DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

Entwicklung einer KI-basierten Reihenfolgestrategie für Hochregallager mit autonomen Fahrzeugen

Development of an AI-based sequencing policy for autonomous vehicle storage and retrieval systems

Franziska Schloz [1]
Thomas Kriehn [2]
Robert Schulz [1]
Markus Fittinghoff [2]

[1] *Institut für Fördertechnik und Logistik*
Fakultät für Konstruktions-, Produktions- und Fahrzeugtechnik
Universität Stuttgart

[2] *Professur Materialfluss und Logistikplanung*
Fakultät Technische Prozesse
Hochschule Heilbronn

Die Nutzung der Künstlichen Intelligenz (KI), insbesondere von Algorithmen des maschinellen Lernens zur Lösung von Problemen in der Praxis und der Forschung ist weit verbreitet und in vielen Bereichen etabliert. Im Bereich der Bildung von Lagerstrategien in Hochregallagern mit autonomen Fahrzeugen (Shuttle-Systemen) besteht jedoch noch eine Forschungslücke. Eines der KI-Verfahren, das in letzter Zeit für Aufmerksamkeit gesorgt hat, ist das Reinforcement Learning mit der Verknüpfung zu Deep Learning. In diesem Beitrag wird eine Möglichkeit aufgezeigt, wie das Deep Reinforcement Learning genutzt werden kann, um eine Auslagerstrategie für Shuttle-Systeme zu entwickeln.

[Schlüsselwörter: Deep Reinforcement Learning, Lagerstrategien, Shuttle-Systeme, Durchsatzoptimierung]

Using Artificial Intelligence (AI) in particular machine learning algorithms to solve problems in practice and research is widespread and established in several areas. However, there is still a research gap in the formation of storage strategies in high-bay warehouses with autonomous vehicles (AVS/RS). One of the AI methods that has recently attracted attention is Reinforcement Learning (RL), which includes deep learning. This paper presents how deep reinforcement learning can be used to develop a sequencing policy for AVS/RS.

[Keywords: Deep Reinforcement Learning, Storage strategies, AVS/RS, throughput optimization]

1 EINLEITUNG

Shuttle-Systeme sind den automatisierten Lägern zuzuordnen, bei welchen die Lagerspiele von autonomen Fahrzeugen und Hebern vorgenommen werden. Dadurch entsteht eine Trennung der vertikalen und horizontalen Förderung, womit sie sich im Bewegungsmuster von kranbasierten Regalbediengeräten (RBG) unterscheiden, die eine Diagonalfahrt durchführen. [VDI13] Vorteilhaft sind der hohe Durchsatz, die Flexibilität sowie der im Vergleich zu RBG niedrige Energieverbrauch, weshalb sie sich in der Praxis etabliert haben. [Irr16]

Je nach Systemausprägung können die Fahrzeuge frei im System verfahren und nutzen damit dieselben Wege und Heber. Die dadurch entstehenden Blockaden führen zu Wartezeiten und somit über die Verlängerung der Spielzeit zu einer Durchsatzreduzierung. [RKH+10]

Wie häufig sich die Fahrzeuge blockieren, ist von der Reihenfolge der Auftragsbearbeitung abhängig, da diese festlegt, zu welchen Zeitpunkten die Fahrzeuge die entsprechenden Wege benutzen. Folglich kann über die Festlegung der Auftragsreihenfolge der Durchsatz erhöht werden.

Eine der Möglichkeiten zur Lösung von Reihenfolgeproblemen ist das Reinforcement Learning (RL). Das RL gehört zu den Verfahren des Maschinellen Lernens, welches durch die Interaktion mit einer Lernumgebung eine Strategie erlernen kann. Durch die Verbindung des RL mit

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

dem Deep Learning (DL) konnten in jüngerer Vergangenheit große Erfolge im Gaming-Bereich erzielt werden [Mar19], wodurch das Deep Reinforcement Learning (DRL) in den Fokus vieler Untersuchungen gerückt ist. [z. B. WRB+18]

In diesem Beitrag wird eine Möglichkeit vorgestellt, wie die Auftragsreihenfolge der Auslagerungen in Shuttle-Systemen mittels DRL optimiert werden kann. Hierfür wurde ein stark vereinfachtes Shuttle-System mit einer Ebene, drei Lagergängen und zwei Shuttle-Fahrzeugen in der Simulationssoftware Plant Simulation abgebildet. Die DRL-Architektur wird unter Nutzung der Open Source Deep Learning Bibliotheken Keras und TensorFlow erstellt.

Im Folgenden werden die Grundlagen und der Stand der Forschung im Bereich der Shuttle-Systeme dargestellt. Anschließend werden das DRL und seine Elemente erläutert. Es folgt die Beschreibung des Simulationsmodells und es wird an einem Beispiel gezeigt, dass die Reihenfolge der Auslageraufträge die Gesamtzeit für die Bearbeitung der Aufträge beeinflusst. Anschließend werden die Architektur des DRL-Agenten und die Schnittstellen mit dem Simulationsmodell beschrieben.

2 Grundlagen und Stand der Forschung von Shuttle-Systemen und Lagerstrategien

2.1 Shuttle-Systeme

Shuttle-Systeme können übergeordnet in vier Kategorien eingeteilt werden. Abhängig von den Bewegungsdimensionen der Fahrzeuge und Heber lassen sich die Arten ungebundene, ebenengebundene, ganggebundene sowie gang- und ebenengebundene Systeme identifizieren.

Bei gang- und ebenengebundenen Systemen kann das Fahrzeug die jeweilige Ebene im jeweiligen Gang nicht verlassen. Die Ladeeinheit wird über einen Heber auf die Zielebene gebracht, dort in einem Pufferplatz zwischengelagert und von dort durch das Fahrzeug an den Lagerplatz transportiert und eingelagert.

Bei ebenengebundenen Systemen können die Fahrzeuge über Quergänge einen Gangwechsel vornehmen. Ein Ebenenwechsel ist nicht möglich, da die Heber, wie bei den gang- und ebenengebundenen Systemen nur Ladeeinheiten, nicht aber die Fahrzeuge transportieren können.

Im Gegensatz dazu gibt es in ganggebundenen Systemen keine Quergänge. Die Fahrzeuge können jedoch über Fahrzeugheber die Ebenen im jeweiligen Gang wechseln.

Bei ungebundenen Systemen sind sowohl Quergänge als auch Fahrzeugheber vorhanden. Die Fahrzeuge können sich damit im gesamten System bewegen und folglich kann jedes Fahrzeug jeden Lagerplatz anfahren.

Während anfangs vor allem ungebundene Systeme installiert wurden, sind heutzutage vermehrt gang- und ebenengebundene Systeme im Einsatz, da diese einen höheren Durchsatz erzielen. [FHL18] Außerdem existieren Mischformen, bei welchen die Fahrzeuge beispielswiese nur bestimmte Ebenen bedienen können oder Heber sowohl Fahrzeuge als auch Ladeeinheiten separat und kombiniert transportieren können.

2.2 Lagerstrategien

Durch die beiden Komponenten Fahrzeug und Heber selbst sowie durch den Bewegungsablauf, der durch die Systemart vorgegeben wird, können verschiedene Lagerstrategien angewendet werden.

In Anlehnung an [Som15] und [TS10] lassen sich Lagerstrategien anhand von sechs Funktionen einteilen:

- Einlagerung bzw. Lagerplatzvergabe
- Auslagerung bzw. Ladeeinheitenauswahl
- Umlagerung
- Reihenfolgebildung bzw. Sequenzierung
- Bestimmung des Befehlszyklus
- Wahl der Fördermittel

Bei der Einlagerung wird einem Artikel ein fester Lagerplatz zugewiesen. Mögliche Strategien sind beispielsweise die Festplatzlagerung, die chaotische Lagerung oder die Zonierung. Zielsetzungen der Einlagerstrategien sind je nach Strategie die Zugriffssicherheit bei einem Ausfall des Systems, die maximale Kapazitätsnutzung und die Durchsatzoptimierung.

Stehen für die Erledigung eines Auslagerauftrages mehrere Artikel zur Verfügung, legt die Auslagerungsstrategie den bestimmten Artikel fest. Gängige Strategien sind first in first out, um eine Überalterung der Artikel zu vermeiden, last in last out, um Umlagerungen zu vermeiden oder die Wahl des Artikels mit dem kürzesten Fahrweg.

Umlagerungen treten systembedingt auf, wenn beispielsweise die Kanäle bei mehrfachtiefer Lagerung nicht artikelrein sind. Der nicht benötigte Artikel, der sich vor dem relevanten Artikel befindet, muss an einen geeigneten Platz umgelagert werden. Umlagerungen können aber auch durchgeführt werden, um das Lager hinsichtlich des Zielkriteriums neu zu sortieren.

Zunächst muss bei der Reihenfolgebildung festgelegt werden, welche Aufträge in die Betrachtung einfließen, da permanent neue Aufträge das System betreten. Bei der blockweisen Reihenfolgebildung wird eine fixe Zeitspanne oder eine fixe Anzahl an Aufträgen gewählt und die Reihenfolge für diesen Auftragsblock gebildet. Erst nach Ab-

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

arbeitung des Blockes bilden die neu eingetretenen Aufträge einen neuen Auftragsblock. Bei der dynamischen Reihenfolgebildung führt jeder neue Auftrag zu einer erneuten Reihenfolgebildung. [HMS+87] Weitere Entscheidungen sind, welche Warteschlange – die der Einlager- oder die der Auslageraufträge – für den folgenden Auftrag ausgewählt wird. Mögliche Strategien sind dabei die Warteschlange zu wählen, die den Auftrag enthält, der am längsten wartet (first come first served) oder der die kürzeste Bedienzeit aufweist (shortest process time). Eine weitere Möglichkeit ist die Priorisierung aller Einlagerungen, bzw. aller Auslagerungen. Bei der Durchführung von Doppelspielen muss die Wahl der Warteschlange alternieren, da sich so Ein- und Auslageraufträge abwechseln. [VTV12] Nach der Wahl der Warteschlange gilt es, die Reihenfolge innerhalb der Warteschlange zu bilden.

Bei der Sequenzierung muss eine bestimmte Reihenfolge eingehalten werden. Eine weiche Sequenzierung erlaubt es die Reihenfolge innerhalb eines Auftragsbündels frei zu wählen, allerdings ist die Reihenfolge der aufeinanderfolgenden Auftragsbündel einzuhalten. Eine harte Sequenzierung gibt sowohl die Reihenfolge der Auftragsbündel als auch die Reihenfolge innerhalb der Auftragsbündel vor.

Die Festlegung des Fahrmodus des RBG bestimmt die Bildung von Arbeitsspielen (Einfach-, Doppel- und Mehrfachspiele) und den Verweilpunkt des RBG. Der Verweilpunkt ist die Position innerhalb des Systems, an welchem das RBG nach einem erledigten Auftrag pausiert, sofern kein Folgeauftrag vorliegt. [HHC+05] Bezogen auf Shuttle-Systeme kann unterschieden werden in die Spielbildung der Heber und der Shuttle-Fahrzeuge. Ebenfalls gilt es den Verweilpunkt für die Heber und die Shuttle-Fahrzeuge festzulegen.

Je nach Art des Shuttle-Systems kann ein Auftrag durch verschiedene Fördermittel ausgeführt werden. In einem ungebundenen System kann beispielsweise jedes Shuttle-Fahrzeug jeden Lagerplatz anfahren und damit jeden Auftrag ausführen. Hierbei gilt es, das Fördermittel zu bestimmen.

Im Bereich der Forschung lassen sich seit 2002 Forschungsarbeiten zu Shuttle-Systemen finden. [Mal02] Untersuchungsgegenstand sind anfangs die Dimensionierung, die Kosten und der Durchsatz. Die erstellten, vorwiegend analytischen Modelle basieren auf angepassten Modellen für Lager mit RBG oder der Abbildung über die Warteschlangentheorie. Später rückt die Einbeziehung von Lagerstrategien in den Fokus der Forschungsarbeiten, wobei aufgrund der damit verbundenen Komplexität häufig eine simulationsbasierte Analyse bzw. Lösungsfindung erfolgt.

In [SKW+17] wird ein Überblick über bestehende Forschungsberichte zu Shuttle-Systemen mit Bezug zu den untersuchten Lagerstrategien gegeben. Reihenfolgestrategien in Shuttle-Systemen werden in der Regel über first come first served abgebildet. In [CV12] wird die Reihenfolge anhand der kürzesten Bedienzeit gebildet. In [EHK+10] erfolgt ein Vergleich der beiden Strategien first come first served und kürzeste Bedienzeit.

[LF18] entwickeln einen routenbasierten Reihenfolgealgorithmus für ungebundene Systeme. Es wird der kürzeste Weg ermittelt und über reservierte Zeitfenster die Bildung einer geforderten Reihenfolge am Zielort ermöglicht.

In [GAH+16] wird eine Möglichkeit beschrieben, wie Informationen über geschätzte Spielzeiten und Ankunftszeiten der Fahrzeuge in einem ungebundenen System genutzt werden können, um den Fahrzeugen Aufträge zuzuweisen. Das Ziel ist das Auftreten von fehlerhaften Reihenfolgen zu verringern und den Durchsatz zu erhöhen.

Es konnten im Bereich von Shuttle-Systemen keine Veröffentlichungen gefunden werden, die die Reihenfolge mittels RL oder DRL bilden.

3 Vorgehen und Elemente des Reinforcement Learning

Das RL ist eine Methode des Maschinellen Lernens, mit dem Ziel ein geeignetes Aktionsmodell bzw. eine Vorgehensweise (Strategie) zu erlernen. Hierbei unterscheidet sich das RL von anderen Methoden des maschinellen Lernens dadurch, dass das Lernen durch die Interaktion des Algorithmus, auch Agent genannt, mit der Umgebung geschieht.

Es lassen sich fünf Hauptelemente des RL identifizieren:

- Aktion (Action)

  Aus allen möglichen Handlungsalternativen wählt der Agent eine aus.

- Belohnung (Reward)

  Positive oder negative Rückmeldung zu einer getätigten Aktion. Durch die Belohnung wird die gewählte Aktion bezogen auf den Zustand, der bei der Wahl der Aktion vorlag, bewertet.

- Zustand (State)

  Situation, die der Agent konkret zu einem bestimmten Moment vorfindet.

- Strategie (Policy)

  Bestimmt die nächste Aktion des Agenten in einem bestimmten Zustand basierend auf der höchsten erwarteten Belohnung.

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

- Umgebung (Environment)

  Welt, mit der der Agent interagiert. Die Umgebung gibt die Rahmenbedingungen vor und transformiert den Zustand und die Aktion des Agenten in den nächsten Zustand.

Für das RL existieren verschiedene Lernmethoden. Eine grundlegende Methode ist das Q-Learning. Hierbei werden sogenannte Q-Werte genutzt, um schrittweise die Strategie und damit das Verhalten des Agenten zu verbessern. Die Q-Werte (Q(S, A)) repräsentieren die Qualität der Aktionswahl (A) in einem Zustand (S) und können in der Q-Tabelle gespeichert und ausgelesen werden. Durch das Agieren des Agenten in der Umgebung werden diese Werte durch die Rückmeldung der Belohnung schrittweise aktualisiert [SB17]:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1,a}) - Q(S_t, A_t)] \quad (1)$$

Mit α als Lernrate, durch welche die neuen Informationen gewichtet werden und somit bestimmt wird, zu welchem Anteil sich der neue Q-Wert aus neuen bzw. alten Informationen zusammensetzt. γ ist der Diskontierungsfaktor, durch welchen zukünftige Belohnungen (R_{t+1}) diskontiert werden. Somit können Belohnungen, die in der Zukunft liegen weniger gewichtet werden, als die sofortige.

Bei Umgebungen mit vielen Zuständen und Aktionen ist das verarbeiten über Q-Tabellen nicht mehr möglich. Stattdessen werden neuronale Netze zur Approximierung der Q-Werte genutzt, welche als Inputfaktoren den Zustand und als Outputfaktoren die Q-Werte für jede mögliche Aktion besitzen.

Abbildung 1 stellt die Struktur und die Schnittstellen des DRL-Agenten mit der Umgebung dar.

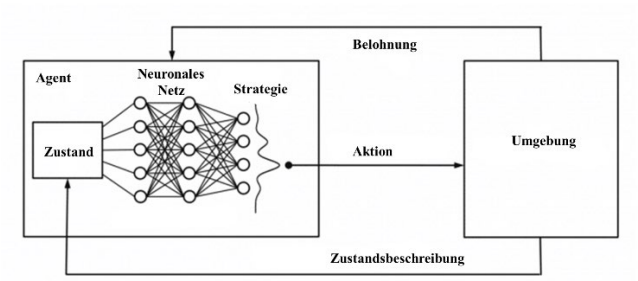

Abbildung 1. DRL-Architektur in Anlehnung an [MAM+16]

Keras bietet den DQN- (Deep-Q-Network) Agenten an, der auf dem Q-Learning aufbaut. Weiterhin stehen andere DRL-Agenten zur Verfügung, die allerdings hier nicht weiter betrachtet werden. Die Agenten unterscheiden sich in der Lernmethode und den Beobachtungs- und Zustandsräumen. [Ker19]

Es können verschiedene Neuronale Netze in den DQN-Agenten implementiert werden. Tiefe neuronale Netze zeichnen sich dadurch aus, dass sie zwischen der Eingabe- und der Ausgabeschicht mehrere verdeckte Schichten besitzen. Gängig ist das Deep Feed Forward (DFF), das Recurrent Neural Network (RNN) und das Convolutional Neural Network (CNN). [ATY+19], [VL19] Bei DFF erfolgt die Informationsweitergabe von der Eingabeschicht bis zur Ausgabeschicht immer vorwärtsgerichtet. Das CNN ist speziell für die Verarbeitung von Bildern oder Audiodaten konzipiert. Mehrere Schichten sind nicht vollsondern nur lokal teilvermascht. Dies verringert den Speicherbedarf und die Trainingszeit. Bei einem RNN liegen Rückkopplungen der Neuronen vor. Ein Neuron kann mit sich selbst, mit einem Neuron aus der gleichen Schicht oder mit einem aus einer anderen Schicht rückgekoppelt sein. Anwendungsbereiche sind beispielsweise Übersetzungen oder Spracherkennung.

4 Modellbeschreibung und Implementierung

4.1 Simulationsmodell

Das erstelle Simulationsmodell ist eine vereinfachte Abbildung einer Ebene eines ebenengebundenen Systems mit drei Lagergängen (Länge 10 m) mit 10 Lagerplätzen pro Gang und einer einfachtiefen Lagerung. Daraus resultieren 60 Lagerplätze. Es existiert ein Quergang mit einer Länge von 16 m, der sich an der Stirnseite des Lagers befindet. Abbildung 2 stellt den Aufbau des Modells dar. Der Auslagerpunkt befindet sich am rechten Ende des Querganges (Punkt 1), der Ausweichpunkt, der als Zielort für die Ausweichfahrten angesteuert wird, befindet sich am linken Ende des Querganges (Punkt 2).

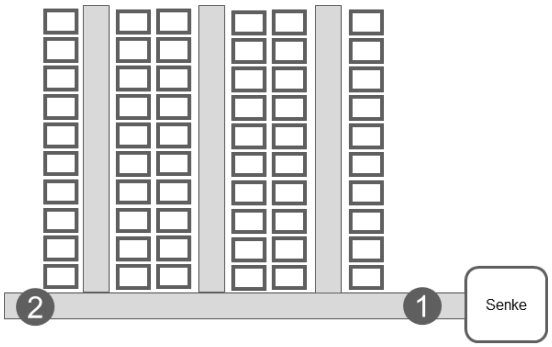

Abbildung 2. Modellstruktur

Auf der Ebene befinden sich zwei Fahrzeuge. Durch den Quergang können beide Fahrzeuge auf der Ebene frei verfahren. Da die Fahrzeuge nicht aneinander vorbeifahren können, jedoch für die Bearbeitung der Aufträge die gleichen Ressourcen (Quergang und die drei Wege der Gänge) nutzen, kommt es zu Blockaden. Die möglichen Blockadesituationen sind in Abbildung 3 dargestellt mit (1) Blockieren im Quergang, (2) Blockieren im Lagergang und (3) Blockieren von Kreuzungen.

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

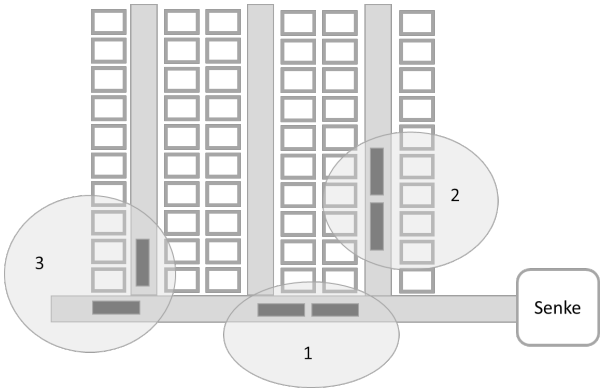

Abbildung 3. Blockiersituationen

Die Steuerung zur Lösung dieser Blockaden ist im Simulationsprogramm folgendermaßen festgelegt:

(1) Blockieren im Quergang:

Zur Auflösung dieser Situation weicht das linke Fahrzeug solange nach links aus, bis das rechte Fahrzeug in den Ziellagergang abbiegen kann. Anschließend steuert das linke Fahrzeug sein ursprüngliches Ziel an.

(2) Blockieren im Lagergang:

Das Fahrzeug, welches sich näher am Quergang befindet, verlässt den Gang und fährt an das linke Ende des Querganges. Dort angekommen macht es sich wieder auf den Weg zum Ziellagerplatz. Sollte die Zeit dieser Ausweichfahrt nicht gereicht haben, damit das im Lagergang verbliebene Fahrzeug seinen Auftrag bearbeitet und den Gang verlassen hat, kommt es zu einer erneuten Blockade und das Fahrzeug unternimmt eine weitere Ausweichfahrt.

(3) Blockieren von Kreuzungen:

Hierbei sind zwei Situationen zu unterscheiden:

(3a) Wenn ein Fahrzeug den Quergang befahren möchte, der Kreuzungsbereich aber von einem anderen Fahrzeug belegt ist, wartet das Fahrzeug im Lagergang, bis der Kreuzungsbereich frei ist.

(3b) Sollte ein Fahrzeug in einen Lagergang einfahren und der Eingang dieses Ganges ist von dem anderen Fahrzeug belegt, begibt sich das Fahrzeug auf dem Quergang zum Ausweichpunkt und steuert wie beim Blockieren im Lagergang danach seinen ursprünglichen Ziellagerplatz an.

Durch die fest programmierte Blockiersteuerung führen die verschiedenen Blockiersituationen zu unterschiedlichen Zeiten zur Auflösung der Situation. Das Blockieren im Lagergang (2) benötigt durch die Ausweichfahrt zum linken Ende des Querganges die längste Zeit, gefolgt von dem Blockieren beim Eintreten in einen Lagergang (3b). Hier erfolgt ebenfalls eine Ausweichfahrt, allerdings mit kürzerem zurückzulegendem Weg bis zum Ausweichpunkt und dadurch mit einer kürzeren Dauer der Ausweichfahrt. Das Auflösen des Blockierens im Quergang (1) benötigt mehr Zeit als das Auflösen des Blockierens an Kreuzungen (3a), da das linke Fahrzeug solange entgegen seinem Zielort fährt, bis das rechte Fahrzeug abbiegt. An der Kreuzung muss das Fahrzeug im Quergang lediglich warten, bis der Kreuzungsbereich frei wird. In Tabelle 1 sind die Blockiersituationen, die Höhe des Zeitverlustes sowie der Grund der Zeiterhöhung zusammengefasst.

Tabelle 1. Übersicht der Blockiersituationen und Auswirkungen

Situation	Dauer	SZE bei	Grund
(1)	++	LF	→
(2)	++++	NQF	→
(3a)	+	GF	warten
(3b)	+++	QF	→

SZE	Spielzeiterhöhung
+,++,+++,++++	Höhe des Zeitverlustes
→	Änderung des Zielortes / der Fahrtrichtung
QF	Fahrzeug auf dem Quergang
GF	Fahrzeug im Lagergang
LF	Linkes Fahrzeug
NQF	Fahrzeug näher am Quergang

Es werden nur Auslageraufträge betrachtet. Anfangs sind alle Lagerplätze gefüllt. Die beiden Fahrzeuge erhalten ihren Fahrauftrag am Auslagerpunkt und lagern nach und nach alle Ladeeinheiten aus. Die Kapazität der Fahrzeuge beträgt eine Ladeeinheit und die Geschwindigkeit 1 m/s.

Dass Blockiereffekte auch in realen Systemen die Spielzeit erhöhen, wurde in [RKH+10] nachgewiesen. Das Blockieren in Lagergängen, im Quergang und an den Kreuzungen hat einen Spielzeitanteil von bis zu 20 %.

4.2 DRL-ARCHITEKTUR

Die DRL-Architektur (siehe Abbildung 5) setzt sich aus dem Simulationsmodell und dem DQN-Agenten zusammen. Als neuronales Netzwerk wird ein DFF mit zwei verdeckten Schichten genutzt. Die DRL-Elemente sind folgendermaßen festgelegt.

- Zustand

  Zu jedem Zeitpunkt, an dem eines der beiden Fahrzeuge meldet, dass es bereit für den nächsten Auftrag ist, übermittelt das Simulationsmodell welcher Auftrag erledigt, in Bearbeitung oder noch zu erledigen ist (Auftragsliste) als Informationen für den Beobachtungsraum an den DQN-

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

Agenten. Abbildung 4 bildet die Matrixdarstellung des Beobachtungsraumes ab.

Auftragsnr.	noch zu erledigen	in Bearbeitung	erledigt
Auftrag 1	0	0	1
Auftrag 2	0	1	0
Auftrag 3	1	0	0
Auftrag 4	1	0	0
…	…	…	…

Abbildung 4. Beobachtungsraum

- Aktion

  Der Agent wählt aus den noch zu erledigenden Aufträgen denjenigen mit dem höchsten Q-Wert als nächsten Auftrag für das anfragende Fahrzeug aus und übermittelt die Auftragsnummer an das Simulationsmodell.

- Belohnungsfunktion

  Zur Ermittlung der zu erhaltenden Belohnung, die der Agent für die Wahl des nächsten Auftrages erhält, wird zunächst die Optimalzeit für die Erledigung des Auftrages ermittelt. Hierfür wird die Simulation vor dem Training des Agenten mit einem Fahrzeug durchgeführt. Da es bei einem Fahrzeug nicht zu Blockaden kommen kann, können so die optimalen Auftragsbearbeitungszeiten pro Auftrag ermittelt werden. Anhand dieser Zeit wird die Aktion bewertet. Da es nicht möglich ist, dass der Auftrag in einer kürzeren Auftragsbearbeitungszeit als der Optimalzeit durchgeführt wird, ist die Belohnung somit bestenfalls null. In diesem Fall entspricht die tatsächliche Bearbeitungszeit der Optimalzeit. Liegen Blockiersituationen vor, verlängert sich die tatsächliche Bearbeitungszeit und die Belohnung wird negativ. Es wurde folgende Belohnungsfunktion gewählt:

  $$r = 100 * (t_{Opt,n} - t_{Ist,n}) \quad (2)$$

  Mit: $t_{Opt,n}$ - Optimalzeit für Auftrag n
  $t_{Opt,Ist}$ - Istzeit für Auftrag n

  Die Belohnung wird durch den DQN-Agenten auf Basis der Inputwerte Optimalzeit und Istzeit des Simulationsmodells berechnet.

- Strategie

  Die zu erlernende Strategie legt die Auftragsreihenfolge fest.

- Umgebung

  Die Umgebung ist das beschriebene Simulationsmodell.

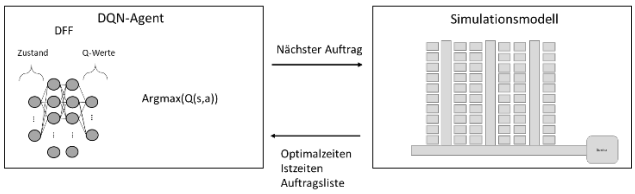

Abbildung 5. DRL-Architektur

5 Einfluss der Reihenfolgeänderung und Vergleichsbasis für den DRL-Agenten

Im Folgenden wird dargestellt, welchen Einfluss die Änderung der Reihenfolge auf die Gesamtbearbeitungszeit der Aufträge hat.

Es wurden 1000 Simulationsläufe durchgeführt, in denen jeweils die kompletten 60 Ladeeinheiten durch eine zufällig generierte Reihenfolge ausgelagert wurden.

Die mittlere Gesamtbearbeitungszeit liegt bei 16:21 min mit einer Standardabweichung von 0:28 min. Der Maximalwert beträgt 17:58 min und der Minimalwert 15:05 min. Die Differenz zwischen der ungünstigsten Reihenfolge und der auf Basis der Simulationsläufe ermittelten optimalen Reihenfolge beträgt damit 2:53 min. Bezogen auf den Maximalwert entspricht das einer Spielzeitreduzierung vom Worst-Case zum Best-Case von 16 %.

Wird das Optimierungspotenzial durch die Differenz zwischen dem durch die Simulationsläufe ermitteltem Optimalwert und dem Mittelwert berechnet, liegt es bei 1:15 min bzw. 7,6 %.

Die auf diese Weise berechnete mittlere Gesamtbearbeitungszeit und das Optimierungspotenzial lassen aufgrund der Vereinfachungen keine Rückschlüsse auf reale Systeme zu. Sie können jedoch für Vergleichszeiten herangezogen werden, um die Performance des DRL-Agenten zu bewerten, da dieser im selben Modell agiert.

Hierfür wurden die Minimalwerte und die Durchschnittswerte der Gesamtbearbeitungszeiten alle 50 Simulationsläufe aktualisiert. Die Minimalzeit verbessert sich von 15:18 min auf 15:05 min, während der Mittelwert nach den ersten 50 Läufen bei 16:24 min liegt und sich dann bei 16:21 min einpendelt. Das Optimierungspotenzial steigert sich von 6,7 % unter Berücksichtigung nur der ersten 50 Läufe auf 7,7 % unter Berücksichtigung der gesamten 1000 Läufe. Die Ergebnisse und das daraus berechnete Optimierungspotenzial sind in Abbildung 6 dargestellt.

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

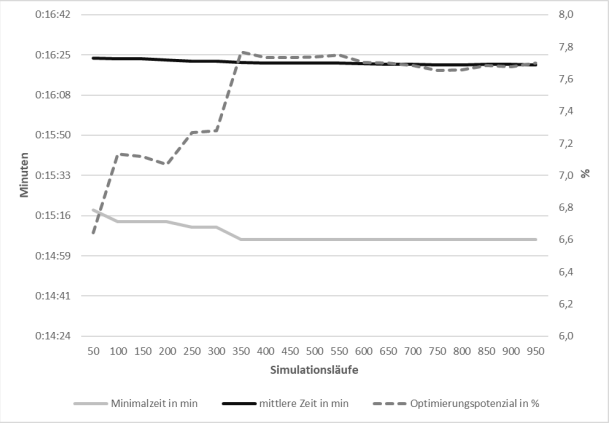

Abbildung 6. Auswertung der Simulationsläufe mit zufälliger Reihenfolge

Im folgenden Forschungsverlauf gilt es den DQN-Agenten auf Basis der beschriebenen Architektur zu implementieren und ihn anhand der Vergleichszeiten zu bewerten.

6 Zusammenfassung und Ausblick

In diesem Beitrag wird eine Möglichkeit vorgestellt, wie durch die Einbeziehung des DRL eine Reihenfolgestrategie für Auslagerungen in Shuttle-Systemen entwickelt werden kann. Hierfür wurde ein Simulationsmodell vorgestellt, welches eine vereinfachte Version eines Shuttle-Systems mit einer Ebene abbildet. Es konnte gezeigt werden, dass die Auslagerreihenfolge einen Einfluss auf die Auftragsbearbeitungszeiten hat. Es wurden weiterhin die Grundlagen des DRL dargelegt und ein Ansatz beschrieben, wie die DRL-Architektur umgesetzt werden kann. Die Implementierung und der Leistungsvergleich erfolgen im weiteren Forschungsverlauf.

Sowohl das Simulationsmodell als auch der DRL-Agent und die RL-Elemente sind zu erweitern.

Es wird ein Simulationsmodell, basierend auf einem realen ebenengebundenen System erstellt, wobei die Anzahl der Fahrzeuge frei wählbar ist. Hierbei sind dann auch Einlagerungen und ggf. Umlagerungen zu berücksichtigen.

Es gilt weitere Belohnungsfunktionen zu bilden und den Effekt auf das Lernen zu analysieren. Zudem beinhaltet der Zustand in der geplanten Architektur nur die noch zu erledigenden Aufträge, er wird um die Lagerpositionen der auszulagernden Güter und die Positionen der anderen Fahrzeuge zum Zeitpunkt der Auftragsanfrage erweitert.

Durch die Einbeziehung von Hebern können auch ungebundene Systeme analysiert werden, hierbei ist zu untersuchen, ob ein Multiagentensystem Anwendung finden muss.

Abschließend ist festzuhalten, dass das DRL im Gaming-Bereich erlernt, die Akteure im Spiel zu steuern. Analog ist denkbar, dass erlernt wird, wie die Fahrzeuge zu steuern sind, beispielsweise zur Auflösung von Blockiersituationen oder, sofern alternative Routen bestehen, zur Findung der besten Route.

Literatur

[ATY+19] Alom, MZ.; Taha, T.; Yakopic, C.; Westberg, S.; Sidike, P.; Nasrin, MS.; Hasan, M.; Van Essen, B.; Awwal, A.; Asari, V.: A State-of-the-Art Survey on Deep Learning Theory and Architectures. In: electronics, 8(3), S. 292 – 359, 2019.

[CV12] Carlo, H.; Vis, I.: Sequencing Dynamic Storage Systems with Multiple Lifts and Shuttles. In: International Journal of Production Economics, 140 (2), S. 844 – 853, 2012.

[EHK+10] Ekren, B.; Heragu, S.; Krishnamurthy, A.; Malmborg, C.: Simulation Based Experimental Design to Identify Factors Affecting Performance of AVS/RS. In: Computers and Industrial Engineering, 58 (1), S. 175 – 185, 2010.

[FHL18] Fottner, J.; Habl, A.; Lienert, T.: Leistungsschub für Shuttle-Systeme. Online: https://www.logistik-fuer-unternehmen.de/2018/Ausgabe-06/Schwerpunkt-Lagerlogistik-und-Kommissioniersysteme/Leistungsschub-fuer-Shuttle-Systeme, 2018. – Abrufdatum: 10.07.2019.

[GAH+16] Gootzen, M.; Adan, I.; Heling, J.; van Wijngaarden, B.: Task Scheduling in a Full Roaming Shuttle System. In: Proceedings of the 2016 Winter Simulation Conference, S. 2844 – 2854, 2016.

[HHC+05] Hu, Y.; Huang, S.; Chen, C.; Hsu, W.; Toh, A.; Loh, C.; Song, T.: Travel Time Analysis of a New Automated Storage and Retrieval System. In: Computers & Operations Research, 32 (6), S. 1515 – 1544, 2005.

[HMS+87] Han, M.; McGinnis, L.; Shieh, J.; White, J.: On Sequencing Retrievals in an Automated Storage/Retrieval System. In: IIE Transactions, 19 (1), S. 56 – 66, 1987.

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

[Irr16] Irrgang, R.: Skalierbar und vielseitig: Shuttles erobern alle Branchen. Online: http://www.materialfluss.de/forder-und-hebetechnik/regalbediengerate/skalierbar-und-vielseitig-shuttles-erobern-alle-branchen/, 2016. – Abrufdatum: 10.07.2019.

[Ker19] Keras-RL Documentation. Online: https://keras-rl.readthedocs.io/en/latest/agents/overview/ – Abrufdatum: 10.07.2019.

[LF18] Lienert, T.; Fottner, J.: Routing-based Sequencing Applied to Shuttle Systems. 21st International Conference on Intelligent Transportation Systems (ITSC), 4.-7.11.2018, Maui, USA.

[Mal02] Malmborg, C.: Conceptualizing Tools for Autonomous Vehicle Storage and Retrieval Systems. In: International Journal of Production Research 40 (8), S. 1807 – 1822, 2002.

[MAM+16] Mao, H.; Alizadeh, M.; Menache, I.; Kandula, S.: Resource Management with Deep Reinforcement Learning. In: HotNets `16 Proceedings of the ACM Workshop on Hot Topics in Networks, S. 50 – 56, 2016.

[Mar19] Marr, B.: Man Vs. Machine: The 6 Greatest AI Challenges To Showcase The Power Of Artificial Intelligence. Online: https://www.forbes.com/sites/bernardmarr/2019/07/08/man-vs-machine-the-6-greatest-ai-challenges-to-showcase-the-power-of-artificial-intelligence/#6bbe3dbe2da2, 2019. – Abrufdatum: 10.07.2019.

[RKH+10] Roy, D.; Krishnamurthy, A.; Heragu, S.; Malmborg, C.: Vehicle Interference Effects in Warehousing Systems with Autonomous Vehicles. 6th annual IEEE Conference on Automation Science and Engineering, 21-24.8.2010, Toronto, Kanada.

[SB17] Sutton, R. und Barto, A.: Reinforcement Learning – An Introduction. The MIT Press, Cambridge u.a., 2017.

[Som15] Sommer, T.: Entwicklung und Bewertung von Lagerstrategien zur Steigerung der Energieeffizienz in automatischen Hochregallagern unter Beachtung des Umschlags. Dissertation an der Universität Stuttgart, 2015.

[SKW+17] Schloz, F.; Kriehn, T.; Wehking, K.; Fittinghoff, M.: Durchsatzoptimierung von Shuttle-Systemen durch situationsabhängige Lagerstrategien. In: Large u. a.: Konferenzband Logistikmanagement - Beiträge zur LM 2017, S. 249 – 257, 2017.

[TS10] ten Hompel, M.; Schmidt, T.: Warehouse Management – Organisation und Steuerung von Lager- und Kommissioniersystemen. 4. Auflage, Springer-Verlag, Berlin und Heidelberg, 2010.

[VDI13] VDI 2692: Shuttle-Systeme für Kleinbehälterlagerung. Verein Deutscher Ingenieure, Beuth-Verlag, Berlin, 2013.

[VL19] Van Veen, F.; Leijnen, S.: The Neural Network Zoo. Online: http://www.asimovinstitute.org/neural-network-zoo/, 2019. – Abrufdatum: 10.07.2019.

[VTV12] Vasili, M.; Tang, S.; Vasili, M.: Automated Storage and Retrieval Systems: A Review on Travel Time Models and Control Policies. In: Manzini, R. (Hrsg.): Warehousing in the Global Supply Chain – Advanced Models, Tools and Applications for Storage Systems, S. 159 – 209, 2012.

[WRB+18] Waschneck, B.; Reichstaller, A.; Belzner, L.; Altenmüller, T.; Bauernhansl, T.; Knapp, A.; Kyek, A.: Optimization of Global Production Scheduling with Deep Reinforcement Learning. In: Procedia CIRP 72, S. 1264 – 1269, 2018.

Franziska Schloz, M.Sc., Wissenschaftliche Mitarbeiterin am Institut für Fördertechnik und Logistik der Universität Stuttgart.
Tel.: +49 (0)711 685 83698
E-Mail: franziska.schloz@ift.uni-stuttgart.de

Thomas Kriehn, M.Eng., Wissenschaftlicher Mitarbeiter an der Professur Materialfluss und Logistikplanung der Hochschule Heilbronn.
Tel.: +49 (0)7131 504 510
E-Mail: thomas.kriehn@hs-heilbronn.de

DOI: 10.2195/lj_Proc_schloz_de_201912_01
URN: urn:nbn:de:0009-14-49751

Univ.-Prof. Dr.-Ing. Robert Schulz, Institutsleiter des Instituts für Fördertechnik und Logistik der Universität Stuttgart.
Tel.: +49 (0)711 685 83770
E-Mail: robert.schulz@ift.uni-stuttgart.de

Prof. Dr.-Ing. Markus Fittinghoff, Inhaber der Professur Materialfluss und Logistikplanung an der Hochschule Heilbronn.
Tel.: +49 (0)7131 504 6802
E-Mail: markus.fittinghoff@hs-heilbronn.de

Adressen:

Institut für Fördertechnik und Logistik
Universität Stuttgart
Holzgartenstraße 15 B
D-70174 Stuttgart

Fakultät Technische Prozesse
Hochschule Heilbronn
Max-Planck-Str. 39
D-74081 Heilbronn