multistart-Heuristik (MSH) erzielte für alle Problemgrößen jeweils die kleinste durchschnittliche relative Abweichung (ARE = Average Relative Error), gefolgt vom modifizierten GRASP-Ansatz von Meisel und Wichmann (GRASP MuW) und dem stapelbasierten Ansatz von Zhang und Kim (SBA ZuK). Bei allen Verfahren sinkt die ARE mit zunehmender Problemgröße. Unter Einbezug aller 6750 Probleminstanzen liegt die ARE der MSH mit 4,08% unter GRASP MuW (4,80%) und deutlich unter SBA ZuK (9,05%). Der durchschnittliche Anteil interner Umladungen (IRR = Internal Rehandle Ratio) gibt an, wie viel Prozent der Rehandle-Container einer Probleminstanz durchschnittlich intern umgeladen werden konnten. Die MSH erzielte für alle Problemgrößen jeweils den höchsten IRR, gefolgt vom GRASP MuW. Da beim SBA ZuK interne Umladungen nicht explizit berücksichtigt werden, führt dieser unabhängig von der Problemgröße zu einem deutlich kleineren IRR. Der IRR steigt bei der MSH von 50,34% bei kleineren Probleminstanzen, über 57,23% bei mittleren Probleminstanzen auf 58,34% bei großen Probleminstanzen. Unter Einbezug aller 6750 Probleminstanzen liegt der IRR der MSH mit 55,31% vor dem GRASP MuW mit 51,23% und deutlich vor dem SBA ZuK (11,39%). Die durchschnittliche Doppelspielrate (DCR = Double Cycling Ratio) gibt an, wie viel Prozent der Aufträge einer Probleminstanz durchschnittlich im Doppelspiel (auf eine Einladung folgte eine Entladung oder auf eine Entladung folgte eine Einladung) ausgeführt wurden. Der SBA ZuK erzielte für alle Problemgrößen jeweils die höchste DCR, gefolgt von der MSH und dem GRASP MuW. Insgesamt steigt die DCR bei allen Verfahren mit zunehmender Problemgröße an. Unter Einbezug aller 6750

[2] Die dem Experiment zugrundeliegenden Bearbeitungszeiten ermöglichen die Berechnung der unteren Schranke nach Meisel und Wichmann (2010). Für die Berechnung der unteren Schranke siehe [4].

Probleminstanzen liegt die DCR des SBA ZuK mit 76,27% vor der MSH mit 73,48% und dem GRASP MuW (70,99%). Die durchschnittlich benötigte Zeit zur Lösung einer Probleminstanz (Zeit) auf einem PC i7-2600K @3,4GHz bleibt selbst bei großen Problemen bei allen Verfahren unter 5 Sekunden.

Größe	Verfahren	*ARE [%]*	*IRR [%]*	*DCR [%]*	Zeit [s]
10x10	SBA ZuK	9.66	10.06	72.64	0.00
	GRASP MuW	5.22	49.62	68.96	0.38
	MSH	4.94	50.34	70.60	0.32
15x15	SBA ZuK	8.84	11.69	77.17	0.00
	GRASP MuW	4.66	52.13	71.18	1.52
	MSH	3.74	57.23	74.26	1.21
20x20	SBA ZuK	8.64	12.41	79.00	0.00
	GRASP MuW	4.52	51.93	72.83	4.88
	MSH	3.54	58.34	75.57	3.71
Gesamt	SBA ZuK	9.05	11.39	76.27	0.00
	GRASP MuW	4.80	51.23	70.99	2.26
	MSH	4.08	55.31	73.48	1.75

Tabelle 1: Gegenüberstellung der Ergebnisse

Bild 2 stellt die ARE für die drei Verfahren in Abhängigkeit vom Anteil an Rehandle-Containern (RR) in den Probleminstanzen dar.

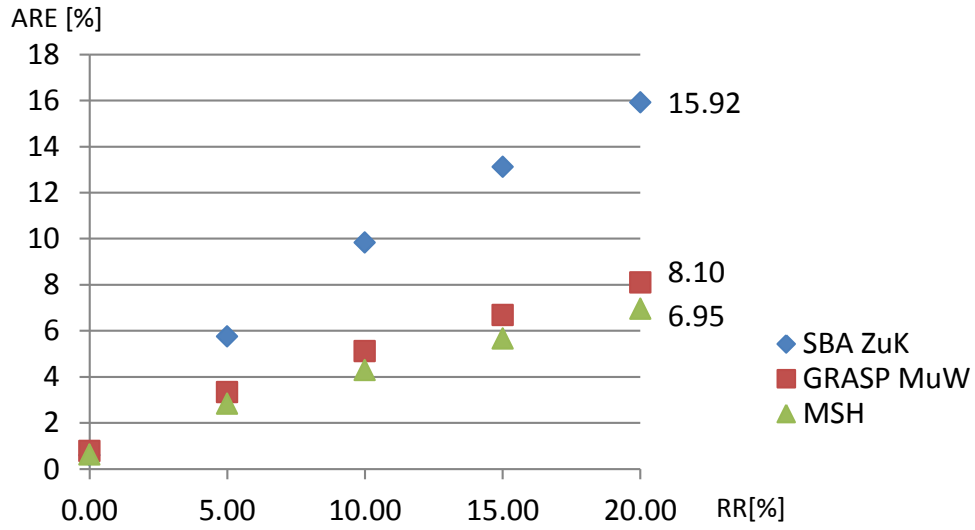

Bild 2: Vergleich der Lösungsgüte in Abhängigkeit vom Anteil an Rehandle-Containern in den Probleminstanzen

Jeder Punkt in der Grafik repräsentiert die von dem jeweiligen Verfahren erzielte ARE für 1350 Probleminstanzen mit entsprechendem Anteil an Rehandle-Containern. Es ist erkennbar, dass die ARE bei allen Verfahren mit zunehmendem RR größer wird. Bei RR=20% liegt die ARE der MSH bei 6,95%, die des GRASP MuW bei 8.10% und die des SBA ZuK bei 15,92%. Bei RR=0% liefern alle Verfahren Lösungen nahe dem Optimum.

6 Zusammenfassung

In diesem Beitrag wurde das Container Sequencing Problem unter Berücksichtigung von Ladeluken betrachtet. Bei der Ermittlung zulässiger Umladesequenzen wurden Doppelspiele und direkte Umladungen von Rehandle-Containern innerhalb der betrachteten Bucht berücksichtigt. Die Problemstellung wurde als gemischt-ganzzahliges Modell formuliert. Zur Lösung des Problems wurde eine Multistart-Heuristik vorgeschlagen. Anhand der Ergebnisse numerischer Experimente wurde die Multistart-Heuristik mit anderen, an die Problemstellung angepassten Lösungsverfahren verglichen. Die Multistart-Heuristik erzielte über alle Problemgrößen hinweg die kleinste durchschnittliche Abweichung des Zielfunktionswertes zur unteren Schranke. Unter Berücksichtigung aller 6750 Probleminstanzen liegt die ARE der Multistart-Heuristik mit 4,08% unter dem des modifizierten GRASP-Ansatzes von Meisel und Wichmann (4,80%) und deutlich unter dem stapelbasierten Ansatz von Zhang und Kim (9,05%). In zukünftigen Forschungsarbeiten könnte das CSPH in die Beladungsplanung von Containerschiffen integriert werden. Ziel wäre es, solche Beladungspläne zu erstellen, die eine möglichst schnelle Abarbeitung der Schiffe ermöglichen. Weiterhin könnte das CSPH in die Planung der Prozesse integriert werden, die an der Bereitstellung der einzuladenden Container beteiligt sind. Dazu zählen Yard-Kran-Operationen sowie die landseitigen Transporte der Container zum Kai-Kran.

7 Literatur

[1] Bierwirth, C; Meisel, F (2010): A Survey of Berth Allocation and Quay Crane Scheduling Problems in Container Terminals, European Journal of Operational Research 202(3): 615-627.

[2] Goodchild, AV; Daganzo, CD (2006): Double-Cycling Strategies for Container Ships and Their Effect on Ship Loading and Unloading Operations, Transportation Science 40(4): 473-483.

[3] Huang, Y; Liang, C; Yang, Y (2009): The Optimum Route Problem by Genetic Algorithm for Loading/Unloading of Yard Crane, Computer & Industrial Engineering 56(3):993-1001.

[4] Meisel, F; Wichmann, M (2010): Container Sequencing for Quay Cranes with Internal Reshuffles, OR Spectrum 32(3):569-591.

[5] Pinedo, ML (2008): Scheduling: Theory, Algorithms, and Systems. 3. Auflage. Springer-Verlag, New York.

[6] Zhang, H; Kim, KH (2009): Maximizing the number of dual-cycle operations of quay cranes in container terminals, Computers & Industrial Engineering 56(3):979-992.