Universität für Bodenkultur Wien

Institut für Produktionswirtschaft und Logistik

Department für Wirtschafts- und Sozialwissenschaften

Optimierungsalgorithmen für das Slicing Tree Verfahren zur Layoutplanung

Masterarbeit von Pascal Perhab

zur Erlangung des akademischen Grades Diplom-Ingenieur

Betreuer: Univ.Prof. Mag.rer.soc.oec. Dr.rer.soc.oec. Manfred Gronalt

Eingereicht am 14.4.2020

I. Eidesstattliche Erklärung

Ich erkläre eidesstattlich, dass ich die Arbeit selbstständig angefertigt habe. Es wurden keine anderen als die angegebenen Hilfsmittel benutzt. Die aus fremden Quellen direkt oder indirekt übernommenen Formulierungen und Gedanken sind als solche kenntlich gemacht. Diese schriftliche Arbeit wurde in gleicher oder ähnlicher Form noch bei keiner anderen Prüferin/ keinem anderen Prüfer als Prüfungsleistung eingereicht.

Wien, am 14.4.2020

Pascal Perhab

II. Abstract

Das Slicing Tree Verfahren ist ein Verfahren zur Layoutplanung und wurde in dieser Masterarbeit zur Optimierung eines Rundholzlagerplatzes angewandt. Für quadratische Zuordnungsprobleme sind Slicing Trees und Optimierungsalgorithmen vielfach untersucht worden. In dieser Arbeit wird die Anwendbarkeit des Slicing Tree Verfahrens auf ein lineares Zuordnungsproblem untersucht und die Leistungsfähigkeit unterschiedlicher Metaheuristiken für dieses Problem evaluiert. Die Optimierung wurde mit drei Metaheuristiken durchgeführt: Hill Climbing, Tabu Search und Simulated Annealing. Im ersten Teil wurden Algorithmen zur Anwendung auf dieses Optimierungsproblem entwickelt. Anschließend wurden die Eingangsparameter der Algorithmen variiert, um ihre Auswirkungen auf die Lösungsgüte zu quantifizieren. Abschließend wurden die Ergebnisse der unterschiedlichen Algorithmen statistisch ausgewertet und verglichen.

The slicing tree method is a method for layout planning and was used in this master thesis for the optimization of operational storage areas. For quadratic assignment problems, slicing trees and optimization algorithms have been investigated many times. In this thesis a linear assignment problem is considered. The optimization was performed with three metaheuristics: Hill Climbing, Tabu Search and Simulated Annealing. In the first part, algorithms were developed to be applied to this optimization problem. Then the input parameters of the algorithms were varied to quantify their impact on the solution quality. Finally, the results of the different algorithms were statistically evaluated and compared.

III. Inhaltsverzeichnis

1 Einleitung

In dieser Arbeit wird die Anwendbarkeit des Slicing Tree Verfahrens auf ein lineares Zuordnungsproblem untersucht und die Leistungsfähigkeit unterschiedlicher Metaheuristiken für dieses Problem evaluiert. Hill Climbing, Tabu Search und ein Simulated Annealing Algorithmus werden an der Optimierung eines Rundholzlagerplatzes angewandt und ihre Ergebnisse hinsichtlich ihrer Leistungsfähigkeit verglichen. Auf dem Rundholzlagerplatz werden die Sortimente (Produkte), nach den unterschiedlichen Bearbeitungsschritten, zwischengelagert. Jedem Produkt wird hierfür ein festgelegter Bereich zugewiesen. Für den Transport von und zu einer Bearbeitungsmaschine werden Flurförderfahrzeuge eingesetzt. Daraus ergibt sich folgender Materialfluss: 1.Bearbeitungsmaschine, Transport zum Lagerplatz, Lagerung am Lagerplatz, Transport zur Weiterbearbeitung, 2.Bearbeitungsmaschine. Zur einfacheren Verständlichkeit wird in dieser Arbeit der Begriff Quelle für die erste Bearbeitungsmaschine und Senke für die zweite Bearbeitungsmaschine verwendet.

1.1 Innerbetriebliche Standortplanung

Die innerbetriebliche Standortplanung oder Layoutplanung beschäftigt sich mit der räumlichen Anordnung der unterschiedlichen Anlagenteile eines Industriebetriebs. Ziel einer solchen Planung ist es, die Kosten und/oder die Zeit für den Transport zwischen den einzelnen Anlagenteilen zu minimieren. Um diese Ziele zu erreichen, sollen die einzelnen Anlagenteile (Maschinen, Lagerplätze und Arbeitsplätze) möglichst optimal auf dem Betriebsgelände oder in der Produktionshalle angeordnet werden. Diese Anordnung der Anlagenteile wird auch als Layoutplan bezeichnet. Optimal bedeutet, dass die Transportwege und damit Durchlaufzeiten auf ein Minimum reduziert werden. Hierfür ist es im Vorfeld notwendig, die Produktionskette genau zu kennen und die innerbetrieblichen Transportkosten der einzelnen Transportsysteme zu erfassen. Idealerweise wird diese Optimierung bereits während der Planungsphase eines Standorts oder Teilbereichs durchgeführt. Eine spätere Änderung ist immer mit weiteren Kosten verbunden, kann aber teilweise notwendig sein, wenn sich der Produktionsprozess verändert.

Das Slicing Tree Verfahren bietet die Möglichkeit, die unterschiedlichen Anforderungen der Anlagenteile, hinsichtlich ihres Flächenbedarfs, zu berücksichtigen. Die Anzahl der möglichen Lösungen ist allerdings zu groß, um den gesamten Lösungsraum nach der besten Lösung zu durchsuchen, wie auch eine Abschätzung des Lösungsraums von Valenzuela & Wang (2002) zeigt. Sie beschreiben den Lösungsraum folgendermaßen:

$$lower\ bound = 2^n n!\, n$$

$$upper\ bound = n!\, 2^n n!\, n$$

Für ein Optimierungsproblem mit 23 Anlagenteilen bedeutet dies, dass der Lösungsraum zwischen $4{,}99*10^{30}$ und $1{,}29*10^{53}$ Lösungen groß ist. Bei einer Rechenzeit von 1ms pro Iteration ergibt dies eine Dauer von $1{,}58*10^{20}$ Jahren bis $4{,}09*10^{42}$ Jahren und damit eine zu lange Zeitdauer, um den gesamten Lösungsraum nach der optimalen Lösung zu durchsuchen. Aus diesem Grund ist es notwendig, einen Algorithmus zur näherungsweisen Lösung von Optimierungsproblemen anzuwenden. Kann diese Heuristik auf unterschiedliche Problemstellungen angewandt werden, wird sie als Metaheuristik bezeichnet. Metaheuristiken sollen den Lösungsraum gezielt nach günstigen Lösungen absuchen. Hierzu wird beispielsweise eine zufällige Startlösung erzeugt und von dieser ausgehend die Nachbarschaft nach besseren Lösungen abgesucht.

Das Einsparungspotenzial für optimierte innerbetriebliche Logistik bewegt sich im Bereich von 10-20% der Logistikkosten. Einsparungen bei den Logistikkosten haben keinen negativen Effekt auf die Qualität und die Geschwindigkeit der Fertigung (vgl. Bauernhansl, et al., 2014). Andere Publikationen zeigen sogar einen Rückgang der Betriebskosten von bis zu 50% (vgl. Drira, et al., 2007). Diese Zahlen hängen sehr stark davon ab, was genau produziert wird und wie dabei der Anteil der innerbetrieblichen Transportkosten an den Gesamtkosten ist. Weil die Bearbeitungsmaschinen in der Holzwirtschaft verhältnismäßig komplex sind und einen großen Platzbedarf aufweisen, ist es nicht möglich die Fertigung an jede Änderung des Produktes anzupassen oder umzubauen. Deshalb muss eine möglichst gute Konfiguration für alle anfallenden Produkttypen gefunden werden und falls möglich, müssen auch schon zukünftige Entwicklungen in der Branche in die Planung miteinbezogen werden. Hingegen kann auf dem

Rundholzplatz leichter eine Umsortierung stattfinden, die die momentanen Bedürfnisse am besten befriedigt. Die bestehenden Sortierpolter können an die momentane Auftragslage angepasst werden. Dies ist nur dann sinnvoll, wenn die Kosteneinsparung durch den Transport die Kosten für die Umsortierung übersteigt.

1.2 Relevanz in der Holzindustrie

Die Anwendungsmöglichkeiten der innerbetrieblichen Standortplanung in der Holzwirtschaft sind sehr vielfältig. Sie kann für die Planung und Optimierung eines Rundholzplatzes sowie anderer Lagerplätze angewandt werden. Weiters kann auch ein gesamtes Sägewerk mit den gewonnenen Erkenntnissen geplant werden. Auch in Platten- und Papierwerken können die Transportkosten zwischen den einzelnen Anlagenteilen minimiert werden.

Auf einem Rundholzplatz kommen unterschiedlichste Transportsysteme zum Einsatz. Im Bereich der Sortierung beim Zuteilen auf die Boxen, werden vorwiegend Längsförderer verwendet. Anschließend bei der Verteilung auf die einzelnen Polter, werden die Bloche entweder mittels Brückenkran, Kranwagen, Großstapler oder Radlader befördert (vgl. Fronius, 1989). Welches Transportmittel verwendet wird, ist wesentlich für die Erstellung des Layouts. Einerseits sollte bei einer Manipulation mit einem Brückenkran die euklidische Distanz für die Berechnung verwendet werden. Andererseits sind für den Transport mit Radladern, Staplern und Kranwagen jeweils ausreichend groß dimensionierte Transportwege im Layout zu berücksichtigen. Die Transportdistanz errechnet sich hier besser mit der rechtwinkeligen Distanz zwischen den beiden involvierten Abteilungen. Transportwege werden zwischen den einzelnen Anlagenteilen eingeplant. Die durchgehenden Guillotineschnitte eignen sich hierfür am besten, es entstehen dadurch keine verwinkelten Transportgassen, sondern geradlinige direkte Transportverbindungen, welche die Manipulation vereinfachen.

1.3 Literaturüberblick

Huka & Gronalt (2018) haben in ihrer Arbeit sehr ausführlich die logistischen Herausforderungen eines Rundholzplatzes beschrieben, wobei sie auch auf die besonderen Anforderungen der Holzindustrie eingegangen sind. Drira et al. (2007) geben in ihrer Arbeit einen Überblick zu den unterschiedlichen Methoden der Layoutplanung. In dieser Arbeit wird

auch darauf eingegangen, dass jeder Industriebetrieb andere Anforderungen an ein Layout hat. Je nach Anwendungsfall sind die Anlagenteile sehr flexibel in ihrem Flächenbedarf oder auch nicht, es können mehrere übereinander liegende Stockwerke erlaubt sein und auch das Transportsystem ist ein wesentlicher Faktor. Scholz (2010) löst in seiner Arbeit ein quadratisches Zuordnungsproblem mit Slicing Trees und einem Tabu Search Algorithmus für ungleich große Anlagenteile. Er erzielt im Vergleich zu anderen Lösungsansätzen ähnliche und bessere Ergebnisse in sehr kurzer Zeit.

Im nachfolgenden Kapitel wird darauf eingegangen wie ein Grundriss für den gesamten Rundholzlagerplatz erstellt werden kann. Anschließend werden Methoden zur Optimierung dieses Grundrisses beschrieben.

2 Material und Methoden

2.1 Datengrundlage

Auf dem bestehenden Werksgelände eines Industriebetriebes soll eine neue Anordnung der Lagerplätze für unterschiedliche Produkte gefunden werden. Hierbei sollen die Lagerplätze möglichst so angeordnet werden, dass die dabei entstehenden Transportkosten so gering wie möglich sind. Der verfügbare Platz für die Lagerplätze ist mit 260m*200m unveränderbar. Die Lagerplätze werden von vorgegebenen Punkten beliefert (Quelle) und müssen an vorgegebene Punkte liefern (Senke). Es gibt Lagerplätze, die direkt beliefert werden. Die dabei anfallenden Transportkosten werden nicht berücksichtigt. Die Lagerplätze müssen eine rechteckige Grundfläche aufweisen, die Größe der Grundfläche ist vorgegeben. Die Grundfläche ist in ihren Abmessungen veränderbar, allerdings muss die kürzeste Seite des Rechtecks 4m betragen. Die für den Transport anfallenden Kosten sind für jedes Produkt unterschiedlich und sind als Kosten pro Jahr und zurückzulegende Meter zu verstehen. Die unterschiedlichen Kosten ergeben sich aus dem Umstand, dass nicht alle Produkte gleich häufig transportiert werden und auch das verwendete Transportmittel unterschiedlich sein kann.

Die Lagerflächen und die Zuordnung der Produkte setzen sich, wie in Tabelle 1 ersichtlich, zusammen. Die Fläche ist die Mindestfläche, die für das jeweilige Produkt zur Verfügung gestellt werden muss. x_{min} und y_{min} beschreiben die Mindestlänge des Lagerplatzes in x- und y-Richtung. In den Spalten Quelle und Senke ist vermerkt von bzw. zu welchem Punkt die Transporte erfolgen. Eine Übersicht zu den Quellen und Senken ist in Tabelle 3 und Tabelle 4 zu finden.

Nr	Name	Fläche [m^2]	x_{min} [m]	y_{min} [m]	Quelle	Senke	Kosten [GE/m]
0	Produkt 1	800	4	4	1	1	7,29
1	Produkt 2	4800	4	4	direkt	4	2,97
2	Produkt 3	2800	4	4	direkt	4	2,16
3	Produkt 4	800	4	4	2	4	9,72
4	Produkt 5	2800	4	4	direkt	3	2,25
5	Produkt 6	3600	4	4	5	6	9,99
6	Produkt 7	800	4	4	direkt	5	6,48
7	Produkt 8	3200	4	4	direkt	5	2,70
8	Produkt 9	2000	4	4	direkt	2	4,32
9	Produkt 10	2000	4	4	direkt	2	4,32

10	Produkt 11	4000	4	4	4	3	2,16
11	Produkt 12	1600	4	4	4	3	1,62
12	Produkt 13	3200	4	4	4	3	2,16
13	Produkt 14	800	4	4	4	2	7,29
14	Produkt 15	3600	4	4	4	2	4,86
15	Produkt 16	3600	4	4	4	2	4,86
16	Produkt 17	400	4	4	4	2	3,78
17	Produkt 18	2000	4	4	direkt	2	3,24
18	Produkt 19	1200	4	4	direkt	2	5,13
19	Produkt 20	400	4	4	direkt	5	1,35
20	Produkt 21	2000	4	4	2	5	2,43
21	Produkt 22	2800	4	4	direkt	5	2,43
22	Produkt 23	400	4	4	direkt	4	4,59
	Summe	49600					

Tabelle 1: Daten der Lagerflächen

Auffallend hierbei ist, dass der benötigte Platz der Lagerflächen kleiner ist als die zur Verfügung stehende Grundfläche: $A_{GESAMT} = 260m * 200m = 52000m^2$. Dies ist notwendig, weil die Bound Curves mit einem maximalen Fehler $\varepsilon_{max} = 0{,}02$ (siehe 2.3) linearisiert werden. Im schlechtesten Fall benötigen die Lagerflächen am Layout somit eine Grundfläche von $A_{LAGER_{max}} = 49600m^2 * 1{,}02 = 50592m^2$. Der zusätzlich zur Verfügung stehende Platz wird bei den ersten platzierten Lagerflächen hinzugefügt. Bessere Ergebnisse werden erzielt, wenn der ungenutzte Flächenbedarf zusätzlichen Dummy Flächen zugeordnet wird. Diese können frei im Layout platziert werden.

Nr	Name	Fläche [m²]	x_{min} [m]	y_{min} [m]	Quelle	Senke	Kosten [GE/m]
23	Dummy	200	0,01	0,01	6	7	0
24	Dummy	200	0,01	0,01	6	7	0
25	Dummy	200	0,01	0,01	6	7	0
26	Dummy	200	0,01	0,01	6	7	0
27	Dummy	200	0,01	0,01	6	7	0
28	Dummy	200	0,01	0,01	6	7	0
	Summe	1200					

Tabelle 2: Daten der Dummy Flächen

Diesen Dummy Flächen werden Kosten von 0GE für den Transport zugeordnet. Zusätzlich besitzen sie eine minimale Größe in x- und y-Richtung von 0,01m, dadurch haben diese Flächen keine Einschränkung hinsichtlich ihrer Abmessungen und können sich auch über die gesamte Länge oder Breite des Layouts erstrecken. Somit ergibt sich ein benötigter Platz für die Lagerflächen und die Dummy Flächen von: $A_{LAGER+DUMMY} = 49600m^2 + 1200m^2 = 50800m^2$, sowie ein maximaler Platzbedarf von: $A_{LAGER+DUMMY_{max}} = 50800m^2 * 1{,}02 = 51816m^2$.

Die Positionen der Quellen und Senken sind in Tabelle 3 und Tabelle 4 ersichtlich.

Quelle	x-Position [m]	y-Position [m]
1	60	200
2	260	70
3	-1	-1
4	60	0
5	200	200
6	-1	-1

Tabelle 3: Daten der Quellen

Senke	x-Position [m]	y-Position [m]
1	60	200
2	260	70
3	220	180
4	240	220
5	40	40
6	200	200
7	-1	-1

Tabelle 4: Daten der Senken

Die Positionswerte von -1 für die Quellen und Senken sind notwendig, um von diesen Punkten ausgehend keine Transporte zu berechnen. Die so markierten Punkte werden auch in der Ausgabe für das CAD Skript ignoriert. Für die korrekte Berechnung der Transportkosten der Dummy Flächen wäre es ausreichend, die jeweiligen Kosten auf 0 zu setzen.

2.2 Verfahren zur Layoutplanung

In der Layoutplanung können mehrere Layoutplanungsmodelle unterschieden werden. Sie unterscheiden sich in der Art, wie die zur Verfügung stehende Grundfläche aufgeteilt wird und den Anlagenteilen zugeordnet wird. Bei einem diskreten Modell wird ein Raster über den Grundriss gelegt, wodurch sich eine Vielzahl von rechteckigen Flächen ergibt. Wird ein kontinuierliches Modell verwendet, verzichtet man auf ein vorgegebenes Raster. Stattdessen werden die Anlagenteile entsprechend ihrem Platzbedarf direkt auf dem Grundriss angeordnet (vgl. Drira, et al., 2007). Weiters gibt es die Möglichkeit, den Grundriss während der Optimierung völlig auszublenden und nur die Nachbarschaftsbeziehungen zu betrachten. Bei diesem Modell stellt die Ableitung eines realisierbaren Layouts bislang allerdings noch ein Problem dar und wurde noch nicht gelöst (vgl. Scholz, 2010).

Koopmans & Beckmann (1957) haben das quadratische Zuordnungsproblem beschrieben. Hierbei handelt es sich um eine diskrete Problemformulierung. In diesem Verfahren wird die zur Verfügung stehende Grundfläche in Rechtecke mit gleichen Abmessungen unterteilt. Jedem dieser Rechtecke wird ein Anlagenteil zugeordnet. Durch das Vertauschen der Anlagenteile wird das Layout verändert und damit verändern sich auch die Kosten. Hierfür müssen die Anlagenteile alle die gleichen Abmessungen aufweisen. Reale Problemstellungen sind schwer bis unmöglich an diese Restriktionen der gleich großen Anlagenteile anzupassen. Abwandlungen dieses Verfahrens bieten die Möglichkeit einen Anlagenteil auf mehrere benachbarte Rechtecke aufzuteilen. Das Raster der Rechtecke kann auch kleiner gewählt werden, um so auf die unterschiedlichen Platzbedürfnisse der Anlagenteile Rücksicht zu nehmen.

Ein Slicing Tree Layout ist eine kontinuierliche Problemformulierung. Die relative Anordnung der Lagerplätze wird durch einen Slicing Tree festgelegt. Hierzu wird die zur Verfügung stehende Grundfläche entsprechend einem Schnittplan in kleinere Abteilung getrennt. Die Grundfläche wird dazu durch einen durchgehenden Schnitt von einer Außenkante zur gegenüberliegenden geteilt. Die Schnitte können in horizontaler oder vertikaler Richtung verlaufen. Die so entstandenen Teilbereiche können durch weitere Schnitte geteilt werden. Dies wird solange durchgeführt, bis jeder Anlagenteil einen entsprechenden Platz zur Verfügung hat (vgl. Otten, 1982).

Eine restriktivere Form eines Slicing Tree Layouts ist das Flexible Bay Layout. Die Grundfläche wird hier in einem ersten Schritt in mehrere Streifen geschnitten. Diese ersten Schnitte sind alle gleichgerichtet, also entweder horizontal oder vertikal. Diese Streifen werden als Bays bezeichnet. Anschließend werden diesen Bays die Anlagenteile zugeordnet. Dadurch entsteht eine Struktur wie bei einem Slicing Tree, mit nur einem erlaubten Schnittrichtungswechsel (vgl. Scholz, 2010).

In dieser Arbeit wird die Slicing Tree Methode angewandt, weil diese im Vergleich mit den anderen vorgestellten Ansätzen eine größere Flexibilität hinsichtlich der Layoutgestaltung bietet. Dadurch können die Anforderungen an einen Lagerplatz (Größe, Abmessungen)

genauer abgebildet werden und man ist nicht durch ein vorgegebenes Raster in der Gestaltung limitiert. Negativ am Slicing Tree Verfahren ist, dass bestehende Installationen auf einem Lagerplatz nicht berücksichtigt werden können. Dies können befestigte Wege, Lichtmasten, Brückenwaagen oder andere für den Betrieb notwendige Einrichtungen sein. Aus diesem Grund ist diese Methode besser für Greenfield Projekte geeignet. Nachfolgend sind die für diese Arbeit wichtigen Eigenschaften des Slicing Tree Verfahrens erklärt.

2.3 Slicing Tree

Bei dem Slicing Tree Verfahren werden die einzelnen Lagerplätze durch horizontale oder vertikale Schnitte voneinander getrennt. Das Slicing Layout bestimmt die genauen Positionen dieser Schnitte und legt damit die Abmessungen der einzelnen Lagerplätze fest. Für die Berechnung werden Bounding Curves, die mit einem maximalen Fehler (ε_{max}) von 2% linearisiert werden, herangezogen (vgl. Scholz, 2010, p. 28ff). Bounding Curves beschreiben den Flächenbedarf und die Abmessungen, die jeder einzelne Lagerplatz benötigt. Dadurch ist es auch möglich, auf spezielle Anforderungen der Lagerplätze Rücksicht zu nehmen. So können auch Anlagenteile berücksichtigt werden deren Flächenbedarf nicht veränderlich ist oder nur in einem sehr kleinen Spielraum.

Für eine einfachere Darstellung und Berechnung der Slicing Trees werden diese in umgekehrter polnischer Notation dargestellt (vgl. Wong & Liu, 1989). Es ist möglich, ein Slicing Layout durch unterschiedliche Slicing Trees zu repräsentieren. Um dies zu verhindern, können skewed Slicing Trees verwendet werden. Ein Slicing Layout wird eindeutig durch einen skewed Slicing Tree repräsentiert (vgl. Wong & Liu, 1989).

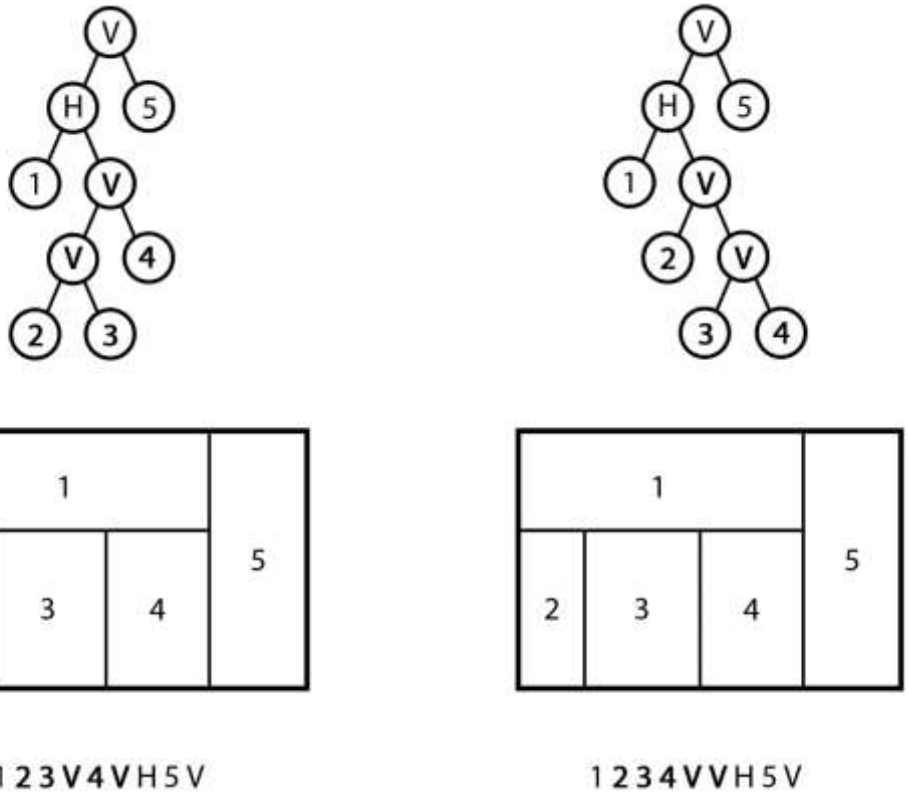


Abbildung 1: Slicing Tree, Slicing Layout und polnische Notation. Links: skewed Slicing Tree. Rechts: non skewed Slicing Tree.

In Abbildung 1 ist ersichtlich, dass zwei unterschiedliche Slicing Trees zu demselben Slicing Layout führen können. Nur wenn skewed Slicing Trees verwendet werden, ist ein Layout eindeutig durch einen Slicing Tree repräsentiert. Wong & Liu (1989) definieren einen skewed Slicing Tree so, dass ein Schnittknoten und sein rechtes Kind niemals die gleiche Schnittrichtung besitzen dürfen.

2.3.1 Nachbarschaft

Um von einem gegebenen Slicing Tree einen neuen abgeänderten Slicing Tree zu erstellen, braucht es Verfahren dafür, wie diese neuen Slicing Trees generiert werden. Es soll möglich sein, jeden beliebigen Slicing Tree durch mehrfache Modifikation einer Ausgangslösung zu erreichen. Scholz, et al. (2009) haben gezeigt, dass mit ihren 4 Modifikationen jeder beliebige Slicing Tree erreicht werden kann.

2.3.2 Modifikation Typ 1

Diese Modifikation wird unter anderem in Valenzuela & Wang (2002), Wong & Liu (1989), Kang & Chae (2017), sowie Scholz, et al. (2009) angewendet. Hier werden zwei Lagerplätze getauscht, wie in Abbildung 2 zu sehen ist. Wenn diese Modifikation an einem skewed Slicing Tree angewandt wird, ist der neue Slicing Tree jedenfalls auch ein skewed Slicing Tree.

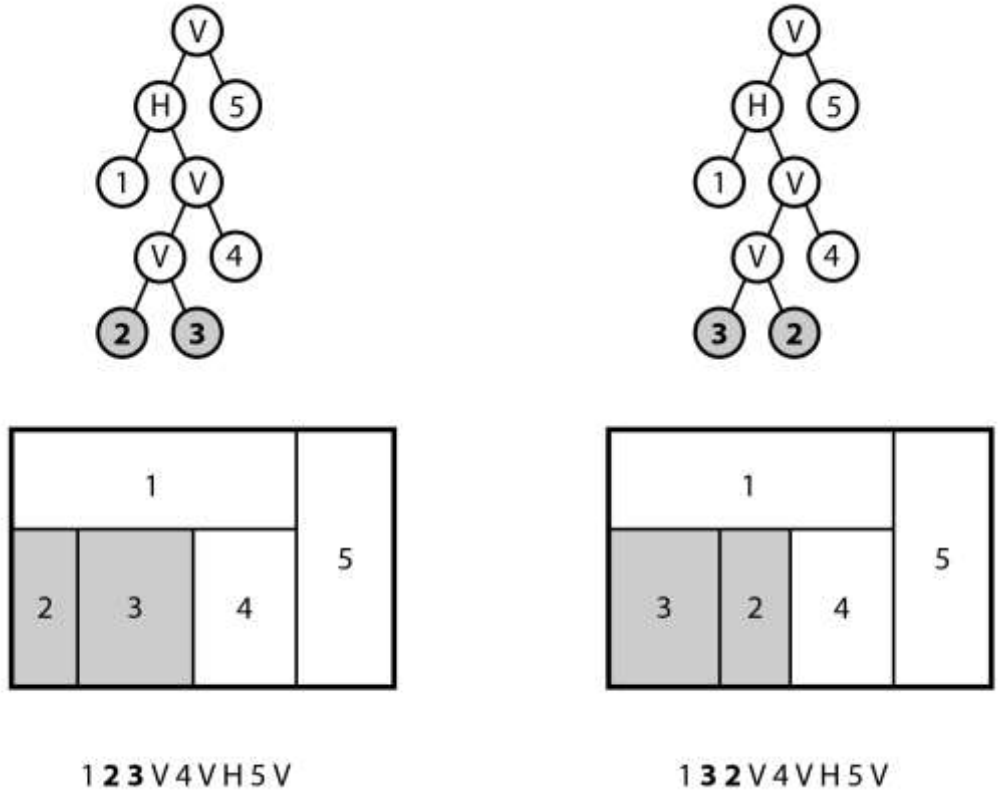


Abbildung 2: Typ1 Modifikation, vertauschen von 2 Lagerplätzen

2.3.3 Modifikation Typ 2

Diese Modifikation wird in Valenzuela & Wang (2002), Wong & Liu (1989), sowie Scholz, et al. (2009) angewendet. Hier wird die Schnittrichtung eines Schnittknotens geändert, aus einem horizontalen Schnitt wird ein vertikaler und aus einem vertikalen ein horizontaler. Bei dieser Modifikation ist es möglich, dass aus einem skewed Slicing Tree ein non skewed Slicing Tree entsteht.

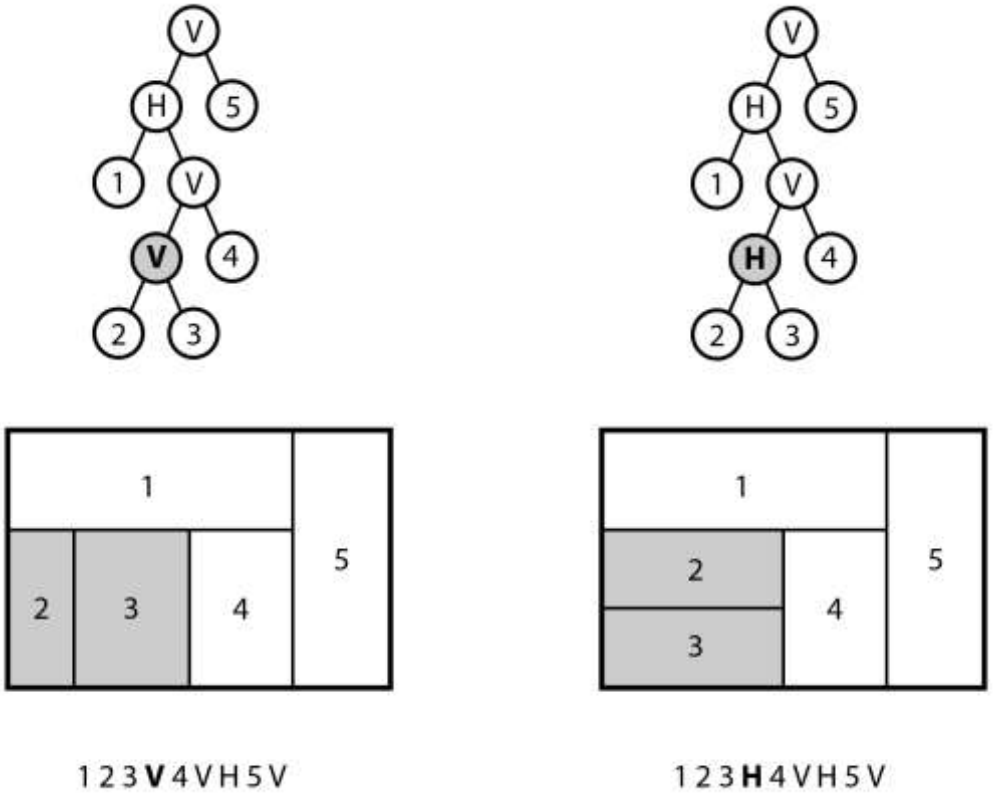


Abbildung 3: Typ2 Modifikation, Schnittrichtungswechsel

2.3.4 Modifikation Typ 3

Diese Modifikation ist eine Abwandlung von Typ 1. Es werden wieder zwei Knoten vertauscht, es muss allerdings zumindest einer der Knoten ein Schnittknoten sein. Die Knoten werden mit ihren gesamten nachfolgenden Knoten an die Stelle des jeweils anderen Knotens verschoben und ebenso umgekehrt (vgl. Scholz, et al., 2009). Wie in Abbildung 4 ersichtlich, ist es möglich, dass durch diese Modifikation ein non skewed Slicing Tree entsteht.

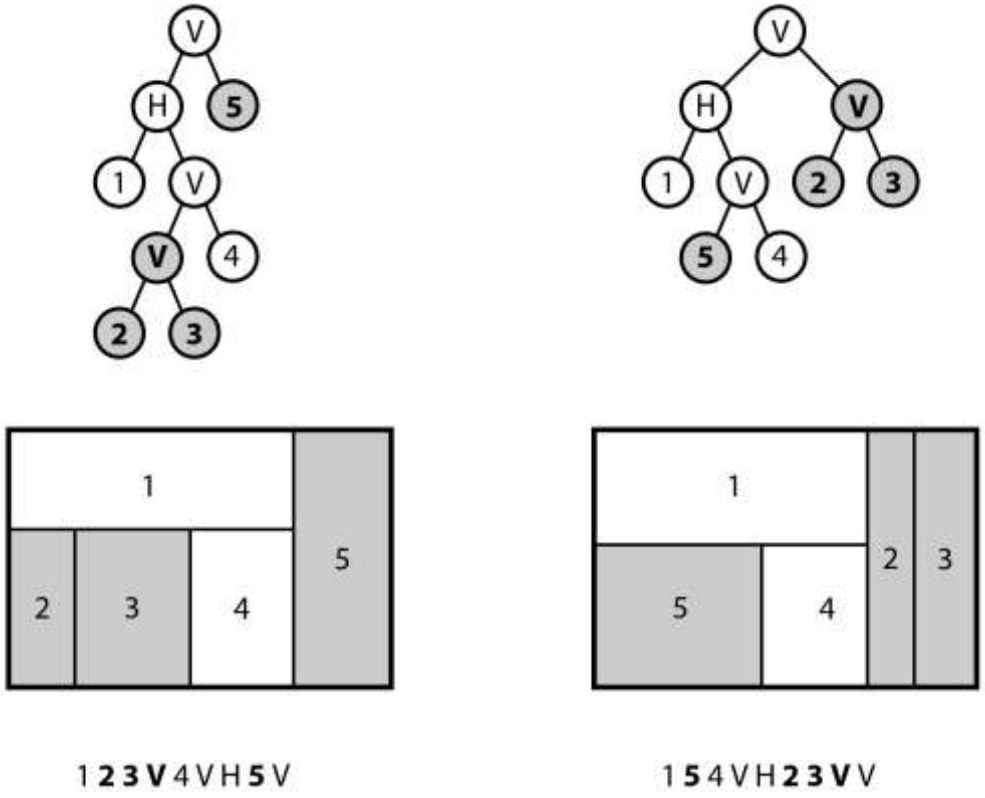


Abbildung 4: Typ3 Modifikation, Vertausch eines Schnittknotens inklusive Anhang mit anderem Schnittknoten oder Abteilung.

2.3.5 Modifikation Typ 4

Bei dieser Modifikation wird ein Knoten des Slicing Trees mit seinen nachfolgenden Knoten an eine andere Stelle des Slicing Trees verschoben. Es ist notwendig, dass der Vater dieses Knotens ebenfalls mitverschoben wird, um einen gültigen Slicing Tree zu erhalten (vgl. Scholz, et al., 2009). Durch diese Modifikation kann ein non skewed Slicing Tree entstehen.

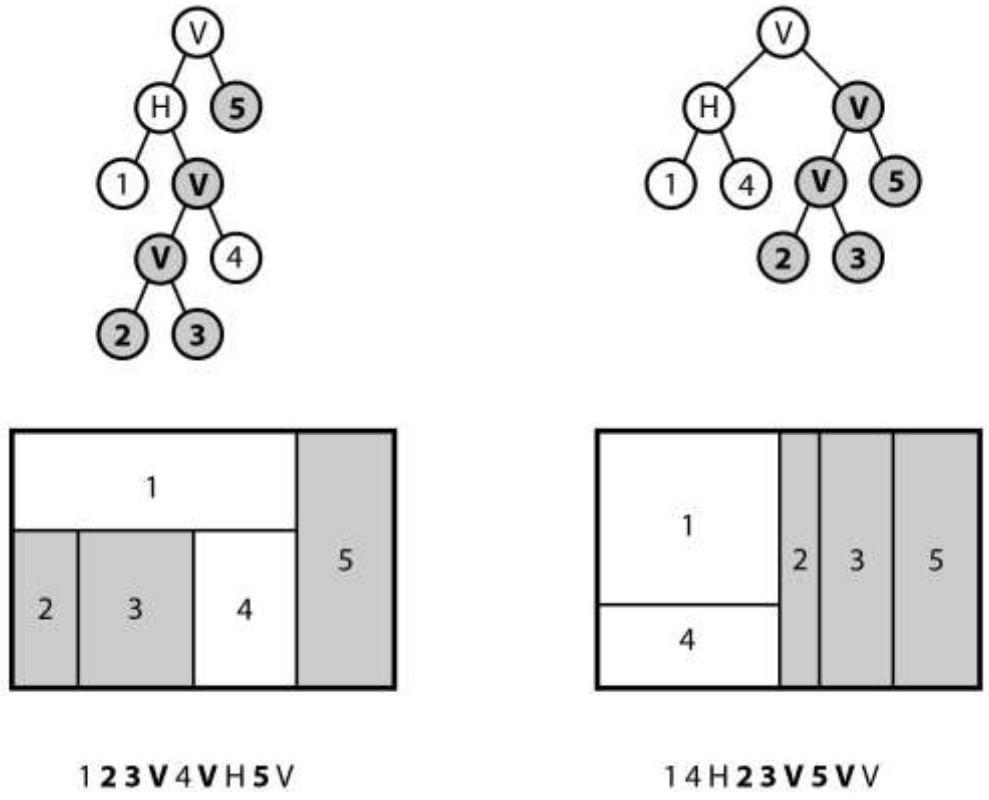


Abbildung 5: Typ4 Modifikation, Verschieben eines Knoten mit eventuell vorhandenem Anhang und seinem Vaterknoten an eine andere Stelle.

2.4 Metaheuristiken

Bei Optimierungsaufgaben sind mehrere Begriffe zu unterscheiden, wobei die Abgrenzung zwischen diesen Begriffen nicht immer ganz eindeutig passiert. Ein *Algorithmus* ist eine Auflistung von Schritten und deren Reihenfolge, um ein Problem zu lösen. Vergleichbar mit einem Rezept. Eine *Heuristik* ist als Herangehensweise an ein konkretes Problem zu verstehen. Eine genaue Abfolge von Einzelschritten ist hier jedoch nicht definiert. Eine *Metaheuristik* kann als allgemeines Konzept zur Lösungsfindung verstanden werden. Sie ist auf unterschiedlichste Probleme anwendbar und sehr allgemein formuliert (vgl. Dittes, 2015). Die in dieser Arbeit

verwendeten Metaheuristiken sind: Hill Climbing, Tabu Search und Simulated Annealing. Die Umsetzung zu konkreten Algorithmen wird in den folgenden Abschnitten erklärt.

2.4.1 Bergsteigeralgorithmus / Hill Climbing

Der Hill Climbing Algorithmus wurde von Lin & Kernighan (1973) beschrieben und wurde für das Traveling-Salesperson Problem angewandt. Der Algorithmus läuft folgendermaßen ab:

```
letzterST = startST;
besterST = startST;
for (int i=0;bessereLoesungGefunden;i++){
     iterationBesterST =  besterSTNachbarschaft(letzterST);
     kostenIterationBester = kosten(iterationBesterST);
     if (kostenIterationBester<kostenBester){
          kostenBester = kostenIterationBester;
          letzterST = iterationBesterST;
          besterST = iterationBesterST;
          bessereLoesungGefunden = true;
     } else {
          bessereLoesungGefunden = false;
     }
}
Ausgabe(besterST);
```

Pseudocode 1: Hill Climbing

Für diese Arbeit wurde der Algorithmus dahingehend adaptiert, dass mit einer vorgegebenen Startlösung begonnen wird. Dies ist notwendig, um die unterschiedlichen Algorithmen miteinander vergleichen zu können. Diese Startlösung ist ein bestimmter Slicing Tree. Von der Startlösung ausgehend wird die gesamte Nachbarschaft, entsprechend den Modifikationen Typ 1-4, generiert. Eine mögliche Einschränkung ist, dass nur skewed Slicing Trees als neue Lösung zugelassen werden. Die Nachbarschaftslösungen werden hinsichtlich ihrer Kosten mit den Kosten der Startlösung verglichen. Wenn die beste Lösung der Nachbarschaft besser als die Startlösung ist, wird diese als neue Startlösung für die nächste Iteration herangezogen. Wenn in der Nachbarschaft keine bessere Lösung gefunden wird, endet der Algorithmus. Es kann durch eine Modifikation der letzten Lösung keine weitere Verbesserung gefunden werden. Das

bedeutet, dass ein lokales Optimum gefunden wurde. Dieses lokale Optimum kann auch gleichzeitig das globale Optimum sein.

Der Hill Climbing Algorithmus findet bereits nach wenigen Iterationen ein lokales Optimum und ist nicht in der Lage dieses zu verlassen und den Lösungsraum weiter zu durchsuchen. Eine Möglichkeit ein lokales Optimum zu verlassen, bietet der später beschriebene Tabu Search Algorithmus. Eine weitere Möglichkeit ist es, nach dem Auffinden eines lokalen Optimums mit einer neuen zufälligen Startlösung, den Hill Climbing Algorithmus neu zu starten und ein weiteres lokales Optimum aufzusuchen. Um den Algorithmus mit den anderen vorgestellten Algorithmen vergleichen zu können und um ihm die gleiche Rechenzeit zukommen zu lassen, wird auch diese Variante untersucht.

2.4.2 Tabu Search

Der Tabu Search Algorithmus wurde von Glover (1986) beschrieben. Er vereint die Fähigkeiten von Hill Climbing schnell eine gute Lösung zu finden, mit der Fähigkeit ein lokales Optimum wieder verlassen zu können. Der Algorithmus läuft folgenderweise ab:

```
letzterST = startST;
besterST = startST;
for (int i=0;currentTime<maxTime;i++){
     iterationBesterST = besterSTNachbarschaft(letzterST);
     kostenIterationBester = kosten(iterationBesterST);
     tabuList.add(typIterationBester,typKnoten);
     if (kostenIterationBester<kostenBester){
          kostenBester = kostenIterationBester;
          besterST = iterationBesterST;
     }
     letzterST = iterationBesterST;
}
Ausgabe(besterST);
```

Pseudocode 2: Tabu Search

Analog zum Hill Climbing Algorithmus wird mit einer vorgegebenen Startlösung begonnen. Auch hier wird die gesamte Nachbarschaft, mit den Modifikationen Typ 1-4, berechnet. Bei jeder berechneten Lösung wird überprüft, ob sie die aktuell beste ist und ob die Modifikation, die zu ihr führt, nicht in der Tabuliste vorhanden ist. Die beste gefundene Lösung wird als Startlösung für die nächste Iteration herangezogen. Weiters wird die Modifikation, die zu ihr

geführt hat, in der Tabuliste ergänzt. Die Größe der Tabuliste wird vorab definiert und ist neben der Anzahl der durchzuführenden Iterationen einer der Parameter, der den Algorithmus beschreibt. Die Tabuliste wird nach dem FIFO (first-in, first-out) Prinzip auf dem aktuellen Stand gehalten. Eine Besonderheit stellen Modifikationen des Typs 4 dar. Sie werden nicht nur für ein Knotenpaar Tabu gesetzt, sondern für den gesamten Slicing Tree. Typ 4 Modifikationen sind verhältnismäßig rechenintensiv und führen nur zu kleinen Verbesserungen an der Lösung (vgl. Scholz, 2010, p. 96). Ein weiterer Eingangsparameter für den Algorithmus ist, ob skewed Slicing Trees als Lösung zugelassen werden oder nicht. Als Aspirationskriterium kann eine global beste Lösung herangezogen werden. Das bedeutet, dass eine global beste Lösung immer als neue Lösung herangezogen wird, unabhängig davon, ob die Modifikation, die zu ihr führt, auf der Tabuliste steht oder nicht.

2.4.3 Simulated Annealing

Simulated Annealing ist eine Analogie zu einem Abkühlungsprozess in der Metallurgie und ist aus dem Metropolis Algorithmus (vgl. Metropolis, et al., 1953) hervorgegangen. Der Algorithmus läuft folgenderweise ab:

```
letzterST = startST;
besterST = startST;
for (int i=0;currentTime<maxTime;i++){
      tempST = randomSTNachbarschaft(letzterST);
      kostenTemp = kosten(tempST);
      if (kostenTemp<kostenLetzter){
            kostenLetzter = kostenTemp;
            letzterST = tempST;
      } else {
            temperatur = calcTemperatur(currentTime, maxTime);
            wahrscheinlichkeit = calcWahrscheinlichkeit(temperatur,
            kostenTemp, kostenLetzter);
            if (random(0, 1) <= wahrscheinlichkeit){
                  kostenLetzter = kostenTemp;
                  letzterST = tempST;
            }
      }
      if (kostenLetzter<kostenBester){
            kostenBester = kostenLetzter;
            besterST = letzterST;
      }
}
Ausgabe(besterST);
```

Pseudocode 3: Simulated Annealing

Von einer beliebigen Startlösung ausgehend wird eine zufällige Nachbarlösung berechnet. Wenn die neue Lösung besser als die vorherige Lösung ist, wird sie übernommen. Mit dieser neuen Lösung wird die nächste Iteration gestartet. Sollte die neue Lösung schlechter als die letzte Lösung sein, wird sie nur mit einer bestimmten Wahrscheinlichkeit akzeptiert.

$$P(ST_{temp}) = e^{-\frac{c_{temp} - c_{letzter}}{T_i}}$$

Aus der Formel geht hervor, dass die Wahrscheinlichkeit für eine Akzeptanz als neue Lösung größer ist, wenn die Differenz zu der letzten gefundenen Lösung kleiner ist. Weiters ist die Wahrscheinlichkeit bei höheren Temperaturwerten ebenfalls höher.

Die Temperatur wird mit fortschreitender Zeit immer weiter verringert und ist eine monoton gegen Null fallende Kurve. Die Form der Temperaturkurve ist damit eine Variable, die Einfluss auf die gefundenen Lösungen hat. Es werden nachfolgend die untersuchten Temperaturkurven beschrieben, die für diese Arbeit herangezogen wurden.

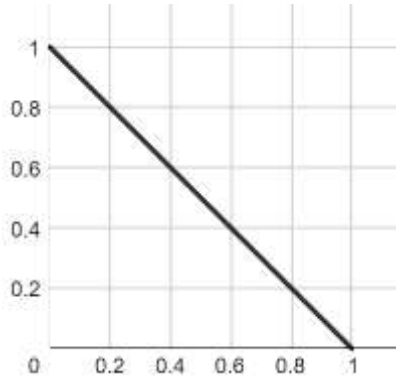


Abbildung 6: lineare Temperaturkurve

Eine linear fallende Funktion der Form: $f_1(x) = -x + 1$

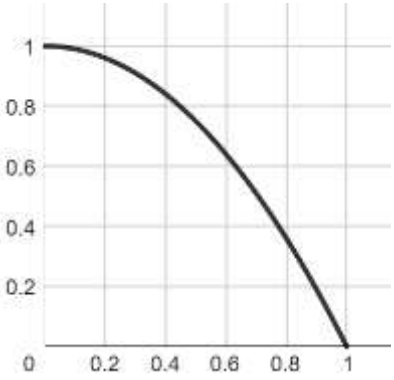


Abbildung 7: degressive Temperaturkurve

Eine degressiv fallende Funktion der Form: $f_2(x) = 1 - x^2$

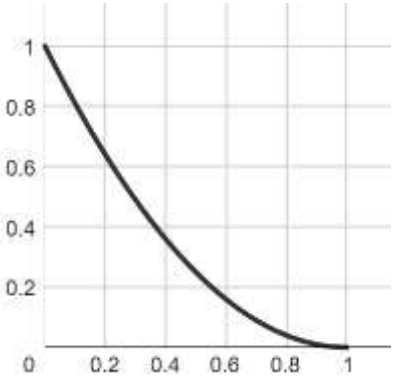


Abbildung 8: progressive Temperaturkurve

Eine progressiv fallende Funktion der Form: $f_3(x) = (-x + 1)^2$, wobei hier der Exponent auch durch andere positive gerade Zahlen ersetzt werden kann, um die Kurve weiter anzupassen. Größere Exponenten bewirken eine schnellere Abkühlung.

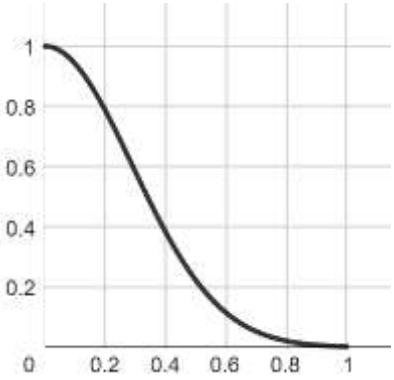


Abbildung 9: S-förmige Temperaturkurve

Die Funktion $f_4(x) = e^{-6x^2}$ kann ebenfalls weiter angepasst werden, durch die Veränderung des Faktors im Exponenten durch eine andere negative Zahl. Kleinere Faktoren resultieren in einer schnelleren Abkühlung.

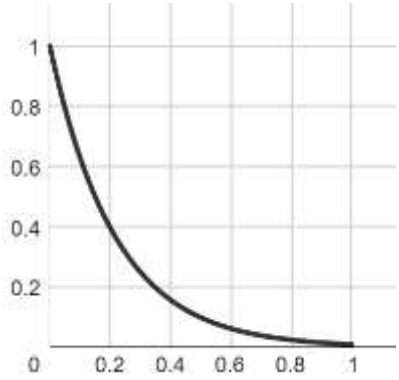


Abbildung 10: exponentieller Zerfall als Temperaturkurve

Die Funktion $f_5(x) = e^{\frac{-x}{\frac{HZ}{\ln(2)}}}$ wird durch den Wert für die Halbwertszeit (HZ) angepasst. Eine kleinere Halbwertszeit resultiert in einer schnelleren Abkühlung.

Der aktuelle Temperaturwert wird aus der jeweiligen Kurve folgenderweise berechnet:

$$T_{Aktuell} = f(x) * (T_{Start} - T_{Ende}) + T_{Ende}$$

$$x = \frac{Laufzeit_{IST}}{Laufzeit_{SOLL}}$$

Daraus ergibt sich eine jeweils monoton fallende Temperaturkurve, die sich in unterschiedlicher Form von einem Startwert (T_{Start}) an einen Endwert (T_{Ende}) annähert. Bei den Kurven 1-4 wird der Endwert auch erreicht.

3 Numerische Experimente

Aus den in Kapitel 2 vorgestellten Methoden zur Layoutgenerierung und Modifikation wird in diesem Kapitel ein Versuchsaufbau erstellt. Weiters werden die Exportmöglichkeiten der gesammelten Daten, zur Analyse der Ergebnisse erklärt.

Um die Qualität der Optimierungsalgorithmen einordnen zu können, wird die obere und untere Schranke für die Problemstellung berechnet. Für die untere Schranke wird die bestmögliche Position der Lagerfläche je Produkt herangezogen, ungeachtet ob die Flächen sich überschneiden. Zur Berechnung der oberen Schranke wird die Lagerfläche so positioniert, dass die Transportkosten möglichst hoch sind. Auch hier wird eine mögliche Überschneidung der Lagerflächen nicht beachtet. Es ergibt sich eine untere Schranke von 10555,64GE und eine obere Schranke von 40271,46GE. Alle Ergebnisse werden sich in diesem Bereich zwischen den beiden Schranken befinden. Die in Tabelle 5 ersichtlichen Werte für $Weg_{min/max}$ und $Kosten_{min/max}$ sind jeweils für einen Transport von der Quelle zum Lagerplatz und weiter zur Senke zu verstehen.

Nr	Name	Weg_{min}[m]	$Kosten_{min}$[GE]	Weg_{max}[m]	$Kosten_{max}$[GE]
0	Produkt 1	7,04	51,33	371,72	2709,81
1	Produkt 2	72,72	215,98	390,58	1160,02
2	Produkt 3	56,61	122,29	410,00	885,60
3	Produkt 4	150,00	1458,00	431,68	4195,96
4	Produkt 5	0,00	0,00	346,82	780,35
5	Produkt 6	17,95	179,33	280,00	2797,20
6	Produkt 7	0,00	0,00	351,18	2275,68
7	Produkt 8	0,00	0,00	321,76	868,76
8	Produkt 9	7,88	34,04	340,57	1471,25
9	Produkt 10	7,88	34,04	340,57	1471,25
10	Produkt 11	340,00	734,40	409,22	883,92
11	Produkt 12	340,00	550,80	418,37	677,75
12	Produkt 13	340,00	734,40	414,75	895,85
13	Produkt 14	270,00	1968,30	593,41	4325,95
14	Produkt 15	270,00	1312,20	529,91	2575,35
15	Produkt 16	270,00	1312,20	529,91	2575,35
16	Produkt 17	270,00	1020,60	609,99	2305,77
17	Produkt 18	14,29	46,29	338,78	1097,64
18	Produkt 19	8,57	43,97	352,04	1805,98
19	Produkt 20	0,00	0,00	359,68	485,57
20	Produkt 21	250,00	607,50	500,30	1215,74
21	Produkt 22	0,00	0,00	325,62	791,25
22	Produkt 23	28,32	129,98	439,97	2019,48
	Summe		10555,64		40271,46

Tabelle 5: Obere und untere Schranke der Transportkosten

3.1 Versuchsaufbau

Für alle Algorithmen wird eine Startlösung benötigt, die laufend verbessert wird. Um die Ergebnisse besser vergleichbar zu machen, werden vorab 20 Slicing Trees generiert, die als Startlösungen für die Algorithmen verwendet werden.

Nr	Slicing Tree
1	18 27 H 22 H 16 H 21 H 24 H 4 H 5 H 0 H 20 6 V H 8 26 H 12 H 1 H 17 25 V 15 3 V 9 7 13 V H 19 H 10 H V 14 H 2 H 28 H V H 11 H 23 H V
2	15 11 16 V H 26 H 23 V 8 H 20 H 6 21 H 3 V H 22 H 17 V 27 H 4 V 12 H 5 V 18 V 28 V 2 25 V H 9 10 V 13 V 14 H V 7 V 19 1 H 0 H V 24 H
3	25 6 V 24 H 13 7 26 H 1 H V 2 V 19 V H 27 H 3 H 16 H 10 H 11 8 H V 17 V 22 V 12 V 18 H 20 14 H V 4 V 23 V 15 H 5 V 0 V 21 9 H V 28 V
4	27 22 V 24 V 26 H 23 H 0 H 4 25 7 H V 5 H 21 V H 6 19 V H 3 28 V H 10 15 H V 18 H 2 V 13 V 20 V 9 H 14 H 12 V 1 H 11 V 16 V 17 V 8 H
5	26 0 16 V H 27 H 20 H 10 V 28 V 17 13 V H 11 H 6 22 H V 14 V 19 V 15 H 24 H 9 5 V H 7 1 H 23 V 8 V H 21 H 25 H 18 H 3 H 2 H 4 H 12 V
6	19 6 H 3 H 21 V 4 20 H 16 V 25 H V 12 13 V 24 V 23 V 18 V 5 H V 15 V 22 V 14 V 26 8 H V 10 0 H V 27 H 2 9 H V 28 H 1 H 11 H 17 V 7 V
7	26 28 V 3 H 5 V 11 V 12 V 21 V 15 V 20 V 8 V 4 V 25 V 18 19 H 9 H V 7 17 V 2 V 14 H 16 H 23 H V 10 6 H V 24 V 1 V 22 27 0 V H 13 H V
8	9 2 18 V 20 V 15 V H 24 V 28 V 26 H 13 H 7 14 V 19 H 4 H V 3 V 23 V 17 6 V H 12 16 H V 0 5 10 V H 22 H 8 H 25 11 V 1 V 21 V H 27 H V
9	21 23 H 9 H 24 V 17 V 14 V 16 V 20 H 1 V 25 V 2 V 28 H 22 11 H 6 V H 4 H 13 H 3 V 10 H 26 V 27 5 H V 15 V 0 19 H V 8 H 7 H 18 12 V H
10	5 8 V 19 V 10 H 9 V 12 H 11 V 0 V 20 V 13 1 V 18 H 6 V 2 3 26 H 22 H V 24 V 15 H V 14 21 H V 28 V 27 25 H 23 V 7 V H V 4 V 17 H 16 V
11	18 27 H 15 V 6 21 H V 11 V 8 V 5 H 12 V 0 22 25 H 28 H V H 10 H 19 V 3 V 24 H 17 4 V H 26 H 7 H 23 H 2 H 9 H 20 H 1 14 V 13 V 16 V H
12	13 22 6 V H 12 V 8 H 17 H 24 H 1 H 15 V 25 H 2 V 10 V 19 V 16 H 20 V 0 V 23 H 26 H 18 H 3 H 28 V 7 V 27 H 11 H 9 H 21 V 14 V 5 V 4 V
13	6 19 V 22 9 V H 26 V 20 V 10 V 3 15 5 2 V H V H 14 V 25 V 12 H 8 H 18 16 V H 27 H 23 H 1 H 24 V 28 H 7 V 4 V 0 H 13 V 17 H 21 H 11 H
14	16 6 H 27 V 4 V 10 V 2 V 17 V 13 H 11 H 22 H 21 20 H V 7 V 23 V 26 V 12 H 14 H 9 18 H 25 8 V H 5 H 15 28 3 V H V 0 V 24 V 1 V 19 V H
15	28 25 H 24 H 22 V 9 V 12 15 V H 0 H 26 V 8 H 6 V 5 V 1 H 19 H 17 10 H V 4 V 7 H 21 H 11 23 14 H V 13 V 3 V 20 V H 2 27 V 16 V 18 V H
16	22 7 V 18 H 14 20 26 V 21 H 9 H V H 0 V 24 V 17 V 1 V 15 V 27 V 3 V 4 V 13 V 2 V 28 V 19 25 6 V 16 H V 12 H 5 H V 8 V 23 H 10 V 11 V
17	16 27 H 18 V 0 V 11 H 13 H 15 21 H V 23 H 8 H 6 H 14 H 12 3 H V 22 V 7 V 19 V 5 H 24 V 9 H 28 V 4 V 26 V 17 V 1 H 25 H 2 V 10 H 20 V
18	25 18 V 2 15 24 H V 3 H V 27 23 16 V 1 V 0 V H V 17 V 6 V 10 V 19 H 7 8 22 H 11 9 H 13 H V 12 V H V 28 H 5 V 4 26 V H 20 14 V 21 V H
19	12 17 25 V H 15 H 0 V 2 V 19 V 14 V 21 H 24 H 13 H 3 H 10 16 22 23 V H V 6 V H 28 H 18 H 1 H 20 26 H V 7 H 27 V 4 V 5 V 9 11 H V 8 V
20	0 9 H 26 H 12 V 15 H 1 V 2 H 25 H 23 V 27 V 22 4 H V 13 H 21 V 5 V 17 8 V 16 V H 19 V 7 V 24 V 11 H 28 H 3 6 H 20 H V 14 10 H 18 H V

Tabelle 6: Slicing Trees der Startlösungen

Um die Startlösungen zu generieren, werden erst die Lagerplätze zufällig verteilt. Anschließend wird an jeder möglichen Stelle für einen Schnittknoten entschieden, ob ein vertikaler, horizontaler oder kein Schnitt eingefügt wird. Dies wird so lange durchgeführt, bis ein gültiger Slicing Tree als Resultat verfügbar ist. Die so gefundenen Slicing Trees werden zu skewed Slicing Trees umgeformt.

Nr	Kosten [GE]
1	29867,76
2	29200,23
3	27856,79
4	24838,34
5	31342,26
6	27783,00
7	24076,61
8	24091,03
9	28586,84
10	27687,05
11	30597,73

Nr	Kosten [GE]
12	28913,99
13	26252,10
14	29041,71
15	28946,78
16	30366,60
17	29859,56
18	26682,23
19	28795,62
20	27962,53

Tabelle 7: Kosten der Startlösungen

In Tabelle 7 sind die Kosten der 20 unterschiedlichen Startlösungen ersichtlich. Sie bewegen sich im Bereich zwischen 24076,61 GE und 31342,26 GE, der Mittelwert ist 28137,43 GE. Die Nullhypothese, dass keine Normalverteilung vorliegt, ist zu verwerfen (siehe Kapitel 8.1). Dies ist wichtig, weil somit davon ausgegangen werden kann, dass die Startlösungen repräsentativ für die Grundgesamtheit aller Lösungen sind.

Alle eingesetzten Algorithmen bekommen ein Zeitfenster von 15 Minuten für jede Startlösung. Das beste Ergebnis, das in dieser Zeit gefunden wird, wird als Vergleichswert herangezogen.

3.1.1 Bergsteigeralgorithmus / Hill Climbing

Für den Hill Climbing Algorithmus wurden folgende Varianten untersucht

- HCla, Hill Climb Algorithmus, der im lokalen Optimum endet und alle Slicing Trees zulässt.
- HCls, Hill Climb Algorithmus, der im lokalen Optimum endet und nur skewed Slicing Trees zulässt.
- HCza, Hill Climb Algorithmus, der nach dem lokalen Optimum mit einer zufälligen neuen Startlösung beginnt und alle Slicing Trees zulässt.
- HCzs, Hill Climb Algorithmus, der nach dem lokalen Optimum mit einer zufälligen neuen Startlösung beginnt und nur skewed Slicing Trees zulässt.

Jede Variante wurde mit jeder der 20 Startlösungen ausgeführt, daraus ergeben sich folgende Versuche:

Abkürzung	Anzahl
HCla	20
HCls	20
HCza	20
HCzs	20
Summe	80

Tabelle 8: Durchgeführte Versuche Hill Climbing

3.1.2 Tabu Search

Die untersuchten Varianten des Tabu Search Algorithmus sind:

- TSaa, Tabu Search Algorithmus, global beste Lösung setzt Tabuliste außer Kraft, alle Slicing Trees
- TSas, Tabu Search Algorithmus, global beste Lösung setzt Tabuliste außer Kraft, nur skewed Slicing Trees
- TSna, Tabu Search Algorithmus, global beste Lösung setzt Tabuliste nicht außer Kraft, alle Slicing Trees
- TSns, Tabu Search Algorithmus, global beste Lösung setzt Tabuliste nicht außer Kraft, nur skewed Slicing Trees

Sowie weitere Varianten für unterschiedliche Längen der Tabuliste. In einem ersten Durchlauf wurden alle Varianten mit Tabulängen von 50 und 200 ausgeführt. In einem zweiten Schritt wurden weitere Tabulängen untersucht, allerdings nur für die besseren Varianten (TSaa und TSna).

Abkürzung	Tabulänge	Anzahl
TSaa	30	20
	50	20
	70	20
	200	20
TSas	50	20
	200	20
TSna	30	20
	50	20
	70	20
	200	20
TSns	50	20
	200	20
Summe		240

Tabelle 9: Durchgeführte Versuche Tabu Search

3.1.3 Simulated Annealing

Simulated Annealing wurde mit den 5 Abkühlkurven aus Kapitel 2.4.3 durchgeführt. Für alle 5 Temperaturkurven wurde eine Starttemperatur von 2000 gewählt und eine Endtemperatur von 0. Nach einem ersten Durchlauf wurde für vielversprechende Lösungen die Halbwertszeit, der Exponent oder der Faktor (y) der Abkühlkurve variiert. In der folgenden Tabelle sind die untersuchten Varianten ersichtlich:

Abkürzung	Temperaturkurve	Variable y	Anzahl
SA1	$f_1(x) = -x + 1$	-	20
SA2	$f_2(x) = 1 - x^2$	-	20
SA3	$f_3(x) = (-x + 1)^y$	2	20
		6	20
SA4	$f_4(x) = e^{-yx^2}$	6	20
		9	20
SA5	$f_5(x) = e^{\frac{\frac{-x}{y}}{\ln(2)}}$	0,05	20
		0,10	20
		0,15	20
Summe			180

Tabelle 10: Durchgeführte Versuche Simulated Annealing

3.2 Java Applikation

Die Algorithmen wurden in einem Java Programm realisiert. Dieses Programm ist inklusive einer Anleitung der Arbeit auf einem Datenträger beigelegt. Das Programm bietet die Möglichkeit, die gewonnen Daten in unterschiedlichen Formaten bereit zu stellen.

3.2.1 CSV Export

Die gefundenen Ergebnisse und alle Zwischenlösungen werden, während der Algorithmus läuft, in eine CSV-Datei geschrieben. Diese Daten wurden zur weiteren Analyse in SPSS importiert. Nachfolgend eine Auflistung der enthaltenen Daten, die Werte in Klammern werden nur bei den zutreffenden Algorithmen gespeichert:

Spalte	Algorithmus	Erklärung
Iteration	HC, TS, SA	In der wievielten Iteration befindet sich der Algorithmus bei dieser Lösung.
LaufzeitGesamt [ms]	HC, TS, SA	Laufzeit seit dem Start des Algorithmus
Modifikation [TYP]	HC, TS, SA	Welche Modifikation hat zu diesem ST geführt
(Knoten 1)	HC, TS	Welcher Knoten wurde verändert (getauscht, verschoben, Schnittrichtungswechsel)
(Knoten 2)	HC, TS	Zweiter Knoten für die Modifikation (-1, bei Schnittrichtungswechsel)
(bessereLoesung)	HC	True wenn eine bessere Lösung im Vergleich zur vorherigen Lösung gefunden wurde.
(bessereLoesung)	TS	True wenn eine bessere Lösung im Vergleich zu den bisherigen Lösungen gefunden wurde.
(Temperatur)	SA	Momentaner Temperaturwert
(Wahrscheinlichkeit)	SA	Die Wahrscheinlichkeit, dass die Lösung akzeptiert wird
Kosten	HC, TS, SA	Kosten der aktuellen Lösung
besteKosten	HC, TS, SA	Kosten der besten bisher gefundenen Lösung
Laufzeit [ms]	HC, TS, SA	Laufzeit seit der letzten Iteration
ST	HC, TS, SA	Slicing Tree der Lösung

Tabelle 11: Enthaltene Daten der CSV-Datei

3.2.2 CAD Export

Nach Abschluss der Berechnungen wird das Layout der besten Lösung in eine SCR-Datei geschrieben. Diese Datei kann mit dem Befehl „SCRIPT“ in AutoCAD importiert werden. Vor dem Ausführen ist darauf zu achten, dass der Objektfang sowie der Raster in AutoCAD deaktiviert sind. Das Skript beinhaltet das gesamte Layout, die Lagerplätze werden als Rechtecke entsprechend den berechneten Abmessungen gezeichnet. Weiters sind die Namen der zugeordneten Produkte, die Quellen, die Senken sowie die Transportwege enthalten.

4 Ergebnisse

Die in Kapitel 3.1 vorgestellten Versuche wurden mit der Java-Applikation durchgeführt. Die ermittelten Ergebnisse werden mit Grafiken dargestellt und auf Besonderheiten der einzelnen Algorithmen wird genauer eingegangen.

4.1 Bergsteigeralgorithmus / Hill Climbing

Der Hill Climbing Algorithmus wird einerseits als lokale Suche (HCl) nach dem Optimum ausgeführt und andererseits nach dem Auffinden eines lokalen Optimums mit einer zufälligen neuen Startlösung neu gestartet (HCz). Die zufällige neue Startlösung ist nicht auf die 20 generierten Startlösungen, beschränkt sondern wird nach demselben Schema wie die 20 Startlösungen neu generiert (siehe Kapitel 3.1) Der zufällige Neustart führt zu signifikant besseren Ergebnissen (siehe Kapitel 8.2). Der Grund hierfür ist, dass in der zur Verfügung stehenden Zeit (15 Minuten) ein Vielfaches an Startlösungen auf mögliche Verbesserungen untersucht wird.

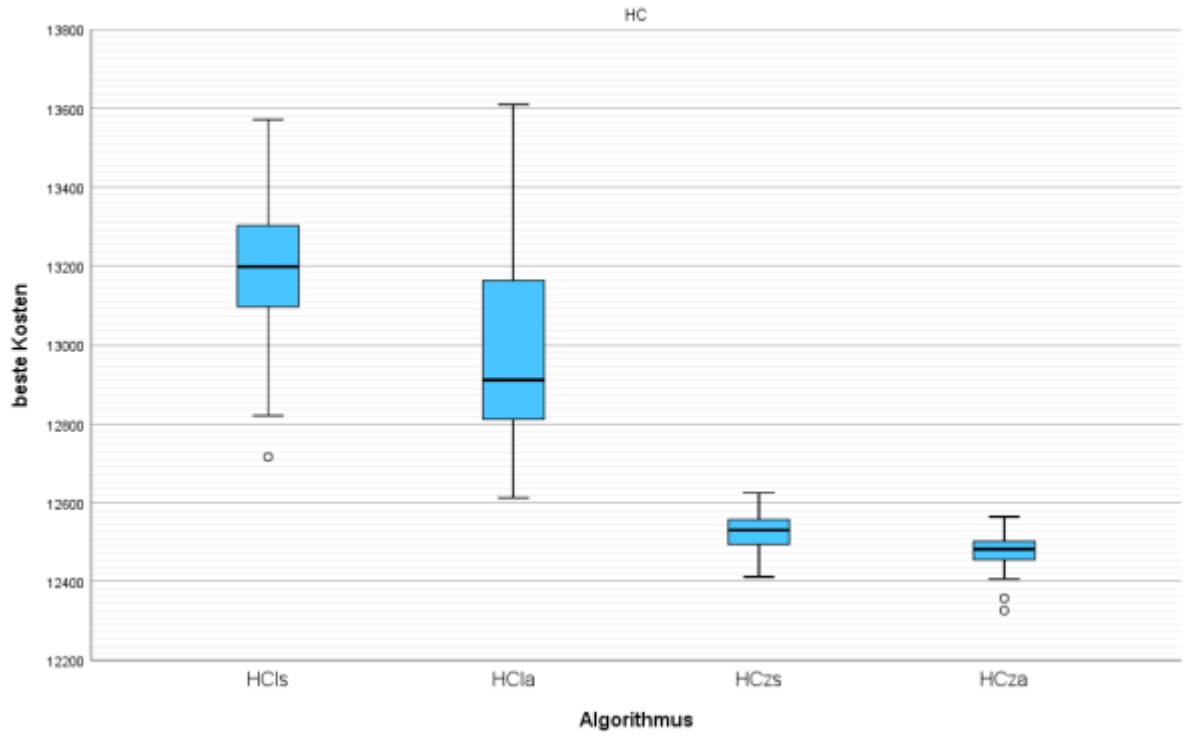


Abbildung 11: Boxplot der Hill Climbing Ergebnisse

4.1.1 Lokale Suche

In Abbildung 12 sieht man den zeitlichen Verlauf der Kosten für die Startlösung 2. Es ist ersichtlich, dass die ersten drei Iterationen zu den gleichen Lösungen führen. Erst danach findet der Algorithmus, welcher die gesamte Nachbarschaft betrachtet (HCla), eine bessere Lösung als HCls.

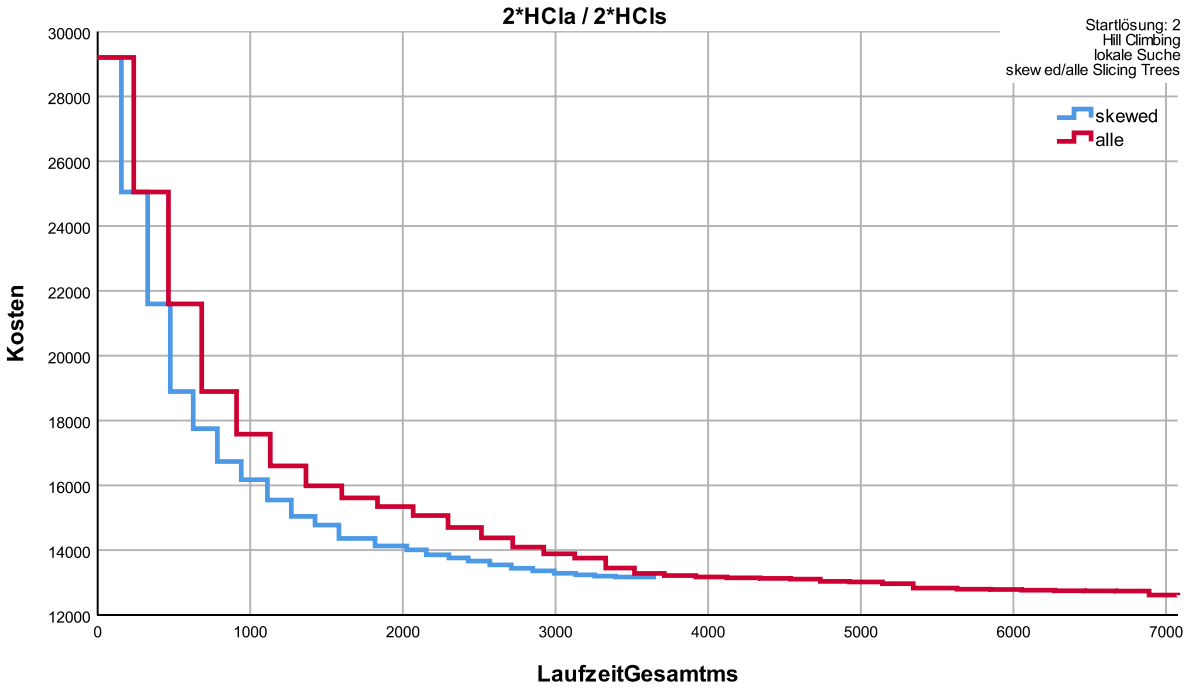


Abbildung 12: Verlauf der Kosten bei lokalem Hill Climbing für die Startlösung 2

Die lokale Suche findet nach durchschnittlich 30,7 Iterationen die beste Lösung. Wenn der Suchraum auf skewed Slicing Trees beschränkt wird, ist ein lokales Optimum schneller gefunden (29,3 Iterationen in 4,5 Sekunden). Wird die gesamte Nachbarschaft durchsucht, sind durchschnittlich 32,1 Iterationen und 7 Sekunden notwendig. Wenn alle benachbarten Slicing Trees betrachtet werden, werden signifikant bessere Lösungen gefunden (siehe Kapitel 8.2) als bei der Einschränkung auf skewed Slicing Trees.

4.1.2 Neustart mit zufälligem Slicing Tree

In Abbildung 13 sieht man den zeitlichen Verlauf des Hill Climbing Algorithmus, der mit einer zufälligen neuen Lösung weiterarbeitet, nachdem ein lokales Optimum gefunden wurde. Es ist ersichtlich, dass viel Rechenzeit mit schlechten Lösungen verbracht wird. Die beste Lösung, mit Kosten von 12326,89 GE, wurde aus einer Startlösung, mit Kosten von 30536,17 GE, generiert.

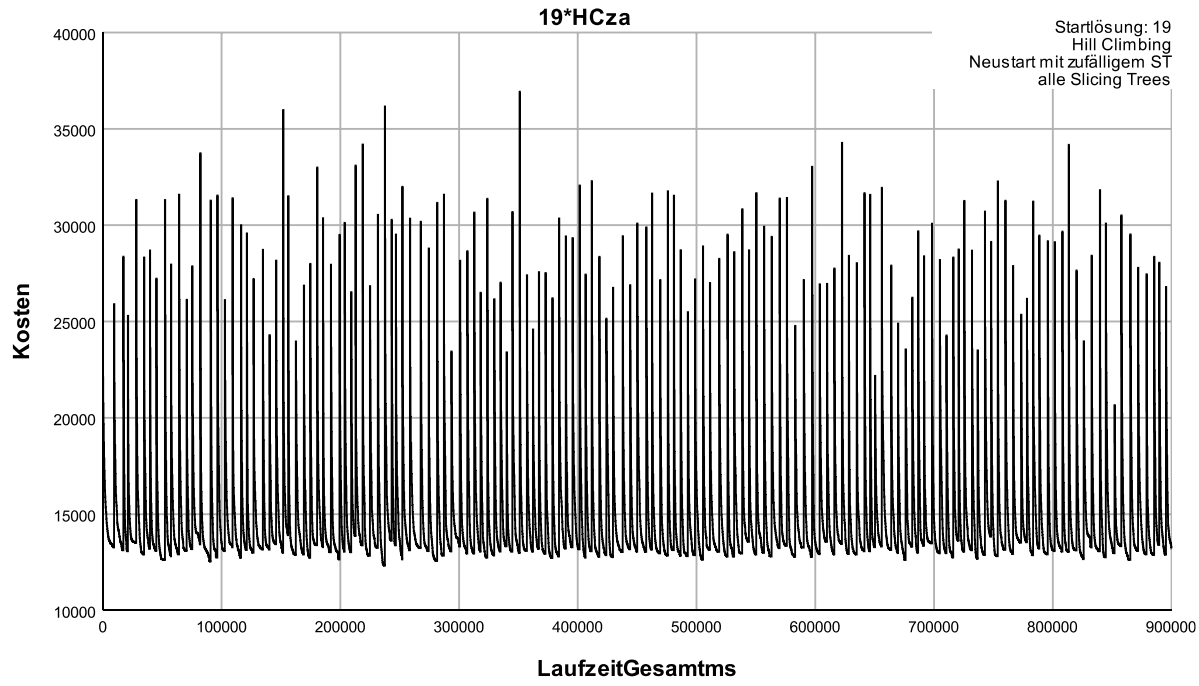


Abbildung 13: Verlauf der Kosten bei Hill Climbing mit Neustart für die Startlösung 19

In Abbildung 14 ist ersichtlich, dass zwischen der Startlösung und dem lokalen Optimum bei der Betrachtung aller benachbarten Slicing Trees (HCza) kein Zusammenhang besteht. Wenn die Nachbarschaft auf skewed Slicing Trees (HCzs) beschränkt wird, liegt eine mittelstarke, negative Korrelation vor (siehe Kapitel 8.4).

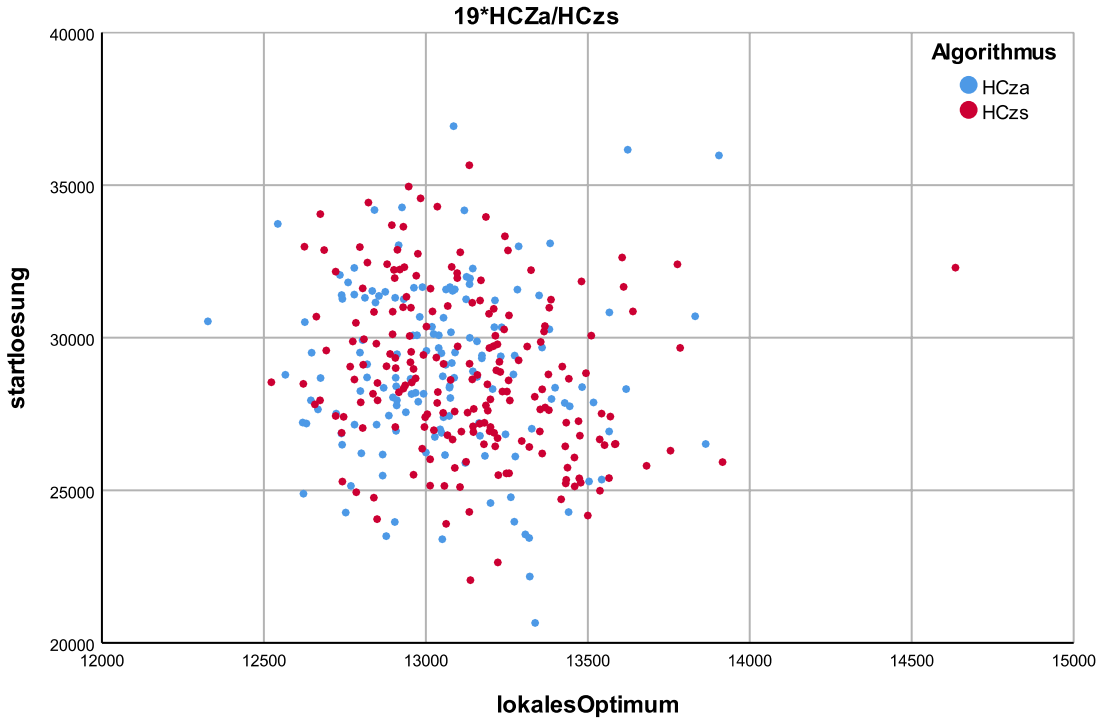


Abbildung 14: Zusammenhang Kosten der Startlösung und lokales Optimum

Der Verdacht, dass eine schlechtere Startlösung dazu führt, dass mehr Iterationen durchgeführt werden müssen, bis ein lokales Optimum erreicht wird und dieses dann besser ist, hat sich nicht bestätigt. Es besteht kein signifikanter Zusammenhang zwischen den Kosten der Startlösung und den Iterationen (siehe Kapitel 8.4). Zwischen dem lokalen Optimum und den Iterationen besteht eine mittelstarke, negative Korrelation. Das bedeutet, wenn mehr Iterationen durchgeführt werden, ist das lokale Optimum tendenziell besser.

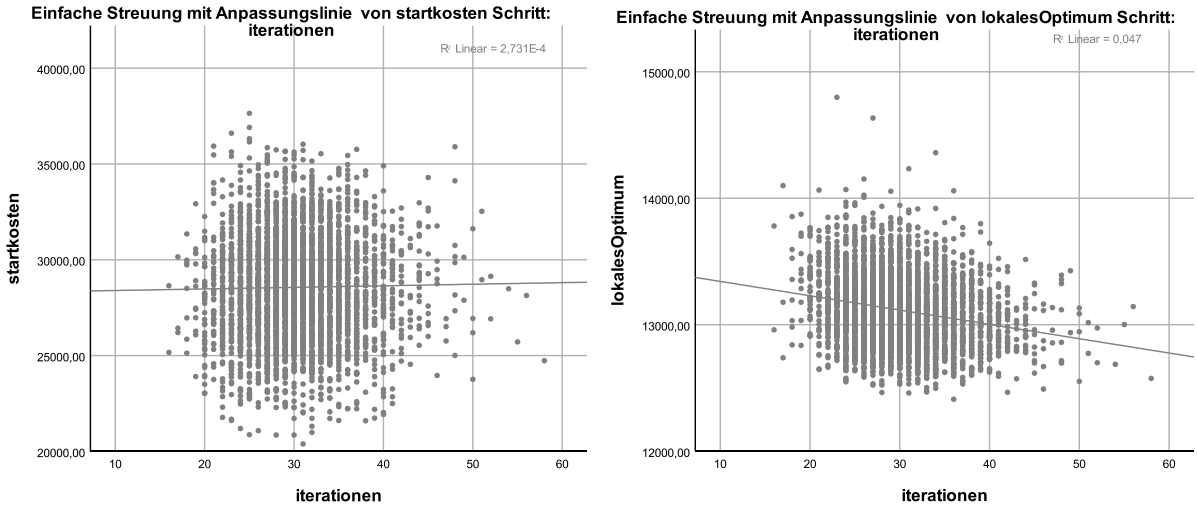


Abbildung 15: Zusammenhang zwischen Iterationen, Kosten der Startlösung und lokales Optimum für HCzs

4.2 Tabu Search

Der Tabu Search Algorithmus wurde in 4 unterschiedlichen Varianten untersucht. Diese Varianten unterscheiden sich durch die Einführung eines Aspirationskriteriums und durch das Beschränken auf skewed Slicing Trees. Hinsichtlich des Aspirationskriteriums wurden 2 Ausprägungen untersucht. Eine, in der das Aspirationskriterium beachtet wurde (TSaa und TSas) und eine andere, in der es nicht beachtet wurde (TSna und TSns). Die zu durchsuchende Nachbarschaft wurde einerseits auf skewed Slicing Trees (TSas und TSns) beschränkt und andererseits wurde die gesamte Nachbarschaft (TSaa und TSna) nach Lösungen durchsucht. Anschließend wurden für vielversprechende Lösungen noch weitere Lösungen mit unterschiedlicher Länge der Tabuliste untersucht.

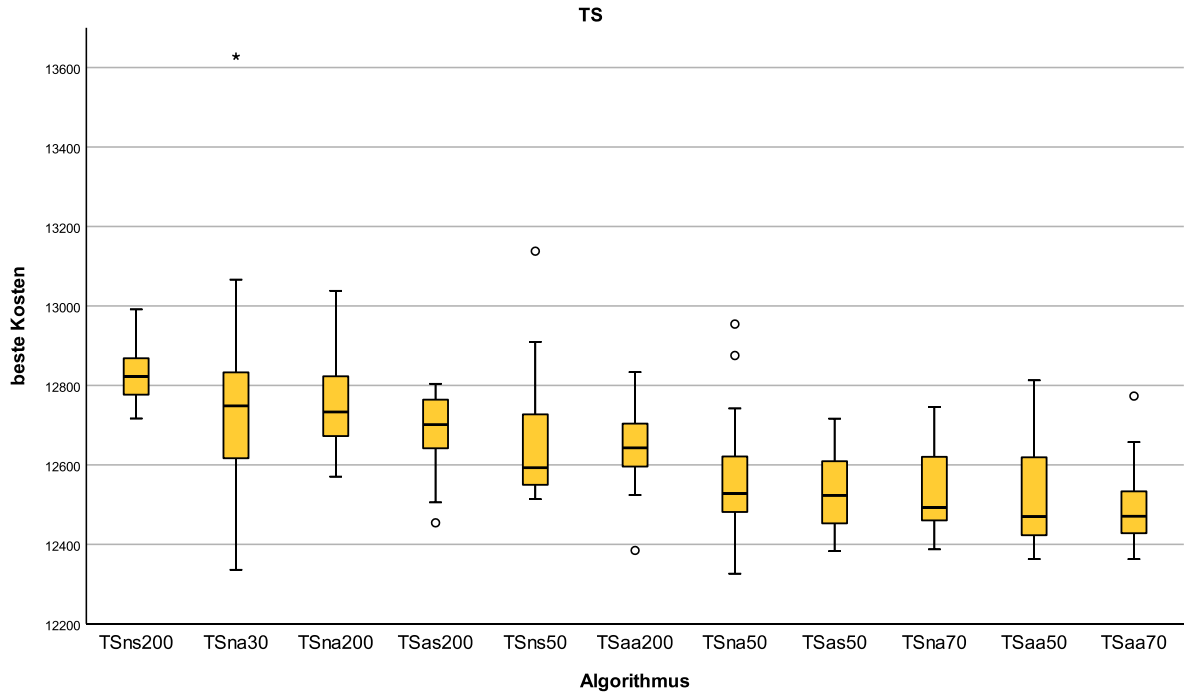


Abbildung 16: Boxplot der Tabu Search Ergebnisse

In Abbildung 17 ist der zeitliche Verlauf des Tabu Search Algorithmus ersichtlich. Anfangs werden sehr schnell Verbesserungen gefunden. Während die Startlösung Kosten von 28947GE verursacht, ist die beste gefundene Lösung bereits nach 78 Iterationen und 15 Sekunden kleiner als 13000GE. Im weiteren Verlauf werden lokale Optima wieder verlassen und auch schlechtere Lösungen akzeptiert. Nach etwas mehr als 3 Minuten und 1085 Iterationen ist eine Lösung gefunden, die im weiteren Verlauf nicht mehr verbessert werden kann. Die Iterationen

benötigen eine Rechenzeit von ca. 0,19 Sekunden, abhängig davon wie groß die Nachbarschaft der letzten Lösung ist. Im zeitlichen Verlauf ist auffällig, dass teilweise längere Phasen mit gleichbleibenden Kosten auftreten. Dies kommt dadurch zustande, dass Lagerflächen, die bereits optimal positioniert sind, mit anderen optimal positionierten Lagerflächen getauscht werden und so zwar eine neue Lösung generiert wird, diese allerdings zu keiner Veränderung der Kosten führt. Am Beispiel in Abbildung 18 ist ersichtlich, dass ein Vertauschen der Lagerfläche für Produkt 12 und 13 zwar ein neues Layout ergibt, dieses aber zu keiner Veränderung der Kosten führt. Auch die Dummy Lagerflächen können beliebig im Layout vertauscht werden, ohne eine Veränderung der Kosten. Bei den Dummy Lagerflächen am linken Rand führt auch ein Schnittrichtungswechsel zu denselben Kosten. Während dieser Phase wird die Tabuliste mit diesen Zügen befüllt und das lokale Optimum wird erst verlassen, wenn sämtliche Züge, die zu denselben Kosten führen, verboten sind.

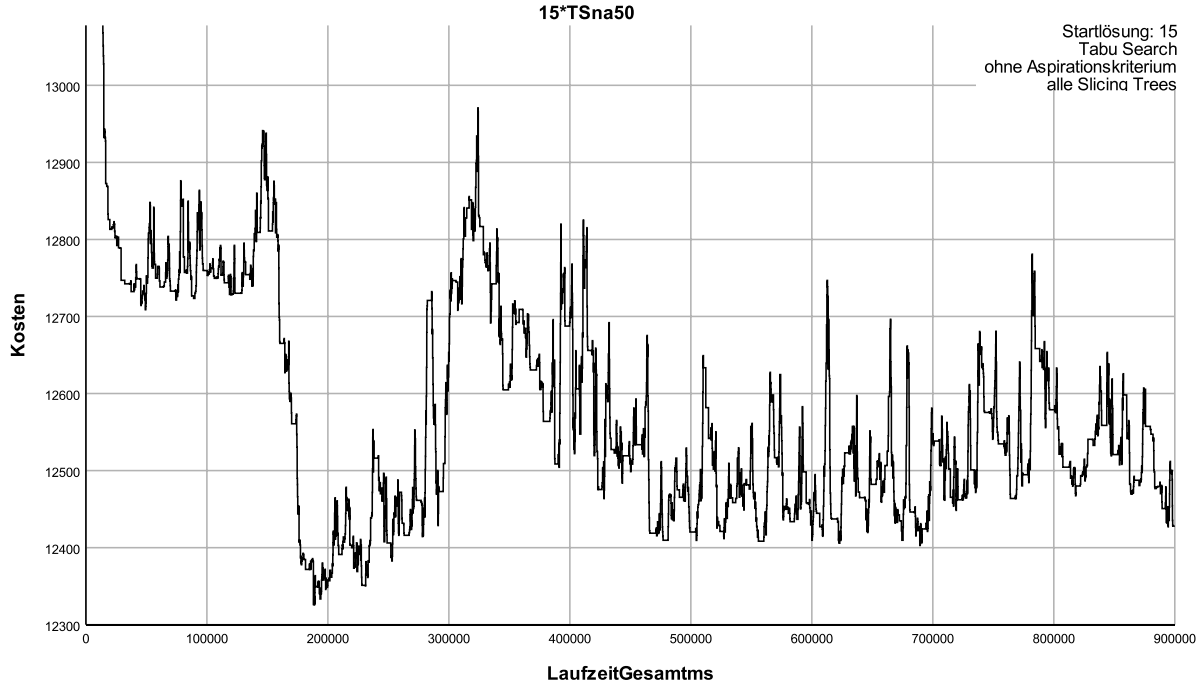


Abbildung 17: Verlauf der Kosten bei Tabu Search Startlösung 15

In Abbildung 18 das beste gefundene Layout der Startlösung 15 mit dem Tabu Search Algorithmus zu sehen. Zum Auffinden dieser Lösung wurde der Tabu Search Algorithmus, mit allen Slicing Trees der Nachbarschaft und ohne Berücksichtigung des Aspirationskriteriums, ausgeführt. Die grün markierten Lagerflächen befinden sich an einer Position, welche die geringst möglichen Transportkosten verursacht. Die Kosten entsprechen der unteren Schranke wie in Kapitel 3 beschrieben. Die Positionierung der grauen Lagerflächen entspricht nicht der optimalen Position. Die Transportwege zwischen Quelle, Lagerfläche und Senke sind hier zur besseren Übersicht als euklidische Distanz dargestellt. Berechnet wird die Distanz immer als rechtwinkelige Distanz (Manhatten-Distanz).

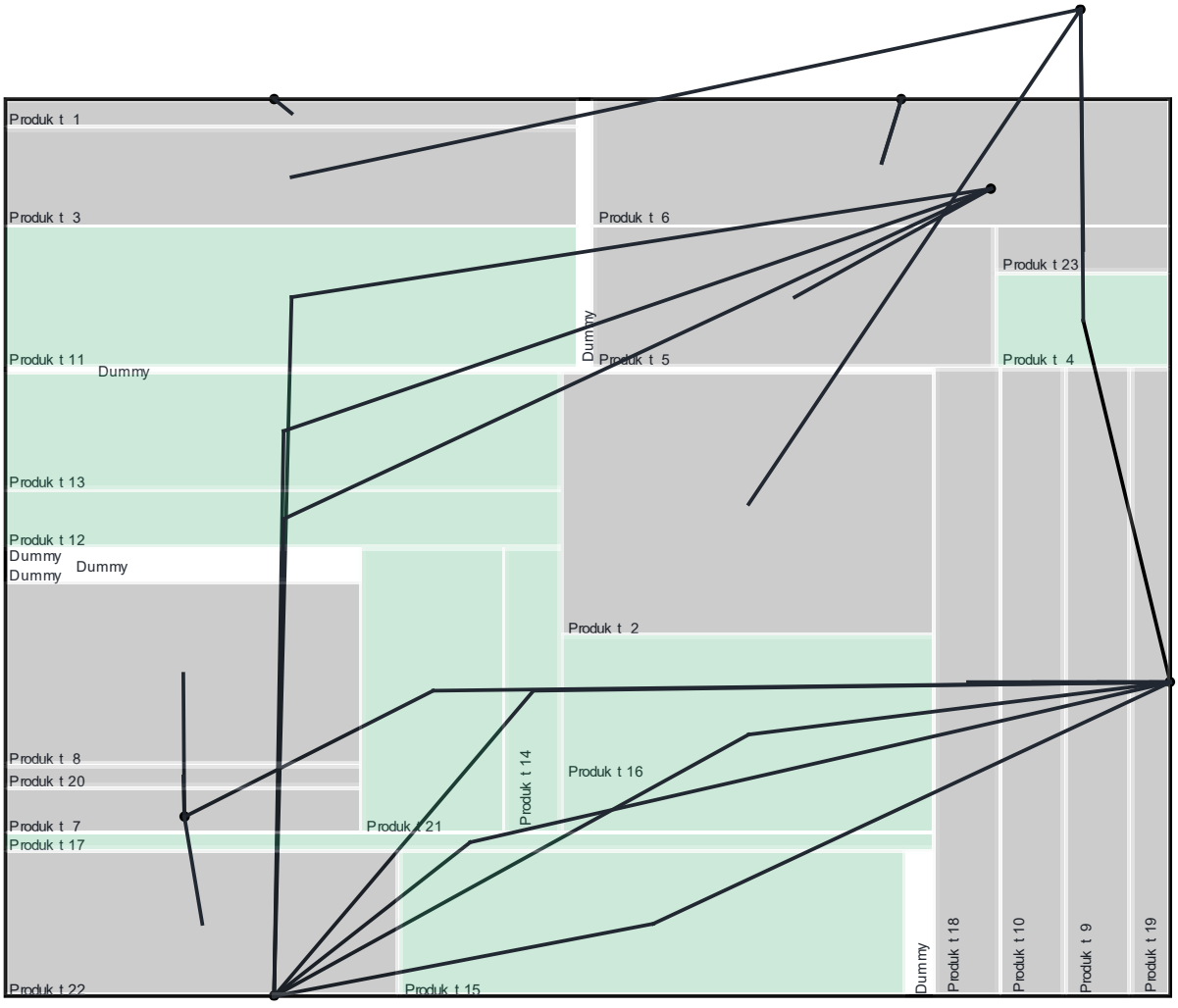


Abbildung 18: Bestes gefundenes Layout zur Startlösung 15 mit Tabu Search

4.2.1 Aspirationskriterium

In Abbildung 19 ist beispielhaft der zeitliche Verlauf der Kosten mit (TSaa) und ohne (TSna) Aspirationskriterium zu sehen. Auffällig ist das Fehlen der starken Verbesserung im Bereich von 180 Sekunden für die Variante mit Aspirationskriterium. In diesem Beispiel schwanken die Werte für die gefundenen Lösungen mit Aspirationskriterium schwächer als ohne.

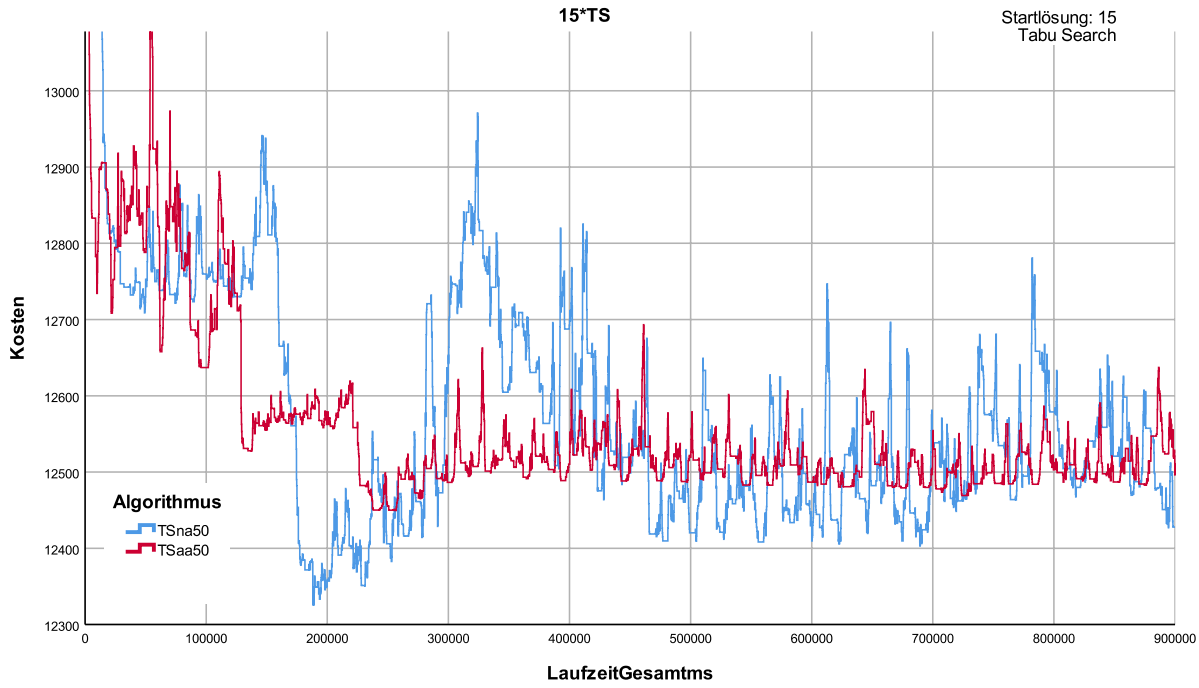


Abbildung 19: Zeitlicher Verlauf mit und ohne Aspirationskriterium für Startlösung 15

In Abbildung 20 sind die Ergebnisse gruppiert nach Aspirationskriterium ersichtlich. Es besteht ein signifikanter Unterschied zwischen den Gruppen (siehe Kapitel 8.5). Tabu Search mit Aspirationskriterium führt zu besseren Ergebnissen.

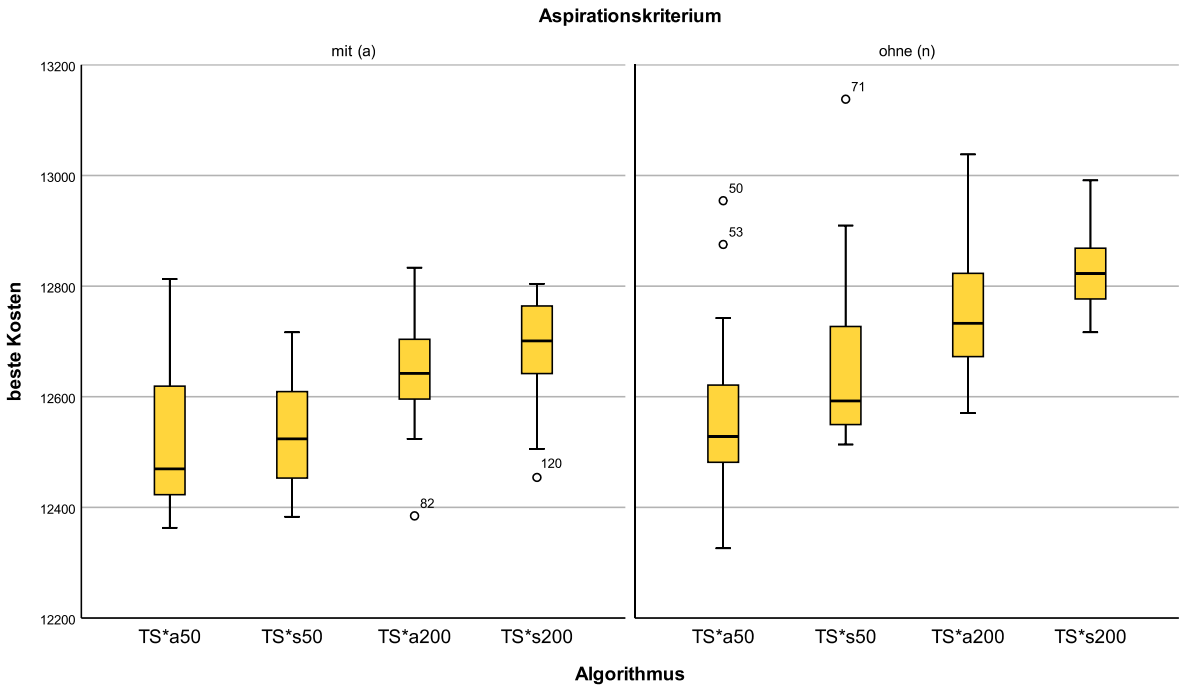


Abbildung 20: Ergebnisse Tabu Search, gruppiert nach Aspirationskriterium

4.2.2 Skewed Slicing Trees

In Abbildung 21 ist der zeitliche Verlauf am Beispiel der Startlösung 15 ersichtlich. Wenn die gesamte Nachbarschaft (TSaa) der zuletzt gefundenen Lösung betrachtet wird, werden geringere Kosten erreicht. Weiters schwanken die Werte der gefundenen Lösungen weniger stark als bei der Einschränkung auf skewed Slicing Trees (TSas).

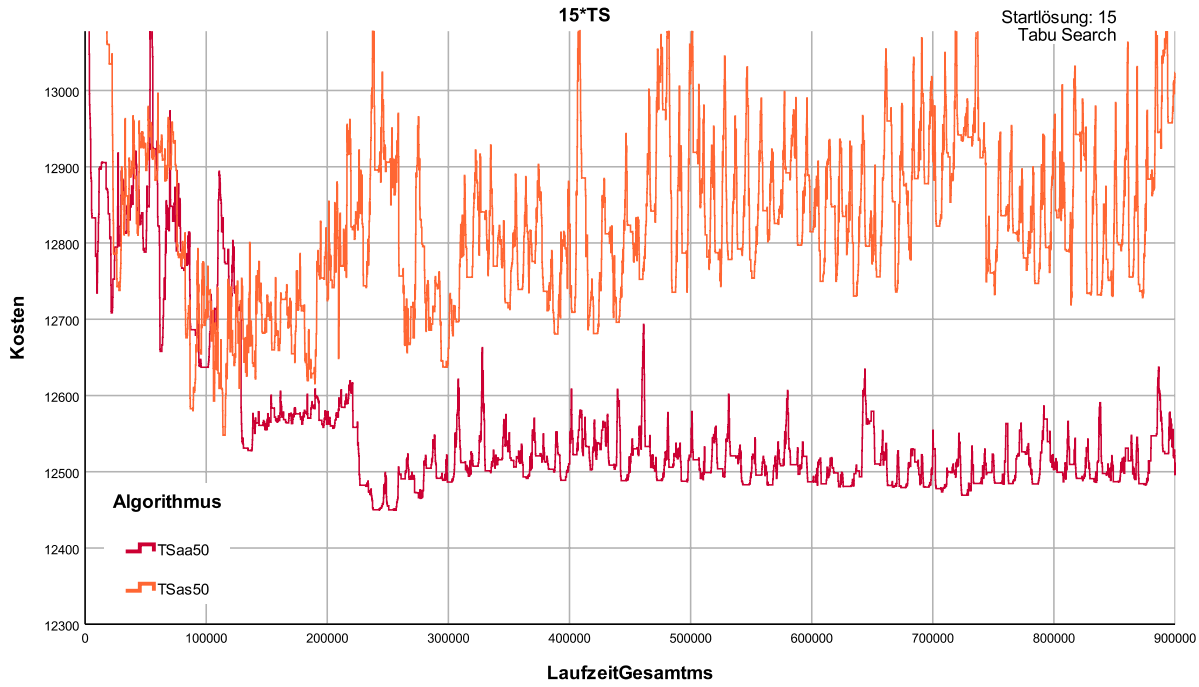


Abbildung 21: Zeitlicher Verlauf für skewed und alle Slicing Trees für Startlösung 15

In Abbildung 22 sind die Ergebnisse, gruppiert nach skewed und alle Slicing Trees, ersichtlich. Es besteht ein signifikanter Unterschied zwischen den Gruppen (siehe Kapitel 0). Tabu Search mit allen verfügbaren Slicing Trees führt zu besseren Ergebnissen.

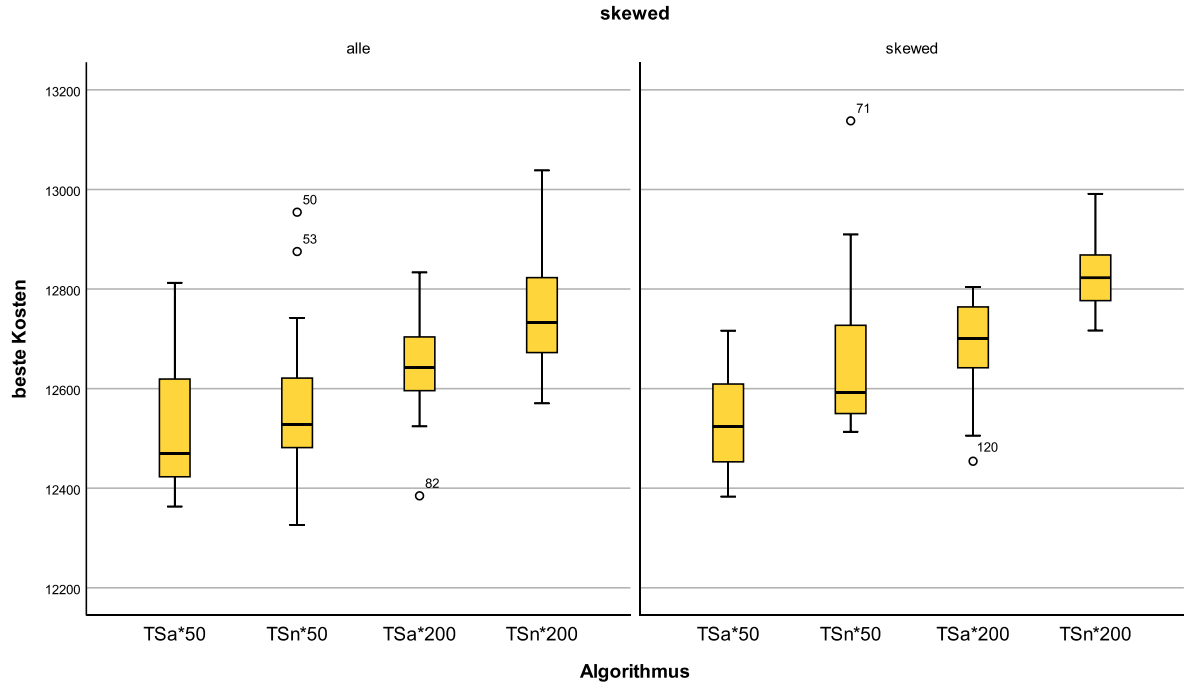


Abbildung 22: Ergebnisse Tabu Search, gruppiert nach skewed ST und alle ST

4.2.3 Tabulänge

Es wurden unterschiedliche Tabulängen untersucht: 30, 50, 70 und 200. Hier wurden nur die Varianten untersucht, welche alle Slicing Trees berücksichtigen (TSna und TSaa), weil diese Varianten bisher zu signifikant besseren Ergebnissen geführt haben. Es hat zwar auch die Variante mit Aspirationskriterium zu signifikant besseren Ergebnissen geführt als die Variante ohne Aspirationskriterium, allerdings wurde die beste Einzellösung ohne Aspirationskriterium gefunden. Aus diesem Grund wurde auch diese Variante mit unterschiedlicher Tabulänge untersucht.

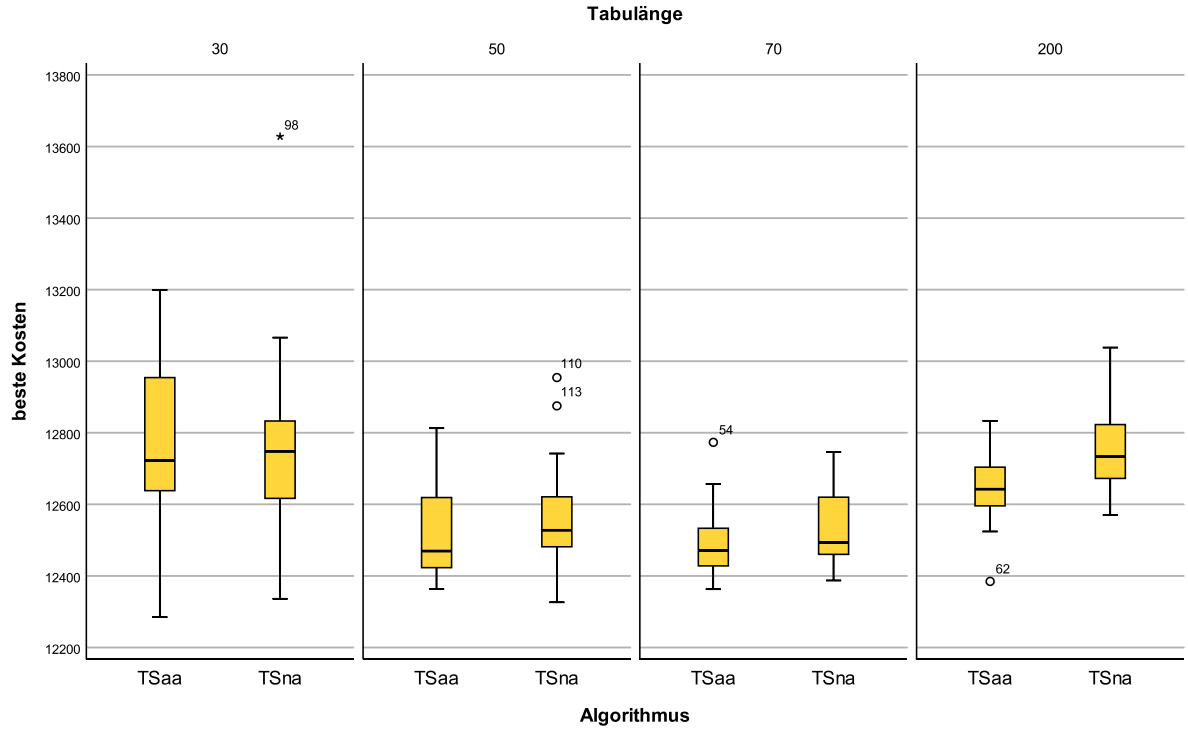


Abbildung 23: Ergebnisse Tabu Search, gruppiert nach Tabulänge

In Abbildung 23 sind die Ergebnisse nach Tabulänge gruppiert ersichtlich. Es ist zu sehen, dass eine zu kleine Tabulänge (30) zu großen Schwankungen in den Ergebnissen führt, weil der Algorithmus ein lokales Optimum nicht mehr verlassen kann. Es wird anfangs ein lokales Optimum gefunden, welches nicht mehr verlassen werden kann, deshalb ist die Qualität der besten Lösung sehr stark von der Startlösung abhängig. Wird die Tabulänge zu groß (200) gewählt, werden lokale Optima übersprungen und so mögliches Optimierungspotential nicht

genutzt. Beispielsweise ist dies in Abbildung 24 für die Startlösung 15 und Tabu Search mit allen Slicing Trees sowie ohne Aspirationskriterium zu sehen.

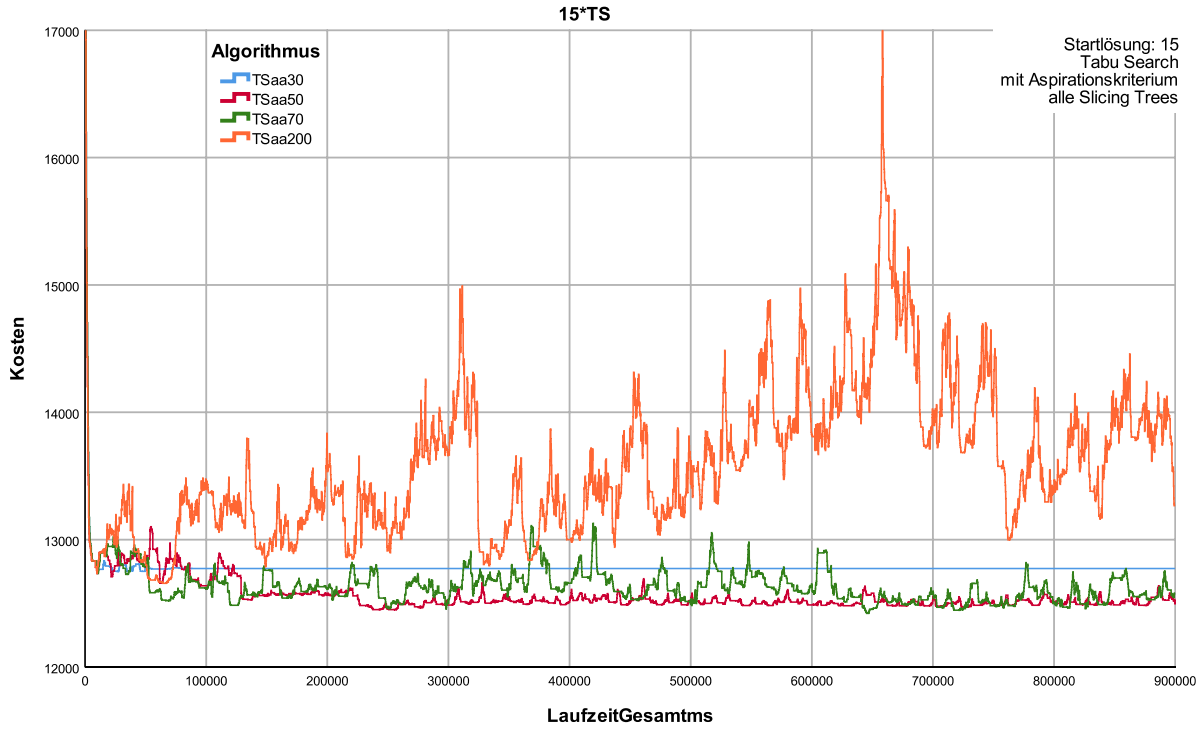


Abbildung 24: Zeitlicher Verlauf für Startlösung 15 mit unterschiedlicher Tabulänge

Die Ergebnisse für die unterschiedlichen Tabulängen weisen einen signifikanten Unterschied auf (siehe Kapitel 8.7). Mit einer Tabulänge von 70 wurden die besten Ergebnisse erzielt.

4.3 Simulated Annealing

Der Simulated Annealing Algorithmus wurde mit 5 unterschiedlichen Temperaturkurven untersucht. Bei guten Ergebnissen wurden die Parameter, für die einzelnen Temperaturkurven, noch weiter optimiert. Allen Varianten ist gemein, dass sie mit einer Starttemperatur von 2000 starten und sich einer Endtemperatur von 0 annähern. Weil der Simulated Annealing Algorithmus eine Zufallskomponente beinhaltet, sind die verwendeten Startlösungen für das Endresultat nicht relevant, es wurden dennoch dieselben Startlösungen wie bei den anderen Algorithmen herangezogen.

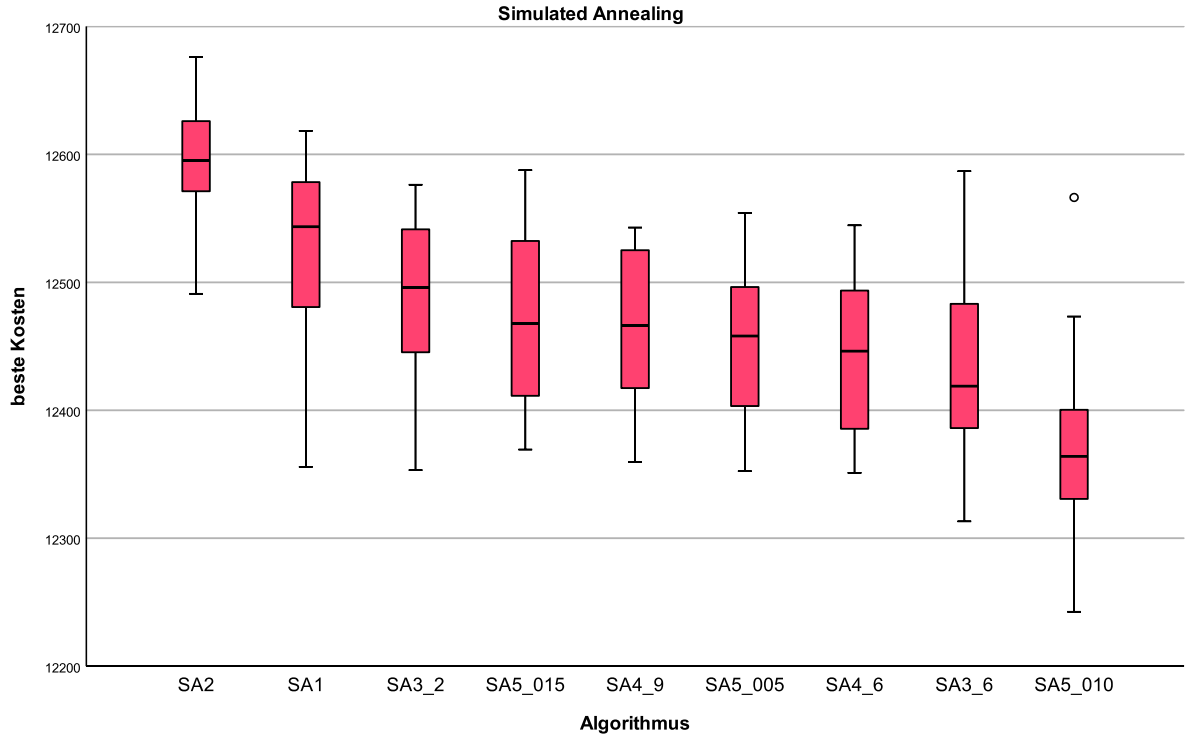


Abbildung 25: Boxplot der Simulated Annealing Ergebnisse

In Abbildung 26 ist der zeitliche Verlauf des Simulated Annealing Algorithmus ersichtlich. Die Startlösung verursacht Kosten in der Höhe von 30367GE, das Optimum wird bei 12242GE gefunden. Das dazugehörige Layout ist in Abbildung 27 zu sehen, unter allen Algorithmen und Variationen ist dieses das beste Layout, das gefunden werden konnte. Im ersten Drittel schwanken die gefundenen Ergebnisse noch sehr stark, weisen aber bereits einen deutlichen Trend nach unten auf. Mit sinkender Temperatur im Zeitverlauf werden schlechtere Lösungen seltener akzeptiert und die Schwankung nimmt ab. Im mittleren Drittel wird die Lösung nur

noch geringfügig verbessert. Im letzten Drittel finden kaum Verschlechterungen der Lösung statt. Im letzten Bereich kommt es auch zu längeren Phasen, in welchen keine neue Lösung gefunden wird, weil die Bedingungen für die Akzeptanz einer neuen Lösung bereits so eng gefasst sind, dass kaum gültige Lösungen produziert werden. Innerhalb der Laufzeit von 15 Minuten werden ca. 500000 Iterationen durchgeführt. Dies ergibt eine mittlere Rechenzeit von 0,0018 Sekunden pro Iteration.

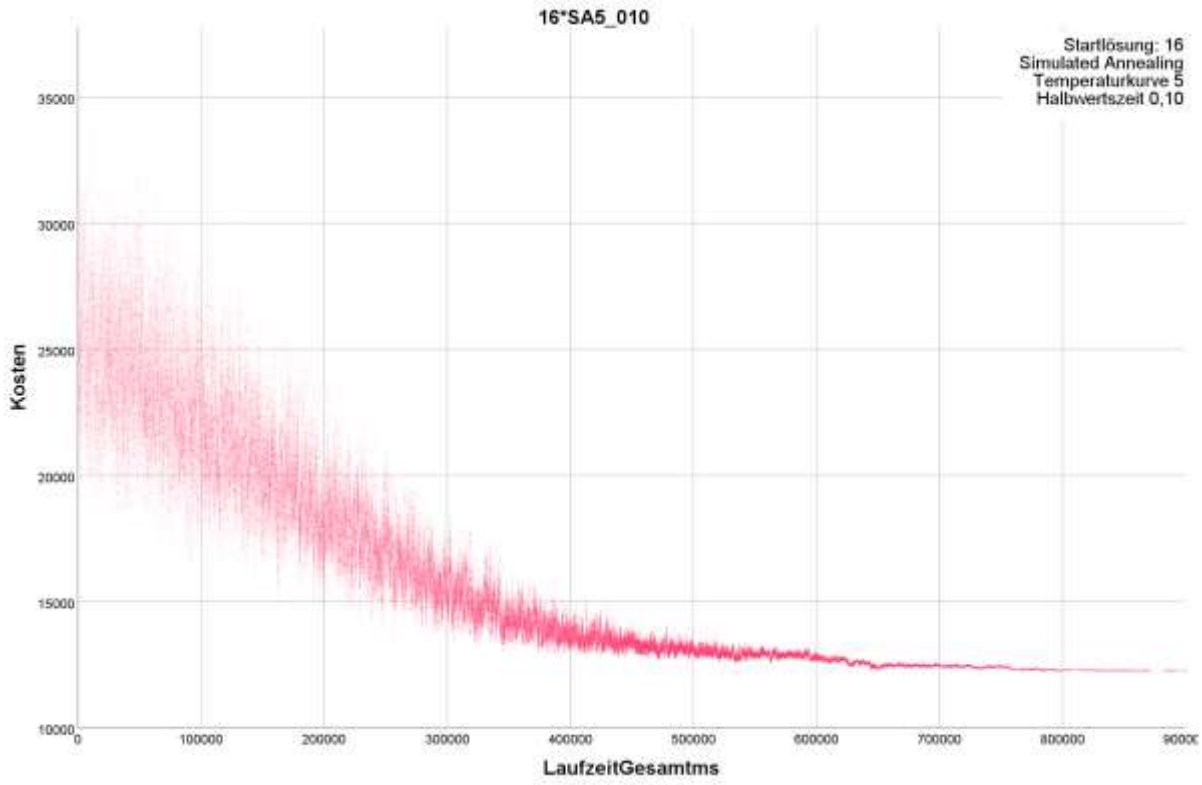


Abbildung 26: Zeitlicher Verlauf von Simulated Annealing für die Startlösung 16

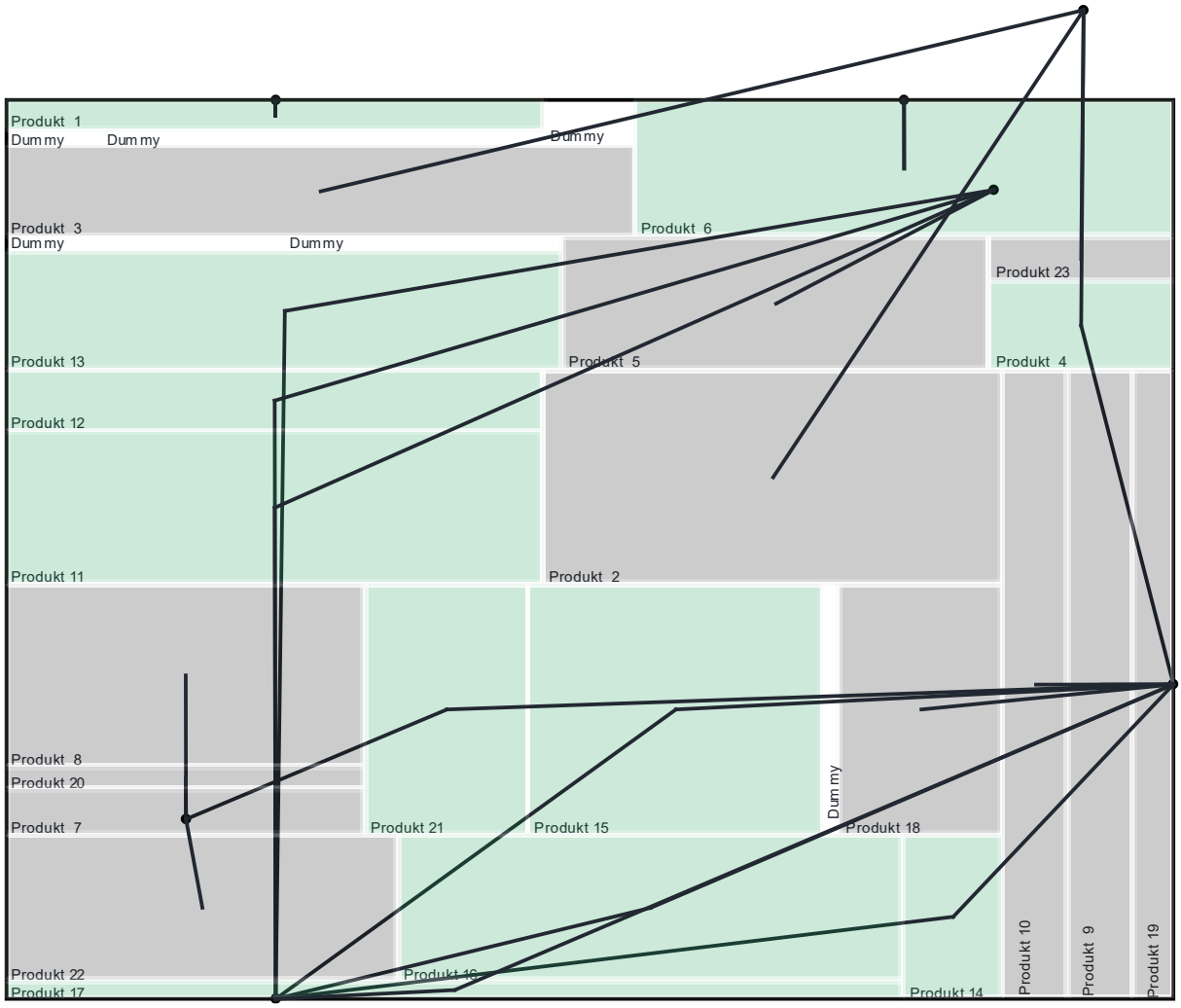


Abbildung 27: Layout zu Simulated Annealing mit Temperaturkurve 5, Halbwertszeit 0,10 und Startlösung 16

Die grün markierten Lagerflächen in Abbildung 27 sind jene Lagerflächen, die optimal platziert wurden, also mit den geringst möglichen Transportkosten unter Berücksichtigung aller Restriktionen. Sämtliche Lagerflächen, deren Quelle und Senke an unterschiedlichen Positionen sind, wurden optimal platziert, zusätzlich auch die Lagerflächen für Produkt 1 und 6. Die Lagerfläche für Produkt 3 könnte in einer manuellen Nachbearbeitung noch näher zur zu beliefernden Senke verschoben werden und an die Position der Dummy Flächen gesetzt werden.

4.3.1 Lineare Temperaturkurve

Mit einer linearen Temperaturkurve finden die Verbesserungen nur sehr langsam und im Laufe der Zeit statt. Den größten Teil der Laufzeit ist keine nennenswerte Einschränkung für schlechtere Lösungen gegeben. Erst im letzten Drittel beginnen die Lösungen besser zu werden. Hier wird zu viel Rechenzeit für schlechte Lösungen aufgewandt. Trotzdem sind die Ergebnisse mit den besten Ergebnissen des Tabu Search Algorithmus zu vergleichen. Die Ergebnisse der unterschiedlichen Startlösungen streuen bedeutend weniger als bei Tabu Search und führen somit zuverlässiger zu guten Resultaten.

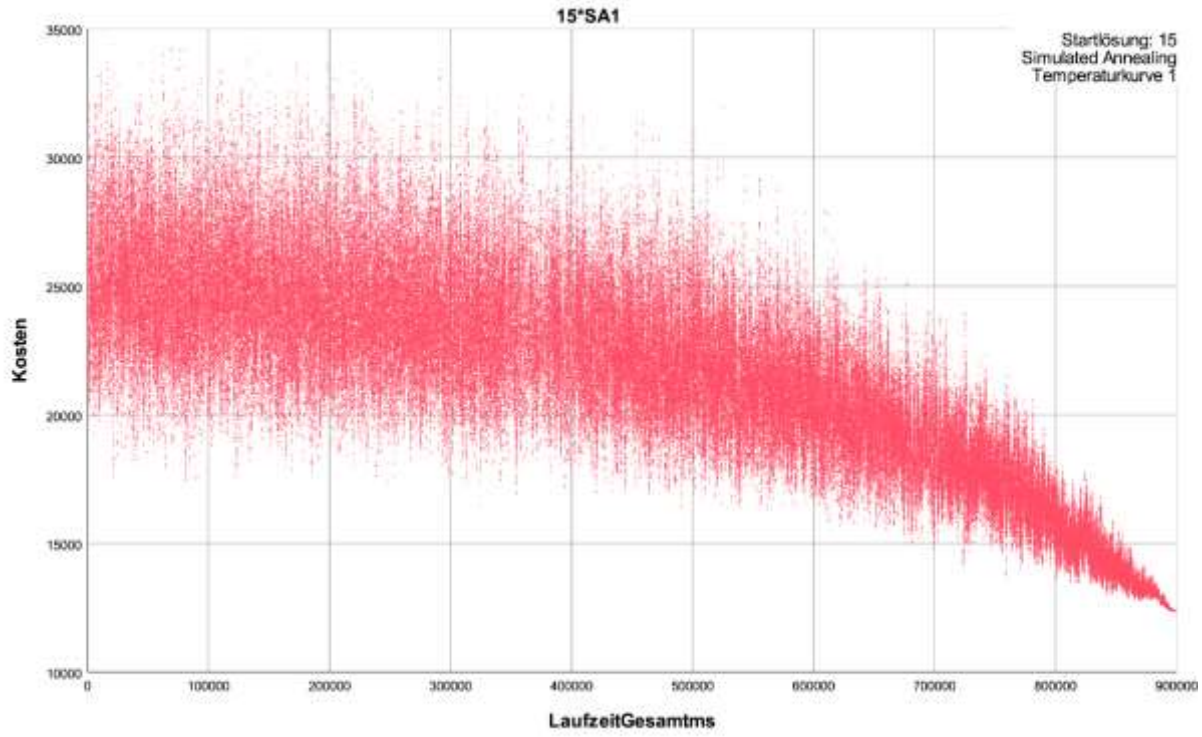


Abbildung 28: Zeitlicher Verlauf von Simulated Annealing für die Startlösung 15 mit einer linearen Temperaturkurve

4.3.2 Degressive Temperaturkurve

Die Ergebnisse der degressiven Temperaturkurve sind vergleichbar mit der Variante mit linearer Temperaturkurve, allerdings werden die Verbesserungen noch später erreicht. Erkennbar ist erst im letzten Fünftel eine Einschränkung und somit Verbesserung der Ergebnisse. Die Ergebnisse bewegen sich verglichen mit Tabu Search im Mittelfeld. Auch hier sind die Ergebnisse für unterschiedliche Startlösungen konstanter, weil die wesentlichen Verbesserungen erst am Ende der Laufzeit erzielt werden.

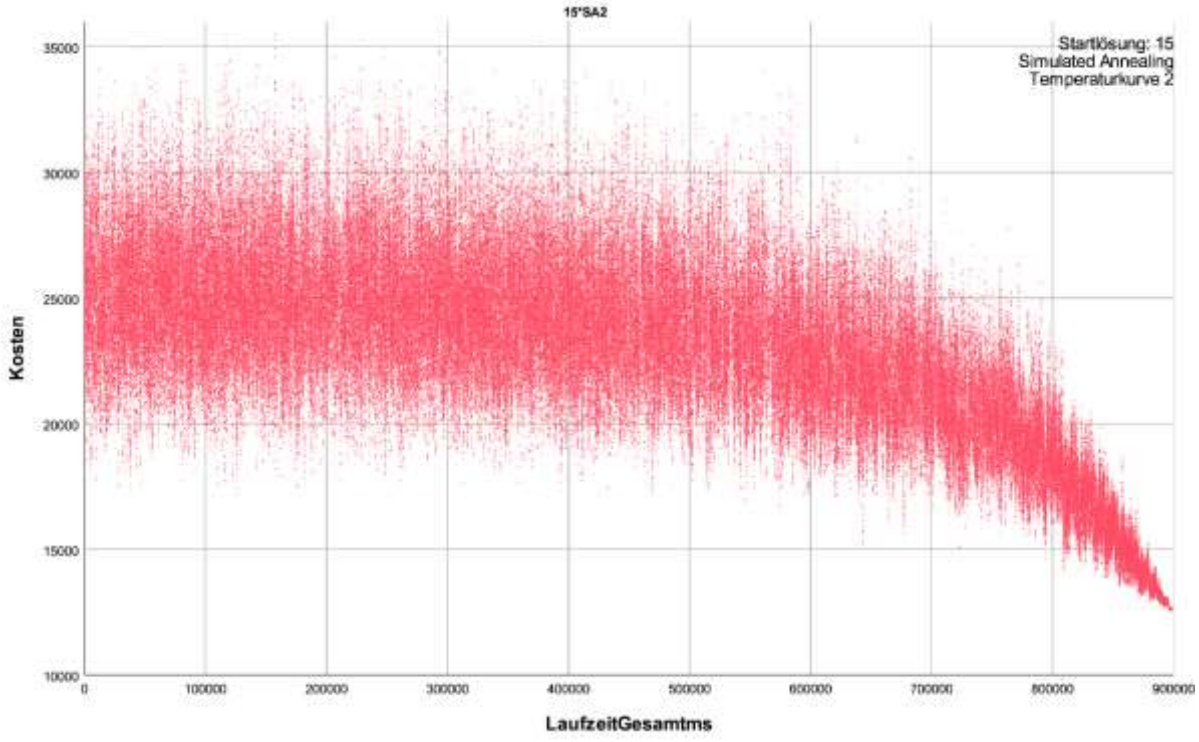


Abbildung 29: Zeitlicher Verlauf von Simulated Annealing für die Startlösung 15 mit einer degressiven Temperaturkurve

4.3.3 Progressive Temperaturkurve

In diesem Beispiel ist erkennbar, dass direkt nach dem Start bereits erste Einschränkungen für schlechtere Lösungen beginnen und somit sehr schnell eine Verbesserung der Ergebnisse eintritt. Dieser Weg setzt sich bis zur Hälfte der Laufzeit fort. Ab der halben Laufzeit sind nur noch geringfügige Veränderungen der Kosten ersichtlich. Die beste Lösung (12364 GE) in diesem Beispiel wird bereits nach ca. 500000 Millisekunden (5/9 der gesamten Laufzeit) gefunden, danach findet keine Verbesserung mehr statt. Zu diesem Zeitpunkt beträgt die Temperatur nur noch 13,45 und lässt keine großen Sprünge in der Lösung zu. Somit kann auch ein lokales Optimum nicht mehr verlassen werden. Ab 800000 Millisekunden wird keine weitere neue Lösung gefunden, weil die Temperatur zu diesem Zeitpunkt bereits kleiner als 0,01 ist.

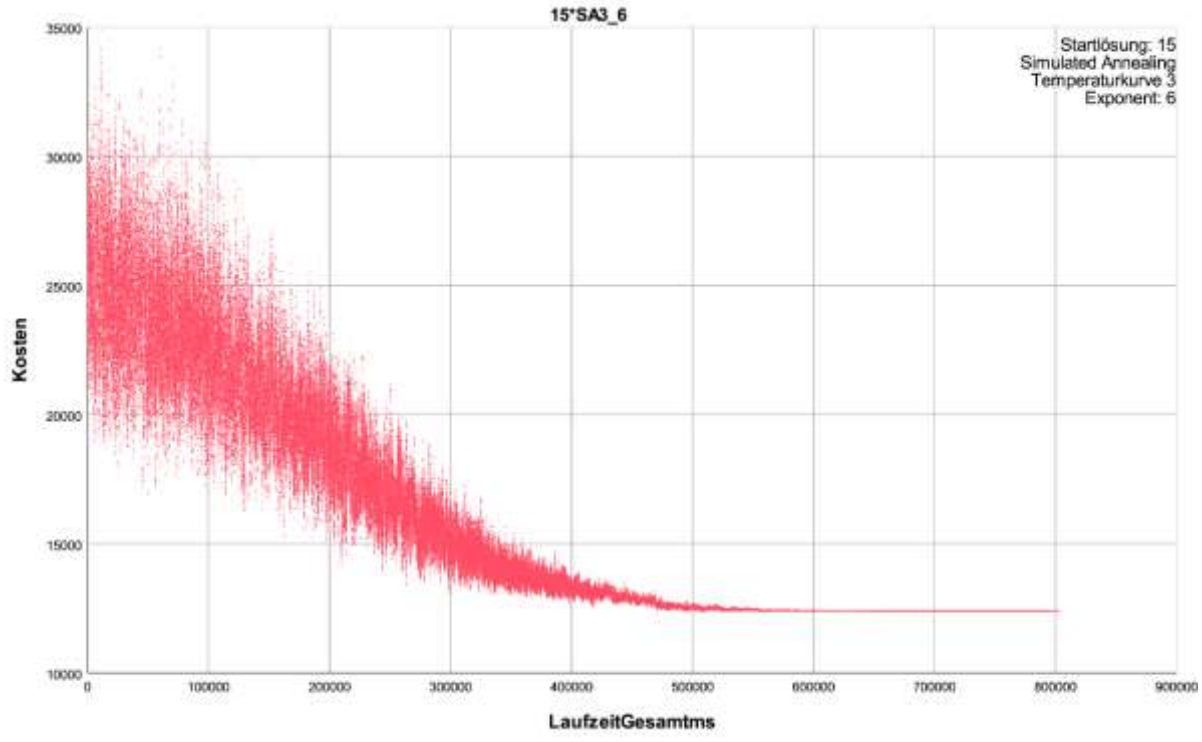


Abbildung 30: Zeitlicher Verlauf von Simulated Annealing für die Startlösung 15 mit einer progressiven Temperaturkurve

4.3.4 S-förmige Temperaturkurve

Die Überlegung hinter einer S-förmigen Temperaturkurve ist, dem Algorithmus anfangs ausreichend Zeit zu geben, um gute Lösungen zu finden und den Lösungsraum möglichst vollständig zu durchsuchen. Anschließend soll durch einen steilen Abfall der Temperatur der Lösungsraum eingegrenzt werden. In der verbleibenden Zeit sollen noch kleine Verbesserungen gefunden werden, welche die Kosten weiter nach unten treiben. Diese Herangehensweise führt zu besseren Ergebnissen als alle bisher untersuchten Temperaturkurven.

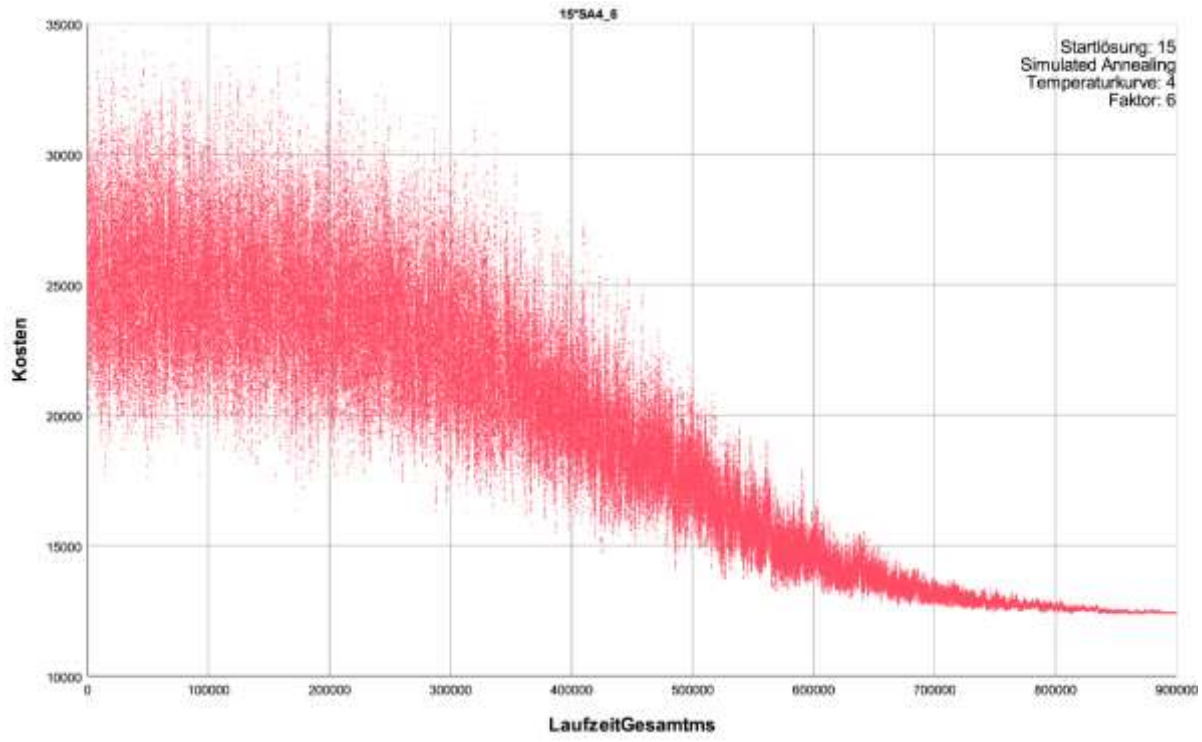


Abbildung 31: Zeitlicher Verlauf von Simulated Annealing für die Startlösung 15 mit einer S-förmigen Temperaturkurve

4.3.5 Exponentieller Zerfall als Temperaturkurve

Diese Temperaturkurve ist mit der progressiven Temperaturkurve in Kapitel 0 vergleichbar, mit einem entscheidenden Unterschied: die Annäherung an die Endtemperatur von 0 geschieht gegen Ende langsamer und lässt so ausreichend Zeit für weitere Verbesserungen. Die Temperatur sinkt bis zur letzten Iteration nur auf 2 ab. Die letzte Verbesserung der Lösung passiert nach ca. 880000 Millisekunden (98% der Gesamtlaufzeit), bei einer Temperatur von 2,3. Diese Temperaturkurve hat zu den besten Ergebnissen geführt.

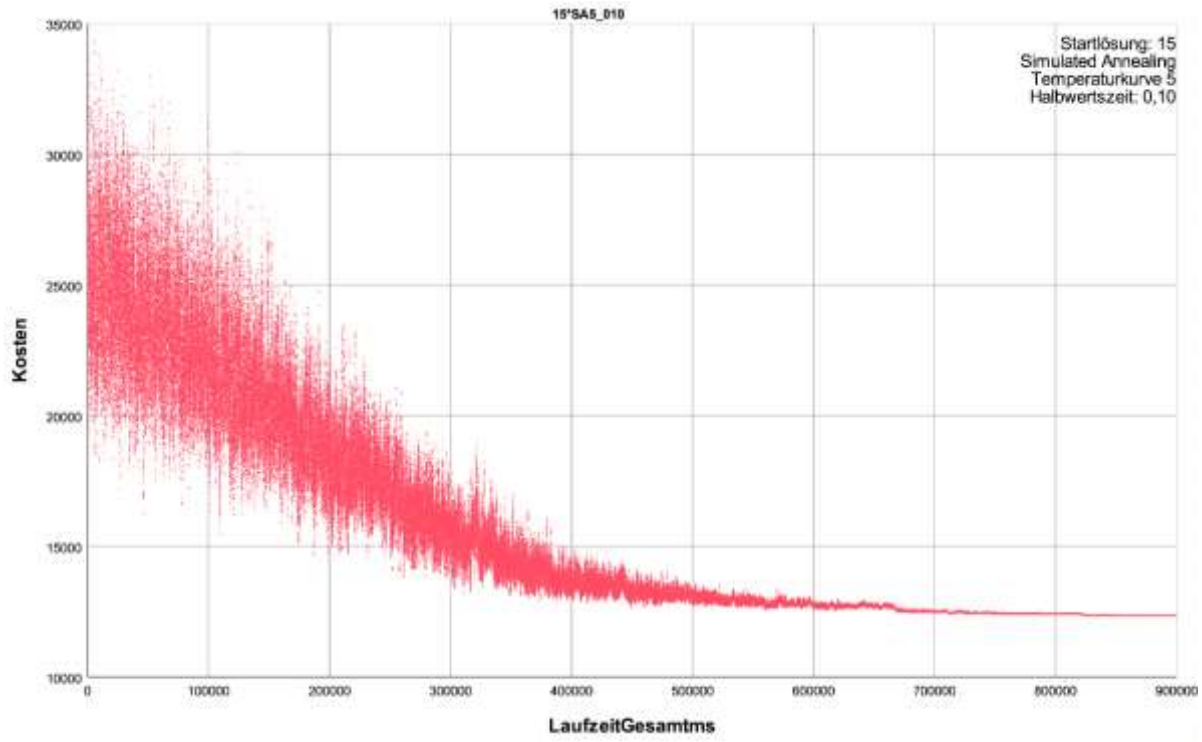


Abbildung 32: Zeitlicher Verlauf von Simulated Annealing für die Startlösung 15 mit einer an den exponentiellen Zerfall angelehnten Temperaturkurve

5 Diskussion

Slicing Trees und dazu korrespondierende Slicing Layouts sind eine Möglichkeit, um Layouts in der Lagerhaltung zu optimieren. Insbesondere wenn die einzelnen Lagerplätze unterschiedliche Restriktionen aufweisen, ist es mit dieser Methode möglich, diese zu berücksichtigen und trotzdem Verbesserungen zu realisieren.

In Abbildung 33 sind die Ergebnisse aller untersuchten Algorithmen und Varianten zu sehen. Bezogen auf den Lösungsraum von 10555,64 GE (0%) bis 40271,46 GE (100%), bewegen sich die gefundenen Lösungen im Bereich von 5% bis 11%. Wenn man die jeweils besten Ergebnisse der einzelnen Algorithmen vergleicht (TSaa70, HCza und SA5_010), so lässt sich sagen, dass Simulated Annealing die besten Ergebnisse hervorbringt.

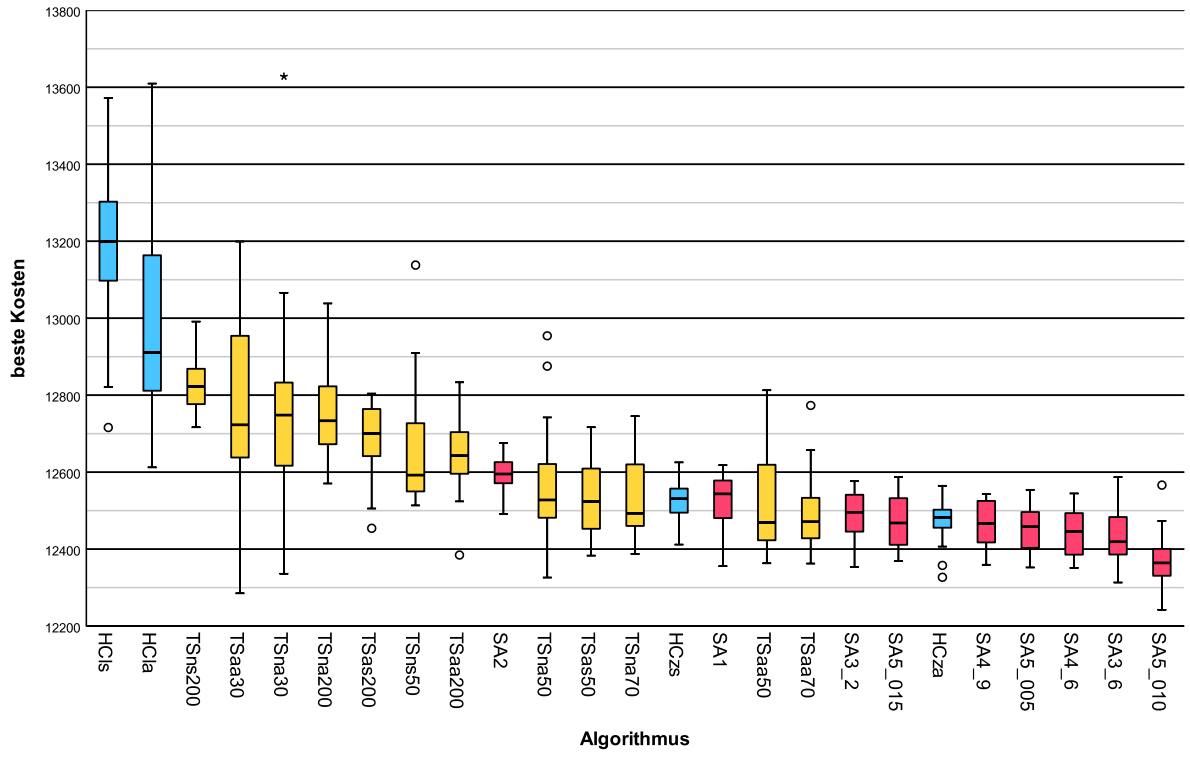


Abbildung 33: Boxplot aller untersuchten Algorithmen

Tabu Search ist für diese Problemstellung nicht gut geeignet, weil zu viele gleiche Lösungen in der Nachbarschaft vorhanden sind und deshalb ein lokales Optimum nicht zuverlässig wieder verlassen werden kann. Es wären weitere Anpassungen am Algorithmus notwendig, um noch bessere Ergebnisse zu erzielen. Dies würde in einer problemspezifischen Heuristik enden, welche im engeren Sinn keine Metaheuristik wäre.

Hill Climbing liefert, wenn nur ein Durchlauf betrachtet wird, die schlechtesten Ergebnisse. Wenn hingegen mehrere Durchläufe mit einem Neustart mit einer zufälligen Startlösung durchgeführt werden, verbessern sich die Resultate außerordentlich. Betrachtet man die gleichen Voraussetzungen hinsichtlich der Rechenzeit, ist Hill Climbing mit Tabu Search vergleichbar.

Simulated Annealing erfordert mehr Aufwand bei der Auswahl der Parameter, liefert dann aber zuverlässig vergleichsweise gute Resultate. Aufgrund der Zufallskomponente im Algorithmus ist er unabhängig von vielen gleichen Lösungen in der Nachbarschaft und findet so ständig Verbesserungen. Eine längere Phase mit hoher Temperatur am Anfang hat sich nicht als positiv herausgestellt, die wesentlichen Verbesserungen werden im letzten Drittel der Laufzeit gefunden.

Werden die besten Ergebnisse der Heuristiken betrachtet, so lässt sich sagen, dass zwischen Hill Climbing und Tabu Search kein signifikanter Unterschied besteht und Simulated Annealing zu signifikant besseren Ergebnissen als alle anderen untersuchten Algorithmen führt. Das beste in dieser Arbeit gefundene Layout ist in Abbildung 34 zu sehen.

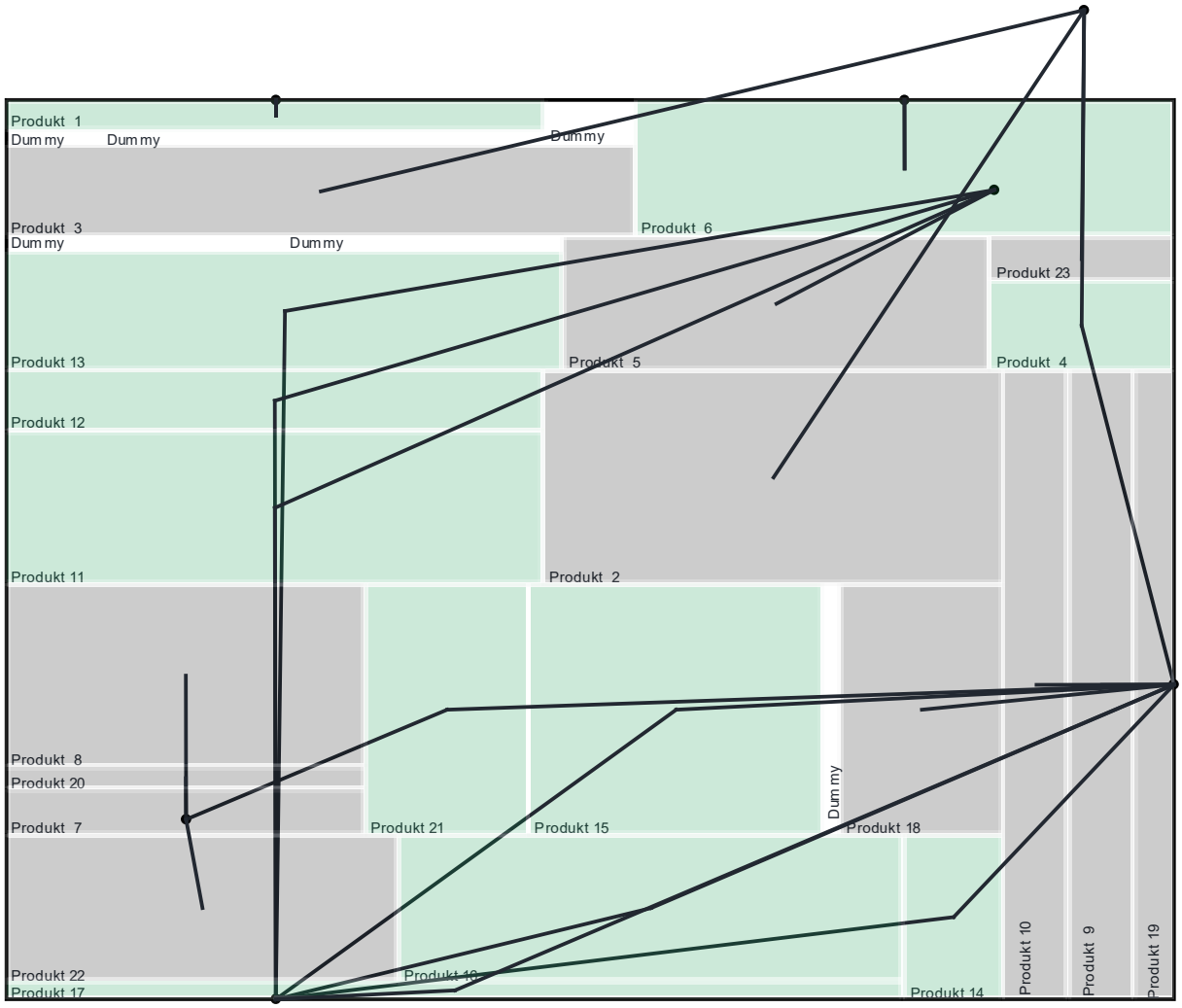


*Abbildung 34: Bestes Layout, das in dieser Arbeit gefunden wurde. Simulated Annealing mit einer an den exponentiellen Zerfall angelehnten Temperaturkurve und einer Halbwertszeit von 0,1 (16*SA5_010).*

6 Zusammenfassung und Ausblick

Das Slicing Tree Verfahren konnte erfolgreich auf ein lineares Zuordnungsproblem eines Rundholzlagerplatzes angewandt werden. Weiters ist im Zuge dieser Arbeit eine Java-Applikation entstanden, mit welcher das Slicing Tree Verfahren auch bei zukünftigen Projekten angewandt werden kann. Bei der Optimierung ergeben sich wesentlich Unterschiede im Vergleich zu einem quadratischen Zuordnungsproblem. Der Tabu Search Algorithmus erzielt für quadratische Zuordnungsprobleme sehr gute Ergebnisse. In dieser Arbeit wurden mit dem Tabu Search Algorithmus keine besseren Ergebnisse als mit einem simplen Hill Climbing Algorithmus erzielt. Simulated Annealing führt bei der vorliegenden Problemstellung zu den besten Ergebnissen. Die beste gefundene Lösung verursacht Transportkosten die um 16% höher sind als die berechnete untere Schranke der Transportkosten. Die vorgestellte Methode kann auch abseits der Holzindustrie zur Layoutplanung für ungleich große Lagerplätze angewandt werden.

Zur Optimierung von bestehenden Anlagen muss das Slicing Tree Verfahren weiter angepasst werden, um auch bestehende und unveränderliche Infrastruktur in der Optimierung berücksichtigen zu können. Auch die Beschränkung auf rechteckige Lagerplätze stellt in der Praxis eine große Einschränkung da. Weiters muss unbedingt ein Vergleich mit einem realisierten Projekt angestellt werden, um die Leistungsfähigkeit auch in der Praxis verifizieren zu können.

7 Literaturverzeichnis

Teile der Kapitel 1.1 und 1.2 wurden so oder ähnlich bereits in meiner Bachelorarbeit *Facility Layout Problem in der Holzwirtschaft* im Jahr 2017 verwendet.

Bauernhansl, T., Vogel-Heuser, B. & Hompel, M. t., 2014. *Industrie 4.0 in Produktion, Automatisierung und Logistik.* Wiesbaden: Springer Vieweg.

Dittes, F.-M., 2015. *Optimierung Wie man aus allem das Beste macht.* Berlin, Heidelberg: Springer Berlin Heidelberg.

Drira, A., Pierreval, H. & Hajri-Gabouj, S., 2007. Facility Layout Problems: A survey. *Annual Reviews in Control*, pp. 255-267.

Fronius, K., 1989. *Der Rundholzplatz.* Leinfelden-Echterdingen: DRW-Verlag.

Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research.*

Huka, M. A. & Gronalt, M., 2018. Log yard logistics. *Silva Fennica.*

Kang, S. & Chae, J., 2017. Harmony search for the layout design of an unequal area facility. *Expert Systems with Applications.*

Koopmans, T. C. & Beckmann, M., 1957. Assignment Problems and the Location of Economic Activities. *Econometrica.*

Lin, S. & Kernighan, B. W., 1973. EFFECTIVE HEURISTIC ALGORITHM FOR THE TRAVELING-SALESMAN PROBLEM.. *Operations Research.*

Metropolis, N. et al., 1953. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics.*

Otten, R. H., 1982. *Automatic floorplan design.* s.l., Institute of Electrical and Electronics Engineers Inc., pp. 261-267.

Scholz, D., 2010. *Innerbetriebliche Standortplanung.* 1. Auflage Hrsg. Wiesbaden: Gabler Verlag.

Scholz, D., Petrick, A. & Domschke, W., 2009. STaTS: A Slicing Tree and Tabu Search based heuristic for the unequal area facility layout problem. *European Journal of Operational Research.*

Valenzuela, C. L. & Wang, P. Y., 2002. VLSI placement and area optimization using a genetic algorithm to breed normalized postfix expressions. *IEEE Transactions on Evolutionary Computation.*

Wong, D. F. & Liu, C. L., 1989. Floorplan design of VLSI circuits. *Algorithmica,* 6, 4(1-4), pp. 263-291.

8 Anhang

8.1 Test auf Normalverteilung der Startlösungen

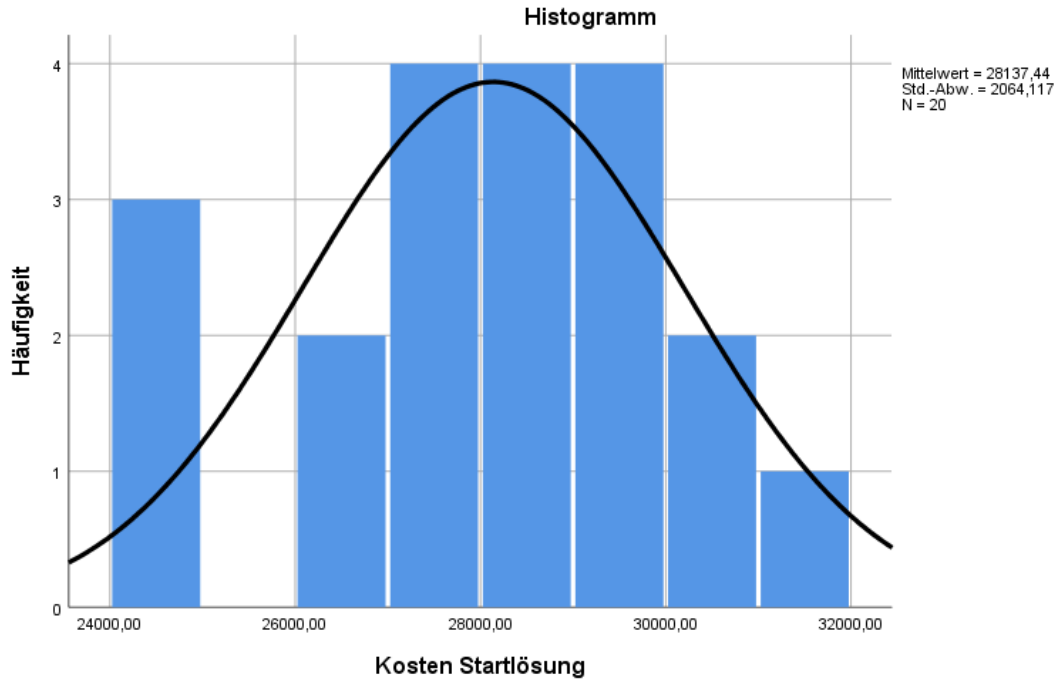
Deskriptive Statistik

			Statistik	Standard Fehler
Kosten Startlösung	Mittelwert		28137,4380	461,55065
	95% Konfidenzintervall des Mittelwerts	Untergrenze	27171,4014	
		Obergrenze	29103,4746	
	5% getrimmtes Mittel		28184,9939	
	Median		28691,2300	
	Varianz		4260580,057	
	Standard Abweichung		2064,11726	
	Minimum		24076,61	
	Maximum		31342,26	
	Spannweite		7265,65	
	Interquartilbereich		2761,29	
	Schiefe		-,709	,512
	Kurtosis		-,104	,992

Tests auf Normalverteilung

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Statistik	df	Signifikanz	Statistik	df	Signifikanz
Kosten Startlösung	,164	20	,167	,932	20	,169

a. Signifikanzkorrektur nach Lilliefors



8.2 ANOVA Hill Climbing lokale Suche und zufälliger Neustart

ONEWAY deskriptive Statistiken

besteKosten

	N	Mittelwert	Std.-Abweichung	Std.-Fehler	95%-Konfidenzintervall für den Mittelwert Untergrenze	Obergrenze	Minimum	Maximum
lokale Suche	40	13081,3883	245,55396	38,82549	13002,8563	13159,9202	12612,41	13609,55
zufälliger Neustart	40	12499,4833	60,84402	9,62028	12480,0244	12518,9421	12326,89	12625,24
Gesamt	80	12790,4358	342,51896	38,29478	12714,2119	12866,6596	12326,89	13609,55

Einfaktorielle ANOVA

besteKosten

	Quadratsumme	df	Mittel der Quadrate	F	Signifikanz
Zwischen den Gruppen	6772268,581	1	6772268,581	211,638	,000
Innerhalb der Gruppen	2495950,969	78	31999,371		
Gesamt	9268219,549	79			

8.3 Zweifaktorielle ANOVA Hill Climbing

Deskriptive Statistiken

Abhängige Variable: besteKosten

variante	skewed	Mittelwert	Std.-Abweichung	N
lokale Suche	alle	12976,7020	241,79290	20
	skewed	13186,0745	205,50226	20
	Gesamt	13081,3883	245,55396	40
zufälliger Neustart	alle	12472,2655	57,37445	20
	skewed	12526,7010	52,41577	20
	Gesamt	12499,4832	60,84402	40
Gesamt	alle	12724,4838	308,75760	40
	skewed	12856,3877	365,23008	40
	Gesamt	12790,4357	342,51896	80

Tests der Zwischensubjekteffekte

Abhängige Variable: besteKosten

Quelle	Quadratsumme vom Typ III	df	Mittel der Quadrate	F	Sig.
Korrigiertes Modell	7240269,255[a]	3	2413423,085	90,446	,000
Konstanter Term	13087619733,990	1	13087619733,990	490475,088	,000
variante	6772268,580	1	6772268,580	253,799	,000
skewed	347973,304	1	347973,304	13,041	,001
variante * skewed	120027,370	1	120027,370	4,498	,037
Fehler	2027950,294	76	26683,557		
Gesamt	13096887953,539	80			
Korrigierte Gesamtvariation	9268219,549	79			

a. R-Quadrat = ,781 (korrigiertes R-Quadrat = ,773)

8.4 Korrelation Startlösung und lokales Optimum

Korrelationen 19*HCza

		lokalesOptimum	startloesung
lokalesOptimum	Korrelation nach Pearson	1	-,036
	Signifikanz (1-seitig)		,330
	N	149	149
startloesung	Korrelation nach Pearson	-,036	1
	Signifikanz (1-seitig)	,330	
	N	149	149

Korrelationen 19*HCzs

		lokalesOptimum	startloesung
lokalesOptimum	Korrelation nach Pearson	1	-,202**
	Signifikanz (1-seitig)		,002
	N	199	199
startloesung	Korrelation nach Pearson	-,202**	1
	Signifikanz (1-seitig)	,002	
	N	199	199

**. Die Korrelation ist auf dem Niveau von 0,01 (1-seitig) signifikant.

Korrelationen HCzs (alle)

		startkosten	lokalesOptimum	iterationen
startkosten	Korrelation nach Pearson	1	-,140**	,017
	Signifikanz (2-seitig)		,000	,303
	N	3893	3893	3893
lokalesOptimum	Korrelation nach Pearson	-,140**	1	-,218**
	Signifikanz (2-seitig)	,000		,000
	N	3893	3893	3893
iterationen	Korrelation nach Pearson	,017	-,218**	1
	Signifikanz (2-seitig)	,303	,000	
	N	3893	3893	3893

**. Die Korrelation ist auf dem Niveau von 0,01 (2-seitig) signifikant.

8.5 Zweifaktorielle ANOVA Tabu Search Aspirationskriterium

Deskriptive Statistiken

Abhängige Variable: beste Kosten

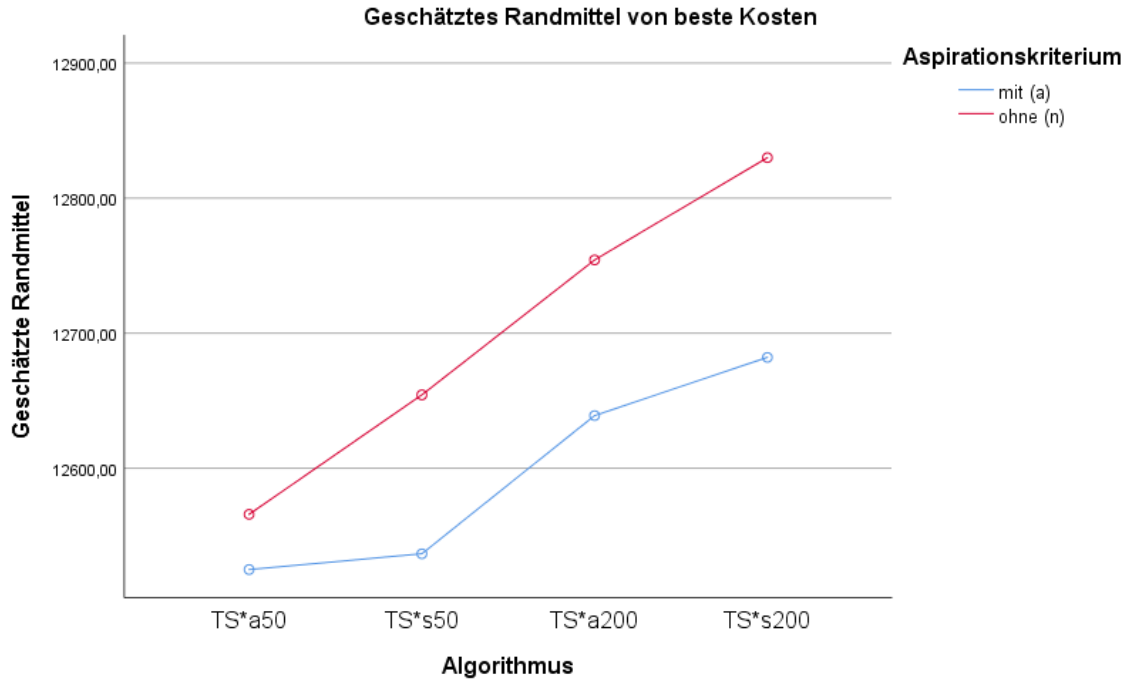
Algorithmus	Aspirationskriterium	Mittelwert	Std.-Abweichung	N
TS*a50	mit (a)	12524,9800	141,98954	20
	ohne (n)	12565,8285	156,43469	20
	Gesamt	12545,4043	148,90282	40
TS*s50	mit (a)	12536,6575	95,14148	20
	ohne (n)	12654,3265	160,41637	20
	Gesamt	12595,4920	143,16755	40
TS*a200	mit (a)	12639,0060	100,71423	20
	ohne (n)	12754,2935	112,63669	20
	Gesamt	12696,6497	120,54250	40
TS*s200	mit (a)	12682,0970	95,68328	20
	ohne (n)	12829,9930	73,73151	20
	Gesamt	12756,0450	112,77083	40
Gesamt	mit (a)	12595,6851	127,20280	80
	ohne (n)	12701,1104	162,91743	80
	Gesamt	12648,3978	154,99377	160

Tests der Zwischensubjekteffekte

Abhängige Variable: beste Kosten

Quelle	Quadratsumme vom Typ III	df	Mittel der Quadrate	F	Sig.
Korrigiertes Modell	1599704,894[a]	7	228529,271	15,647	,000
Konstanter Term	25597114502,753	1	25597114502,753	1752624,383	,000
algorithmus	1092914,614	3	364304,871	24,944	,000
aspiration	**444579,334**	**1**	**444579,334**	**30,440**	**,000**
algorithmus * aspiration	62210,946	3	20736,982	1,420	,239
Fehler	2219963,070	152	14605,020		
Gesamt	25600934170,717	160			
Korrigierte Gesamtvariation	3819667,964	159			

a. R-Quadrat = ,419 (korrigiertes R-Quadrat = ,392)



8.6 Zweifaktorielle ANOVA Tabu Search skewed und alle Slicing Trees

Deskriptive Statistiken

Abhängige Variable: beste Kosten

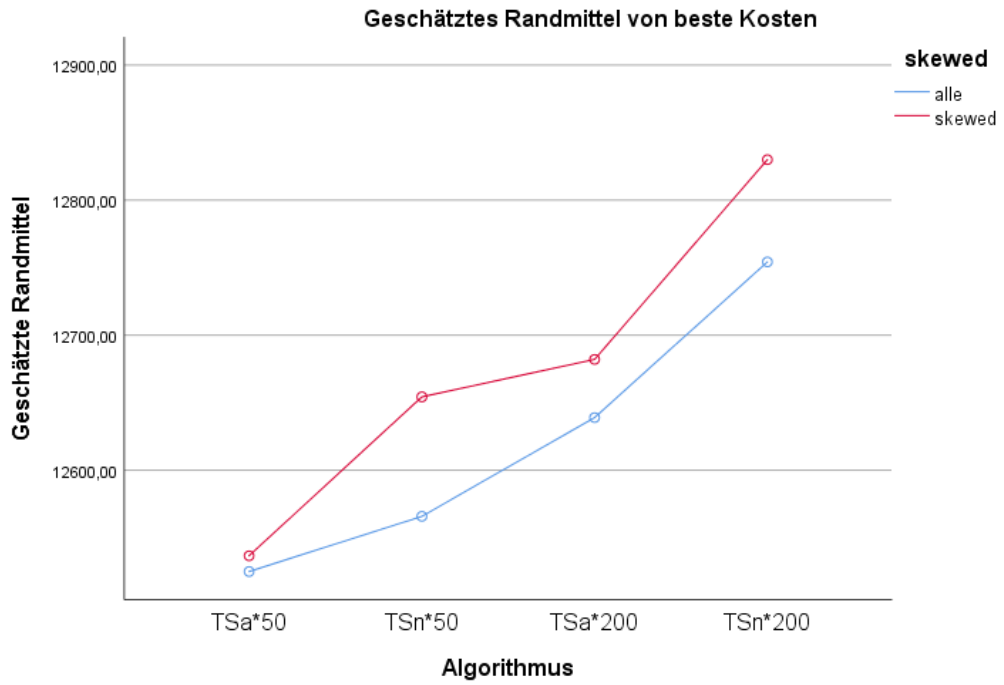
Algorithmus	skewed	Mittelwert	Std.-Abweichung	N
TSa*50	alle	12524,9800	141,98954	20
	skewed	12536,6575	95,14148	20
	Gesamt	12530,8188	119,44415	40
TSn*50	alle	12565,8285	156,43469	20
	skewed	12654,3265	160,41637	20
	Gesamt	12610,0775	162,68723	40
TSa*200	alle	12639,0060	100,71423	20
	skewed	12682,0970	95,68328	20
	Gesamt	12660,5515	99,38821	40
TSn*200	alle	12754,2935	112,63669	20
	skewed	12829,9930	73,73151	20
	Gesamt	12792,1432	101,48234	40
Gesamt	alle	12621,0270	154,62913	80
	skewed	12675,7685	151,40270	80
	Gesamt	12648,3978	154,99377	160

Tests der Zwischensubjekteffekte

Abhängige Variable: beste Kosten

Quelle	Quadratsumme vom Typ III	df	Mittel der Quadrate	F	Sig.
Korrigiertes Modell	1599704,894[a]	7	228529,271	15,647	,000
Konstanter Term	25597114502,753	1	25597114502,753	1752624,383	,000
algorithmus_skew	1444149,808	3	481383,269	32,960	,000
skewed	**119865,273**	**1**	**119865,273**	**8,207**	**,005**
algorithmus_skew * skewed	35689,813	3	11896,604	,815	,488
Fehler	2219963,070	152	14605,020		
Gesamt	25600934170,717	160			
Korrigierte Gesamtvariation	3819667,964	159			

a. R-Quadrat = ,419 (korrigiertes R-Quadrat = ,392)



8.7 Zweifaktorielle ANOVA Tabulänge

Deskriptive Statistiken

Abhängige Variable: beste Kosten

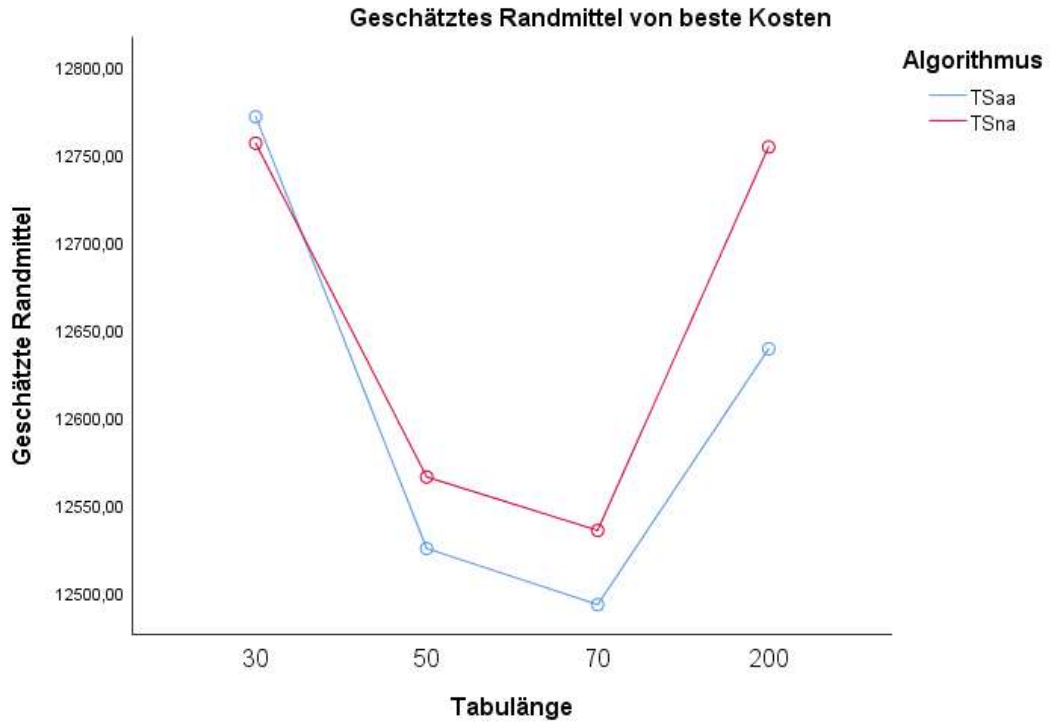
Algorithmus	Tabulänge	Mittelwert	Std.-Abweichung	N
TSaa	30	12771,5178	250,38974	20
	50	12524,9800	141,98954	20
	70	12492,9066	98,97847	20
	200	12639,0060	100,71423	20
	Gesamt	12607,1026	191,91992	80
TSna	30	12756,4100	280,01218	20
	50	12565,8285	156,43469	20
	70	12535,2950	105,73172	20
	200	12754,2935	112,63669	20
	Gesamt	12652,9567	203,02456	80
Gesamt	30	12763,9639	262,29839	40
	50	12545,4043	148,90282	40
	70	12514,1008	103,34291	40
	200	12696,6497	120,54250	40
	Gesamt	12630,0297	198,26655	160

Tests der Zwischensubjekteffekte

Abhängige Variable: beste Kosten

Quelle	Quadratsumme vom Typ III	df	Mittel der Quadrate	F	Sig.
Korrigiertes Modell	1888951,278[a]	7	269850,183	9,405	,000
Konstanter Term	25522823907,087	1	25522823907,087	889525,584	,000
algorithmus	84104,173	1	84104,173	2,931	,089
tabusize	**1719102,978**	**3**	**573034,326**	**19,971**	**,000**
algorithmus * tabusize	85744,126	3	28581,375	,996	,396
Fehler	4361278,982	152	28692,625		
Gesamt	25529074137,347	160			
Korrigierte Gesamtvariation	6250230,259	159			

a. R-Quadrat = ,302 (korrigiertes R-Quadrat = ,270)



8.8 ANOVA der besten Varianten der Algorithmen

ONEWAY deskriptive Statistiken

Kosten

	N	Mittelwert	Std.-Abweichung	Std.-Fehler	95%-Konfidenzintervall für den Mittelwert		Minimum	Maximum
					Untergrenze	Obergrenze		
HCza	20	12472,26550	57,374452	12,829317	12445,41343	12499,11757	12326,890	12564,580
SA5_010	20	12372,67000	70,073915	15,669004	12339,87440	12405,46560	12242,460	12566,360
TSaa70	20	12492,90658	98,978473	22,132259	12446,58323	12539,22994	12362,683	12773,302
Gesamt	60	12445,94736	92,729528	11,971331	12421,99278	12469,90194	12242,460	12773,302

Einfaktorielle ANOVA

Kosten

	Quadratsumme	df	Mittel der Quadrate	F	Signifikanz
Zwischen den Gruppen	165347,693	2	82673,847	13,780	,000
Innerhalb der Gruppen	341979,474	57	5999,640		
Gesamt	507327,167	59			

Mehrfachvergleiche

Abhängige Variable: Kosten

	(I) V2	(J) V2	Mittlere Differenz (I-J)	Std.-Fehler	Signifikanz	95%-Konfidenzintervall	
						Untergrenze	Obergrenze
Tukey-HSD	HCza	SA5_010	99,59550000*	24,49416236	,000	40,6523030	158,5386970
		TSaa70	-20,64108350	24,49416236	,678	-79,5842805	38,3021135
	SA5_010	HCza	-99,59550000*	24,49416236	,000	-158,5386970	-40,6523030
		TSaa70	-120,2365835*	24,49416236	,000	-179,1797805	-61,2933865
	TSaa70	HCza	20,64108350	24,49416236	,678	-38,3021135	79,5842805
		SA5_010	120,23658350*	24,49416236	,000	61,2933865	179,1797805
Bonferroni	HCza	SA5_010	99,59550000*	24,49416236	,000	39,1760682	160,0149318
		TSaa70	-20,64108350	24,49416236	1,000	-81,0605153	39,7783483
	SA5_010	HCza	-99,59550000*	24,49416236	,000	-160,0149318	-39,1760682
		TSaa70	-120,2365835*	24,49416236	,000	-180,6560153	-59,8171517
	TSaa70	HCza	20,64108350	24,49416236	1,000	-39,7783483	81,0605153
		SA5_010	120,23658350*	24,49416236	,000	59,8171517	180,6560153

*. Die Differenz der Mittelwerte ist auf dem Niveau 0.05 signifikant.

8.9 Ergebnisse des Hill Climbing Algorithmus

Nr	HCIa	HCIs	HCza	HCzs
1	12835,33	12820,94	12449,86	12468,17
2	12612,41	13170,44	12564,58	12531,73
3	13159,39	13354,66	12477,48	12466,31
4	12979,12	13181,83	12494,96	12548,58
5	12930,61	13572,06	12517,05	12584,03
6	13167,21	13218,13	12558,17	12460,54
7	12891,86	13087,24	12437,19	12491,97
8	12872,55	12938,33	12467,68	12503,80
9	12709,52	12716,05	12501,88	12625,24
10	12720,81	13297,49	12473,19	12550,88
11	12892,12	13063,38	12482,01	12580,31
12	12790,25	13308,14	12461,27	12531,21
13	13609,55	13510,77	12495,91	12533,54
14	13220,09	13136,58	12357,68	12411,80
15	12833,00	13244,84	12503,05	12523,64
16	13018,92	13209,85	12406,73	12563,60
17	13105,68	13107,48	12484,21	12497,81
18	13185,48	13217,64	12482,76	12607,68
19	13270,26	13378,08	12326,89	12522,83
20	12729,88	13187,56	12502,76	12530,35

8.10 Ergebnisse des Tabu Search Algorithmus

Nr	TSaa30	TSaa50	TSaa70	TSaa200	TSas50	TSas200
1	12576,83	12383,90	12448,67	12542,87	12449,49	12669,13
2	12384,73	12384,73	12384,73	12384,73	12511,51	12678,41
3	12887,30	12437,82	12455,20	12733,05	12513,80	12710,31
4	12285,08	12363,10	12435,00	12524,09	12647,69	12768,52
5	12611,63	12416,41	12490,98	12619,10	12456,44	12804,25
6	12722,99	12447,59	12485,81	12667,01	12514,99	12768,86
7	12722,85	12498,61	12530,62	12608,53	12592,12	12795,18
8	12604,71	12469,55	12536,04	12680,50	12564,44	12632,66
9	12696,17	12609,51	12596,63	12634,66	12540,55	12650,98
10	12664,73	12598,32	12657,34	12720,81	12532,35	12728,12
11	12674,35	12628,89	12362,68	12598,76	12504,59	12580,46
12	12779,30	12520,68	12405,59	12531,12	12628,15	12720,72
13	13020,92	12798,94	12502,74	12601,47	12716,47	12777,70
14	13199,29	12759,02	12773,30	12650,62	12383,11	12505,66
15	12734,31	12449,55	12421,14	12657,14	12547,89	12655,67
16	12711,80	12640,67	12401,62	12783,09	12626,45	12691,28
17	13075,20	12469,67	12512,73	12593,00	12428,24	12570,21
18	13183,09	12812,73	12563,48	12686,89	12714,26	12759,65
19	13165,76	12429,71	12456,42	12833,37	12449,05	12719,89
20	12729,31	12380,20	12437,41	12729,31	12411,56	12454,28

Nr	TSna30	TSna50	TSna70	TSna200	TSns50	TSns200
1	12404,75	12590,33	12459,74	12639,36	12513,59	12850,67
2	12727,13	12480,82	12560,42	12719,54	12591,65	12991,30
3	12630,45	12509,95	12387,39	12757,78	12582,84	12854,09
4	12820,97	12483,72	12535,81	12660,80	12560,45	12785,82
5	12684,90	12529,83	12480,31	13038,22	12593,31	12810,48
6	12798,37	12389,18	12430,08	12655,46	12721,24	12758,01
7	12464,91	12541,72	12642,27	12877,21	12594,58	12863,76
8	12742,47	12482,16	12492,43	12684,21	12664,49	12834,47
9	12961,21	12455,32	12494,08	12728,84	12752,28	12768,05
10	13066,15	12954,43	12563,66	12821,81	12733,04	12735,83
11	12959,23	12728,91	12728,07	12720,14	13137,90	12862,88
12	12840,88	12604,34	12428,95	12916,83	12909,61	12785,32
13	12754,17	12875,38	12745,59	12799,12	12868,31	12873,24
14	12336,08	12528,23	12460,87	12737,38	12545,62	12739,86
15	12603,28	12326,12	12491,21	12762,67	12583,83	12877,15
16	12758,86	12742,21	12658,49	12824,22	12600,10	12933,92
17	12446,49	12527,27	12409,94	12570,44	12531,95	12716,60
18	13627,91	12637,85	12661,40	12686,23	12554,02	12950,68
19	12675,17	12426,76	12477,06	12862,67	12522,67	12803,97
20	12824,82	12502,04	12598,13	12622,94	12525,05	12803,76

8.11 Ergebnisse des Simulated Annealing Algorithmus

Nr	SA1	SA2	SA3_2	SA3_6
1	12568,50	12675,73	12514,27	12491,46
2	12438,51	12645,49	12486,85	12461,70
3	12607,54	12491,01	12498,53	12393,33
4	12553,99	12572,02	12544,16	12408,09
5	12546,19	12570,39	12538,08	12423,98
6	12522,48	12619,77	12442,44	12408,64
7	12585,78	12626,66	12493,65	12361,56
8	12485,07	12596,79	12558,72	12586,84
9	12540,59	12597,86	12542,00	12413,88
10	12476,33	12574,43	12517,11	12313,17
11	12580,07	12593,36	12466,63	12520,95
12	12532,64	12579,81	12479,41	12493,74
13	12471,50	12517,98	12374,52	12360,60
14	12496,13	12512,82	12576,28	12380,32
15	12355,70	12575,04	12570,50	12364,38
16	12583,53	12642,35	12353,21	12391,90
17	12433,65	12534,67	12540,73	12474,93
18	12618,35	12601,08	12418,93	12461,40
19	12576,58	12676,02	12448,17	12463,69
20	12558,37	12625,34	12392,46	12546,00

Nr	SA4_6	SA4_9	SA5_005	SA5_010	SA5_015
1	12496,14	12366,34	12528,27	12317,65	12412,92
2	12419,68	12419,71	12509,49	12369,17	12501,75
3	12455,29	12459,82	12428,58	12364,93	12467,31
4	12476,23	12486,12	12483,12	12372,09	12505,68
5	12442,77	12389,68	12475,64	12322,63	12401,39
6	12447,07	12470,55	12457,35	12415,00	12543,74
7	12368,96	12541,98	12444,16	12380,64	12422,63
8	12355,77	12445,20	12378,06	12356,10	12438,38
9	12387,31	12511,09	12554,20	12437,00	12454,34
10	12351,10	12533,00	12355,69	12439,54	12497,22
11	12366,03	12367,35	12432,82	12319,86	12399,88
12	12508,70	12461,69	12457,03	12359,59	12409,70
13	12485,92	12414,90	12466,98	12340,03	12391,58
14	12491,02	12540,72	12355,10	12385,76	12556,35
15	12383,74	12539,85	12459,63	12338,74	12560,67
16	12544,56	12517,25	12524,35	12242,46	12555,97
17	12445,50	12511,99	12458,75	12566,36	12520,92
18	12529,86	12454,33	12544,43	12363,19	12587,69
19	12502,44	12542,78	12360,26	12289,42	12468,64
20	12445,35	12359,41	12352,36	12473,24	12369,28