Fachhochschule der Wirtschaft
-FHDW-
Mettmann

Bachelorthesis

Thema:

Factoringportfoliokonzentration – Berechnung und grafische Darstellung für die fidis GmbH

Prüfer:
Markus Borschbach
Marvin Chibuzo Offiah

Verfasser:
Steven Datzmann
Suitbertusstr. 20
40223 Düsseldorf

Wirtschaftsinformatik
Software Engineering

Eingereicht am:
14.07.2014

Executive Summary

Die vorliegende Arbeit behandelt die algorithmische Zusammenstellung von Portfolios von Posten (Rechnungen) unter Limitierungen, welche die Zusammensetzung beschränken. Hierbei wird darauf eingegangen, zu welcher Problemklasse diese Aufgabe gehört, welche Probleme sich bei der Lösung derartiger Optimierungsprobleme ergeben und inwiefern diese Umgangen werden können.

Die Komplexität des vorliegenden Problems wird analysiert und weitestgehend reduziert, sodass es auf ein Rucksackproblem mit Nebenbedingungen zurückgeführt wird. Auf Basis dieser Reduktion werden bekannte Lösungen für das Rucksackproblem vorgestellt und für die Problemlösung verwendet.

Das Ergebnis dieser Übertragung sind zwei konzeptionierte Algorithmen welche für das vorliegende Optimierungsproblem Näherungslösungen liefern. Bei einer anschließenden Umsetzung wird gezeigt wie diese mittels JavaScript umgesetzt werden können und warum ein dynamischer Algorithmus nicht praxistauglich ist.

Zudem wird gezeigt wie das Ergebnis-Portfolio grafisch, mittels der Einbindung von Nutzerinteraktion, so dargestellt werden kann, dass zum einen eine intuitive und übersichtliche Gesamtdarstellung einen Überblick gibt und zum anderen der Informationskontext dennoch vollständig erhalten bleibt.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

ABS	Asset Backed Securities
APG	Absolutes Limit Posten / Gesamt
APL	Absolutes Limit Posten / Land
CSV	Comma Seperated Values
GL	Gesamtlimit
GLL	Gesamtlimit: Land
GLV	Gesamtlimit: Verkäufer
HTML	HyperText Markup Language
KLL	Kombiniertes Limit Land
KLV	Kombiniertes Limit Verkäufer
NPC	NP Complete
RLG	Relatives Limit Land / Gesamt
RVL	Relatives Limit Verkäufer / Land
RVG	Relatives Limit Verkäufer / Gesamt
RPL	Relatives Limit Posten / Land
RPG	Relatives Limit Posten / Gesamt

1 Einführung

Asset Backed Securities, kurz ABS sind eine Form der Liquiditätssteigerung von Gläubigern indem die Zahlungsforderungen verbrieft werden. Diese verbrieften Zahlungsforderungen gegenüber dem Schuldner werden daraufhin an eine Bank verkauft.[1]

Dadurch erhält der Gläubiger bereits einen Großteil des Geldes der Forderung im Voraus, ohne auf ein Zahlungsziel von bis zu neunzig Tagen oder länger aufgrund eines Zahlungsverzugs zu warten. Die Bank, der die Forderung verkauft wurde, schüttet hierbei einen Großteil der Forderung direkt aus. Der Rest wird, als Sicherheitseinbehalt später, nach Tilgung der Schuld durch den Gläubiger, ausgezahlt. Gegenüber der Bank wird für diesen Service eine Grundpauschale zuzüglich eines Prozentsatzes der Forderungssumme fällig.

Für die Bank ist diese Form des Kredits sehr lukrativ, es wird, abzüglich des Sicherheitseinbehalts, die Summe eines zuvor festgelegten Postens aufgekauft und die Gebühren, sowie ein prozentualer Anteil der Forderung selbst können als Gewinn anfallen. Diese Art von Kreditvergabe lohnt sich für die meisten Banken erst ab einem sehr hohen Volumen an Forderungen welche angekauft werden. Diese Forderungen stammen daher von unterschiedlichen Verkäufern und aus unterschiedlichen Ländern. Dafür ist die Bonität jedes Landes bzw. jedes Verkäufers einzeln zu betrachten, da ein Forderungsausfall einem Verlust der Forderungssumme abzüglich des Sicherheitseinbehalts entspricht.

Aufgrund dieses zu tragenden Risikos ist es unumgänglich eine Risikostrategie anzuwenden, mit der es möglich ist, so viele Posten anzukaufen, wie es das Budget erlaubt, und zugleich dafür Sorge zu tragen, dass das Risiko von ausfallenden Zahlungen minimiert wird, indem das Rating Beachtung findet.

[1] Vgl. Heldt, Cordula., (o. J.), Online im Internet.

1.1 **Problemstellung**

Aufgrund strikter Sicherheits- und Investitionsrichtlinien ist die optimale Auswahl ankaufbarer Posten aus einem zur Verfügung stehenden Angebot ein nicht triviales Problem für Banken. Es müssen Regeln in Form von relativen sowie absoluten Limiten in der Zusammenstellung des anzukaufenden Portfolios beachtet werden, bei einem maximalen Ankauf von „Realwert". Der Realwert definiert sich durch den Wert des Postens in Relation zu dem Ranking des Postens. Hierbei kann die Erfüllung eines Limits die Nichterfüllung eines anderen verursachen. Dieses Problem ist schon ab einer geringen Zahl von Posten nicht mehr ohne Unterstützung durch Computer lösbar. Auch auf Basis von computergestützter Berechnung kann die Rechenzeit bei größeren Mengen von Posten aufgrund der Komplexität der verknüpften Limite exponentiell steigen. Daher ist eine Annäherung an das Optimum aus wirtschaftlicher Sicht von Bedeutung. Die grafische, nachvollziehbare, Darstellung eines anzukaufenden Portfolios ist ebenfalls nicht trivial, da die Zusammenhänge der Limite ebenfalls visualisiert werden müssen.

Ausgangslage für derartige Ankaufentscheidungen ist eine Postenliste und eine Limitliste. Jeder angebotene Posten hat jeweils einen Wert, einen Verkäufer und ein Land von welchem aus er angeboten wird. Die Limite sind Regeln für die Zusammenstellung des Portfolios. Es können feste Limitierungen für den Gesamtwert des Ankaufs, den Ankauf pro Käufer und den Ankauf pro Land vorgegeben werden. Zudem können relative Limitierungen, in Form von Prozentangaben gemacht werden. Diese beziehen sich auf Posten, Verkäufer oder Land und limitieren deren Wert auf einen prozentualen Anteil eines höheren Limits.

Zur Veranschaulichung ein einfaches Beispiel, bei dem alle Posten ein Rating von 1.0 haben:

Absolute Limite: 20€ pro Verkäufer,

20€ pro Land und

40€ Gesamtlimit

Relative Limite:

1 Posten dürfen nicht mehr als 50% des Verkäuferlimits oder 25% des Länderlimits ausschöpfen.
2 Alle Posten eines Verkäufers die angekauft werden dürfen nicht mehr als 50% des Länderlimits und 25% des Gesamtlimits ausschöpfen.

3 Alle Posten die von einem Land angekauft werden dürfen nicht mehr als 50% des Gesamtlimits ausschöpfen.

Abbildung 1: Beispielportfolio, angebotene Posten

Posten	Verkäufer	Land	Wert	Posten	Verkäufer	Land	Wert
1	A	DE	15 €	7	B	DE	15 €
2	A	DE	5 €	8	B	DE	5 €
3	A	DE	5 €	9	B	DE	5 €
4	A	UK	15 €	10	B	UK	15 €
5	A	UK	5 €	11	B	UK	5 €
6	A	UK	5 €	12	B	UK	5 €

Quelle: eigene Darstellung

Bei diesem Beispiel ist eine vollständige Gesamtlimit-Ausschöpfung durch folgende Posten möglich: 2, 3, 5, 6, 8, 9, 11, 12, da diese summiert 40€ ergeben. Gleichzeitig werden mit dieser Auswahl alle definierten Limitierungen eingehalten. Jeder der Posten 1, 4, 7 und 10 alleine darf nicht angekauft werden, weil der Betrag des Postens mit 15€ bereits das Verkäuferlimit verletzt, wonach ein Posten maximal 50% des Ankaufslimits pro Verkäufer ausmachen darf, mithin maximal 10€.

1.2 Zielsetzung

Ziel dieser Bachelorarbeit ist es das in der Problemstellung angeschnittene und später unter Randbedingungen ausformulierte Optimierungsproblem algorithmisch auf zwei Arten zu lösen. Anschließend soll das Ergebnis dieses Algorithmus grafisch dargestellt werden. Dazu wird das gegebene Problem zuerst eingegrenzt, analysiert und abstrahiert. Nach der Abstraktion können zwei etablierte Lösungsmethoden angewandt werden. Daraus wird dann ein theoretischer Entwurf für eine Implementierung der beiden Algorithmen abgeleitet. Die Umsetzung dieses Entwurfes ist ebenfalls Teil dieser Arbeit, hier sollen die Grundlagen der Darstellung von Massendaten mit umgesetzt werden. Hierbei soll grundlegend aufgrund der exponentiellen Art des Problems und somit problematischen Berechnung, keine exakte Lösung, sondern eine Annäherung an das Optimum erreicht werden. Im Abschluss wird zudem ein Vergleich zwischen den beiden Algorithmen gezogen, zum einen in wie weit diese praxistauglich sind und zum anderen welcher sich als bessere Lösung für die Praxis eignet.

1.3 Aufbau und Struktur der Arbeit

Diese Arbeit gliedert sich in sechs Kapitel. Im Anschluss an diese Einleitung werden in Kapitel 2 zunächst die Randbedingungen für das vorliegende Problem definiert. Dann werden die Grundlagen zur Lösung von Optimierungsproblemen, im speziellen der kombinatorischen Optimierung, erarbeitet. Es wird erläutert wie Optimierungsprobleme definiert sind und kategorisiert werden. Des Weiteren werden Notationen vorgestellt mit denen die Effizienz von Algorithmen messbar und somit vergleichbar wird. Zudem werden Techniken erläutert mit denen Massendaten grafisch aufbereitet werden können und dabei der Informationskontext erhalten bleibt.

Darauf aufbauend wird in Kapitel 3 das vorliegende Problem analysiert, auf Basis der erläuterten Grundlagen wird das Problem modelliert und erläutert. Daraufhin wird das Problem in Teilprobleme zerlegt die im Anschluss, soweit möglich, reduziert und ebenfalls Analysiert werden. Anschließend werden Lösungsansätze vorgestellt, im speziellen der Aufbau und die Funktionsweise des Rucksackproblems und zwei Lösungsvarianten dieses Problems. Zudem wird die Methode des „Divide and Conquer“ Paradigmas erläutert und wie sich das Rucksackproblem damit lösen lässt.

Im vierten Kapitel wird der Algorithmus konzeptioniert, die Teilprobleme, die im vorherigen Kapitel reduziert und analysiert wurden, werden hier durch Teilalgorithmen gelöst, welche dann zu einem Algorithmus für das Gesamtproblem zusammengefügt werden. Hierbei findet zudem eine Betrachtung der Komplexität des konzeptionierten Algorithmus statt. Und es wird konzeptioniert wie mit den Randbedingungen eine Darstellung in Anlehnung an die Paradigmen der Darstellung für Massendaten möglich ist.

Das fünfte Kapitel behandelt, anschließend an die Konzeptionierung, die Umsetzung des Algorithmus und die grafische Darstellung anhand des konkreten Anwendungsfalls. Zum Abschluss dieses Kapitels werden die Ergebnisse von zehn Portfolioberechnungen betrachtet.

Schließlich wird im sechsten Kapitel ein Fazit gezogen, ob bzw. wie weit die entwickelten Algorithmen den Anforderungen gerecht werden, und es wird ein Ausblick hinsichtlich der Verwendbarkeit gegeben. Zudem wird hier diskutiert inwieweit die Algorithmen und deren Umsetzung erweiterbar sind.

2 Grundlagen

In diesem Kapitel werden die Grundlagen auf die sich diese Bachelorarbeit stützt definiert, erläutert und abgegrenzt. Die Randbedingungen für die Lösung des eingangs beschriebenen Problems werden definiert. Es wird festgelegt welche Daten für die Problemlösung Teil dieser Arbeit sind und in welchem Eingabe- bzw. Ausgabeformat diese vorliegen oder erzeugt werden. Ebenso werden Methoden zur Bewältigung des Problems der Darstellung von Massendaten erörtert.

2.1 Randbedingungen

Die folgenden Randbedingungen dienen einer Einschränkung der Problemstellung, da generelle Optimerungsprobleme in ihrer Variation unüberschaubar vielfältig sind. Aufgrund dessen werden im Folgenden die genauen Parameter definiert.

Eingabe / Ausgabe:

Als Ausgangslage für jedwede Verarbeitung dienen zwei tabellarische Listen im Dateiformat CSV, d.h. einer Textdatei in der die Werte mit Anführungszeichen eingegrenzt und Kommata separiert (Comma Seperated Values). Datensätze werden mit Zeilenumbrüchen abgegrenzt. Dieses Format eignet sich deshalb, weil es keine binären, proprietären, Kodierungen enthält wie bspw. XLSX, Microsofts Excel Format, oder vergleichbare Dateiformate[2]. Zudem ist es durch seinen minimalistischen Aufbau auf allen Hardwareplattformen mit allen Texteditoren les- und modifizierbar.

In der einen Datei sind alle für die Berechnung relevanten Posten, sowie deren Eigenschaften enthalten. In der zweiten Liste sind alle Limitierungen enthalten. Diese beschreiben, welchen Einschränkungen das Portfolio unterliegt.

Das Ergebnis der Berechnung ist ebenfalls eine Liste in CSV-Format, hier werden alle Posten aufgelistet die den errechneten optimalen Ankauf ermöglichen.

Zudem wird die grafische Ausgabe des Portfolios in HTML aufbereitet ausgegeben.

Eigenschaften der Posten:

Alle Posten haben das gleiche, vollständige, Eigenschaftenset. Dieses besteht aus einer eindeutigen Identifikationsnummer die pro Liste nur einmalig vergeben werden

[2] Vgl. Shafranovich, Y., (2005), S. 2, Online im Internet.

darf. Dieser dient der Identifizierung des Datensatzes für die Verarbeitung. Einem Wert des Postens, wobei hier kein Währungszusatz enthalten ist. Als Zusatz zum Wert des Postens ein Rating, dies ist ein Multiplikator zwischen 0,0 und 1,0. Einem Verkäufer, dem dieser Posten zugeordnet ist, und einem Land, von dem aus der Posten zum Ankauf angeboten wird. Der vollständige Datensatz hat somit folgenden Aufbau:

Tabelle 1: Beispiel Datensatz Postenliste

ID	Wert	Rating	Verkäufer	Land
1	256	0,95	BSP GmbH	DE

Quelle: eigene Darstellung

Portfoliolimitierungen:

Die Zusammensetzung des Portfolios unterliegt, wie in der Einführung erwähnt, bestimmten Limitierungen. Da diese die Grundlage für die Berechnung des Ankaufs liefern müssen sie ebenfalls immer vollständig vorliegen.

Limitierungen sollen in zwei verschiedenen Arten erfolgen können, zum einen absolut und zum anderen relativ. Die absolute Limitierung bezieht sich auf die summierten Werte der unterliegenden Werte. So limitiert das absolute Verkäuferlimit die summierten Werte der Posten eines einzelnen Verkäufers, das absolute Länderlimit die summierten Werte der Posten eines einzelnen Landes und das absolute Gesamtlimit die summierten Werte aller Posten.

Im Gegensatz dazu stützen sich die relativen Limite nicht auf eine vorher festgelegte Summe, sondern auf einen prozentualen Anteil eines höheren Limits. So bezieht sich das relative Länderlimit auf die Gesamtsumme, ein relatives Länderlimit von 20% bedeutet demnach dass die summierten Werte aller Posten eines beliebigen Landes maximal 20% der Gesamtsumme ausschöpfen dürfen. Zur Veranschaulichung der weiteren Limitierungsmöglichkeiten wird nachfolgend der tabellarische Aufbau der Limitdatei dargestellt:

Tabelle 2: Aufbau Limitdatei

Limitierung	Wertangabe	Bedeutung
Gesamtlimit	Absolute Summe	Gesamtsumme aller Posten
Länderlimit	Absolute Summe	Summe aller Posten eines Landes

Verkäuferlimit	Absolute Summe	Summe aller Posten eines Verkäufers
Land / Gesamt Limit	Prozentangabe	Relative Ausschöpfung der Gesamtsumme je Land
Verkäufer / Land Limit	Prozentangabe	Relative Ausschöpfung der Ländersumme je Verkäufer
Verkäufer / Gesamt Limit	Prozentangabe	Relative Ausschöpfung der Gesamtsumme je Verkäufer
Posten / Gesamt	Prozentangabe	Relative Ausschöpfung der Gesamtsumme pro Posten
Posten / Land	Prozentangabe	Relative Ausschöpfung der Ländersumme pro Posten

Quelle: eigene Darstellung

Umsetzung:

Für die Umsetzung wird die Programmiersprache JavaScript genutzt. Dies geschieht aus dem Grund, dass die Darstellung des Ergebnisses neben einer Liste aus anzukaufenden Posten auch ein HTML Dokument beinhalten soll. Dieses HTML Dokument kann mittels Nutzung von JavaScript verschiedenste Nutzerinteraktionen verarbeiten und die Darstellung des Dokumentes entsprechend ändern[3].

Durch die Nutzung von JavaScript für die Berechnung des Portfolios ist somit kein Sprachenwechsel bei verschiedenen Bereichen der Programmierung notwendig.

Um dies zu realisieren wird node.js genutzt. Dies ist prinzipiell ein Webserver-Entwicklungspaket, welches auf dem V8 JavaScript Interpreter von Google basiert. Dieses Paket ist in Module aufgeteilt welche ähnlich den Bibliotheken von anderen Programmiersprachen funktionieren. Somit stehen grundlegende Funktionen wie Dateioperationen direkt zur Verfügung. Ebenfalls wird dadurch die Erzeugung des HTML Dokuments vereinfacht, es wird keine umwandelnde Schnittstelle zwischen den Daten des Algorithmus und denen des darstellenden Dokuments benötigt.

Zudem sind durch Netzwerkmodule Möglichkeiten gegeben die Skalierbarkeit durch die Vernetzung von mehreren Computern zu erreichen.

[3] Vgl. o. V./ „W3C“, (2014), Online im Internet.

2.2 Kombinatorische Optimierung

Allgemeine, lineare, Optimierung behandelt die Suche einer optimalen Lösung unter einschränkenden Bedingungen, dies ist mathematisch gesehen der Extremwert einer Funktion. Hierbei ist die Funktion die Beschreibung des Problems und die einschränkenden Bedingungen definieren unter welchen Kriterien in dieser Funktion ein Extremum gesucht wird.[4]

In wirtschaftlichem Kontext ist dies beispielsweise die günstigste Betankung auf einer LKW Route zwischen zwei Lagerhäusern, wobei hier für jede Tankstelle ein Umweg gemacht wird. Das bestimmende Merkmal hierbei ist die lineare Nebenbedingung[5]. In diesem Beispiel das Verhältnis Kosten zu Umweg.

Kombinatorische Optimierung hingegen behandelt die Lösung von Problemen die neben der eigentlichen Problemstellung weitere einschränkende Nebenbedingungen bieten, die exponentiellen Lösungsaufwand darstellen. Durch diese Definition können mittels kombinatorischer Optimierung viele realitätsnahe Probleme, insbesondere wirtschaftliche, gelöst werden. Ein wirtschaftliches Problem dieser Art ist das „Travelling Salesman Problem“, bei diesem Problem geht es um einen Händler, der von einer Start-Stadt aus n verschiedene Städte aufsucht, um dort zu handeln und anschließend zum Start zurückkehrt. Hierbei soll die zurückgelegte Distanz minimiert werden[6]. Hier ist die Reise zwischen den Städten, also die Routenfindung, die eigentliche Problemstellung und die Minimierung der zurückgelegten Distanz die Nebenbedingung, welche sich wiederum auf das Ergebnis der gesuchten Route bezieht.

Derartige Probleme können nicht mittels einer simplen Funktion beschrieben und unter Maxima, bzw. Minima Betrachtung gelöst werden. Ein simples Verfahren um eine Lösung zu finden wäre es jede erdenkliche Kombination durchzuprobieren und die zurückgelegte Strecke mit der vorherigen zu vergleichen, ist die getestete kürzer als die vorherige wird nun diese mit der nächsten verglichen.

Bei diesem Ansatz fällt allerdings relativ früh auf das dies, selbst ohne eine Stadt zweimal zu besuchen, schon n! zu prüfende Strecken sind. Bei jeder weiteren Stadt die betrachtet wird würde sich die Fakultät um eins erhöhen. Bei 15 Städten und einer Wegberechnungszeit von 0,0001 Sekunde würde die Rechenzeit die zu vergleichenden Routen zu berechnen in Tagen folgendes betragen:

---

[4] Vgl. Hamacher, Horst W./ Müller, Stefanie (o. J.), S. 6, Online im Internet.
[5] Vgl. Korte, Bernhard/ Vygen, Jens (2012), S. 9.
[6] Vgl. Bellman, Richard (1962) S. 61.

$$\frac{15! \times 0{,}0001}{60 \times 60 \times 24} = 1513{,}51$$

Es ist klar ersichtlich dass eine solche Rechenzeit unter keinerlei Umständen Praktikabel ist. Somit ist diese simple Variante nicht praktikabel. Mit dieser Art von Problemen beschäftigt sich die kombinatorische Optimierung, mittels Reduktion des Problems, auf weniger komplexe Probleme und der Anwendung von Näherungsverfahren, kann die Rechenzeit auf eine für die Praxis geeignete Zeitspanne gesenkt werden.

2.3 **Definitionen**

Im Folgenden werden einige Klassifizierungen und Notationen der Komplexitätstheorie definiert, auf die sich diese Arbeit bezieht. Diese Notationen dienen dazu Algorithmen in Komplexitätsklassen aufzuteilen und sie damit miteinander vergleichbar zu machen.

2.3.1 Landau Notation

Die Landau Notation, oder auch „Big O Notation“ ist eine Notation der Zahlentheorie und wurde von Edmund Landau eingeführt. Obwohl es weitere zur O Notation gehörige, technisch exaktere, Notationen wie die Ω und Θ Notationen gibt[7], wird sich diese Arbeit auf die O Notation beschränken.

Sie kann dazu verwendet werden, um durch Ihre Abstraktion Algorithmen unabhängig von der eigentlichen Implementierung hinsichtlich des benötigten maximalen „Aufwandes“ miteinander zu vergleichen[8]. Dieser „Aufwand“ beschreibt die maximale Anzahl der notwendigen Berechnungsschritte, indirekt die vermutliche, nicht aber exakte, Laufzeit eines Programms, welches den gegebenen Algorithmus implementiert. Die daraus resultierende Formel kann vereinfacht als Skalierbarkeit eines Algorithmus in Relation zur Eingabemenge gesehen werden.

Hierbei ist bei einer mathematischen Repräsentation und bei einer polynomialen Laufzeit des vorliegenden Algorithmus der größte Exponent in Abhängigkeit von der Problemgröße entscheidend, da dieser die obere Komplexitätsgrenze der Verarbeitung und dadurch den benötigten Aufwand angibt. Kleinere Exponenten und Konstanten können bei dieser Betrachtung vernachlässigt werden, da deren Auswirkungen bei größeren Eingabemengen kaum bemerkbar sind.[9]

---

[7] Vgl. Cormen, Thomas H. u.a. (2010), S. 65.
[8] Vgl. Esponda, Margerita, (2012), S.11, Online im Internet.
[9] Ibid., S.17, Online im Internet.

Sei n die Eingabemenge so ist bei einem Algorithmus A mit folgendem verhalten:

$$f(n) = 32n^3 + 16n^2 + 8n + 32$$

Schon bei kleinen Eingabegrößen ist bemerkbar dass der erste Exponent die Laufzeit dominiert.

Tabelle 3: Aufwandsverhalten gemessen mit O Notation

Eingabegröße	$32n^3$	$16n^2$	Gesamt
n = 4	2.048	256	2.368
n = 8	16.384	1.024	17.504

Quelle: eigene Darstellung

Es ist klar ersichtlich dass sich die anderen Exponenten kaum auf die Gesamtzahl auswirken, daher reicht es aus, um die obere Komplexitätsgrenze zu definieren, zu beschreiben dass dieser Algorithmus A ein Element von $O(n^3)$ ist. Denn $O(n^3)$ bezeichnet eine Komplexitätsklasse und ein Algorithmus ist Element dieser Menge, wenn seine Laufzeit mit der Formel $f(n)$ abgeschätzt werden kann.

Daher kann vereinfacht kann auch $f(n) = O(n^3)$ geschrieben werden. Hierbei ist zu beachten dass das Gleichheitszeichen nicht der mathematischen Repräsentation von gleich entspricht, sondern eine Komplexitätsangabe der Funktion ist[10].

Die gängigsten Verhaltensweisen werden nun kurz erläutert:

So beschreibt $O(1)$ einen Algorithmus der für jede gegebene Eingabe immer die gleiche Zeit benötigt, das ist der theoretische Optimalfall. Dies ist bspw. eine einfache zeilenweise Textausgabe, unabhängig von der Größe des Textes ist die Ausführungszeit konstant.

$O(n)$ beschreibt einen Algorithmus dessen Ausführung sich linear zur Eingabe verhält, das „worst-case-scenario“. Dies ist bei einer linearen Suche, z.B. einer Bedingungsüberprüfung auf einen exakten Wert innerhalb einer Schleife gegeben, da dort maximal jeder Wert einmal geprüft wird.

Folglich beträgt für das Durchschreiten eines zweidimensionalen Arrays der Aufwand $O(n^2)$, da mindestens eine Schleife pro Dimension des Arrays notwendig ist.

[10] Vgl. Dobler, Heinz/ Pomberger, Gustav, (2008), S. 249

$O(log\ n)$ ist ein logarithmisches Verhalten des Aufwandes. Ein Beispiel dafür ist die binäre Suche. Die binäre Suche benötigt eine vorsortierte Liste und fängt in der Mitte der Liste an, ist der geprüfte Wert „größer“ oder „kleiner“ als der gesuchte Wert wird die Prozedur für diesen Bereich wiederholt. Auf diese Weise wird in jedem Schritt der weiter zu durchsuchenden Teil der Liste halbiert. Ist der geprüfte Wert „gleich“ dem gesuchten Wert wird die Suche beendet.

Wie eingangs beschrieben, kann mit diesem Verfahren der maximale Aufwand von zwei Algorithmen verglichen werden. Zur Veranschaulichung zeigt die folgende Tabelle die maximalen Ausführungsschritte von linearer und binärer Suche in einer vorsortierten Liste im Vergleich:

Tabelle 4: binäre und lineare Suche im Vergleich durch O Notation

Listengröße	Binäre Suche O(log n)	Lineare Suche O(n)
1.000	3	1.000
10.000	4	10.000
1.000.000	6	1.000.000

Quelle: eigene Darstellung

So wird durch die O Notation direkt ersichtlich dass die binäre Suche um ein vielfaches schneller ist als die lineare Suche. Je größer die zu durchsuchende Liste, desto stärker wird dieses bemerkbar. Somit kann die O Notation zur Messbarkeit der Skalierung eines Algorithmus genutzt werden und wie die maximale Ausführungszeit sich zu einer gegebenen Eingangsmenge an Daten verhält.

2.3.2 Polynomialzeit

Polynomialzeit ist eine Möglichkeit um die „härte“ von Entscheidungsproblemen zu messen und unterscheidet ob ein gegebenes Problem in einer polynomial zur Eingabegröße wachsenden Zeit lösbar ist oder nicht. Es wird zwischen zwei Basisklassen für die Kategorisierung von Problemen unterschieden: P und NP[11].

[11] Vgl. Cormen, Thomas H. u.a. (2010), S. 1059.

Die Komplexitätsklasse P beschreibt Probleme für die gilt dass sie in polynomial wachsender Zeit lösbar sind, also beispielsweise einen Aufwand von $O(n^k)$ mit einem Konstanten k haben[12]. Diese Art von Problemen wird als „leicht“, im Gegensatz zu NP aufgefasst da die Skalierung bei großen Eingabemengen in gleichem Verhältnis erhalten bleibt[13].

Zur Komplexitätsklasse NP gehörig sind „harte“ Probleme, diese sind nicht in polynomial wachsender Zeit lösbar, sondern nur verifizierbar. Dies ist der Fall wenn die Konstante k aus dem vorherigen Beispiel $O(n^k)$ variabel ist, somit wird der Aufwand Exponentiell und ist nun nicht mehr nur von der Eingabegröße n abhängig. Verifizierbar besagt hierbei, dass es möglich ist, ein gegebenes „Zertifikat“ einer Lösung, eine vermutliche Lösung in polynomialer Zeit auf Korrektheit zu Prüfen.[14]

Somit ist die Komplexitätsklasse P zugleich in NP enthalten, da alle Probleme in P nicht nur in polynomialer Zeit verifizierbar, sondern auch lösbar sind.

Für eine P / NP Untersuchung muss das vorliegende Problem zuerst in ein Entscheidungsproblem, mit den Antwortmöglichkeiten Wahr oder Falsch, umformuliert werden. Formal wird dieses Problem dann als Sprache L definiert. Die formale Frage besteht dann daraus, ob ein Algorithmus mit einer höchstens polynomial wachsenden Anzahl an Schritten ermitteln kann ob $L \in P$ oder $L \notin P$ gilt.[15]

Neben den beiden Klassen P und NP existiert noch der Sonderfall NPC (NP Complete), bzw. NP Vollständig. Dieser Sonderfall beschreibt ein Problem welches sich in NP befindet und welches sich nur in andere Probleme welche sich ebenfalls in NP befinden transformieren lässt. Bei Problemen dieser Sonderkategorie ist es bisher nicht nachweisbar ob ein Algorithmus existiert, welcher das Problem in polynomialer Zeit lösen kann. Man nimmt heute aber allgemein an, dass es für NP vollständige Probleme keine Algorithmen gibt, die diese in polynomialer Zeit lösen.

[12] Vgl. Cormen, Thomas H. u.a. (2010), S. 1060.
[13] Vgl. Wanka, Rolf, (2006), S. 4.
[14] Vgl. Cormen, Thomas H. u.a. (2010), S. 1060.
[15] Vgl. Wanka, Rolf, (2006), S. 5.

2.3.3 Approximation

In der praktischen Anwendung ist eine exakte Lösung eines Problems nicht immer erforderlich. Häufig ist eine annähernd optimale Lösung eines Problems bereits ausreichend um diese praktisch nutzen zu können. Aufgrund dessen werden Approximationsmethoden genutzt um in einem vertretbaren Zeitrahmen eine ausreichende Lösung zu finden.

Approximationsmethoden zeichnen sich dadurch aus, dass die Genauigkeit der Lösung zugunsten von Geschwindigkeit aufgegeben wird. Im idealen Fall verläuft dieser Kompromiss linear, so dass eine „freie“ Wahl zwischen Genauigkeit und benötigter Rechenleistung bzw. Zeit besteht. Wenn dieser Kompromiss nicht linear ist muss die fehlende Genauigkeit gegenüber der Praxistauglichkeit abgewogen werden.

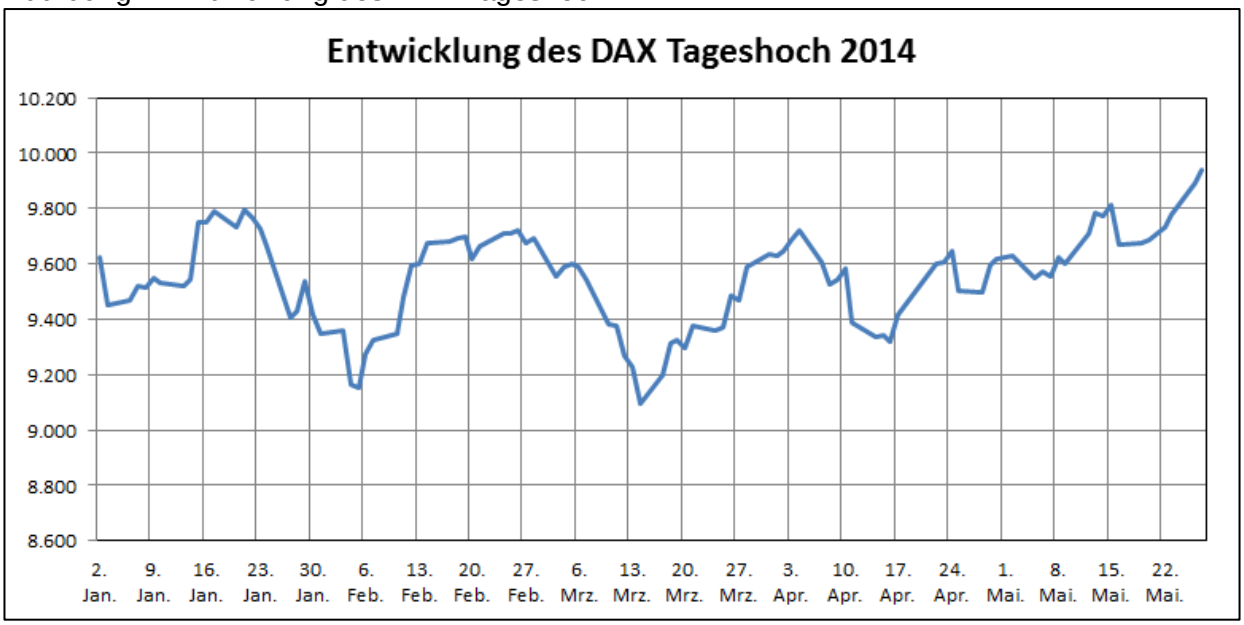
2.4 Grafische Darstellung von Massendaten

Die Darstellung von Daten ist bereits ab einer geringen Zahl von Datensätzen ein nicht triviales Problem. Hunderte von Einträgen sind nicht mehr übersichtlich darstellbar und können schwer, bzw. nicht mehr in einen zusammenhängenden Kontext gebracht werden. Zudem ist die Textausgabe in Zeilenform bzgl. der darzustellenden Masse an Datensätzen stark limitiert. Es können höchstens ein paar hundert Zeilen auf einem Bildschirm gleichzeitig dargestellt werden.[16]

Diese Limitierungen können umgangen werden, indem die Daten mittels grafischer Hilfsmittel wie z.B. Diagrammen dargestellt werden. Durch diese grafische Verdichtung von Daten zu Information können auch große Mengen von Datensätzen in einem Zusammenhang angezeigt werden.

Zum Beispiel die Tageshoch-Werte des DAX vom zweiten Januar bis Ende Mai des Jahres 2014, in Tabellarischer Form, nach Monaten Gruppiert[17]. Bei einer großen Menge an Datensätzen ist ein direkter Verlauf des DAX Tageshöchstwertes in Relation zu den Datumsangaben im Zusammenhang nicht mehr direkt klar ersichtlich. Hier können aufbereitende Hilfsmittel, wie Informationsreduktion und eine grafische Darstellung als Liniendiagramm den Zusammenhang der Daten einfach verständlich darstellen:

Abbildung 2: Entwicklung des DAX Tageshoch



Quelle: eigene Darstellung auf Basis der Daten von Finanzen.net (2014)

[16] Vgl. Keim, Daniel A. (o. J.) S. 1.

[17] Vgl. Anhang 1: DAX Tageshoch Werte nach Monaten.

Diese Art der Darstellung bietet den Vorteil dass neben dem Gesamtzusammenhang der Daten auch direkt die wichtigsten Informationen ersichtlich sind. Durch die vorgenommene Erweiterung des Liniendiagrammes um Hilfslinien, welche jeweils siebentägige Intervalle abgrenzen, kann die Interpretation noch weiter erleichtert werden. So ist der Verlauf des DAX Tageshochs an bestimmten Bezugspunkten klar erkennbar.

Der Nachteil hingegen ist dass durch diese komprimierende Art der Informationsdarstellung exakte Daten, wie der genaue Tages Höchstwert am 11. Februar entfallen. Diese sind in dem Diagramm nicht mehr abzulesen. Zudem ist es nicht möglich erweiterte Informationen zu diesem Datensatz, wie einen Zeitstempel, anzuzeigen. Daher kann eine einfache Diagrammdarstellung nur als Kompromiss, nicht aber als Lösung der Informationsdarstellung von Massendaten angesehen werden.

Visual Information Seeking Mantra

Eine Lösung für dieses Problem ist die Nutzung des „Visual Information Seeking Mantra“ Paradigmas. Dieses Paradigma stützt sich grundlegend darauf dass die Bandbreite von Informationspräsentationen mit grafischer Darstellung am höchsten ist, da Nutzer sich Bilder einfach einprägen können und somit Änderungen an diesen schnell und intuitiv wahrnehmen.[18] Um den vorhin beschriebenen Informationsverlust zu verhindern und es dem Nutzer so zu ermöglichen auf die Daten zuzugreifen welche bei einem „Diagram-Kompromiss“ nicht mehr genau dargestellt werden, gibt es weiterführende Schritte mit denen die Nutzerinteraktion in die Informationsdarstellung eingebunden wird:

Overview / Zoom

Zuerst soll eine Übersicht über den gesamten Datenbestand gegeben werden. Dafür wird ein Diagramm verwendet. Um dieses für große Datenmengen übersichtlicher zu gestalten als eine rein statische Darstellung wie in Abbildung 1 wird Nutzerinteraktion in Form von Navigationstools eingebunden. Mit diesen soll der Nutzer von der Gesamtübersicht auf Teilbereiche des gesamten Bestandes zoomen können. Da bei einer solchen Darstellung logischerweise die Daten, welche außerhalb des „Gezoomten“ liegen nicht mehr angezeigt werden, sind weitere Navigationstools notwendig. So sol-

[18] Vgl. Shneiderman, Ben, (1996), S. 336f, Online im Internet.

len die beiden Achsen des Diagrammes verschiebbar sein, sodass der Nutzer eine Übersicht eines von ihm gewählten Informationsbestandes hat, er allerdings auch einfach auf angrenzende Daten zugreifen kann.[19]

Filter

Der Schritt des Filterns soll den Datenbestand mittels vom Nutzer bestimmbaren Abfragen direkt einschränken. Dadurch wird ihm die Möglichkeit gegeben, sich auf die für ihn wichtige Daten zu fokussieren und für seine Betrachtung irrelevante Daten auszublenden.[20]

Diese Abfragen des Nutzers unterteilen sich in zwei Arten des Filterns. Eine Möglichkeit ist das Browsing. Hierbei bestimmt der Nutzer durch direkte Selektion welche Teilmenge der Daten er betrachten möchte. Die andere Möglichkeit ist das Querying. Hierbei definiert der Nutzer die Eigenschaften der Teilmenge die er zu betrachten wünscht.[21]

Hierbei ist der Übergang zwischen beiden Möglichkeiten des Filterns fließend. So sind auch Mischformen denkbar, wie eine vorab Selektion mittels Querying und einer Browsing Auswahl um das Ergebnis genauer zu spezifizieren.

Details-on-Demand

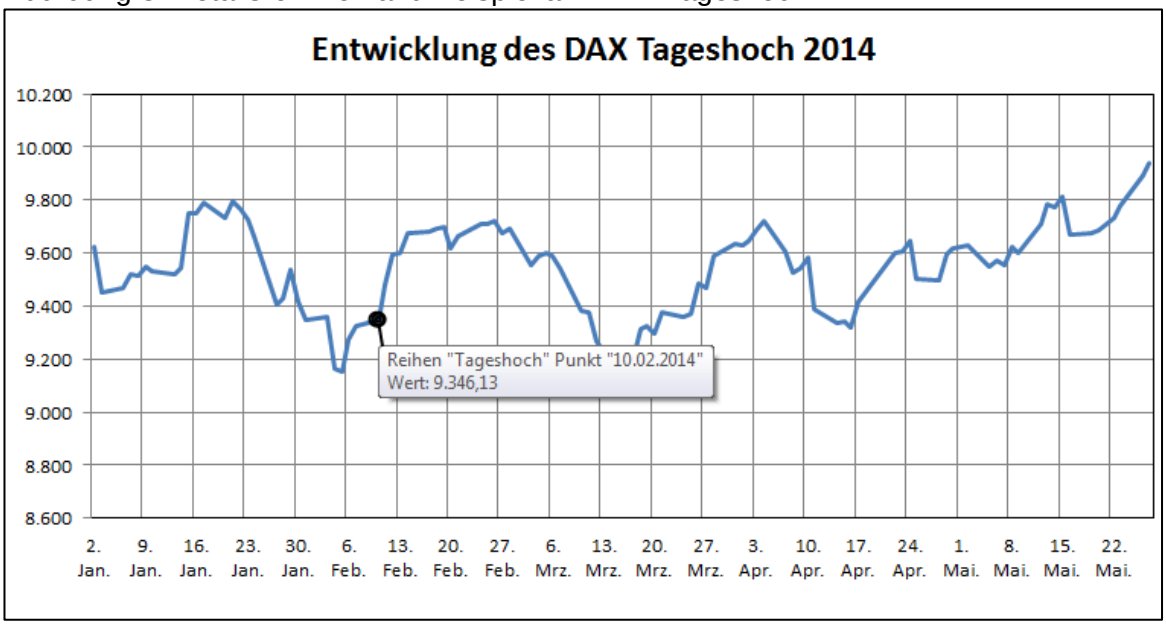
Sobald der Initiale Datenbestand durch Vorauswahl, Filterung und Zoomen soweit eingegrenzt wurde dass die gleichzeitig dargestellten Informationen übersichtlich dargestellt werden können sollen erweiterte Details angezeigt werden. Bei zuvor komprimierten Daten kann so nun z.B. der komplette Datensatz in der Detailansicht dargestellt werden.[22]

Dadurch kann das vorher als „Kompromiss“ behandelte Problem als gelöst angesehen werden, da die Übersichtlichkeit erhalten wird, ohne dass Informationen verloren gehen. So könnte das vorherige Beispiel um Details-on-Demand erweitert werden, indem der gesamte Datensatz eingeblendet wird sobald sich der Mauszeiger über einem Punkt des Diagramms befindet.

---

[19] Vgl. Shneiderman, Ben, (1996), S. 339, Online im Internet.
[20] Vgl. Shneiderman, Ben, (1996), S. 339, Online im Internet.
[21] Vgl. Keim, Daniel A. (o. J.) S. 14.
[22] Vgl. Shneiderman, Ben, (1996), S. 340, Online im Internet.

Abbildung 3: Details-on-Demand Beispiel am DAX Tageshoch



Quelle: eigene Darstellung auf Basis der Daten von Finanzen.net (2014)

Der Zusammenhang des vorherigen Beispiels ist eine einfache Datenstruktur, zu je einer Datumseinheit existiert ein dazugehöriger Tageshöchstwert. Diese Datenstruktur ist eindimensional[23], benötigt allerdings bereits eine zweidimensionale grafische Repräsentation um den Zusammenhang visualisieren zu können. Die Implementierung des Visual Information Seeking Mantras erhöht den Aufwand diese Struktur exakt darzustellen nochmals.

Für komplexere Beziehungen zwischen verschiedenen Daten kann das grundlegende Visual Information Seeking Mantra erweitert werden. So bietet es sich an, bei zwei- oder multidimensionalen Daten, erweiterte grafische Techniken, wie Farbunterschiede, zu nutzen um bestimmte Relationen darzustellen.

[23] Vgl. Keim, Daniel A. (o. J.) S. 5.

3 Problemanalyse und Lösungsansätze

Mithilfe der erläuterten Grundlagen wird im Folgenden das in dieser Bachelorarbeit vorliegende Problem der Portfoliokonzentration Modelliert, analysiert und anschließend in Teilprobleme zerlegt. Aufgrund dieser Vorarbeit können danach Kombinatorische Lösungsverfahren abstrahiert und auf das Problem angewandt werden.

3.1 Modellierung des Problems

Das in der Problemstellung formulierte und in den Randbedingungen spezifizierte Problem bezieht sich auf Posten welche jeweils eine Wertigkeit, ein Rating, einen Verkäufer und ein Land als Eigenschaften haben. Die einzelnen Eigenschaften sind wie folgt definiert:

$$Wertemenge\ W = \{w\ \in \mathbb{R}\ |w >\ 0\}$$

$$Ratingmenge\ R = \{r\ \in \mathbb{R}\ |\ 0 < r < 1\}$$

$$Verkäufermenge\ V = \{v\ \in \mathbb{Z}\ |v > 0\}$$

$$Ländermenge\ L = \{l\ \in \mathbb{Z}\ |l > 0\}$$

Zur einfacheren Darstellung werden Verkäufer und Länder für die Problemmodellierung als ganzzahlige Werte kodiert. Hierbei steht jede Zahl fest für ein Land bzw. für einen Verkäufer. Die Wertangabe wird als reelle Zahl definiert, da sie eine gängige Währungszahl mit zwei Dezimalstellen angibt. Das Rating wird ebenfalls als reelle Zahl mit zwei Dezimalstellen definiert, mit der Einschränkung dass dieser Wert zwischen 0,0 und 1,0 liegen muss, da es ein Multiplikator ist. Dieser dient zur Errechnung des Realwertes des Postens, der Realwert wird im Folgenden als $RW = w \times r$ bezeichnet.

Die Posten selbst können nun als Mathematisches Quadrupel dargestellt werden:

$$P = \{p_i\ |\ i\ \in\ \mathbb{N}\}$$

$$p_i = (w_i, r_i, v_i, l_i)\ |\ w_i \in W, r_i\ \in R, v_i \in V, l_i \in L$$

In der weiteren Ausarbeitung wird davon ausgegangen, dass es immer mehr Posten gibt, als die Portfoliolimitierungen zulassen. Somit gilt Zielmenge ⊆ Postenmenge. Die Zielmenge wird wie folgt definiert:

$$Q = \{q_i\ |\ i\ \in\ \mathbb{N}\}\ \subseteq P$$

$$q_i = (w_i, r_i, v_i, l_i)\ |\ w_i \in W, r_i\ \in R, v_i \in V, l_i \in L$$

Die absoluten Limitierungen Gesamtlimit (GL), Länderlimit (GLL) und Verkäuferlimit (GLV), welche die Zusammensetzung des Portfolios bestimmen, unterliegen der Regelung $\{x \in \mathbb{R} \mid x > 0\}$, da es Währungsangaben sind. Die relativen Limitierungen Land / Gesamt (RLG), Verkäufer / Gesamt (RVG) und Verkäufer / Land (RVL) werden als prozentualer Anteil mit $\{x \in \mathbb{R} \mid 0 < x \leq 100\}$ definiert. Die Ausschöpfung eines Limits für eine Zielmenge wird als Variable gleichen Namens mit Suffix a definiert.

Das Hauptproblem ist eine Maximierung des Realwertes aller Posten:

$$\sum_{p_i \in Q}^{n} r_i w_i \; Maximal$$

unter der Einschränkenden Nebenbedingung des Gesamtlimits welches nicht überschritten werden darf:

$$GLa = \sum_{p_i \in Q}^{n} w_i \leq GL$$

Die absoluten Limitierungen sind Einschränkungen bei bestimmten Ländern und Verkäufern. Der bestimmende Index ist als l bzw. v definiert. Zur besseren Darstellung wird dieser nicht als Laufindex definiert, diese Limitierung ist allerdings für alle Länder bzw. Verkäufer vorgesehen. So sind diese Limitierungen per Definition „kleinere Arten“ der zuvor bestimmten Nebenbedingung:

Absolutes Limit: Für jedes Land l aus der Ländermenge L gilt

$$GLLa = \sum_{\substack{p_i \in Q \\ und\ li=i}}^{n} w_i \leq GLL$$

Das absolute Länderlimit bezieht sich auf eine limitierende Summe, ist also eine kleinere Version der Portfolioberechnung selbst, als Einschränkung gilt GLL, hier hat der Verkäufer keine Relevanz für die Summierung.

Die Verkäuferlimitierung verhält sich genauso wie die vorhergehende Länderlimitierung, bei dieser Betrachtung ist analog dazu das Land irrelevant.

Absolutes Limit: Für jeden Verkäufer v aus der Verkäufermenge gilt

$$GLVa = \sum_{\substack{p_i \in Q \\ und\ vi=v}}^{n} w_i \leq GLV$$

Bei den relativen Limitierungen verhält es sich ähnlich, allerdings wird anstelle einer vorher fest definierten Summe ein Multiplikator definiert, welcher mit dem Gesamtlimit bzw. dem Länderlimit verrechnet wird.

Relatives Limit: Für jedes Land l aus der Ländermenge L gilt

$$RLGa = \sum_{\substack{p_i \in Q \\ und\ li=l}}^{n} w_i \leq GL \times \frac{RLG}{100}$$

Relatives Limit: Für jeden Verkäufer v aus der Verkäufermenge gilt

$$RVGa = \sum_{\substack{p_i \in Q \\ und\ vi=v}}^{n} w_i \leq GL \times \frac{RVG}{100}$$

Relatives Limit: Für jeden Verkäufer v aus der Verkäufermenge und jedes Land l aus der Ländermenge gilt

$$RVLa = \sum_{\substack{p_i \in Q \\ und\ vi=v \\ und\ li=l}}^{n} w_i \leq GLL \times \frac{RVL}{100}$$

Die relative Postenlimitierungen, Relatives Limit Posten / Gesamt (RPG) und Relatives Limit Posten / Land (RPL) nehmen hier eine Sonderstellung ein, sie beziehen sich nicht auf ein Summenlimit, sondern direkt auf eine Wertigkeit eines jeden Posten und beschreiben Prozentangaben, daher sind sie als $\{x \in \mathbb{R} \mid 0 < x \leq 100\}$ definiert. Als Bedingungen gelten demnach für jeden Posten pi:

$$w_i \leq GL \times \frac{RPG}{100}$$

Und für jeden Posten pi aus dem Land l

$$w_i \leq GLL \times \frac{RPL}{100}$$

Posten bei denen festgestellt wird, dass diese Bedingung nicht erfüllt ist sind direkt vom Portfolio ausgeschlossen und somit für weitere Berechnungen und Prüfungen nicht mehr relevant.

3.2 **Analyse der Teilprobleme**

Die zuvor definierten Teilprobleme sind hierarchisch anzuordnen, da sie sich weitestgehend auf die gleichen Werte beziehen oder einander ausschließen. So macht es keinen Sinn einen Posten bei der Auswahl darauf zu prüfen ob er Teil der Zielmenge ist, wenn er bei den Prüfungen „Absolutes Limit: Verkäufer“ und „Relatives Limit: Verkäufer / Gesamt“ gegen eines der beiden verstößt. Dies ist darauf zurückzuführen dass die geringere Summe einschränkend genug ist, sodass die höhere Summe des anderen Limits keinerlei Relevanz mehr hat:

Tabelle 5: Teilproblemreduktion

Gesamtes Limit:	Absolutes Limit: Verkäufer	Relatives Limit: Verkäufer / Gesamt	Bestimmendes Limit:
1.000	500	20% (200)	200
1.000	400	50% (500)	400

Quelle: eigene Darstellung

Das in diesem Fall am weitetesten einschränkende Limit bestimmt eine Art „Schranke“ dafür, ab welchem Wert der entsprechende Posten Teil des Portfolios sein darf. Deshalb würde bei dem Beispiel in Zeile eins das relative Limit von 20% mehr einschränken als das absolute Limit. Die Summe von 500 wird nie erreicht, da dies gegen das Relative Limit von 200 verstoßen würde. Die gleiche Konstellation findet sich ebenfalls bei der absoluten und bei der relativen Limitierung der Länder wieder.

Durch eine Vorberechnung des relativen Länderlimits und des relativen Verkäuferlimits können die Bedingungen vereinfacht werden. Die errechneten Werte können bei einem Vergleich zum korrespondierenden absoluten Limit zu einem zusammengefasst werden, da nur das kleinere Limit gültig ist.

Kombiniertes Limit: Land (KLL)

Vorberechnung:

$$KLL = Minimum\left\langle GL \times \frac{RLG}{100} \middle| GLL \right\rangle$$

Dann deckt das kombinierte Limit beide Bedingungen ab, welche für jedes Land l aus der Ländermenge gilt:

$$\sum_{\substack{p_i \in Q \\ und\ li=l}}^{n} w_i \leq KLL$$

Dies kann analog dazu ebenfalls für das Verkäuferlimit geschehen, wobei hier die Werte von Gesamtlimit: Verkäufer und Relatives Limit: Verkäufer / Gesamt, sowie relatives Limit: Verkäufer / Land zu einem Limit zusammengefasst werden können:

Kombiniertes Limit: Verkäufer (KLV)

Vorberechnung:

$$KLV = Minimum \left\langle GV \times \frac{RVG}{100} \middle| GLV \middle| GLL \times \frac{RVL}{100} \right\rangle$$

Dies führt zu folgender Kombinierter Bedingung, welche für jeden Verkäufer v aus der Verkäufermenge in Kombination mit jedem Land l aus der Ländermenge gilt:

$$\sum_{\substack{p_i \in Q \\ und\ vi=v \\ und\ li=l}}^{n} w_i \leq KLV$$

Die Postenbedingungen RPG und RPL können nicht zu einem kombinierten Limit zusammengefasst werden, da sie sich jeweils auf das Länder, bzw. Gesamtlimit und somit nicht die gleichen Summen beziehen. Um einheitliche, zu überprüfende, Limitierungen zu erhalten können sie für die weitere Bearbeitung auch als absolute Summen definiert werden:

$APG = GL \times \frac{RPG}{100}$ und $APL = GLL \times \frac{RPL}{100}$

Zudem können sie vom Hauptproblem ausgeschlossen werden. Da sie eine Bedingung an jeden einzelnen Posten stellen, kann die Postenliste im voraus geprüft werden, ob gegen eine der beiden Bedingungen verstoßenden wird. Die verstoßenen Posten müssen dann nicht mehr betrachtet werden, da sie direkt vom Ankauf ausgeschlossen sind.

Auf Basis der Problemanalyse werden im Folgenden Techniken vorgestellt mit denen diese bewältigt werden können. Hierbei wird Bezug auf die bisherigen Ergebnisse genommen. Diese Techniken werden zuerst generell vorgestellt und im Anschluss daran auf das vorliegende Portfoliooptimierungsproblem abstrahiert um aufzuzeigen dass sie anwendbar sind. Die Anwendung selbst findet im Kapitel Konzeption statt.

3.3 **Das Rucksackproblem**

Als „Rucksackproblem“ wird allgemein eine Problemstellung bezeichnet bei der eine Menge von Gegenständen mit Eigenschaften, in Form von Gewicht und Wert, in einen abstrakten Rucksack gepackt werden sollen. Dieser ist hinsichtlich seiner Tragfähigkeit, d.h. dem maximalen Gewicht welches erreicht werden darf, limitiert. Hierbei geht es darum dass unter Einschränkung dieses maximalen Traggewichts des Rucksacks der Wert der Gegenstände maximiert wird. Diese abstrakte Formulierung des Rucksackproblems kann auf viele Probleme übertragen werden bei denen aus einer Menge von Elementen mit Eigenschaften eine Teilmenge unter Nebenbedingungen gesucht wird, wie bspw. Investments. Dort ist der Rucksack die Investitionssumme und der Wert einer Investition der „Return on Investment“, welcher maximiert werden soll.[24]

Die Gegenstände (I) können als Tupel dargestellt werden mit den ganzzahligen Eigenschaften Gewicht (G) und Wert (W), die gesamte Menge an Gegenständen ist somit eine Menge (GI) von den Tupeln I:

$$I = (W, G)$$

$$G = \{g \in \mathbb{Z} \,|\, g > 0\}$$

$$W = \{w \in \mathbb{Z} \,|\, w > 0\}$$

Gesucht wird nun eine Teilmenge TGI von I, welche unter der Bedingung:

$$\sum_{i \in TGI}^{n} g_i \leq Gesamtgewicht$$

zugleich eine Maximierung der addierten Werte der Gegenstände erreicht:

$$\sum_{i \in TGI}^{n} w_i \; Maximal$$

Es ist klar ersichtlich dass es nicht praktikabel ist jede mögliche Kombination von Teilmengen zu bilden und zu vergleichen. Da auch hier, wie bei dem „Travelling Salesman“ in Kapitel 2.3, durch die Komplexität von $O(n!)$ der Zeitaufwand für diese Berechnung nicht polynomial beschränkt ist.

Das Rucksackproblem ist NP-Vollständig, da eine Generalisierung dieses Problems, das „Subset-Sum“ Problem bereits als NP-Vollständig bewiesen wurde. Somit ist für eine praxistaugliche Lösung eines solchen Problems nur ein Approximationsverfahren oder ein Pseudopolynomiales Lösungsverfahren denkbar.[25]

---

[24] Vgl. Martello, Silvano/ Toth, Paolo, (1990), S. 1f.

[25] Vgl. Korte, Bernhard/ Vygen, Jens (2012), S. 493.

3.4 Greedy-Algorithmus

Diese Art von Algorithmen zeichnet sich dadurch aus dass eine situationsabhängige optimale Entscheidung getroffen wird und somit eine Summe von optimalen Teilergebnissen das Gesamtergebnis darstellt. Allerdings ist dies nicht dazu geeignet ein optimales Gesamtergebnis zu erreichen. Dies ist darauf zurückzuführen dass nicht alle möglichen Lösungswege beachtet werden, sondern nur der mit den jeweils besten Teilergebnissen.[26]

So kann für ein einfaches Rucksackproblem als „Greedy-Heuristik“ ein relativer Wert aus Gewicht und Wert des Gegenstandes berechnet werden und die Gegenstände nach diesem relativen Wert absteigend sortiert werden. Auf Basis dieses relativen Wertes $relW = \frac{W}{G}$ wird dann der Rucksack gefüllt:

Tabelle 6: Rucksackproblem, Greedy-Beispiel

Gegenstand	Wert	Gewicht	Relativer Wert
1	2	1	2,0
2	3	3	1,0
3	2	2	1,0
4	2	2	1,0

Quelle: eigene Darstellung

Bei einem maximalen Gewicht des Rucksacks von 5 Einheiten würde der Greedy-Algorithmus in diesem Beispiel die Gegenstände eins und zwei auswählen, weil er die sortierte Liste absteigend prüft. Somit wird die maximale Tragfähigkeit des Rucksacks nicht erreicht und somit auch nicht den optimalen Wert der Gegenstände.

Die Optimale Lösung wäre es die Gegenstände eins, drei und vier zu wählen, da der Gesamtwert dann 6 im Gegensatz zu dem Greedy-Ergebnis von 5 beträgt. Diese Form der Annäherung ist zwar nicht optimal, bietet aber ein schnelles Lösungsverfahren für dieses komplexe Problem. Da für jedes n nur ein Rechenschritt für die Berechnung des relativen Wertes erforderlich ist, die Liste mit einer ebenfalls von n abhängigen Komplexität sortierbar ist und die Auswahl im schlechtesten Fall ebenfalls mit n Schritten zu treffen ist, ergibt sich eine Gesamtkomplexität von $O(n)$ für diesen Lösungsansatz.

[26] Vgl. Cormen, Thomas H. u.a. (2010), S. 425.

3.5 Divide and Conquer

Das Paradigma „Divide and Conquer“ beschreibt eine weitere generelle Vorgehensweise, um komplexe Probleme zu lösen. Im Gegensatz zum inkrementellen Verfahren, bei dem das Problem von einem Ausgangspunkt aus schrittweise gelöst wird, geht es bei diesem Paradigma darum das Problem rekursiv zu lösen. Hierbei ist die Voraussetzung für die Anwendung dass das Problem in gleichartige Teilprobleme zerlegt werden kann. Die Lösung dieser Teilprobleme führt somit zur Lösung des Gesamtproblems.[27]

Um dieses Paradigma anwenden zu können müssen auf das zu lösende Problem drei Schritte angewandt werden. Der erste Schritt ist die Bestimmung der Teilprobleme, diese Teilprobleme müssen kleinere Formen des Hauptproblems sein und folglich nach gleichem Schema lösbar sein. Zweiter Schritt ist die „Beherrschung“ des jeweiligen Teilproblems, das Problem wird, falls es klein genug um es zu lösen, direkt gelöst. Andernfalls wird das vorliegende Teilproblem erneut mit Schritt eins behandelt. Sobald es keine Teilprobleme mehr gibt die nicht mehr direkt gelöst werden können, wird im letzten Schritt, der „Vereinigung“, die Lösung des Gesamtproblems aus den Lösungen der Teilprobleme zusammengesetzt.[28]

Eine Anwendungsmöglichkeit dieses Paradigmas für das Rucksackproblem ist die dynamische Programmierung. Diese unterteilt das vorliegende Rucksackproblem in kleinere Rucksackprobleme und löst diese rekursiv. Hierbei werden die Resultate der Berechnungen in Arrays zwischengespeichert. Aus diesen lässt sich dann nach der Verarbeitung ableiten, welche Gegenstände ein Optimum erreichen.[29]

Die dynamische Lösung des Rucksackproblems ist prinzipiell eine „Branch and Bound“ Variante. Hierbei werden bei einem „wachsenden“ Rucksack alle möglichen Füllungen des Rucksacks berechnet, welche eine Wertsteigerung bedeuten, angefangen bei einer Rucksackgröße von eins. Davon ausgehend wird der Rucksack bei jedem Schritt um eine Einheit erweitert und geprüft, ob ein Gegenstand hinzugefügt werden kann. Wenn dies der Fall ist, wird in einer „Wertmatrix“ der nun entstandene Rucksackwert festgehalten und in einer „Entscheidungsmatrix“ ein Wahrheitswert festgehalten, welcher angibt das eben dieser Gegenstand Teil der Auswahl ist oder nicht. Auf diese Weise enthält die Entscheidungsmatrix nach der Verarbeitung jede mögliche Konstella-

[27] Vgl. Cormen, Thomas H. u.a. (2010), S. 31.
[28] Ibid, S. 67.
[29] Vgl. Wanka, Rolf, (2006), S. 68f.

tion der Rucksackfüllungen. Zur Veranschaulichung die Lösung des Greedy-Beispiels mittels der dynamischen Variante:

Abbildung 4: Rucksackproblem, Dynamisierungs-Matrizen

W	1	2	3	4	5
1	2	2	2	2	2
2	0	0	3	5	5
3	0	2	4	4	5
4	0	2	4	5	6

E	1	2	3	4	5
1	1	1	1	1	1
2	0	0	1	1	1
3	0	1	1	1	1
4	0	1	1	1	1

Quelle: eigene Darstellung

Das Greedy-Beispiel umfasste einen Rucksack mit einer Kapazität von fünf und einer Auswahl zwischen vier Gegenständen. Wie benötigen daher zwei gleich große Matrizen deren Spaltenzahl gleich der Rucksackkapazität und deren Zeilenzahl gleich der Anzahl der Gegenstände ist.

Die Wertmatrix wird nun nach folgendem Schema gefüllt: Die Spalte gibt die momentane Rucksackkapazität an und die Zeile den zu evaluierenden Gegenstand. Für diesen Gegenstand wird nun die Frage gestellt, ob er in die aktuelle Rucksackkapazität passt und noch ein Rest Kapazität übrig ist. Ist der Gegenstand passend und es ist kein Rest vorhanden, wird der Wert des Gegenstandes an dieser Position in die Wertematrix eingetragen und an der gleichen Koordinate in der Entscheidungsmatrix eine 1. Ist der Gegenstand nicht passend, wird entsprechend eine 0 eingetragen.

Falls es einen Rest gibt wird verglichen, ob der Aktuelle Gegenstandswert addiert mit dem Wert eines kleineren Rucksacks, welcher den Restwert füllt, größer als der direkt zuvor errechnete Rucksackwert ist. Dieser befindet sich eine Zeile über dem momentan zu berechnenden Wert (In Abbildung 4 ist dieser für Spalte 4, Zeile 3 dementsprechend in Spalte 4, Zeile 2). Ist der zuvor errechnete Wert größer wird in der Entscheidungsmatrix eine 0 festgehalten, ist er kleiner eine 1.

Um herauszufinden welche Kombination die optimale Lösung darstellt wird nun die Entscheidungsmatrix genutzt.

Ausgehend von der maximalen Spalten- und Zeilenposition wird nun der Rucksack gefüllt. An dieser Koordinate wird eine Fallunterscheidung getroffen. Ist eine 1 vorhanden wird der entsprechende Gegenstand eingepackt und die Koordinate für die nächste Prüfung ergibt sich aus der aktuellen Spaltenzahl minus dem Gewicht des eingepackten Gegenstandes. Zudem ist die neue, zu prüfende, Zeile die aktuelle Zeilenzahl minus 1. Falls an der zu prüfenden Koordinate eine 0 vorliegt, wird der Gegenstand

nicht eingepackt. Die neue Koordinate ist dann gleich der vorherigen, wobei die Zeilenzahl um eins reduziert wird.

Durch dieses Vorgehen wird im Greedy-Beispiel, bei Nutzung des dynamischen Verfahrens, der Gegenstand zwei nicht eingepackt. Somit wird die optimale Lösung von 6 erreicht.

4 Konzeption

Im Folgenden wird ein Konzept erarbeitet auf dessen Basis die Umsetzung des Algorithmus beginnen kann. Dazu werden zuerst Teilalgorithmen entworfen, welche spezifische Teile des Problems lösen können. Anschließend werden diese Teilalgorithmen in einem Algorithmus zusammengefasst der eine vollständige Lösung des Problems darstellt. Hierbei wird die eigentliche Berechnung des Portfolios auf zwei Arten gelöst, diese Teilalgorithmen sollen austauschbar sein um sie hinterher vergleichen zu können. Zum Abschluss werden Optimierungsmöglichkeiten für die Umsetzung aufgezeigt.

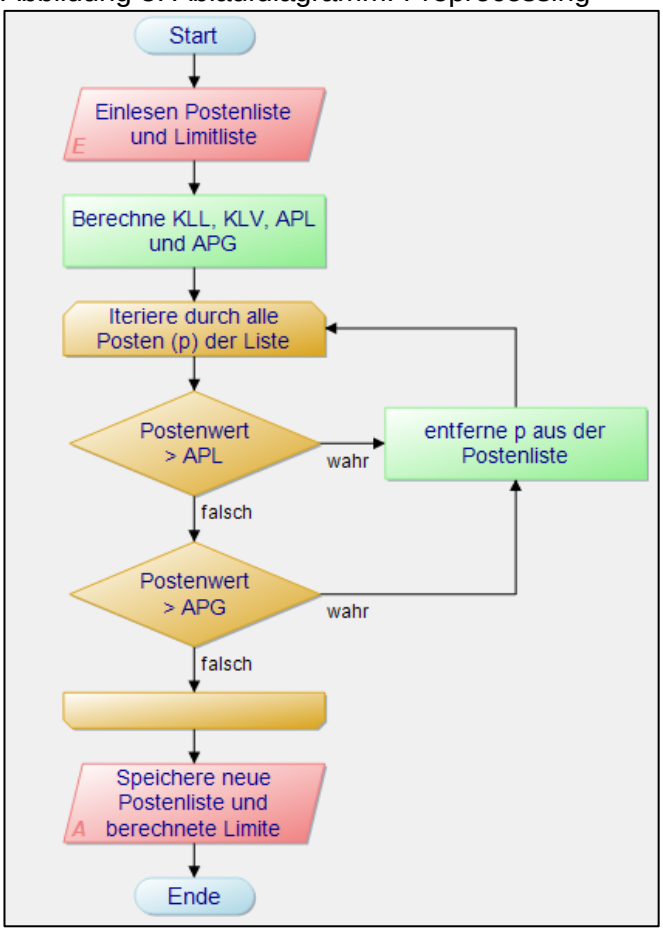
4.1 Teilalgorithmen

Im Folgenden werden einzelne Teile der Gesamtlösung konzeptioniert, dies geschieht in Form von Programmablaufplänen und einer Erläuterung weshalb sich dieser zur Umsetzung eignet.

Vorverarbeitung:

Als „Preprocessing“, eine vorab Verarbeitung der Daten bevor die eigentliche Ermittlung des Portfolios stattfindet, können die kombinierten Limitierungen für die weitere Verarbeitung berechnet werden:

Abbildung 5: Ablaufdiagramm: Preprocessing



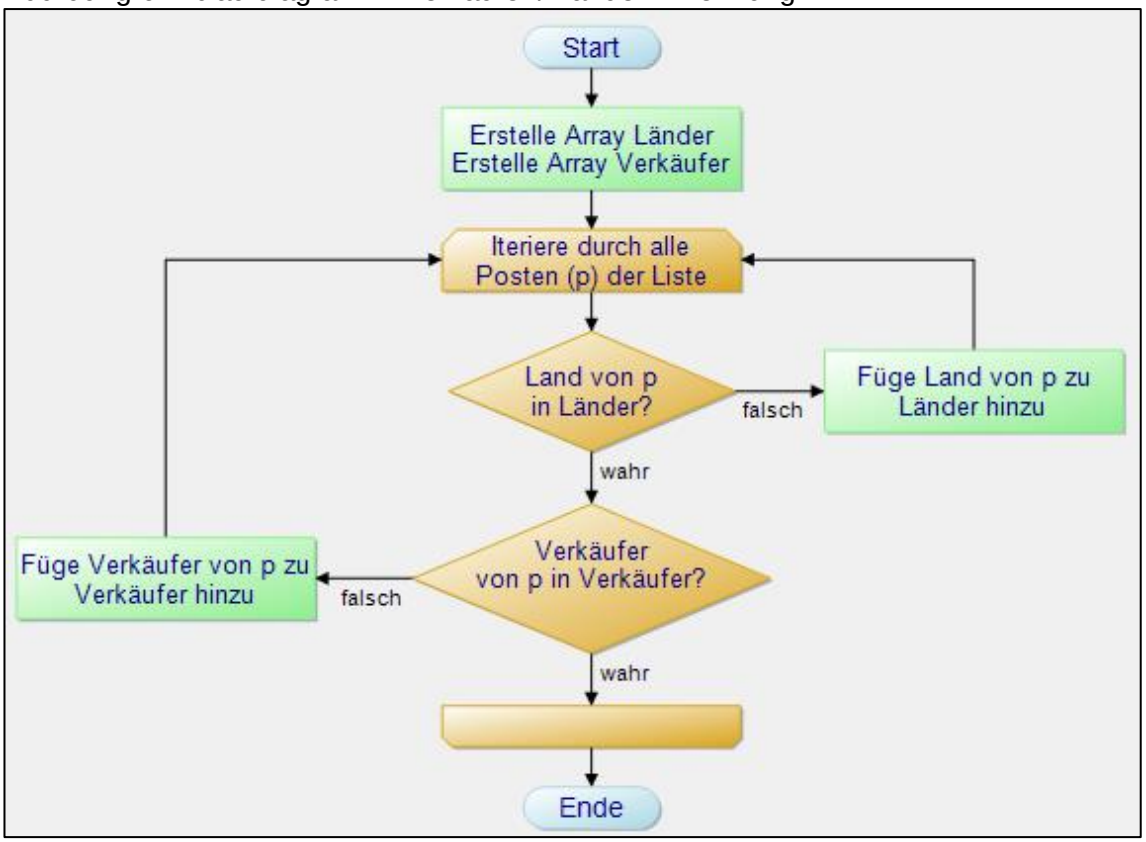
Quelle: eigene Darstellung

Da die Postenbedingungen APG und APL vom Hauptproblem ausgeschlossen werden können und direkte Ausschlusskriterien für das Portfolio darstellen kann diese Prüfung direkt auf die Postenliste angewandt werden und reduziert damit direkt die für eine weitere Verarbeitung zu berücksichtigende Datenmenge.

Erkennung der vorhandenen Länder und Verkäufer:

Als nächstes müssen für eine weitere Prüfung der Posten zuerst alle Verkäufer und Länder welche sich in der gelieferten Postenliste befinden, ermittelt werden. Dies ist notwendig, da diese vorab nicht bekannt sind und sich je nach Liste unterscheiden können. Dies kann dadurch realisiert werden, dass zuerst zwei Teillisten erstellt werden in denen die gefundenen Länder bzw. Verkäufer gespeichert werden. Um die unterschiedlichen Länder bzw. Verkäufer zu finden, wird erneut durch die gesamte Postenliste iteriert und jeder Wert, der noch nicht in einer der Listen vorhanden ist, der entsprechenden Liste hinzugefügt:

Abbildung 6: Ablaufdiagramm: Verkäufer / Länder Erkennung



Quelle: eigene Darstellung

Auf Basis dieser Länder- und Verkäufererkennung können dann die erstellten Arrays um Variablen zur Zählung der jeweiligen Limitausschöpfung erweitert werden. In der folgenden Darstellung wird angenommen dass die Arrays jeweils eine Zählervariable ihres entsprechenden Limits haben.

Greedy Variante:

Die Greedy Variante ist eine Adaption der Greedy-Lösung des Rucksackproblems. Dieses ist, wie in Kapitel 4.2 beschrieben, für eine praktische Nutzung abzuwägen, da es eine sehr schnelle Annäherung bietet, diese Annäherung allerdings sehr ungenau sein kann. Wie zuvor beschrieben wurde, basiert dieser Lösungsansatz auf einem relativen Wert, welcher sich aus dem Quotienten von Wert und Gewicht bildet.

Dies kann auf das vorliegende Portfoliooptimierungsproblem abstrahiert werden. Die limitierende Gewichtung ist hierbei der Wert des Postens, wobei der Realwert, welcher sich aus der Multiplikation von Wert und Rating ergibt, zu maximieren ist.

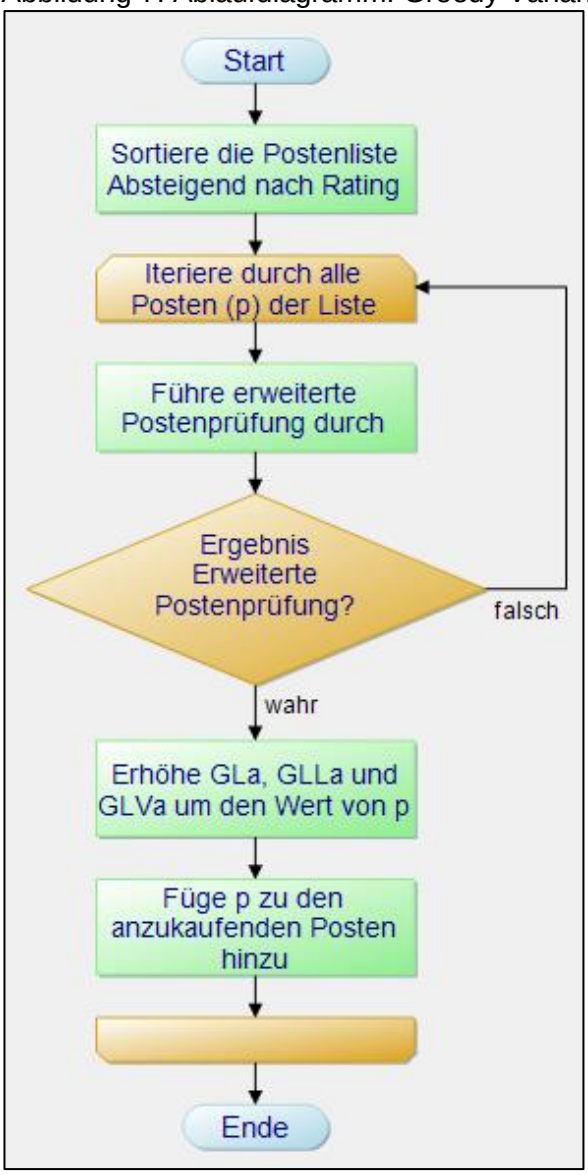
Die Übertragung des Relativen Wertes aus der Greedy-Rucksackproblem-Lösung ergibt hierbei wiederum das Rating, sodass weitere Berechnungen für einen relativen Wert entfallen.

$$relW = W \times R \; / \; W$$

Für den folgenden Programmablaufplan wird eine Vorinitialisierung der Variablen für die Limitauschöpfung: GLa, sowie je Land GLLa und je Verkäufer GLVa mit 0 aus Darstellungsgründen vorausgesetzt. Diese Variablen dienen dazu während eines Schleifendurchlaufs über die Postenliste die jeweilige Ausschöpfung des entsprechenden Limits festzuhalten.

Zuerst findet eine absteigende Sortierung der Postenliste nach Rating-Wert statt. Danach wird für jeden Posten der sortierten Postenliste die erweiterte Postenprüfung durchgeführt. Dies dient dazu festzustellen ob ein Hinzufügen des Postens in das Portfolio möglich ist oder nicht:

Abbildung 7: Ablaufdiagramm: Greedy Variante



Quelle: eigene Darstellung

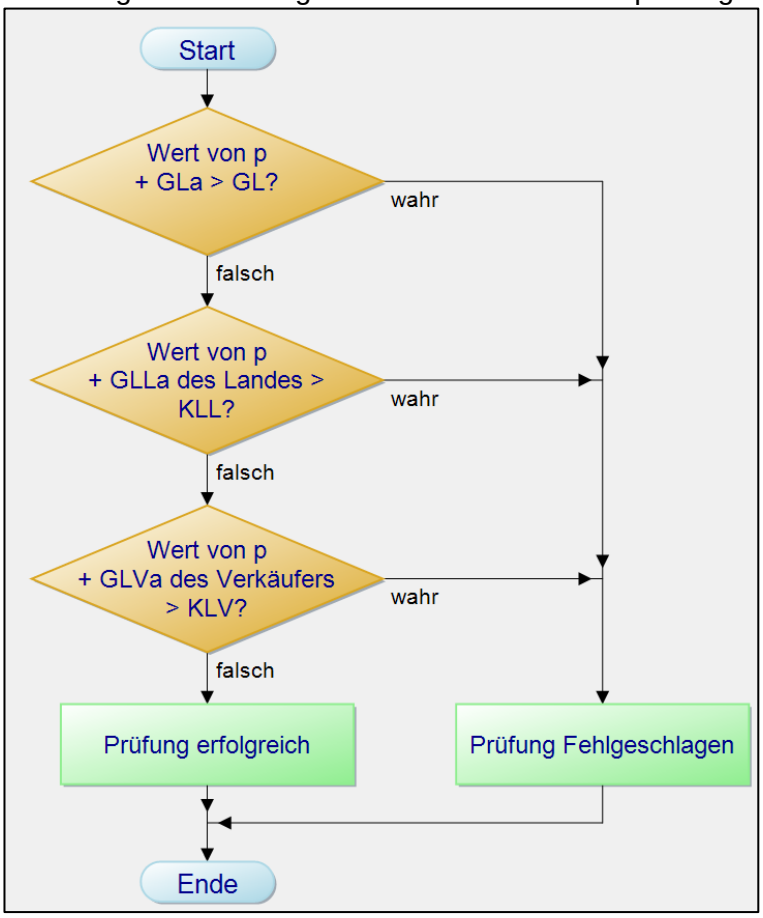
Sobald diese Schleife durchlaufen ist sind alle Posten geprüft worden und das Portfolio „optimal“ gefüllt. Die Einschränkung hierbei ist ein unvermeidlicher Fehler der bei unterschiedlichen „Gewichtungen“ und gleichem relativen Wert, hier dem Rating auftritt.

So besteht die Möglichkeit dass ein Posten mit einem hohen Rating mehr Gesamtlimit verbraucht als mehrere Posten kleineren Wertes mit gleichem Rating.

Da allerdings keine Einschränkung bezüglich einer Verteilung des Postenankaufs gegeben ist wird dieser Fehler später unter „Fazit und Ausblick“ weiter diskutiert.

Die erweiterte Postenprüfung überprüft ob die momentane Ausschöpfung eines Limits bei Hinzufügen des Wertes des Postens das Gesamtlimit oder eines der kombinierten Limite überschreitet:

Abbildung 8: Ablaufdiagramm: Erweiterte Postenprüfung



Quelle: eigene Darstellung

Falls die erweiterte Postenprüfung den Wert „Prüfung erfolgreich“ erreicht, liegt bei Hinzufügen des Postens keine Limitüberschreitung vor.

Daraufhin wird die Ausschöpfung der Limite um den Betrag des nun hinzugefügten Postens erhöht. Der Posten wird daraufhin zur Liste der anzukaufenden Posten hinzugefügt.[30]

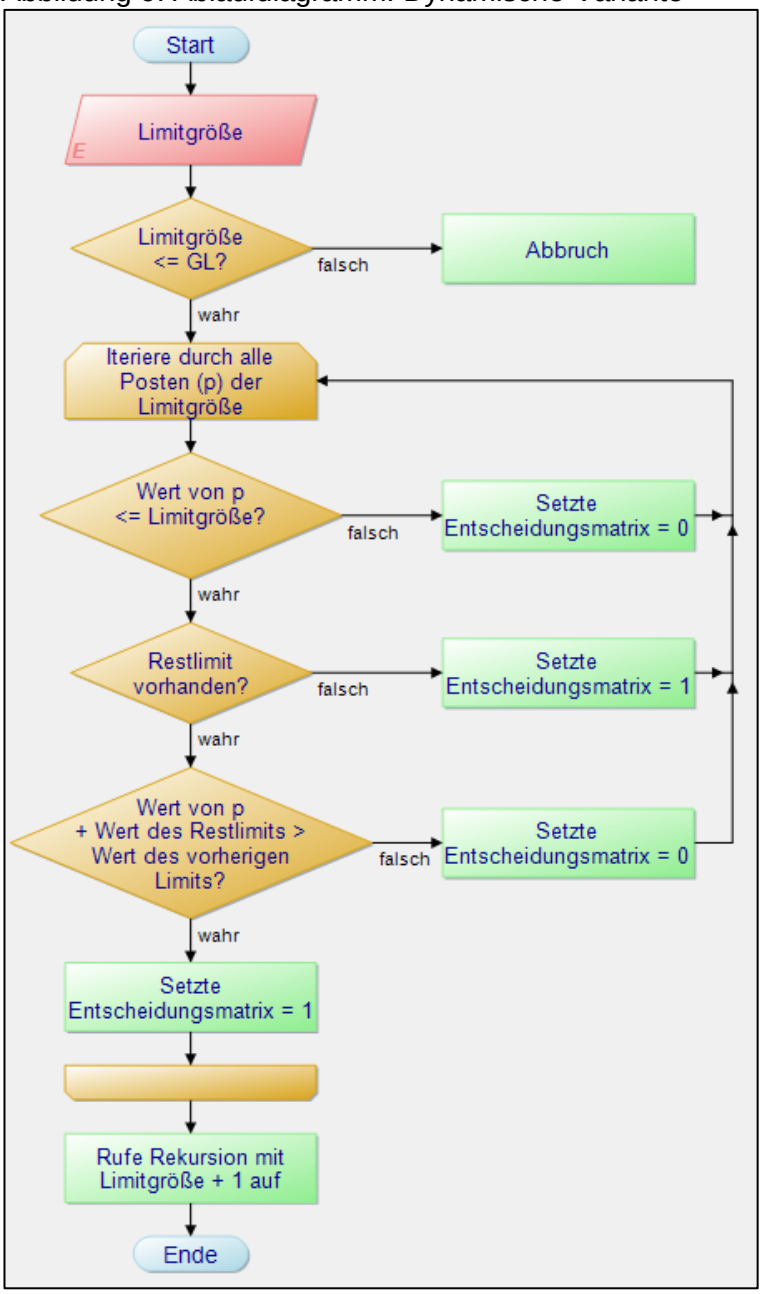
[30] Vgl. Abbildung 7: Ablaufdiagramm: Greedy Variante.

Dynamische Variante:

Die dynamische Variante ist in zwei Teile untergliedert. Zuerst werden zwei Dynamisierungsmatrizen erstellt, welche zweidimensionalen Arrays der Größe Gesamtlimit mal Postenzahl entsprechen.

Diese Arrays werden dann mittels einer rekursiven Funktion, welche Initial mit Limitgröße 1 startet, gefüllt:

Abbildung 9: Ablaufdiagramm: Dynamische-Variante



Quelle: eigene Darstellung, in Anlehnung an Kapitel 3.3

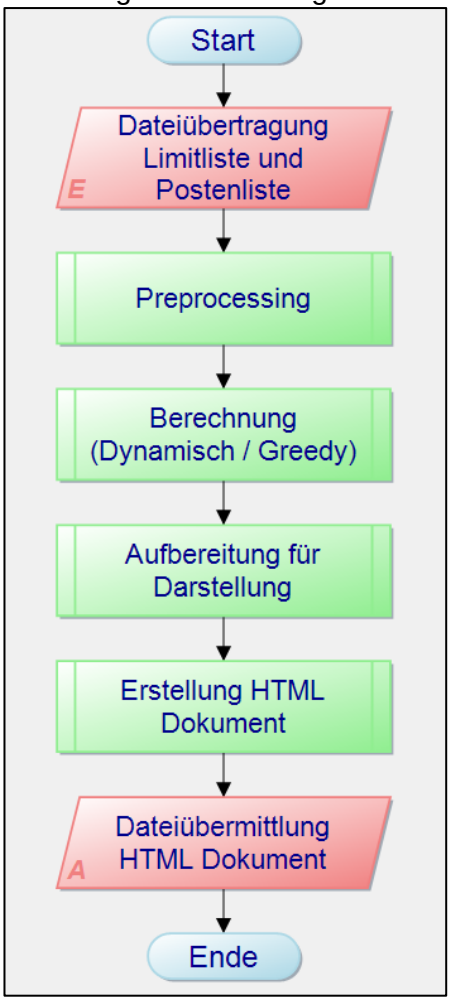
Hierbei wird zuerst geprüft ob das angegebene Limit größer als das Gesamtlimit ist. Dies dient als Abbruchkriterium der Rekursion.
Das gesamte Array dieser Limitgröße wird daraufhin durchlaufen. Dieses steht für jeden Posten der Postenliste. Nun wird für jeden Posten geprüft ob er Teil dieser Limitausschöpfung sein kann. Wenn dem so ist, wird, wie in Kapitel 4.3 beschrieben, evaluiert ob ein Rest besteht und ob der momentane Posten, addiert mit dem errechneten Wert des Restes, größer als die vorherige Berechnung ist.
Dementsprechend wird dann eine 0 oder eine 1 in die Entscheidungsmatrix an der betroffenen Stelle gesetzt (Dies ist aus Darstellungsgründen in der Abbildung abgekürzt mit „Setze Enscheidungsmatrix“). Im Anschluss daran ruft dieser Algorithmus sich selbst mit einer um eins inkrementierten Limitgröße auf, so dass auf diese Art jede Limitgröße berechnet wird.
Die Ermittlung der anzukaufenden Posten ist nun, in Anlehnung an den Lösungsansatz, in Verbindung mit den Portfoliobedingungen möglich. Die Entscheidungsmatrix wird, ausgehend vom Maximalen Limit und der Position des letzten Postens, durchlaufen. Bei jedem Iterationsschritt wird geprüft ob an diese Koordinate eine 1 steht. Ist dies der Fall wird die erweiterte Postenprüfung, gleich der des Greedy-Algorithmus, durchgeführt. Ist eine der beiden Bedingungen nicht erfüllt, wird die neue Limitposition sowie die Postenposition um 1 dekrementiert.
Sind beide Prüfungen erfolgreich, wird analog zur Greedy-Variante, die Ausschöpfung der betroffenen Limite festgehalten und der Posten den Anzukaufenden hinzugefügt. Der nächste Iterationsschritt ist dann für die Position Postenstelle – 1, sowie Limitwert abzüglich des Postenwertes.[31]

[31] Vgl. Anhang 2: Ermittlung der Dynamisch berechneten Posten.

4.2 Aufbau des gesamten Algorithmus

Der Ablaufplan des gesamten Algorithmus setzt sich aus der Aneinanderreihung der Teilalgorithmen zusammen, welche von einem Rahmenprogramm aus aufgerufen werden. Dieses Rahmenprogramm beinhaltet die Kommunikationsschnittstelle zum Nutzer und verwaltet die initiale Dateiübertragung der Postenliste und der Limitliste. Zudem dient es dazu das Ergebnis des Algorithmus zu übermitteln:

Abbildung 10: Ablaufdiagramm: Gesamtplan



Quelle: eigene Darstellung

Hierbei sollen die einzelnen Abschnitte weitestgehend modular aufgebaut werden. Dies kann bspw. durch Kapselung in Funktionen und eine Kommunikation über Schnittstellen realisiert werden.

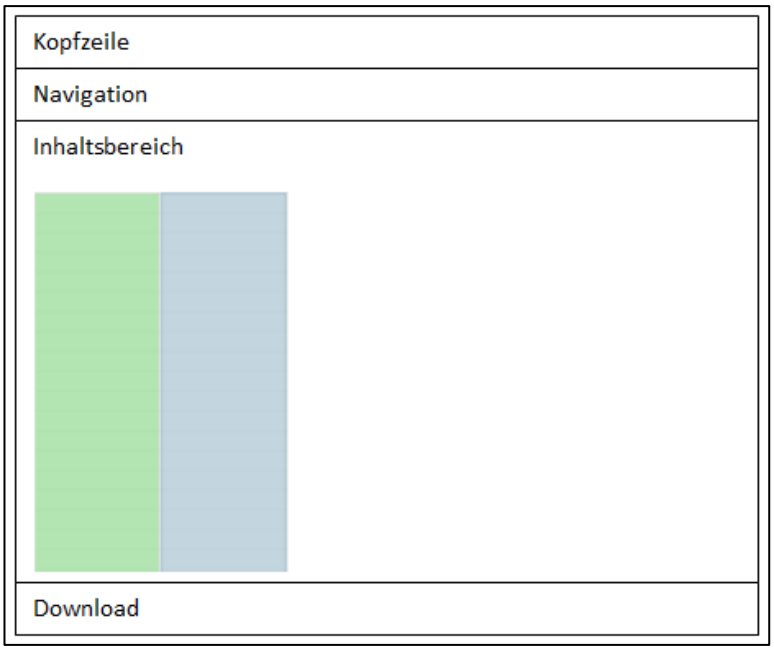
4.3 Darstellung der Ergebnisse

Die Darstellung der Ergebnisse soll, wie unter den Randbedingungen definiert, als HTML Dokument realisiert werden.

Da HTML ein Tag basiertes Dokumentenformat für Webseiten ist und mittels JavaScript der Aufbau eines solchen Dokumentes zu Lasten von Rechenzeit frei veränderbar ist, gibt es kaum Einschränkungen hinsichtlich des Aufbaus des Berichtes.

Die grundlegende Darstellung des Berichtes kann sich somit an einer Traditionellen Webseite orientieren:

Abbildung 11: Konzept: Portfoliobericht Aufbau



Quelle: eigene Darstellung

HTML-Aufbau:

Auf der oberen linken Ecke eine Headline, in dieser wird angezeigt dass dies ein Portfoliobericht ist.

Darunter ein Navigationsbereich, in diesem soll der Nutzer die Möglichkeit haben zwischen einzelnen Formen der Informationsdarstellung zu wählen und einzelne Informationskategorien aufzurufen.

Darunter der Inhaltsbereich in dem die Informationen dargestellt werden. Dieses erfolgt textuell oder grafisch und beinhalt Informationstexte und Diagramme, welche den Kontext der Daten wiedergeben.

JavaScript-Funktionen:

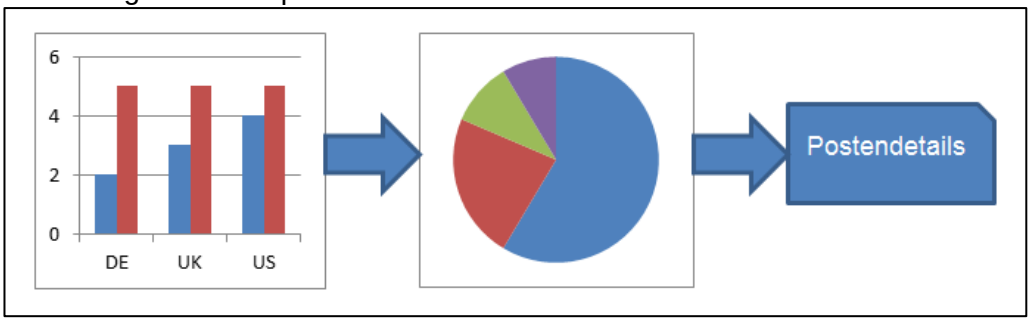
Wie zuvor, in Kapitel 2.4, beschrieben sind bildliche Unterschiede einfach wahrnehmbar. Diese Fähigkeit ist allerdings nicht zufriedenstellend gegeben wenn, wie bei traditionellen Webseiten die Bilder durch Scrollen erreichbar sind. Wichtige Bezugspunkte in Hinsicht auf bspw. die Größe der Grafik gehen durch die Bewegung des Bildes dabei verloren. Um diese Fähigkeit zu unterstützen kann der Inhaltsbereich des Portfolioberichtes in Höhe und Breite fest limitiert werden. Wenn alle Grafiken in einer festen Größe und an einer festen Position angezeigt werden, gehen solche Bezugspunkte nicht verloren.

JavaScript kann dies unterstützen indem sogenannte Tabs genutzt werden. Dadurch wird der gesamte Inhalt des Berichtes „gestapelt“ und der Nutzer kann über Navigation auswählen, welchen dieser Stapel er betrachten möchte. Ein Scrollen wäre dann nur bei langen Texten, innerhalb eines Tabs, erforderlich. Dies kann ebenfalls dazu genutzt werden das Paradigma „Overview / Zoom“ bzw. Filtern umzusetzen, indem die einzelnen Navigationsbereiche vorgefilterte Informationen enthalten, welche jeweils eine Übersicht darstellen. Durch einen Klick auf einen Teil diese Übersicht kann dann ein Filter angewandt werden, der auf Basis dieser Auswahl eine neue Grafik mit detaillierteren Informationen einblende.

Zusätzlich zur Umsetzung dessen kann JavaScript ebenfalls dazu genutzt werden weitere Paradigmen wie „Details-On-Demand“ umzusetzen, indem bei einem Mouseover über einer Grafik zusätzliche Informationen eingeblendet werden.

Somit ergibt sich für den Grafischen Inhaltsbereich folgende Inhaltsdarstellung:

Abbildung 12: Konzept: Informationsmodell



Quelle: eigene Darstellung

5 Umsetzung

In diesem Kapitel wird eine Umsetzung des konzeptionierten Algorithmus beschrieben. Die Umsetzung der Vorverarbeitung, Berechnung und Erstellung der Ausgabe wird mit Node.js umgesetzt. Der Node.js Server dient hierbei als Programm, wobei die Steuerung über den Browser stattfindet.

Zudem wird auf Basis des „Visual Information Seeking Mantras“ eine grafische Darstellung des Outputs erstellt und erläutert wie die einzelnen Schritte dieses Paradigmas mit HTML und JavaScript umgesetzt werden können.

Hier werden nur Ausschnitte der Umsetzung mittels Abbildungen aufgezeigt, der vollständige Quelltext ist im digitalen Anhang enthalten.

Grundlegende Aufbau Schemata von HTML und JavaScript sind hierbei nicht Teil der Arbeit. Der Fokus liegt auf den problemlösenden Teilen der Umsetzung.

5.1 Rahmenprogramm

Das Rahmenprogramm für die Umsetzung ist ein einfacher Node.js Webserver der auf Port 8080 zuhört, so kann die GUI über einen beliebigen Browser mittels http://localhost:8080 aufgerufen werden:

Abbildung 13: Umsetzung: Rahmenprogramm

```
1  var formidable = require('formidable'),
2      http = require('http'),
3      fs   = require('fs-extra');
4
5
6  var initport = 8080;
7
8  var app = http.createServer(function (req, res) {
9
10     |
11
12 }).listen(initport);
13
```

Quelle: eigene Darstellung

Hierbei werden vor Start des eigentlichen Servers die grundlegenden Module importiert welche für eine weitere Interaktion notwendig sind.

So wird http für eine generelle Interaktion mit Browsern benötigt. Das http Modul stellt diese durch Unterstützung des http Protokolls her und erlaubt es mit Browsern zu kommunizieren und stellt die benötigten Header des Protokolls bereit.[32]

Das fs-extra Modul ist eine Erweiterung des Standard fs Moduls von Node.js. Dies bietet durch Import des fs Moduls die Standard Dateioperationen, wie Kopieren und I/O Streams. Zudem sind rekursive Dateioperationen gegeben und ein einfacher Umgang mit den Standard Dateioperationen wird ermöglicht.[33]

Das Modul formidable stellt Funktionen bereit um Formulardaten welche von Browsern mittels POST und GET übermittelt werden zu parsen. Zudem bietet es Funktionen um Datei Uploads zu managen.[34]

Somit ist dieses Modul essentiell notwendig um den Browser als Programm-GUI nutzen zu können, da ein komplexer Funktionen-Overhead zur Verarbeitung von HTML Formularen und Uploads entfällt.

Die Funktion ***http.createServer()*** mit einer Funktionsübergabe als Parameter und den beiden Parametern req und res als Parameter für die übergebene Funktion startet den Server.

Der Parameter req (Request) stellt eine Schnittstelle zu den Anfragedaten dar und res (Response) für die Antwortdaten. Antwortdaten beinhalten den http Header sowie den zu übermittelnden Inhalt.

Eine Verarbeitung der Requests findet innerhalb der übergebenen Funktion statt. Hierbei wird eine Fallunterscheidung getroffen, ob ein eingehendes HTML Formular Teil des Requests ist oder nicht. Falls kein Formular gesendet wurde wird die Programm-GUI als HTML Formular übermittelt. Wenn ein Formular gesendet wurde wird die Verarbeitung der übermittelten Daten begonnen, sobald es vollständig übertragen wurde:

Abbildung 14: Umsetzung: Rahmenprogramm

```
32      form.on('end', function(fields, files) {
33          //Temporary location of our uploaded files
34          var file_items = this.openedFiles[0].path;
35          var file_limits = this.openedFiles[1].path;
```

Quelle: eigene Darstellung

[32] Vgl. o. V./ „http modul“, (o. J.), Online im Internet.
[33] Vgl. o. V./ „fs-extra modul“, (o. J.), Online im Internet.
[34] Vgl. o. V./ „formidable modul“, (o. J.), Online im Internet.

Durch das Event ***form.on('end')*** ist es möglich auf die übertragenen Formularfelder und angehängten Dateien zuzugreifen. Die Dateien werden automatisch in einem temporären Verzeichnis abgelegt und können im lokalen Zugriffsbereich der übergebenen Funktion adressiert werden.

Für das Preprocessing, nach Kapitel 5.1 „Vorverarbeitung" ist es notwendig dass die Posten in einer programminternen Datenstruktur vorhanden sind. Dies ist Aufgabe des Rahmenprogramms. Daher schließt sich an die Adressierung der temporären Dateien das Parsing des CSV Formates an. Die CSV Dateien werden in ein Objekt-Array für die Verarbeitung umgewandelt:

Abbildung 15: Umsetzung: CSV Parsing

```
56          /*
57           * read items available for purchase and construct a datastructure from csv
58           */
59          var items = '';
60          //we got items to process
61          if(file_items)
62              items = fs.readFileSync(file_items,'utf8');
63          //split by lines
64          items = items.split(/\n/);
65
66          //declare object array for records from file
67          var itemlist = new Array();
68
69          //iterate through file
70          for(var i = 0; i < items.length; i++){
71              //strip away quotes
72              items[i] = items[i].replace(/"/g, "");
73              //strip away linebreaks
74              items[i] = items[i].replace('\r','');
75              //split line to get attributes
76              items[i] = items[i].split(',');
77              //ignore blank lines, we only accept complete records
78              if(items[i].length == 5)
79                  //create a record for further processing
80                  itemlist.push({ id:items[i][0],
81                                  w: parseFloat(items[i][1]),//force float parsing
82                                  r: parseFloat(items[i][2]),//force float parsing
83                                  v: items[i][3],
84                                  l: items[i][4]
85                  });
86          }
87          //throw away csv headline, it would interrupt further processing
88          itemlist.splice(0, 1);
```

Quelle: eigene Darstellung

Um die Datei in ein Objekt-Array umzuwandeln wird die übermittelte temporäre Datei zuerst im utf8 Format eingelesen. Dieses Format dient der Unterstützung von Sonderzeichen. Anschließend wird die Datei bei Zeilenumbrüchen aufgetrennt um die einzelnen Datensätze in einem Array zu erhalten. Die Abfrage in Zeile 78 dient dazu sicherzustellen dass nur vollständige Datensätze als Objekt hinzugefügt werden.

5.2 Vorverarbeitung

Die Vorverarbeitung ist in eine Funktion gekapselt und nimmt als Parameter ein Posten-Objekt und ein Limit-Objekt an. Diese Objekte enthalten die jeweils zu Objekt-Arrays aufbereiteten Daten der hochgeladenen Dateien.

Als erstes müssen für die Vorverarbeitung die relativen Postenlimitierungen errechnet werden. Diese sind später dafür ausschlaggebend ob ein Posten vor Portfolioerstellung bereits von der Berechnung ausgeschlossen werden kann:

Abbildung 16: Umsetzung: Limitberechnung

```
382     //calculate relative limitations for preprocessing check
383     var APG = limits.GL     * (limits.RPG / 100);
384     var APL = limits.GLL    * (limits.RPL / 100);
385
386     //perform limit combination
387     //combined country limit
388     var KLL = limits.GL * (limits.RLG / 100);
389     //find the lowest value
390     if(KLL > limits.GLL)
391         KLL = limits.GLL;
392
393     //combined seller limit
394     var KLV = limits.GL * (limits.RVG / 100);
395     //check if GLV is smaller
396     if(KLV > limits.GLV)
397         KLV = limits.GLV;
398     //check if relative limit on seller / country is smaller
399     if(KLV > limits.GLL * (limits.RVL / 100))
400         KLV = limits.GLL * (limits.RVL / 100);
```

Quelle: eigene Darstellung

Die Kombinierten Limitierungen, sowie APG und APL werden, wie in Kapitel 3.2 „Analyse der Teilprobleme“ beschrieben, errechnet, so dass die nachfolgenden Prüfungen vereinfacht werden:

Abbildung 17: Umsetzung: Datenreduktion

```
402     //preprocess item array
403     for(var i = 0; i < items.length; i++){
404         //find the ones that can never be purchased and erase them
405         if(items[i].w > APG || items[i].w > APL){
406             items.splice(i, 1);
407             //reduce counter, we modified the Array index
408             i--;
409         }else{
410         //this item needs to be processed, calculate its real value
411             items[i].rw = (items[i].w * items[i].r).toFixed(2);
412         }
413     }
414     //sort items by rating
415     items.sortByProp('r');
```

Quelle: eigene Darstellung

Hier wird für jeden Posten der kompletten Postenliste abgefragt ob dieser gegen die, zuvor errechneten, Limitierungen APG oder APL verstößt. Ist dies der Fall wird der Posten aus der Postenliste entfernt. Ist der Postenwert unterhalb der Limitierungen wird der Realwert, für spätere Auswertungen berechnet und dem Posten als Eigenschaft hinzugefügt.

Anschließend wird die Postenliste nach dem Rating-Wert absteigend sortiert. Nachdem nun alle Posten, welche nicht Teil des Portfolios sein können entfernt wurden können, die in der Postenliste enthaltenen Länder und Verkäufer identifiziert werden:

Abbildung 18: Umsetzung: Länder / Verkäufer Erkennung

```
425     //iterate through all items and identify how many sellers / countrys we have
426     //and initialize them with 0
427     for(var i = 0; i < items.length; i++){
428         if(!countrys[items[i].l])
429             countrys[items[i].l] = 0;
430         if(!sellers[items[i].v])
431             sellers[items[i].v] = 0;
432     }
```

Quelle: eigene Darstellung

Dies geschieht durch ein weiteres Durchlaufen der Postenliste, wobei hier bei jedem Posten geprüft wird ob in dem jeweiligen Länder- und Verkäuferarray bereits ein Eintrag mit dem dazugehörigen Schlüssel enthalten ist. Der Schlüssel ist hierbei die Länderkennung bzw. der Name des Verkäufers. Wenn dies nicht der Fall ist, wird ein entsprechender Eintrag angelegt und mit 0 belegt, so entsteht ein assoziatives Array nach folgendem Schema: **[‘Land1‘ : 0, ‘Land2‘ : 0, ….]**

Dies dient dazu das später die Limitausschöpfungen über den jeweiligen Namen als Index erreichbar sind und ein komplexes Mapping von Bezeichnern auf Arrayindizes entfällt. Als Rückgabewert wird von der Vorverarbeitung ein Objekt, welches die Verarbeiteten Daten als Arrays in Eigenschaftswerten enthält, zurückgegeben.

Zur Veranschaulichung eine Abbildung der Datenstruktur, die Punkte beziehen sich auf weitere Einträge in den jeweiligen Arrays:

Abbildung 19: Umsetzung: Ergebnis der Vorverarbeitung

```
{ items:
   [ { id: '2', w: 5, r: 1, v: 'A', l: 'DE', rw: '5.00' },
     ................
     { id: '12', w: 5, r: 1, v: 'B', l: 'UK', rw: '5.00' } ],
  limits:
   [ GL: '40',
     ................
     KLV: 20 ],
  countrys: [ DE: 0, UK: 0 ],
  sellers: [ A: 0, B: 0 ],
  itemcount: 8 }
```

Quelle: eigene Darstellung

5.3 **Berechnung (Greedy)**

Die Greedy-Variante des Portfolioalgorithmus nimmt als Parameter die vorverarbeiteten Daten an. Hierbei werden, um die Übersicht zu erhalten, die einzelnen Eigenschafts-Arrays des Parameter-Objektes im lokalen Sichtbarkeitsbereich mit den entsprechenden Bezeichnern neu referenziert. So enthalten items, limits, etc. die entsprechenden Werte der vorverarbeiteten Daten. Zudem wird eine Variable GLa angelegt um die Ausschöpfung des Gesamtlimits festzuhalten und ein Array um die ID der Posten festzuhalten welche angekauft werden sollen.

Abbildung 20: Umsetzung: Greedy Algorithmus

```
278     //iterate through all items
279     for(var i = 0; i < items.length; i++){
280         //check if it would break GL
281         if(!(items[i].w + GLa > limits.GL)){
282             //check if it would break its own KLL
283             if( !(items[i].w + countrys[items[i].l] > limits.KLL)){
284                 //check if it would break its own KLV
285                 if( !(items[i].w + sellers[items[i].v] > limits.KLV)){
286                     //increase global limit exhaustion
287                     GLa += items[i].w;
288                     //increase specific country exhaustion
289                     countrys[items[i].l] += items[i].w;
290                     //increase specific seller exhaustion
291                     sellers[items[i].v] += items[i].w;
292                     //item can be purchased, save its id
293                     itemsToBuy.push(items[i].id);
294                 }
295             }
296         }
297     }
```

Quelle: eigene Darstellung

Die Umsetzung des Programmablaufplanes aus Kapitel 5.1 „Greedy-Variante“ ist ohne Änderungen möglich. Innerhalb einer Schleife werden alle Posten der, absteigend nach Rating-Wert, sortierten Postenliste darauf geprüft ob eine Addition des Postenwertes zur Ausschöpfung eine Überschreitung des Limits verursachen würde. Hierbei wird zuerst auf das Gesamtlimit und damit GLa vs. GL geprüft, da eine Überschreitung des Gesamtlimits keine weitere Prüfung erforderlich macht. Folgend wird geprüft ob eine Überschreitung von GLLa oder GLVa, adressiert über ***countrys[Land des Postens]*** bzw. ***sellers[Verkäufer des Postens]***, bei Addition des Wertes stattfinden würde.

Sobald ein Posten jede dieser Bedingungen erfüllt werden die entsprechenden GLLa und GLVa Variablen um den Wert des Postens erhöht. Zum Abschluss wird die ID des Postens in das Ergebnis-Array eingefügt.

5.4 **Berechnung (dynamisch)**

Die dynamische Variante ist nicht Praktikabel, da sie nicht ohne einen immensen Speicheraufwand umsetzbar ist. Für die Umsetzung wurde zuerst die Vorverarbeitung gestartet und folgend die Dynamisierungsmatrizen für 2187 Posten bei einem Gesamtlimit von 100.000 initialisiert. Dies scheitert allerdings schon bei der Initialisierung:

Abbildung 21: Umsetzung: Dynamischer Algorithmus Initialisierung

```
Server running at 0.0.0.0:8080
27.03
53.93
100.00
Preprocessing took: 6 miliseconds
Preprocessing cut: 2187 down to: 2187 items, thats: 100.00% of the original valu
e left
Greedy got: 190 items
Greedy selection took: 2 miliseconds
Totaltime: 8 miliseconds
FATAL ERROR: CALL_AND_RETRY_2 Allocation failed - process out of memory
```

Quelle: eigene Darstellung

Zurückzuführen ist dieser Abbruch auf den Speicherbedarf dieses Algorithmus. Die Initialisierung von Wertmatrix und Entscheidungsmatrix würde im optimalen Fall, ohne Verwaltungsoverhead der Variablen, bei einfachen Boolean Variablen, bei diesem Test, folgendes betragen:

$$\frac{100.000 \times 2178 \times 2}{2^{23}} = 51{,}9\ MB$$

Dieser Speicherbedarf wird allerdings weit überschritten da JavaScript einen Verwaltungsoverhead aufgrund seiner schwachen Typisierung hat.

5.5 Aufbereitung

Da die grafische Darstellung, angelehnt an das Information Seeking Mantra Umgesetzt wird, ist eine Aufbereitung der errechneten Daten notwendig. Dies ist darauf zurückzuführen, dass für die Darstellung Chart.js genutzt wird. Auf Chart.js wird im Folgekapitel näher eingegangen.

Chart.js benötigt Daten für die Erzeugung von Balkendiagrammen in Form von separaten Arrays, jeweils eines für die anzuzeigenden Titel, eines welche die aktuellen Balkenwerte und eines welches die Maximalwerte der Balken enthält.

Um diese Arrays bereitzustellen werden die berechneten Daten des Portfolioalgorithmus aufbereitet:

Abbildung 22: Umsetzung: Datenaufbereitung, Array Auftrennung, GLLa

```
446     //split GLLa into three arrays, one containing names, one exhaustions
447     var cnames = new Array();
448     var cexhaustions = new Array();
449     var climits = new Array();
450     for(var name in processedData.exhaustions.GLLa){
451         //exclude array addons from split
452         if(name != 'sortByProp' && name != 'contains'){
453             cnames.push(name);
454             cexhaustions.push(processedData.exhaustions.GLLa[name].toFixed(2));
455             climits.push(processedData.limits.GLL);
456         }
457     };
458
459     //construct GLLa datasets for Charts.js
460     var gllaData =  {   labels : cnames,
461                         exhaustions : cexhaustions,
462                         limits : climits
463                     };
```

Quelle: eigene Darstellung

Die GLLa und GLVa Werte des Objektes welches vom Algorithmus erstellt wurde werden innerhalb einer Schleife „aufgetrennt“. Es gibt jeweils ein Array für jede der von Chart.js benötigten Information. Innerhalb der Schleife werden die Namen, bzw. Länder, deren GLLa, bzw. GLVa Wert und das jeweilige maximale Limit dem entsprechenden Array hinzugefügt. Anschließend daran werden diese drei Arrays für je GLLa und GLVa in einer Objektstruktur zusammengefasst. Dies dient später dazu diese Daten übersichtlich im HTML Dokument einzubetten.

Zusätzlich ist es erforderlich die Posteninformationen des Portfolios aufzubereiten. Zuvor wurde vom Berechnungsalgorithmus ein Array erzeugt, in welchem die Posten-ID eines jeden Postens gespeichert wurde der Teil des berechneten Portfolios ist. Dies ist

notwendig, um eine strikte Trennung von Darstellung und Berechnung möglich zu machen. Das Ergebnis wird bereits der Liste dieser IDs vollständig repräsentiert. Für einen Portfoliobericht, in welchem nach den Prinzipien des Information Seeking Mantras ein Zoomen möglich sein soll und „Details-On-Demand“ verfügbar sein sollen ist es allerdings notwendig, dass die grundlegende Informationseinheit Posten ebenfalls zu berücksichtigen ist.
Um dies zu realisieren werden die vollständigen Daten eines jeden Postens benötigt der Teil des berechneten Portfolios ist:

Abbildung 23: Umsetzung: Aufbereitung der Posten

```
485     var boughtItems = new Array();
486     var boughtRW = 0;
487     //iterate through the whole itemList, as it contains all information about items
488     for(var i = 0; i < itemList.length; i++){
489         //check if the iterator is also in the processed item list
490         if (processedData.items.contains(itemList[i].id) == true) {
491             //add this item to bought items and increase real value accordingly
492             boughtItems.push(itemList[i]);
493             boughtRW += parseFloat(itemList[i].rw);
494         }
495     }
```

Quelle: eigene Darstellung

Dadurch dass während der Datenaufbereitung zum einen die vollständige Postenliste sowie die ID-Liste der angekauften Posten adressierbar sind, ist es möglich innerhalb einer Schleife, welche die gesamte Postenliste durchläuft, zu prüfen, ob die ID des Postens Teil der angekauften Liste ist. Sobald dies der Fall ist, wird der vollständige Datensatz dieses Postens einem Array hinzugefügt, welches wiederum später im HTML Dokument eingebettet wird. Zugleich wird während des Schleifendurchlaufs aufsummiert, welcher Realwert angekauft wurde.

5.6 Grafische Darstellung

Nach der Datenaufbereitung des vorherigen Kapitels liegt nun folgende Datenstruktur vor:

Abbildung 24: Umsetzung: Datenstruktur Aufbereitete Daten

```
{ limitData: { labels: 'Gesamtlimit', exhaustions: 40, limits: 40 },
gllaData:
 { labels: [ 'DE', 'UK' ],
   exhaustions: [ '20.00', '20.00' ],
   limits: [ '20', '20' ] },
glvaData:
 { labels: [ 'A', 'B' ],
   exhaustions: [ '20.00', '20.00' ],
   limits: [ '20', '20' ] },
boughtItems:
 [ { id: '2', w: 5, r: 1, v: 'A', l: 'DE', rw: '5.00' },
   ...........
   { id: '12', w: 5, r: 1, v: 'B', l: 'UK', rw: '5.00' } ],
boughtRW: '40.00' }
```

Quelle: eigene Darstellung

Es gibt zwei Arrays welche direkt für eine Darstellung als Diagramm geeignet sind und ein Array, welches alle Daten über die angekauften Posten enthält. Das Array der angekauften Posten kann nicht weiter vorverarbeitet werden, da die genaue Informationsdarstellung vom Nutzer gewählt wird. So würde es zu einer Datenredundanz führen, wenn die angekauften Posten in vorsortierten Arrays für jede Auswahl des Nutzers vorliegen würden. Der Rechenaufwand der Darstellung kann somit in das HTML Dokument ausgelagert werden.

Die Erstellung des Dokuments erfolgt in einzelnen Schritten vom Rahmenprogramm, indem die benötigten Scripte vom Dateisystem eingelesen und innerhalb von Script-Tags aneinandergereiht werden. Das gleiche geschieht ebenfalls mit den CSS Dateien, welche für das Seitenlayout genutzt werden. So entsteht folgender Aufbau:

Abbildung 25: Umsetzung: HTML Struktur

```
<html>
<head>
    <script> +Chart.js+ </script>
    <script> +SimpleTabs.js+ </script>
    <style> +SimpleTabsStyle.css+ </style>
    <style> +PortfolioStyle.css+ </style>
</head>
    <body>

    <-- HTML Markup für Simpletabs mit -->
    <-- Canvas Elementen für Grafiken  -->

    <script> +DisplayControl.js+ </script>
</body>
</html>
```

Quelle: eigene Darstellung

Zuerst werden die für die Darstellung benötigten Scripte Chart.js und SimpleTabs.js in die Head Sektion des HTML Dokumentes geschrieben. Da diese vor jeglichem Script, welches darauf zugreift, vom Browser geparst werden müssen.

Als nächstes werden die CSS Dateien in Style-Tags übertragen, damit, bereits vor Übersenden von Layout bedingten HTML-Tags deren Darstellungsinformationen vorliegen. Darauf folgend wird das für den Seitenaufbau benötigte Markup erstellt und übertragen. Dieses besteht aus den Div Elementen und Listeninformationen, welche für SimpleTabs benötigt werden, um einen Seitenaufbau mittels Tabs zu ermöglichen. Hierbei wird zugleich der Inhalt der Tabs mit übertragen. Dies ist für den Tab „Portfolioinformationen“ ein Informationstext, welcher Eckdaten wie die Anzahl der Portfolioposten, die erreichte Ausschöpfung und den berechneten Realwert beinhaltet. Die anderen Tabs enthalten als Inhalt jeweils ein Canvas Element, um Diagramme Zeichnen zu können und ein Div Element, um die Legende eben dieser Diagramme anzuzeigen. Unter den für die Tabs benötigten Elemente ist ein Link, welcher keine Verknüpfung und nur eine Beschriftung enthält.

Nachdem diese Daten übertragen wurden wird die eingangs beschriebene Datenstruktur übertragen. Dies wird dadurch realisiert, dass Variablendeklarationen serverseitig in Strings geschrieben werden und diese zusammen mit der JSON Notation der jeweiligen Objekte bzw. Objektarrays zu vollständigen Scriptzeilen verkettet werden. Diese Variablen werden dann zusammen mit dem eigentlichen Script zur Nutzerinteraktion übertragen. (DisplayControl.js)

DisplayControl.js

Die Steuerung der Interaktion des Dokumentes gliedert sich in verschiedene Funktionen. Als erstes wird Chart.js konfiguriert. Animationen werden abgeschaltet, da diese für die darzustellende Menge an Informationen nicht zuverlässig funktionieren.

Als nächstes werden Adressierungsvariablen für die Canvas Elemente gesetzt und die im vorherigen Kapitel beschriebenen Arrays zu DataSets für Chart.js aufbereitet. Diese Aufbereitung besteht daraus dass Farbinformationen und Beschriftungen für die Balken hinzugefügt werden. Anschließend daran werden die Übersichtsdiagramme mittels Chart.js erzeugt und die dazugehörigen Legenden in die dafür im Markup vorgesehen Elemente eingefügt.

Die Download Funktionalität wird ebenfalls bei der Initialisierung bereitgestellt indem der Referenzparameter des Links, welcher vorher ohne Funktion war, durch eine URI kodierte Datei ersetzt wird und die dazugehörigen Attribute auf Download, den Dateinamen und die Kodierung text/csv gesetzt werden.

Die URI Kodierung geschieht mittels der Standard JavaScript Funktion ***encodeURI-Component()***, welche durch die Funktion ***createCSV()*** die zu kodierende Datei erhält. Diese Datei wird aus der, im HTML Dokument als Array vorhandenen, Postenliste des Portfolios erstellt:

Abbildung 26: Umsetzung: CSV Erstellung

```
183     //construct CSV headline at initialization
184     var CSV = '"ID","Wert","Rating","Verkäufer","Land"\n';
185
186     //iterate through all bought items
187     for (var i = 0; i < boughtItems.length; i++) {
188         //reference for comprehensive usage
189         var item = boughtItems[i];
190         //create a new CSV row
191         CSV += '"'+ item.id+'", "'+item.w+'", "'+
192                     item.r+'", "'+item.v+'", "'+item.l+'"\n';
193     }
```

Quelle: eigene Darstellung

Die Funktion ***createCSV()*** konstruiert aus dem Array der Portfolioposten einen String welcher im CSV-Format kodiert ist, indem zuerst eine Kopfzeile erstellt wird. Anschließend daran werden in einem Schleifendurchlauf alle Eigenschaften eines Postens in eine CSV Zeile umgewandelt. Diese werden wiederum miteinander verkettet. So entsteht ein String der dem Aufbau einer CSV Datei entspricht. Als Funktionalität steht

nun so ein Download-Link zur Verfügung welcher die Postenliste des Portfolios als CSV-Datei anbietet.

Die Interaktionssteuerung wird dadurch realisiert dass beide Canvas Elemente je einen „onclick“ Event Listener zugewiesen haben und zwei Variablen den Status der jeweiligen Diagramme speichern, deren Initialwert false ist. Sobald einer der Event Listener einen Klick registriert wird abgefragt ob die dazugehörige Statusvariable false ist. Wenn dies der Fall ist befindet sich das Diagramm im Übersichtsmodus und soll nun in die Detailansicht wechseln.

Die Detailansicht wird erstellt indem zuerst mittels der Chart.js Funktionalitäten ermittelt wird welcher Balken angeklickt wurde. Diese Information wird dann, zusammen mit der Information welches Diagramm betroffen war, an die Postensuche übergeben:

Abbildung 27: Umsetzung: Postensuche

```
115     //iterate through all bought items and collect the ones matching
116     var foundItems = new Array();
117     for(var i = 0; i < boughtItems.length; i++){
118         //determine if this items property matches desired value
119         if(boughtItems[i][option.prop] == option.value)
120             foundItems.push(boughtItems[i]);
121     }
```

Quelle: eigene Darstellung

Diese ermittelt in einem Schleifendurchlauf, welcher alle Posten des Portfolios durchläuft, alle Posten die den Suchkriterien ***[Übersichtseigenschaft] = [gewünschter Wert]*** entspricht und speichert diese in einem Array. Dieses wird von der Funktion anschließend zu einem DataSet aufbereitet und zurückgegeben.

Daraufhin wird das existierende Chart (Diagramm) zerstört und mittels des neuen DataSets das „DetailChart“ erstellt. Dieses ist ein Doughnut Diagramm, da das ursprüngliche Kreisdiagramm der Konzeption aufgrund der Linienverläufe bei größeren Datenmengen nicht mehr übersichtlich ist.

Die Legende des Charts hat durch das „DetailChart“ keine Funktion mehr. Der Platz wird dazu genutzt, eine Übersicht anzuzeigen, in welcher Detailansicht sich der Nutzer befindet, und einzublenden, wie er wieder in die Übersicht gelangen kann.

Nachdem die neue Legende erstellt wurde, wird die Statusvariable auf true gesetzt und der Übergang zur Detailansicht ist abgeschlossen.

Im Fall das die Variable bei einem „onclick“ Event true ist, wird erneut das existierende Chart zerstört und ein neues erstellt. Dieses wird mit den bei der Initialisierung aufbe-

reiteten Daten erstellt und die dazugehörige Legende ebenso wie bei der Initialisierung. Anschließend wird die Statusvariable wieder auf false gesetzt. Somit ist die initiale Übersicht wiederhergestellt.

Die Funktionalität „Details-On-Demand“ wird dadurch bereitgestellt dass für die DetailCharts, welche als Doughnut-Diagramme vorhanden sind, die Tooltip Funktionen von Chart.js genutzt werden. Hier ist keine weitere Umsetzung notwendig, da die einzelnen Datensätze vorher bereits bei der Postensuche entsprechend aufbereitet werden:

Abbildung 28: Umsetzung: Aufbereitung der gefundenen Posten

```
126     var colorswitch = '#C8C8C8';
127     for(var i = 0; i < foundItems.length; i++){
128         var labeltext = '';
129         //determine which labeltext to use for tooltips
130         if(option.prop == 'l')
131             labeltext = 'Item: '+foundItems[i].id+' angeboten von: '+
132                         foundItems[i].v+'\n Wert: ';
133         else
134             labeltext = 'Item: '+foundItems[i].id+' angeboten aus: '+
135                         foundItems[i].l+'\n Wert: ';
136
137         //add item to dataset
138         detailDataset.push({
139                 value: foundItems[i].w,
140                 color: colorswitch,
141                 highlight: "#FF5A5E",
142                 label: labeltext
143         });
```

Quelle: eigene Darstellung

Die Datensätze werden, je nachdem aus welcher Suche sie stammen, entsprechend aufbereitet. So wird bei dem DetailChart der Länderübersicht der Verkäufer bei der Aufbereitung hinzugefügt und bei der Verkäuferübersicht das Land.

Die Anzeige dieser Informationen wird daraufhin von Chart.js übernommen, bei einem Mouseover über die Segmente des Doughnut Diagramms werden diese Texte als Tooltips eingeblendet[35].

[35] Vgl. Anhang 3: Beispielportfolio: Tooltips.

5.7 **Ergebnisse**

Im Folgenden werden die Ergebnisse der Umsetzung präsentiert und erläutert. Hierbei wurden zwei unterschiedliche Limitlisten und fünf Postenlisten mit Zufallsposten genutzt. Die vollständigen Testdaten, die tabellarische Analyse der Testdaten inklusive der dazugehörigen HTML Portfolioberichte, sind im digitalen Anhang einzusehen. Um die Berechnung zu testen wurden zwei Limitlisten mit folgenden Einträgen erstellt:

Tabelle 7: Ergebnisse: Limitliste 1

GL	GLL	GLV	RLG	RLV	RVG	RPG	RPL
200.000	30.000	25.000	35	60	26	15	10

Quelle: eigene Darstellung

Tabelle 8: Ergebnisse: Limitliste 2

GL	GLL	GLV	RLG	RLV	RVG	RPG	RPL
300.000	32.000	20.000	25	50	30	20	25

Quelle: eigene Darstellung

Die Limitierungen wurden so gewählt, dass möglichst unterschiedliche Ergebnisse daraus resultieren. So ist zum Beispiel die Spanne zwischen GLL und GLV in Limitliste 1 mit 5.000 weitaus kleiner als 12.000 bei Limitliste 2. Die Postenlisten sind so gewählt, dass Engpässe bzgl. der Ausschöpfungsmöglichkeiten entstehen, so enthält Postenliste 5 nur vier Länder, die anderen allerdings neun. Diese Maßnahmen bilden damit keine praxisnahen Daten ab. Allerdings kann der Algorithmus so umfangreicher getestet werden:

Tabelle 9: Ergebnisse: Berechnung Limitliste 1

Testposten	GLa	KLLa	KLVa	Entfernt	Selektiert
Liste 1	98.428,53	36,46%	98,42%	7.976	51
Liste 2	172.092,52	63,74%	98,34%	8.065	89
Liste 3	199.599,76	73,93%	88,71%	7.997	97
Liste 4	199.749,86	73,98%	88,78%	4.013	103
Liste 5	119.330,54	99,44%	44,19%	4.015	62

Quelle: eigene Darstellung in Anlehnung an Testdatenanalyse

Die Ergebnisse für Limitliste 1 erreichen eine Ausschöpfung von durchschnittlich 99% bei den am weitesten einschränkenden Limitierungen. So wurde das Gesamtlimit bei den Listen drei und vier maximal ausgeschöpft. Bei den übrigen Listen war dies nicht möglich, da entweder das kombinierte Länder- oder Verkäuferlimit maximal ausgeschöpft wurde. Ein Ankauf darüber hinaus ist logischerweise nicht möglich. Hierbei ist zudem auffällig, dass die niedrig angesetzten Limitierungen von RPL und RPG eine starke Reduktion der zu verarbeitenden Daten bewirken.

Dieser Zusammenhang ist durch die grafische Darstellung in den HTML Portfolioberichten ebenfalls einfach nachzuvollziehen, da das Balkendiagramm „Limitausschöpfung nach Land“ eine maximale Ausschöpfung angibt[36]. Dem entgegen zeigt das Balkendiagramm „Limitausschöpfung nach Verkäufer“ keinerlei Ausschöpfung nahe des Maximums[37].

Ein ähnliches Ergebnis liefert Limitliste 2 in Kombination mit den gleichen fünf Testpostenlisten:

Tabelle 10: Ergebnisse: Berechnung Limitliste 2

Testposten	GLa	KLLa	KLVa	Entfernt	Selektiert
Liste 1	78.209,00	27,16%	97,76%	3.023	20
Liste 2	136.114,24	47,26%	97,22%	2.950	32
Liste 3	176.876,70	61,42%	98,26%	3.042	48
Liste 4	175.923,29	61,08%	97,74%	1.535	47
Liste 5	125.817,30	98,29%	69,89%	1.492	34

Quelle: eigene Darstellung in Anlehnung an Testdatenanalyse

Diese Portfolioberechnungen haben aufgrund eines höher angesetzten Gesamtlimits eine weitaus höhere Datenmenge nach der Vorverarbeitung zur Folge. Aufgrund der größeren Differenz zwischen GLL und GLV wird hier keine maximale Ausschöpfung des Gesamtlimits erreicht. Die Ausschöpfung der dominierenden Limitierung ist dennoch maximal (KLL bei Liste 5 und KLV bei den restlichen Listen). Durchschnittlich wurden 98% Ausschöpfung des dominanten Limits erreicht.

[36] Vgl. Anhang 4: Beispielportfolio: Limitausschöpfung nach Land.

[37] Vgl. Anhang 5: Beispielportfolio: Limitausschöpfung nach Verkäufer.

6 Fazit und Ausblick

Das Ziel dieser Arbeit war das Konzipieren und Umsetzen von zwei Algorithmen zur Zusammenstellung eines Portfolios von Posten welches unter einschränkenden Limitierungen, hinsichtlich seiner Komposition, eine Maximierung des Realwertes erreicht. Zudem sollte das Ergebnis dieser Algorithmen grafisch so aufbereitet werden, dass die Zusammenhänge der Limitierungen sowie wichtige Eckdaten ersichtlich sind. In diesem Rahmen wurden zuerst die Randbedingungen, unter denen die Problemstellung gelöst werden soll, formuliert und definiert. Anschließend wurden Definitionen und Notationen vorgestellt, wie derartige Optimierungsprobleme definiert sind und wie Algorithmen, die diese lösen, vergleichbar werden. Hierbei wurde auch darauf eingegangen welche grundlegenden Probleme bei dieser Art von Optimierung auftreten können und wie diese, mittels Approximation, Umgangen werden können.

Hierbei wird bereits bei der Formalisierung der Problemstellung erkennbar dass es sich bei dem vorgestellten Portfolioproblem um ein kombinatorisches Optimierungsproblem mit einer Vielzahl von Nebenbedingungen handelt. Um dieses einfacher zu abstrahieren und auf bereits bekannte Lösungsverfahren übertragen zu können, wurde das vorliegende Problem analysiert. Dadurch konnten einige Nebenbedingungen des Problems aus dem Optimierungsalgorithmus eliminiert und in eine Vorverarbeitung ausgelagert werden. Übrig blieb ein Problem, welches dem Rucksackproblem mit zwei Nebenbedingungen entspricht. Zudem wurden Paradigmen erläutert, wie die spätere Darstellung des Portfolios dem gesetzten Ziel gerecht werden kann.

Im Anschluss an die Problemanalyse wurden etablierte Lösungsmethoden für das Rucksackproblem vorgestellt. Hierbei wurde eine Vorauswahl bezüglich der Methoden getroffen, um eine dynamische und eine heuristische Lösung mit relativem Wert (gierige-Heuristik) als Annäherung zu erhalten. Diese Lösungsmethoden wurden daraufhin abstrahiert und auf das zuvor reduzierte Problem angewandt. So entstanden zwei Algorithmen welche das vorliegende Problem lösen. Zudem wurde ein Konzept erarbeitet wie die vorgestellten Paradigmen der Darstellung, innerhalb der Randbedingungen, umgesetzt werden können.

Die, an die Konzeptionierung anschließende, Umsetzung erläutert wie die beiden Algorithmen mittels JavaScript umgesetzt wurden. Es dient ein Node.js Server als berechnendes Programm und ein Browser, welcher sich mit diesem verbindet, als grafische Benutzeroberfläche. Die Umsetzung des gierigen Algorithmus wurde hier vollständig

erläutert. Die Umsetzung des dynamischen Ansatzes wurde abgebrochen, da der Speicheraufwand in diesem Fall zu groß ist, um praxisnahe Datenmengen verarbeiten zu können. Die gierige Annäherung, welche bei der Vorstellung der Lösungsansätze aufgrund ihrer Ungenauigkeit, als „unter praxisbezogenen Bedingungen“ abzuwägen galt, kann als praxistauglich eingestuft werden, da die Ausschöpfungen des am meisten einschränkenden Limits im Durchschnitt 98%, unter Berücksichtigung der Postenmenge erreicht.

Die zuvor vorgestellten Paradigmen der Darstellung von Massendaten wurden ebenfalls umgesetzt und deren Realisierung mit JavaScript sowie die Aufbereitung der Daten für die Darstellung mit Diagrammen erläutert. Der Informationszusammenhang bleibt hierbei erhalten und die wichtigsten Eckdaten sind direkt ersichtlich. Allerdings ist die Skalierbarkeit bzgl. vielen Verkäufern und Ländern nicht gegeben[38]. Abschließend zum Fazit können die gesetzten Ziele als teilweise gelöst angesehen werden. Es wurden zwei Algorithmen entworfen, welche den Zielsetzungen gerecht werden, wovon allerdings nur einer praxistauglich ist und umgesetzt wurde. Zudem wurde die grafische Darstellung realisiert und erfüllt die gesetzten Anforderungen.

Ausblick

Bei einer abschließenden Betrachtung des Ergebnisses ist ersichtlich dass eine Vielzahl von Möglichkeiten besteht die vorliegende Lösung zu erweitern und zu verbessern.

So lässt sich zum Beispiel der gierige Algorithmus verbessern, indem die Vorverarbeitung der Daten nicht nur eine einfache Sortierung absteigend nach dem Rating-Wert eines Postens vornimmt, sondern zusätzlich nach der Nebenbedingung des Postenwertes sortiert. Damit kann die Ungenauigkeit dieses Algorithmus reduziert werden.

Des Weiteren ist ein neues Konzept bzgl. des dynamischen Algorithmus denkbar, eine Auslagerung der Dynamisierungsmatrizen in bspw. Dateistreams, könnte diesen umsetzbar machen.

Die grafische Darstellung ist ebenfalls erweiterbar. So ist es möglich die Paradigmen, welche genutzt wurden, anders anzuwenden. Wenn weitere Zwischenschritte bei der Selektion oder andere Diagrammformen zum Einsatz kommen, ist es möglich, die Skalierbarkeit bei großen Verkäufer und Ländermengen zu realisieren.

[38] Vgl. Anhang 6: Länderskalierung.

Anhang

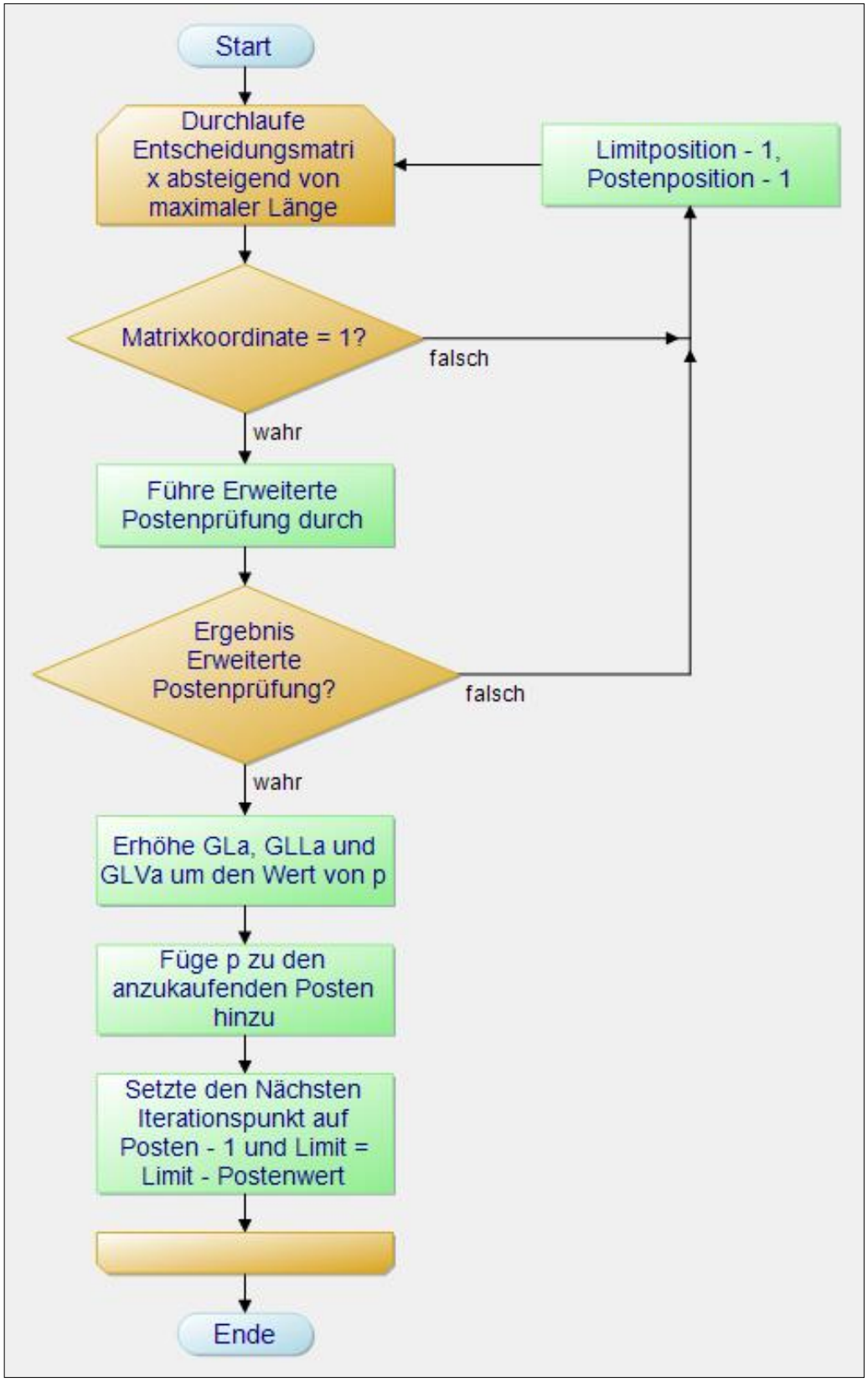
Anhangsverzeichnis

Anhang 1: DAX Tageshoch Werte nach Monaten

Januar		Februar		März		April		Mai	
Datum	Tageshoch	Datum	Tageshoch	Datum	Tageshoch	Datum	Tageshoch	Datum	Tageshoch
31.01.2014	9.346,79	28.02.2014	9.692,08	31.03.2014	9.634,82	30.04.2014	9.618,98	27.05.2014	9941,79
30.01.2014	9.415,48	27.02.2014	9.672,82	28.03.2014	9.587,19	29.04.2014	9.596,42	26.05.2014	9893,81
29.01.2014	9.537,63	26.02.2014	9.720,66	27.03.2014	9.469,39	28.04.2014	9.496,84	23.05.2014	9779,59
28.01.2014	9.427,59	25.02.2014	9.710,95	26.03.2014	9.488,74	25.04.2014	9.501,91	22.05.2014	9734,14
27.01.2014	9.402,72	24.02.2014	9.708,94	25.03.2014	9.372,07	24.04.2014	9.645,06	21.05.2014	9709,91
24.01.2014	9.664,24	21.02.2014	9.666,15	24.03.2014	9.358,95	23.04.2014	9.607,84	20.05.2014	9685,56
23.01.2014	9.728,78	20.02.2014	9.618,85	21.03.2014	9.376,94	22.04.2014	9.602,57	19.05.2014	9676,52
22.01.2014	9.765,63	19.02.2014	9.695,86	20.03.2014	9.296,90	17.04.2014	9.417,82	16.05.2014	9670,90
21.01.2014	9.794,05	18.02.2014	9.690,97	19.03.2014	9.325,93	16.04.2014	9.318,97	15.05.2014	9810,29
20.01.2014	9.733,40	17.02.2014	9.682,19	18.03.2014	9.315,07	15.04.2014	9.344,85	14.05.2014	9772,09
17.01.2014	9.789,89	14.02.2014	9.677,53	17.03.2014	9.197,81	14.04.2014	9.339,17	13.05.2014	9783,72
16.01.2014	9.747,37	13.02.2014	9.600,64	14.03.2014	9.094,24	11.04.2014	9.390,44	12.05.2014	9710,34
15.01.2014	9.747,40	12.02.2014	9.594,85	13.03.2014	9.226,96	10.04.2014	9.581,48	09.05.2014	9602,86
14.01.2014	9.540,51	11.02.2014	9.478,77	12.03.2014	9.267,10	09.04.2014	9.542,31	08.05.2014	9622,30
13.01.2014	9.519,30	10.02.2014	9.346,13	11.03.2014	9.375,29	08.04.2014	9.525,94	07.05.2014	9554,35
10.01.2014	9.529,70	07.02.2014	9.323,24	10.03.2014	9.382,98	07.04.2014	9.608,20	06.05.2014	9571,63
09.01.2014	9.549,50	06.02.2014	9.274,46	07.03.2014	9.543,24	04.04.2014	9.721,50	05.05.2014	9548,17
08.01.2014	9.516,26	05.02.2014	9.154,72	06.03.2014	9.587,44	03.04.2014	9.689,52	02.05.2014	9627,38
07.01.2014	9.518,72	04.02.2014	9.166,98	05.03.2014	9.599,00	02.04.2014	9.645,60		
06.01.2014	9.468,80	03.02.2014	9.357,58	04.03.2014	9.590,11	01.04.2014	9.631,06		
03.01.2014	9.453,48			03.03.2014	9.554,17				
02.01.2014	9.620,93								

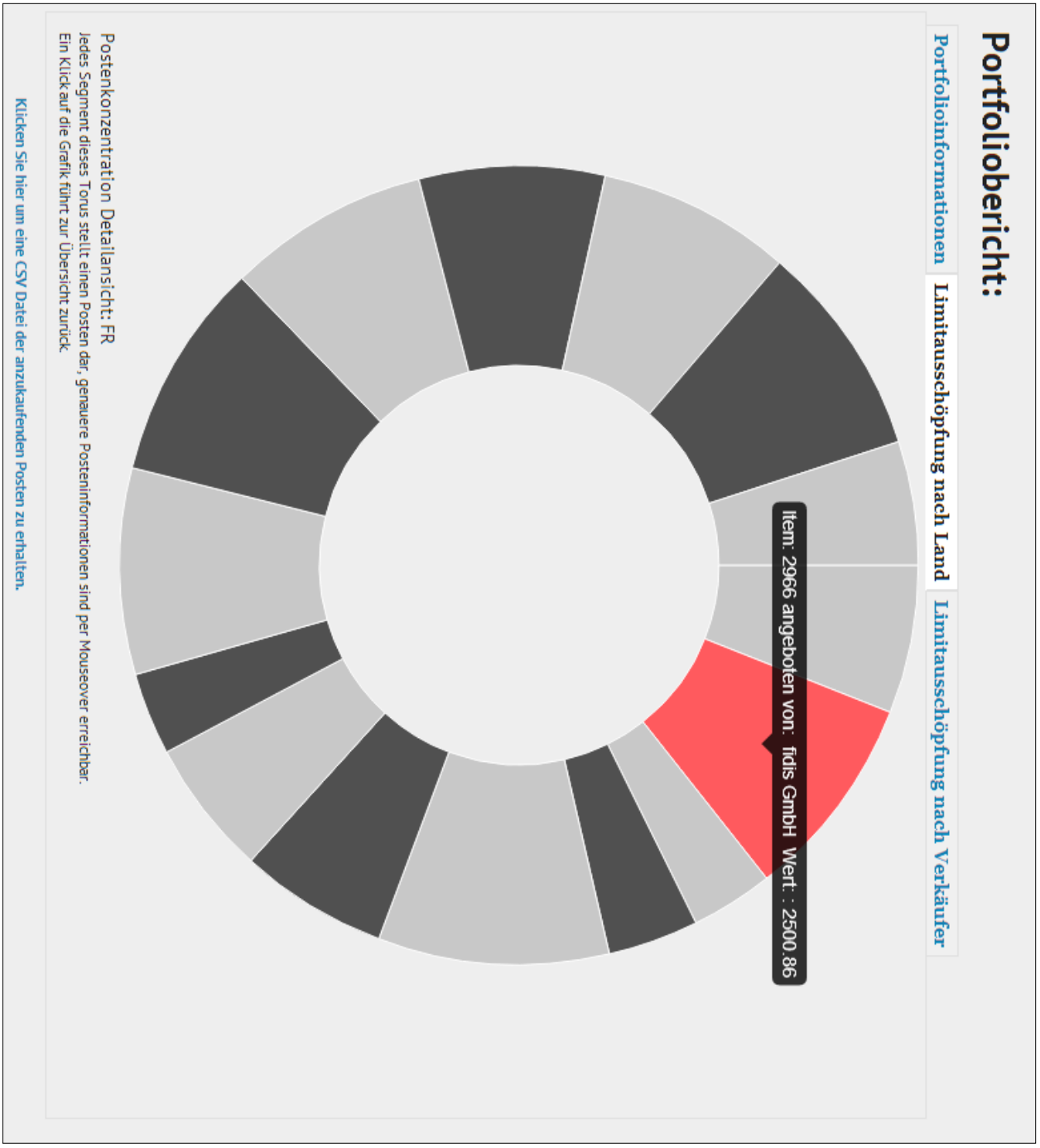
Quelle: eigene Aufarbeitung der Daten von Finanzen.net, Online im Internet

Anhang 2: Ermittlung der Dynamisch berechneten Posten



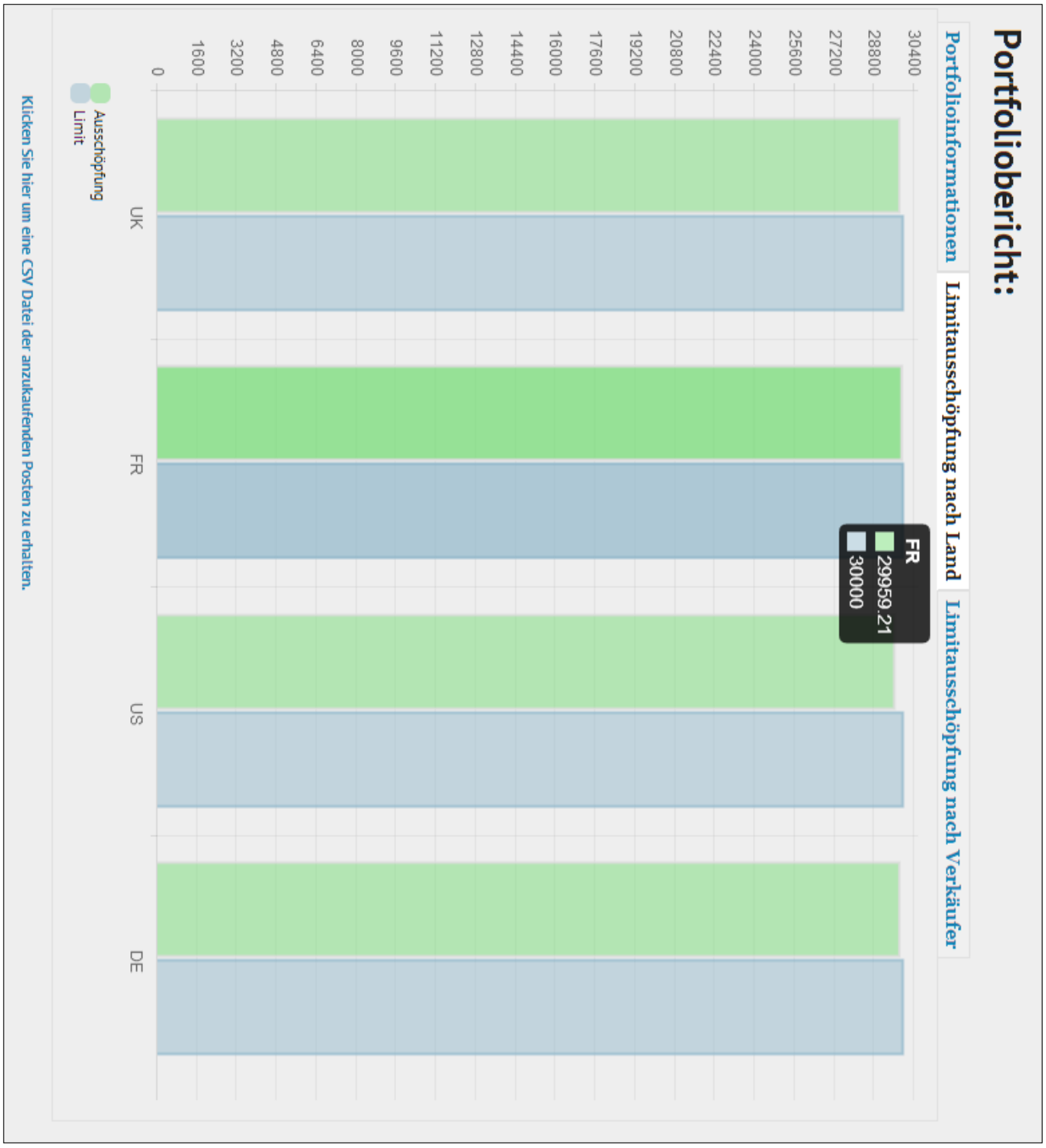
Quelle: eigene Darstellung

Anhang 3: Beispielportfolio: Tooltips



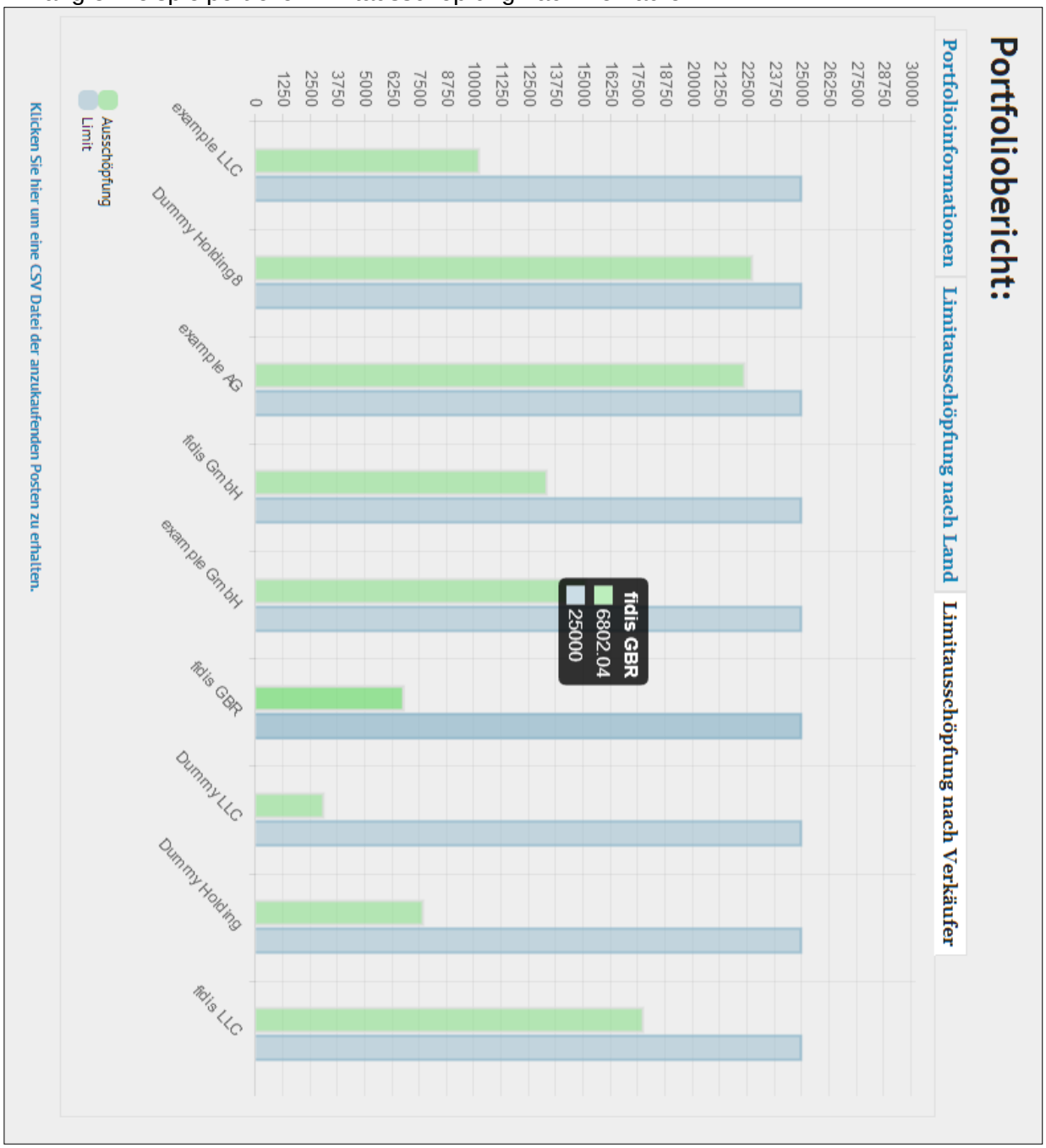
Quelle: eigene Darstellung

Anhang 4: Beispielportfolio: Limitausschöpfung nach Land



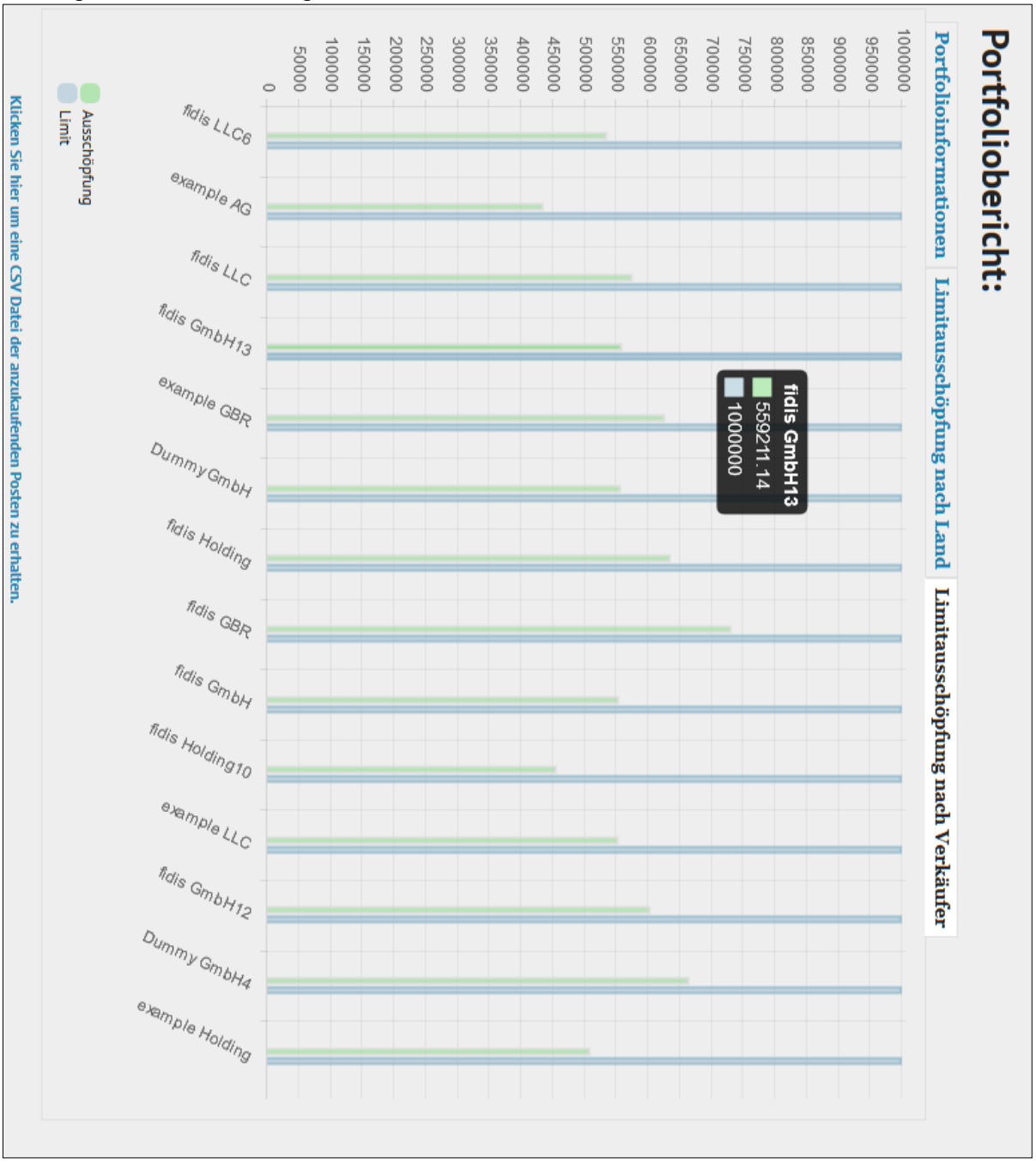
Quelle: eigene Darstellung

Anhang 5: Beispielportfolio: Limitausschöpfung nach Verkäufer



Quelle: eigene Darstellung

Anhang 6: Länderskalierung



Quelle: eigene Darstellung

Quellenverzeichnis

Monographien:

Blazewicz, Jacek/ Ecker, Klaus H./ Pesch, Erwin/ Schmidt, Günter/ Weglarz, Jan (2001), Scheduling Computer Manufacturing Processes, 2. Auflage.

Bellman, Richard (1962), Dynamic Programming treatment of the Travelling Salesman Problem, in Journal of the ACM, Volume 9, Issue 1.

Cormen, Thomas H./ Leiserson, Charles E./ Rivest, Ronald/ Stein, Clifford (2010), Algorithmen – Eine Einführung, 3. Auflage.

Dobler, Heinz/ Pomberger, Gustav (2008), Algorithmen und Datenstrukturen. Eine systematische Einführung in die Programmierung.

Korte, Bernhard/ Vygen, Jens (2012), Kombinatorische Optimierung Theorie und Algorithmen, 2. Auflage

Martello, Silvano/ Toth, Paolo (1990), Knapsack Problems Algorithms and Computer Impllementations.

Wanka, Rolf (2006), Approximationsalgorithmen Eine Einführung.

Internetquellen:

Held, Cordula (o. J.), Asset Backed Securities (ABS), Online im Internet: PURL:

http://wirtschaftslexikon.gabler.de/Archiv/3577/asset-backed-securities-abs-v8.html

Hamacher, Horst W./ Müller, Stefanie (o. J.), Lineare Optimierung im Mathematikunterricht, Online im Internet: PURL:

http://optimierung.mathematik.uni-kl.de/mamaeusch/projekte/wimstems/opt1.pdf

Keim, Daniel A. (o. J.), Datenvisualisierung und Data Mining, Online im Internet: PURL:

http://fusion.cs.uni-magdeburg.de/pubs/spektrum.pdf

Esponda, Margerita (2012), Analyse von Algorithmen – Die O Notation, Online im Internet: PURL:

http://w3.inf.fu-berlin.de/lehre/SS12/ALP2/slides/V6_Rekursion_vs_Iteration_ALP2.pdf

Shafranovich, Y. (2005), Cmmon Format and MIME Type for Comma-Seperated Values (CSV) Files, Online im Internet: PURL:

http://tools.ietf.org/html/rfc4180#page-2

Shneiderman, Ben (1996), The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations, Online im Internet: PURL:

http://www.cs.ubc.ca/~tmm/courses/old533/readings/shneiderman96eyes.pdf

Ohne Verfasser / Internetquellen:

„fs-extra modul“ (o. J.), Online im Internet: PURL:

https://www.npmjs.org/package/fs.extra

„formidable modul“ (o. J.), Online im Internet: PURL:

https://www.npmjs.org/package/formidable

„http modul“ (o. J.), Online im Internet: PURL:

http://nodejs.org/api/http.html

„W3C“ (2014), Online im Internet: PURL:

http://www.w3.org/TR/html5/

Abrufdatum:

Alle Internetquellen wurden zuletzt am 11.07.2014 aufgerufen.

Ehrenwörtliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Bachelorthesis selbständig angefertigt habe. Es wurden nur die in der Arbeit ausdrücklich benannten Quellen und Hilfsmittel benutzt. Wörtlich oder sinngemäß übernommenes Gedankengut habe ich als solches kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form ganz oder teilweise noch keiner Prüfungsbehörde vorgelegen.

Ort, Datum

Unterschrift