TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Verfahren zur automatischen Grundrissgenerierung für Gebäude mit mehreren Funktionseinheiten

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines Diplom-Ingenieurs unter der Leitung von

Privatdozent Dipl.-Ing. Dr. techn. Gabriel Wurzer

E 259 - 01

Digitale Architektur und Raumplanung

eingereicht an der Technischen Universität Wien

Fakultät für Architektur und Raumplanung

von

Christoph Rößler B.Sc.

Martrikelnummer: 00527432

Wien, am 21.05.2021 Christoph Rößler

Kurzfassung

Die Diplomarbeit befasst sich mit der algorithmischen Generierung und Optimierung von Grundrissen im architektonischen Entwurf.

Obwohl das Entwerfen von Grundrissen ein sehr zeitaufwändiger Prozess ist, wird in der architektonischen Praxis nur selten von der Möglichkeit zur automatischen Generierung jener Gebrauch gemacht. Ein Grund liegt in der gewohnten Arbeitsweise der Architekturschaffenden, die eine weitgehende Kontrolle über den räumlichen Allokationsprozess erfordert und die mit bisherigen generativen Verfahren nur schwer zu erreichen ist.

In der vorliegenden Arbeit wird ein Verfahren vorgestellt, bei dem Funktionen bzw. Räume in Gruppen eingeteilt werden und diese dann vom Benützenden mittels korrespondierender Attraktoren intuitiv innerhalb einer Gebäudegeometrie zugewiesen und anschließend vom entwickelten Algorithmus verteilt und optimiert werden.

Abstract

The thesis deals with the algorithmic generation and optimization of floor plans in architectural design.

Although the design of floor plans is a very time-consuming process, the option of generating them automatically is rarely used in architectural practice. One reason is the common way how architects work, which requires extensive control over the spatial allocation process, which has been difficult to achieve with current generative methods.

In the present work, a method is proposed in which functions or rooms are divided into groups and these are then intuitively assigned by the user by means of corresponding attractors within a building geometry and then distributed and optimized by the developed algorithm.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, keine anderen als die angegeben Quellen und Hilfsmittel benutzt und die den benutzen Quellen wörtlich und inhaltlich entnommenen Stellen als solche erkenntlich gemacht habe.

Wien, 21.05.2021 Christoph Rößler

Inhaltsverzeichnis

Einleitung

Das Entwerfen von Gebäuden ist grundsätzlich immer auch eine Form der Optimierung, bei der versucht wird, für verschiedene, teils gegensätzliche Ziele eine optimale Lösung zu finden. Dies findet in Form eines iterativen Prozesses statt, bei dem durch ein Trial and Error Verfahren versucht wird, die für ein spezifisches Entwurfsproblem optimale Variante heraus zu destillieren.

Meist wird dabei in hierarchischer Form vorgegangen, das heißt, es wird zuerst ein Gebäudetypus ausgehend vom städtebaulichen Kontext und im Bezug auf die Funktion des Gebäudes erarbeitet und definiert. In Kapitel 1 wird daher überblicksmäßig auf die Definitionsmöglichkeiten unterschiedlicher Gebäudetypologien eingegangen und auf ihre Eignung für die automatische Grundrissgenerierung überprüft.

Im nächsten Schritt wird versucht, ein vorgegebenes Raumprogramm innerhalb der Gebäudeform so zu verteilen (Abbildung 1), dass ein annähernd optimaler Zustand in Bezug auf die Lage der Räume im Gebäude und die Lage der Räume/Funktionen zueinander erreicht wird .

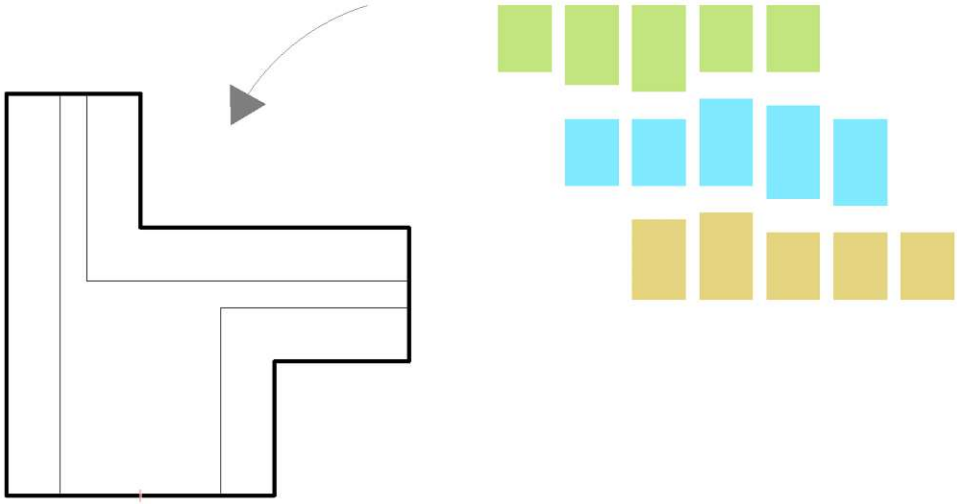

Abbildung 1: Verteilen von Funktionen innerhalb einer Geometrie, Eigene Darstellung

Geometrisch betrachtet, wird dabei eine vorhandene Form in Subformen unterteilt und die gewählte Unterteilungs- bzw. Verteilungsabfolge einer Validierung

unterzogen. Wenn das Ergebnis nicht zufriedenstellend ist, wird eine andere Unterteilungsvariante ausprobiert und diese wieder nach den Kriterien Lage der Räume im Gebäude und Lage der Räume zueinander validiert, bis eine zufriedenstellende Lösung gefunden wird. In Kapitel 2 wird ein Überblick über gängige Generierungs- und Optimierungsverfahren gegeben.

In Kapitel 3 wird ein vom Autor entwickeltes Verfahren zur automatischen Grundrissgenerierung vorgestellt, das die oben beschriebene Vorgangsweise in ein algorithmisches Verfahren übersetzen soll. Das Verfahren überlässt dabei das Finden der Gebäudeform den ArchitektInnen und soll diese bei der Platzierung der Räume bzw. Funktionen innerhalb der Gebäudegeometrie unterstützen. Anhand von Fallstudien soll anschließend die Performance des Algorithmus gegliedert nach Gebäudetypologien untersucht werden.

In Kapitel 4 wird das entwickelte Verfahren zur Diskussion gestellt und dessen Vor- und Nachteile abgewogen und in Kapitel 5 einer Schlussbetrachtung unterzogen.

1 Gebäudetypologien

Eine strukturierte Möglichkeit sich einer architektonischen Entwurfsaufgabe zu nähern ist, das zu entwerfende Gebäude ausgehend von typologischen Kriterien in Bezug auf städtebaulichen Kontext, Erschließungssysteme und Nutzung zu betrachten. Je nach Typologie stellen sich dabei unterschiedliche Anforderungen an die Architekturschaffenden und daraus folgend unterschiedliche Anforderungen an die verwendeten Entwurfswerkzeuge.

In diesem Kapitel werden Kriterien für Gebäude definiert, anhand derer bestimmt werden kann, ob sich ein Gebäude zur automatischen Generierung von Grundrissen eignet oder nicht. Dabei werden zuerst die Kriterien definiert, danach verschiedene Möglichkeiten der typologischen Einteilung von Gebäuden erläutert und anschließend nach den erstellten Kriterien (Kapitel 1.1), passende Gebäudetypologien ausgewählt (Kapitel 1.3).

In Kapitel 3.4 werden dann die ausgewählten Gebäudetypologien einer Fallstudie unterzogen.

1.1 Kriterien an die Gebäudetypologie

1.1.1 Kriterien im städtebaulichen Kontext

Erstes Kriterium ist die städtebauliche Einbindung bzw. Dimension des Gebäudes. In Kapitel 1.2.1 werden die grundlegenden städtebaulichen Figuren und deren Eigenschaften beschrieben und in Kapitel 1.3 ausgewählt, welche Figuren als geeignet betrachtet werden. Bewertet werden:

- Gebäudevolumen
- Außenerschließung des Gebäudes
- Parzellierung bzw. Parzellengrößen

1.1.2 Kriterien nach Erschließungssystem

Als nächstes werden Gebäude anhand des Erschließungssystem auf ihre Eignung für generative Verfahren untersucht. Dazu wird in Kapitel 1.2.2 ein Überblick über Erschließungssysteme gegeben und in Kapitel 1.3 bewertet. Die Kriterien sind:

- Anzahl der vertikalen Festpunkte (Treppen, Aufzüge) pro Gebäude
- Raum in Quadratmeter, der pro vertikalem Festpunkt erschlossen wird

1.1.3 Kriterien nach Nutzung

Um automatische Grundrissgenerierungsverfahren sinnvoll einzusetzen, ist auch eine gewisse Komplexität des Raumprogramms vonnöten. In Kapitel 1.2.3 werden die wichtigsten Nutzungstypologien erläutert und in Kapitel 1.3 validiert. Kriterien nach Nutzung sind:

- Kleinteiligkeit des Raumprogramms
- Anzahl der zu platzierenden Räume / Funktionen je Geschoß
- Durchschnittliche Bruttogeschossfläche nach Nutzungstypologie

1.2 Typologisierung

1.2.1 Typologisierung im städtebaulichen Kontext

Der Block

Der Block (Schneider, 2020, S 2.17 - 2.19) ist die älteste städtische Bebauungsform und stellt eine von allen Seiten durch Straßen umschlossene Gebäudegruppe dar. Er besteht meist aus mehreren Parzellen und wird von der Straßenseite erschlossen (Abbildung 2).

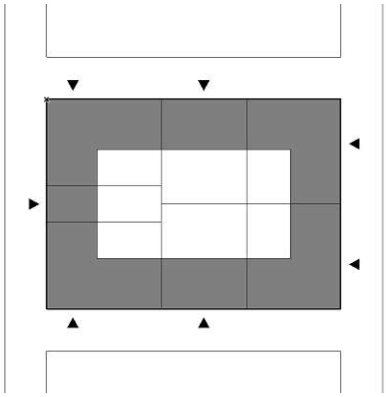

Abbildung 2: Der Block, Eigene Darstellung

Die Reihe

Die Reihe (Schneider, 2020, S 2.19 - 2.20) gehört ebenso wie der Block zu den ältesten städtebaulichen Bebauungstypologien. Hierbei werden Gebäude bzw. Parzellen entlang einer Straßenflucht in additiver Form, in geschlossener oder offener Bauweise, angeordnet und jede Parzelle einzeln erschlossen. Oft sind den jeweiligen Parzellen auch private Grünräume als Abstandsflächen zur öffentlichen Straße zugeordnet (Abbildung 3).

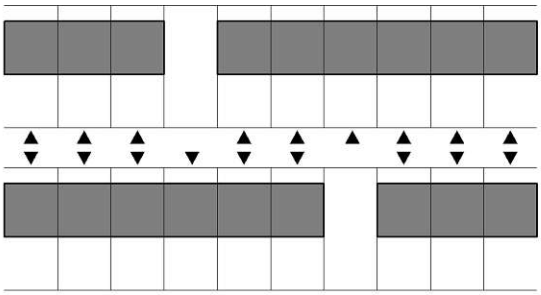

Abbildung 3: Die Reihe, Eigene Darstellung

Die Zeile

Die Zeile (Schneider, 2020, S 2.21 2.22) wird wie die Reihe durch eine additive Anordnung von Baukörpern entlang einer Straßenflucht gebildet. Unterschied ist, dass bei der Zeile die Gebäude in offener Bauweise errichtet werden und deren Stirnseiten zur Straße orientiert sind. Zwischen den Gebäuden sind meist Grünflächen und die Erschließungswege angeordnet (Abbildung 4).

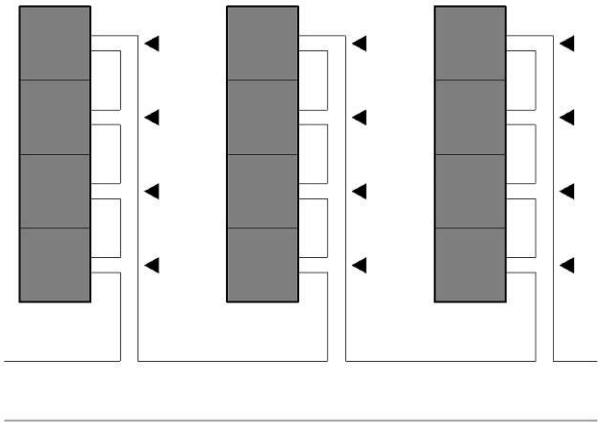

Abbildung 4: Die Zeile, Eigene Darstellung

Der Hof

Der Hof (SchneideR, 2020, S.22) als städtebaulicher Typ kann sowohl in kleinstrukturierten Parzellen und Gebäuden als auch im Geschosswohnbau Anwendung finden. Ein wesentlicher Unterschied zum Block ist, dass er über den Hof erschlossen wird und dieser als halbprivate bzw. gemeinschaftliche Fläche genutzt wird (Abbildung 5).

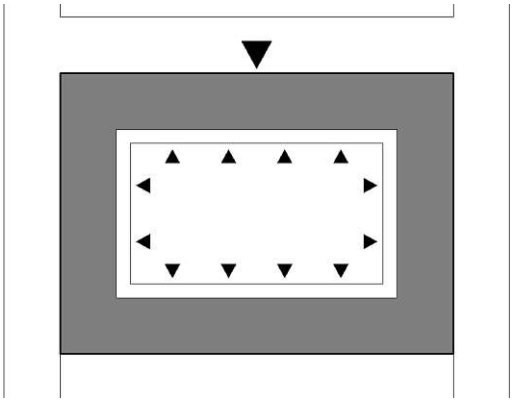

Abbildung 5: Der Hof, Eigene Darstellung

Der Solitär

Der Solitär (Schneider, 2020, S. 2.23) ist ein alleinstehendes Gebäude, das im städtebaulichen Kontext oft eine Sonderstellung bezüglich der Nutzung einnimmt. Historisch betrachtet waren dies zumeist Kirchen, Rathäuser oder Theater. Seit dem 20. Jahrhundert werden Solitäre auch für Wohn- und Bürohäuser sowie für Schulen und Heime verwendet (Abbildung 6).

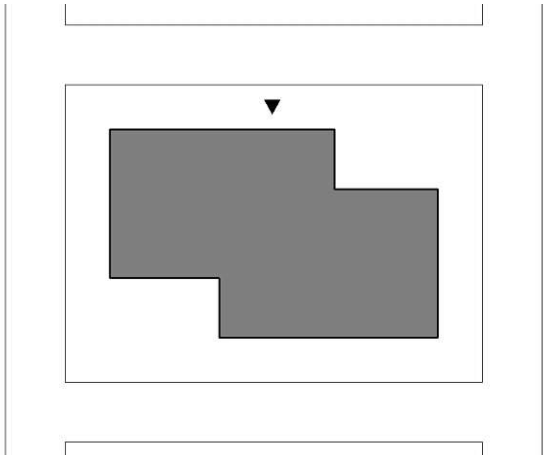

Abbildung 6: Der Solitär, Eigene Darstellung

1.2.2 Typologisierung nach Erschließung

Grundsätzlich ist bei der Erschließung (Neufert, 2012, S. 289) von Gebäuden zwischen Einzelhaus-, Reihen- und Stapelerschließung zu unterscheiden. Einzel- und Reihenerschießung erfolgen über die Aussenanlagen und liegen folgend nicht im Gebäude. Daher wird folgend die Stapelerschließung genauer beleuchtet.

Spänner

Spänner (Neufert, 2012, S. 290) sind eine weit verbreitete Form der Erschließung vor allem im Wohnbau. Sie reichen von Einspänner (Eine Einheit pro Geschoß) über Zwei- und Dreispänner (Abbildung 7.a) bis hin zu Mehrspänner (Abbildung 7.b), die meist an solitären Bauvolumen angewendet werden.

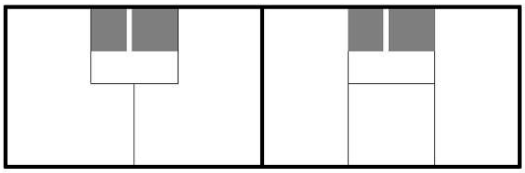

Abbildung 7.a: Spänner, Eigene Darstellung

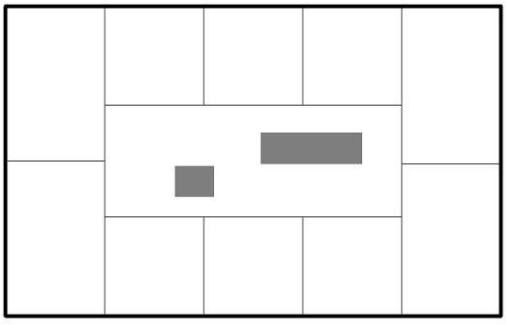

Abbildung 7.b: 10-Spänner, Eigene Darstellung

Gangerschließungen

Gangerschließungen (Neufert, 2012, S. 291) sind eine weit verbreitete Möglichkeit, Gebäude zu erschließen, bei der ein oder mehrere vertikale Festpunkte verbunden werden bzw. von dort ausgehen. Sie werden für viele Nutzungstypologien, vom Wohnbau über den Büro- und Hotelbau bis zu Bauten der öffentlichen Verwaltung und Bildungsbauten, angewandt.

Einbündige Gänge (Neufert, 2012, S. 291 und S. 479) (Abbildung 8) werden im Wohnbau meist als außenliegender Laubengang, oft in Verbindung mit Maisonettewohnungen, angedacht. Bei Bürobauten sind einbündige Erschließungen aus Wirtschaftlichkeitsgründen selten. Oft kommen einbündige Gangerschließungen in Alten- und Pflegeheimen sowie Beherbergungsgebäuden vor.

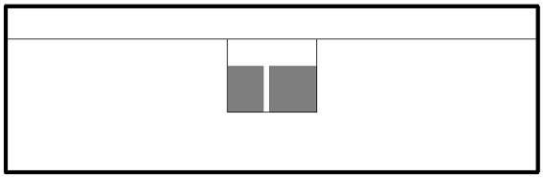

Abbildung 8: Einbündige Gangerschließung, Eigene Darstellung

Zweibündige Gänge (Neufert, 2012, S. 291 und S. 479) (Abbildung 9) werden im Wohn- und Bürobau in der Regel als innenliegender Mittelgang konzipiert. Es ist eine sehr wirtschaftliche Form der Erschließung, bei der die vertikalen Festpunkte, also Stiegenhäuser und Aufzüge, eine möglichst große Flächen abdecken können. Zu beachten ist, dass der Fluchtweg je vertikalem Festpunkt nicht länger als 35 Meter sein darf.

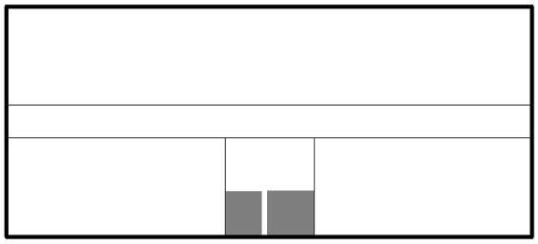

Abbildung 9: Zweibündige Gangerschließung, Eigene Darstellung

Dreibündige Gänge (Neufert, 2012, S. 479) sind weit verbreitet im Bürobau und vor allem im Bürohochhaus. Sie bieten die Möglichkeit, große Versorgungszonen in der Mitte anzuordnen (Abbildung 10). Durch Lösungen mit Glaswänden können mittig auch Besprechungsräume angeordnet werden und so eine höhere Gebäudetiefe umgesetzt werden.

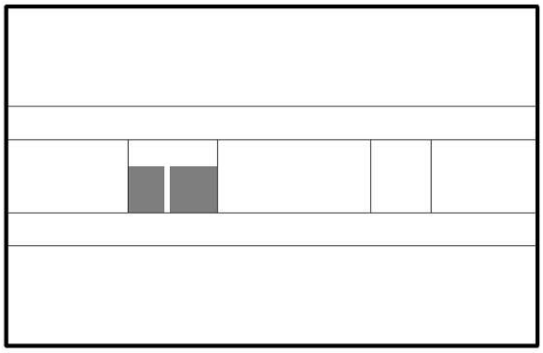

Abbildung 10: Dreibündige Gangerschließung, Eigene Darstellung

Sonderformen

Neben den eindeutig zuweisbaren Stapelerschließungstypologien gibt es auch eine Reihe von Mischformen, bei denen je nach Bauaufgabe und Nutzung ein-, zwei- und dreibündige Erschließungstypen gemischt werden (Abbildung 11).

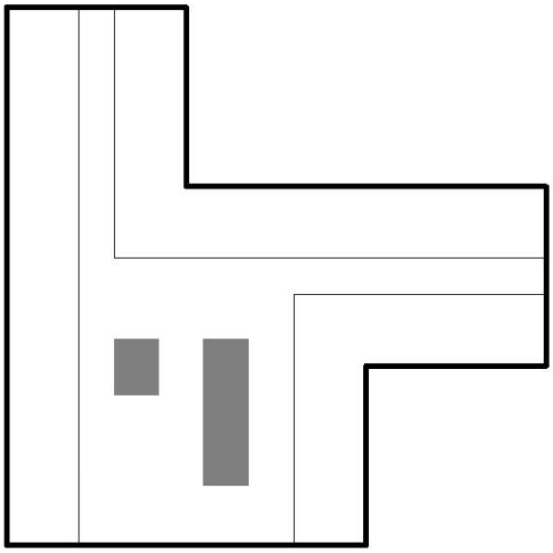

Abbildung 11: Sonderform der Gangerschließung, Eigene Darstellung

1.2.3 Typologisierung nach Nutzung

Folgend wird ein ausgewählter Überblick von Gebäudetypologien im Bezug auf die Nutzung gegeben.

Freistehende Einfamilienhaus

Das freistehende Einfamilienhaus (Neufert, 2012, S. 293) ist die einfachste Form eines Wohngebäudes. Parzellierung und Erschließung sind in der Regel speziell auf diesen Typ ausgerichtet. Jede Einheit wird separat erschlossen und das Raumprogramm beschränkt sich meist auf Vorraum, Wohnzimmer, Küche, Bad und private Zimmer der BewohnerInnen.

Geschoßwohnungsbau

Geschosswohnungen (Neufert, 2012, S. 298 - 302) bilden eine Wohnbautypologie, bei der Wohnungen vertikal gestapelt werden. Um die Wohnungen erschließen zu können, sind vertikale Festpunkte (Stiegenhaus, Aufzug) notwendig. Neben den Wohnungen und deren internes Raumprogramm werden, vor allem im Erdgeschoß und den Untergeschoßen, zusätzliche Anforderungen an das Raumprogramm im Bezug auf von allgemeine horizontale Erschließungsflächen, Gemeinschaftsräume sowie Lager und Parkräume gestellt. Wichtiger Aspekt im Geschosswohnungsbau ist auch die städtebauliche Einbindung (siehe Kapitel 1.2.1)

Büro- und Verwaltungsbau

Der Büro- und Verwaltungsbau (Neufert, 2012, S. 470 - 477) unterliegt ständigen Veränderungen, sowohl technischer und organisatorischer Natur, als auch im Bezug auf das Verhältnis der Menschen zur Arbeit. Man unterscheidet grundsätzlich zwischen Großraumbüro und Zellenbüro, wobei es hier auch Hybridformen wie zum Beispiel Gruppenbüros oder Hotelling-Offices gibt. Gemeinsam haben alle, dass hohe Anforderungen an die Flexibilität gestellt werden. So sollen Wande jederzeit versetzbar sein, die Tischanzahl und Dichte leicht verändert und auch die Angestellten flexibel verschiedenen Arbeitsplätzen zugewiesen werden können.

Grundsätzlich lässt sich ein Bürobau in Verkehrsflächen, Versorgungsflächen und die tatsächlichen Arbeitsflächen gliedern (Abbildung 12).

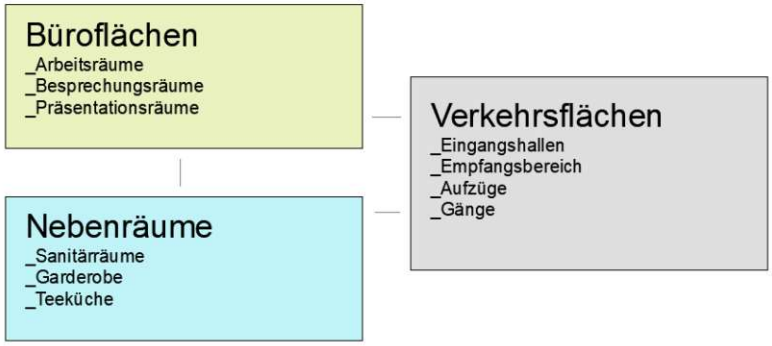

Abbildung 12: Funktionsdiagramm Bürobau, Eigene Darstellung

Hochhaus

Das Hochhaus (Neufert, 2012, S. 482) stellt eine Sonderform in Bezug auf die vertikale Erschließung dar, bei der wesentlich mehr Fläche im Verhältnis zur Gesamtfläche benötigt wird (Abbildung 13). Im städtebaulichen Kontext kann das Hochhaus als in die Höhe erweiterter Block (Amerika) oder als Solitär mit dem Zweck der Repräsentation (Europa) angesehen werden.

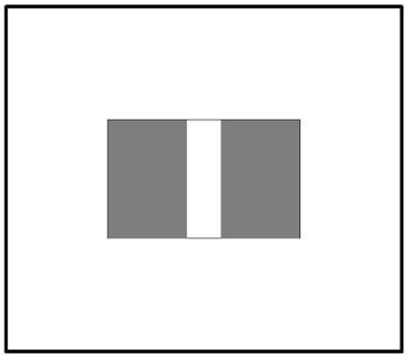

Abbildung 13: Erschließung Hochhaus, Eigene Darstellung

Beherbergung

Unter den Begriff Beherbergungbauten (Neufert, 2012, S. 328 - 334) fallen unter anderem Studentenheime, Alten- und Pflegeheime sowie Hotelbauten. Allen ist gemeinsam, dass es einerseits großzügige Gemeinschaftsräume, andererseit sehr kleinteilige Privaträume gibt (Abbildung 14). Städtebaulich gesehen können sie sowohl Teil eines Blocks (Hotel), ein ganzer Block oder ein Solitär sein.

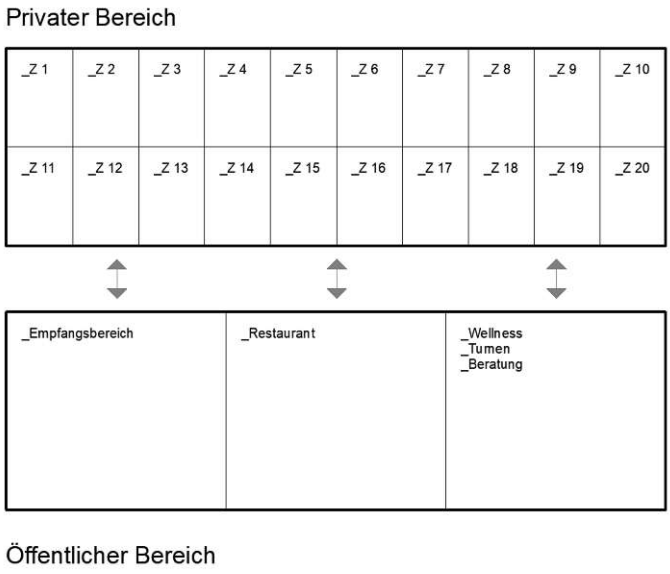

Abbildung 14: Funktionsdiagramm Beherbergung, Eigene Darstellung

Bildungsbauten

Bildungsbauten (Neufert, 2012, S. 349 - 375) reichen von Kindertagesstätten über Volks- und Mittelschulen zu Gymnasien (Primärer und Sekundärer Sektor) sowie Universitäten, Fachhochschulen und Forschungseinrichtungen (Tertiärer Sektor).

Im Primären und Sektor Sektor stellen die wichtigsten Bereiche die Garderobe, die Aula, das Schulrestaurant, das Lehrerzimmer sowie die Unterrichtsräume und Fachunterrichtsräume dar.

Im Tertiären Sektor sind neben den Hörsälen, Seminarräumen und der Mensa auch eine Reihe von Arbeitsräumen der Institute sowie fachspezifische Forschungräume wie Laboratorien zu berücksichtigen. Aufgrund des Umfangs des Raumprogramms sind tertiäre Bildungseinrichtungen oft in einem Campus (Abbildung 15) organisiert oder auf mehrere Gebäude innerhalb einer Stadt verteilt, wobei die Gebäude in die städtische Blockstruktur integriert sind, an sich aber oft als Solitäre funktionieren.

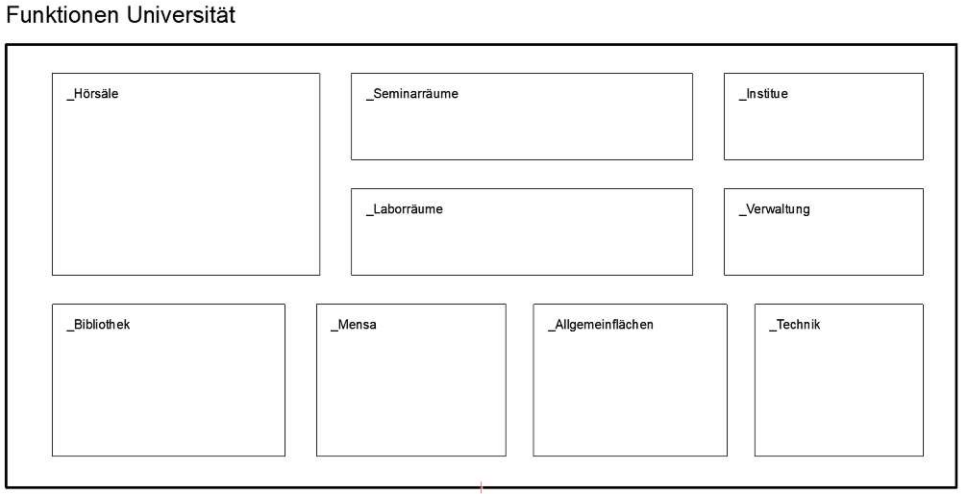

Abbildung 15: Funktionsdiagramm Universität, Eigene Darstellung

Krankenhäuser

Krankenhäuser (Neufert, 2012, S. 535 - 561) gehören zu den komplexesten Bauaufgaben und bedürfen daher speziellen Vorbereitungen in der Findung des Raumprogramms. Eine Möglichkeit ist es, mittels einer Halbmatrix (Abbildung 16) Verbindungen zwischen verschiedenen Abteilungen herauszuarbeiten. Im städtebaulichen Sinne ist ein modernes Krankenhaus immer als Solitär zu betrachten. Es ist darauf zu achten, getrennte Zufahrten und Eingänge für Besucher und Patienten, Personal und Wirtschaftsverkehr, sowie Rettungsfahrzeuge zu planen.

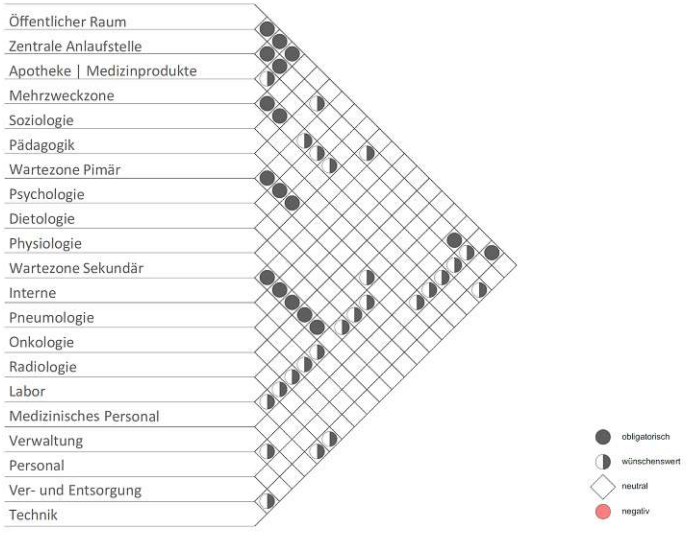

Abbildung 16: Halbmatrix, Eigene Darstellung

Gewöhnlich werden entlang einer oder mehrerer Magistralen die Funktionsbereiche angeordnet. In vertikaler Hinsicht sind im unteren Bereich die Ver- und Entsorgung und der Untersuchungs- und Behandlungsbereich untergebracht. In den oberen Etagen wird meist der Pflegebereich mit vielen Zimmern geplant.

1.3 Auswahl von geeigneten Gebäudetypen

1.3.1 Städtebauliche Kriterien

	Block	Reihe	Zeile	Hof	Solitär
Gebäudevolumen	+	−	+	+	+
Erschließung	−/+	−	−	−	+
Parzellierung	−/+	−	−	−	+
Gesamt	+	−	−	−	+

Nach rein städtebaulichen Kriterien eignen sich der **Solitär** und der **Block**, wenn er nur aus einer Parzelle besteht, am besten für automatische Grundrissgenerierungsverfahren.

1.3.2 Kriterien nach Erschließung

	Spänner	Gang	Sonderformen
Anzahl vertikaler Festpunkte	−/+	+	+
Fläche in m² je Festpunkt	−/+	+	+
Gesamt	−/+	+	+

Im Bezug auf die Erschließung ist der Einsatz von automatischen Grundrissgenerierungsverfahren für **Gangtypologien** und **Sonderformen** am geeignetsten. Bei Spännertypen ist die Anzahl der Raumeinheiten und die Bruttogeschossfläche entscheidend, ob automatisierte Verfahren sinnvoll einsetzbar sind.

1.3.3 Kriterien nach Nutzung

	EFH	Wohnbau	Büro	Hochhaus
Kleinteiligkeit des Raumprogramms	+	−	+	+
Anzahl der zu plazierenden Räume	−	+	+	+
Durchschnittliche BGF	−	−/+	−/+	+
Gesamt	−	−	+	+

	Beherbergung	Bildung	Krankenhaus
Kleinteiligkeit des Raumprogramms	+	+	+
Anzahl der zu plazierenden Räume	+	−/+	+
Durchschnittliche BGF	−/+	−/+	+
Gesamt	+	−/+	+

Aus dem Blickwinkel der Nutzung stechen vor allem **Büro-**, **Hochhaus-**, **Beherbergung-** und **Krankenhaustypologien** als sinnvolle Einsatzmöglichkeit hervor.

1.4 Gewählte Typologien für die Fallstudien in Kapitel 3.4.

- **Büro** als Block in U-Form, mit zweibündiger Gangerschließung (Abbildung 17)

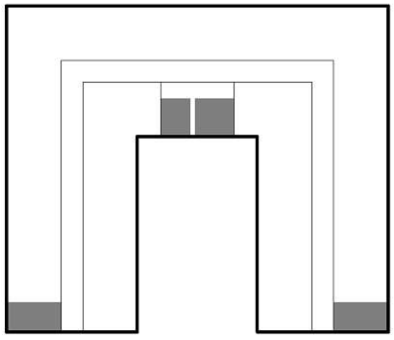

Abbildung 17: Block in U-Form, Eigene Darstellung

- **Hochhaus** als Solitär mit dreibündiger Gangerschließung (Abbildung 18)

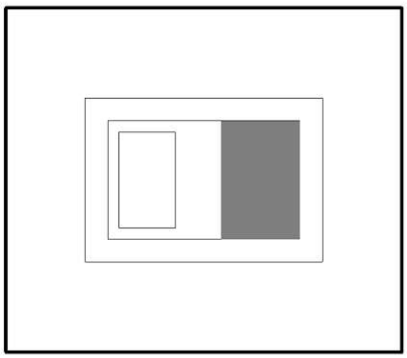

Abbildung 18: Hochhaus, Eigene Darstellung

- **Altenheim** als offener Block mit einbündiger Gangerschließung (Abbildung 19)

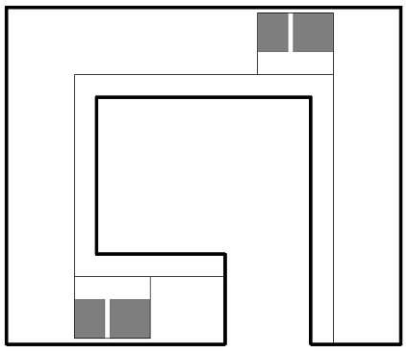

Abbildung 19: Offener Block, Eigene Darstellung

- **Krankenhaus** als Solitär mit Sonderform der Erschließung (Abbildung 20)

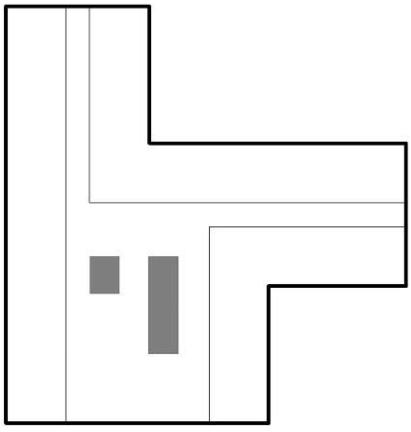

Abbildung 20: Sonderform der Erschließung, Eigene Darstellung

2 Verfahren der automatischen Grundrissgenerierung und Optimierung

Wie eingangs erwähnt, geht es bei der automatischen Generierung von Grundrissen darum, ein Raumprogramm innerhalb einer Geometrie zu verteilen bzw. aus der entstandenen Raumanordnung eine Gebäudeform abzuleiten. Dafür ist ein Regelwerk notwendig, das definiert, wie und in welcher Reihenfolge die Räume bzw. Funktionen angeordnet werden und ein Regelwerk, das es ermöglicht, dieses dann zu evaluieren und zu optimieren.

In diesem Forschungsbereich spricht man dabei vom generativen Teil und vom optimierenden Teil.

Der generative Teil legt dabei fest, wie räumliche Verteilung bzw. Teilung geregelt wird und wo diese stattfinden soll. Die Reihenfolge in der das geschieht, lässt sich letztendlich in einem Graphen darstellen und ist eine topologische Fragestellung, denn sie hat Einfluss auf die Lage des Raumes bzw. Funktion im Gebäude und die Beziehung zu den anderen Räumen/Funktionen.

Der optimierende Teil definiert dabei das Regelwerk, wie die platzierten Elemente und die entstandene Lösung evaluiert und wie die Input-Parameter verändert werden, um eine neue Lösung zu generieren. Die Evaluierung erfolgt über eine Ziel- oder Fitnessfunktion. Dabei werden meist verschiedene, teils entgegengesetzte Ziele zur Evaluierung herangezogen.

Folgend wird zuerst ein Überblick über gängige generative Verfahren gegeben und danach verschiedene Optimierungs- und Evaluierungstrategien vorgestellt.

2.1 Generative Verfahren

2.1.1 Der k-d Baum

K-d Bäume (Bentley, 1975) sind binäre Suchbäume, wobei jeder Knoten ein k-te Menge an Werten halten kann. Häufig kommen solche Suchbäume für eine nähester Nachbar Abfrage bei relationalen Datenbank vor.

Bei der Verwendung zur Generierung bzw. Unterteilung von Grundrissen wird ein k-d Baum verwendet, um ein Rechteck in einem rekursiven Verfahren in Subrechtecke zu unterteilen. Meist wird dabei ein Basis-Rechteck mit zufällig gesetzten Punkten befüllt. Je nach Unterteilungstrategie wird dann eine Schnittebene an einer bestimmten Stelle vollzogen.

Ein gängiges Unterteilungverfahren für die generative Grundrisserzeugung ist, dass ein Mittelwert aus allen Punkten des Basis-Rechteckes gebildet wird und an dieser Stelle die erste Teilung stattfindet (Knecht et al, 2010, S. 238-253). Aus den Punkten in den jeweils neu entstandenen Rechtecken wird wieder ein Mittelwert gebildet und dort unterteilt. Dieser Prozess wird solange wiederholt, bis in jedem Rechteck nur mehr ein Punkt vorhanden ist (Abbildung 21).

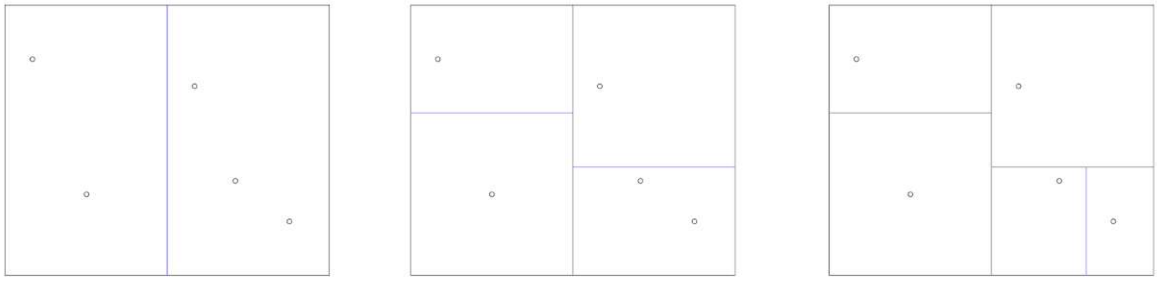

Abbildung 21: Teilungsabfogle mittels kd-Baum, eigene Darstellung

Ein Nachteil der k-d Bäume ist, dass gewisse geometrische Teilungen nicht vollzogen werden können und die resultierende Geometrie immer ein Rechteck ist.

2.1.2 Das Voronoi

Ein Voronoi-Diagramm (Coates, 2005) ist ein mathematisches Verfahren zur Unterteilung eines geometrischen Raumes. Dabei wird der geometrische Raum mit einer bestimmten Anzahl von Punkten befüllt. Folgend ist jeder Region im Raum ein Punkt als Zentrum zuweisbar. Dort wo das nicht der Fall ist, entstehen die Schnittlinien zwischen den Räumen, woraus das Voronoi-Diagramm entsteht (Abbildung 22).

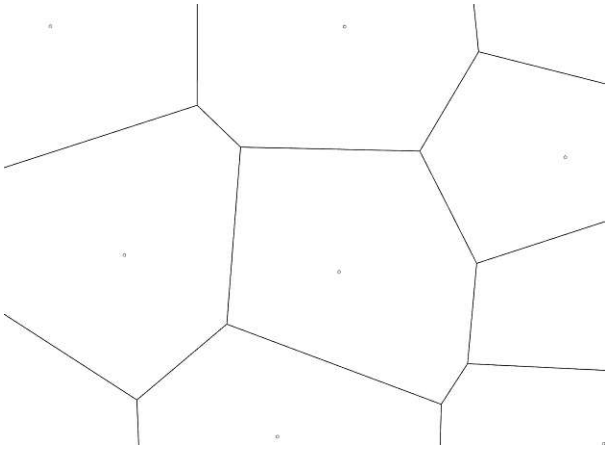

Abbildung 22: Teilung durch Voronoi-Verfahren, eigene Darstellung

Für die Generierung von Grundrisslösungen kann dieses Verfahren auch in hierarchischer Form angewandt werden, das heißt, dass zuerst der gesamte Raum so geteilt wird und in den nächsten Schritten die jeweiligen Subräume.

2.1.3 Der zelluläre Automat

Bei einem zellulären Automat (Neuman, 1963) wird ein vorhandener Raum durch eine regelmäßige Teilung in Zellen diskretisiert. Dabei hat jede Zelle eine endliche Nachbarschaft und eine Zustandsmenge, welche durch eine Überführungsfunktion in eine andere Zustandsmenge überführt wird. Jede Zelle kann abhängig vom Status der Nachbarzellen den Status ändern (Abbildung 3).

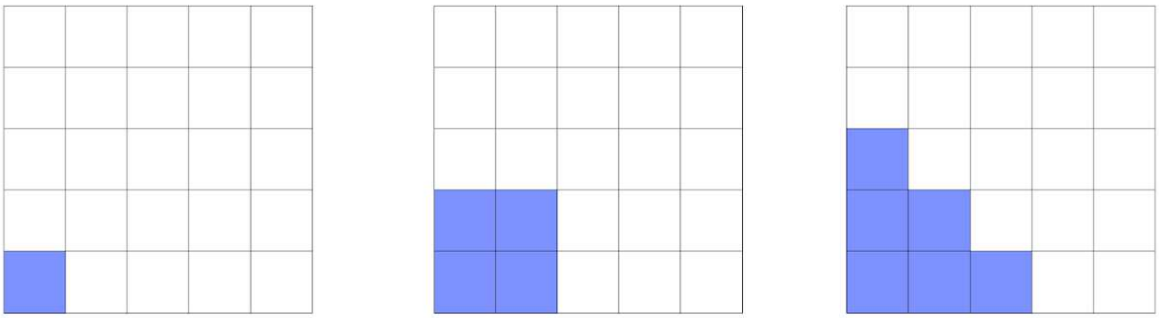

Abbildung 23:. Zellulärer Automat, Eigene Darstellung

Dadurch wird in einem Art Wachstumsverfahren Zelle für Zelle die Allokation der Funktionen in einem endlichen Raum erreicht.

Bei einem zellulären Automat unterscheidet man zwischen der Von-Neumann-Nachbarschaft (Abbildung 24.a) und der Moore-Nachbarschaft (Abbildung 24.b)

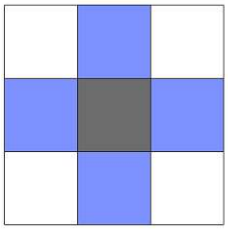

Abbildung 24.a: Von-Neumann-Nachbarschaft, Eigene Darstellung

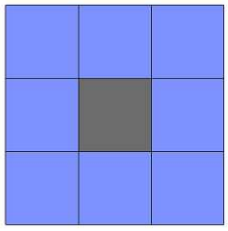

Abbildung 24.b: Moore-Nachbarschaft, Eigene Darstellung

2.1.4 Die Shape Grammar

Shape Grammar (Stiny et al, 1971, S. 1460–1465) ist eine Methode zur Erzeugung geometrischer Figuren (Shapes). Dabei wird ein Regelwerk festgesetzt, bei dem aus einer Ausgangsform neue Formen erzeugt werden können. Um einen solchen Prozess durchzuführen, braucht es mindestens eine Startregel, eine Translationsregel und eine Abbruchregel. In der Architektur werden Shape Grammars hauptsächlich zur Generierung neuer Gebäude- und Stadtformen verwendet.

Eine einfache Implementierung einer Shape Grammar wäre zum Beispiel die Definition eines Ausgangs-Quadrats. Eine Translationsregel könnte sein, dass der

Mittelpunkt des neuen Quadrates an einem zufällig ausgewähltem Eck- oder Mittelpunkt einer Seitenkante des Ausgangs-Quadrates positioniert wird. Ist jeder Eckpunkt mindestens einmal als Startpunkt zufällig gewählt worden, wird der Prozess beendet (Abbildung 25).

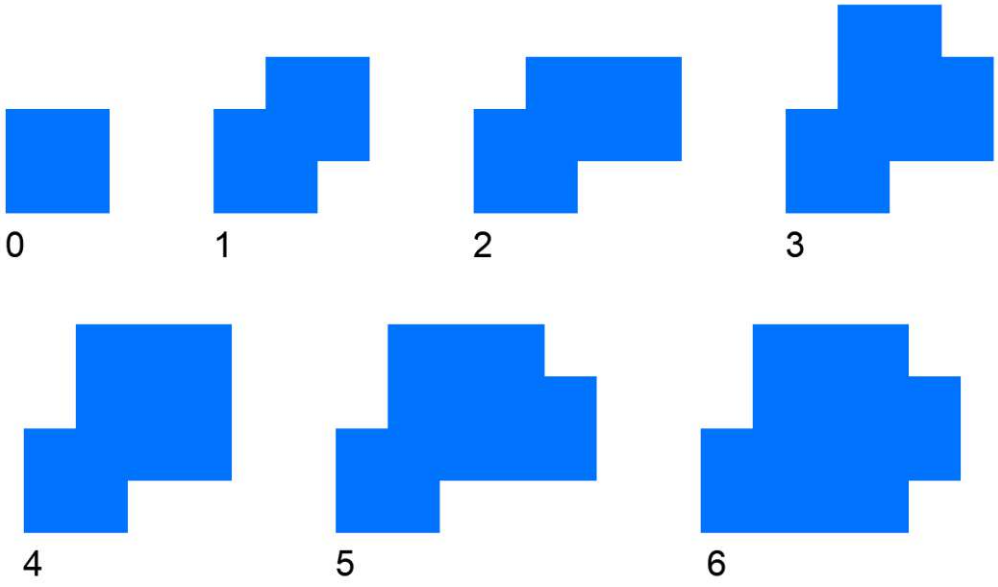

Abbildung 25: Formfindung mittels Shape Grammar, Eigene Darstellung

2.1.5 Die agentenbasierten Verteilungsalgorithmen

„Ein Agent ist ein Computersystem, das sich in einer bestimmten Umgebung befindet und welches fähig ist, eigenständige Aktionen in dieser Umgebung durchzuführen, um seine (vorgegebenen) Ziele zu erreichen.“ (Michael Wooldridge, Intelligent Agents: The Key Concepts, 2002, S. 5)

Agentenbasierte Simulationsverfahren haben in der Informatik einen großen Einsatzbereich und eine logische Nähe zum zellulären Automaten und zur Shape Grammar. Das Wesen eines Agenten ist, dass er Operationen nach festgelegten Regeln autonom, also ohne zusätzlich erforderlichen User-Input, ausführen kann und zumindest eine minimale Form der Reaktion auf die Umwelt, das heißt einen gewissen Kontextbezug aufweist. Klassische Einsatzbereiche liegen in der Simulation von Besucherströmen beim möglichst schnellen Entleeren von großen Räumen bzw. großen Menschenansammlungen.

Bei der Generierung von Grundrissen werden sie als Verteilungsverfahren bestimmter Funktionen bzw. Geometrien im Raum verwendet. Ein Agent hat einen Startpunkt und führt dort eine Aktion aus. Danach bewegt sich der Agent entlang eines je nach Regelwerk festgelegten Vektors solange, bis eine Abbruchbedingung erreicht ist. An der neuen Position versucht der Agent eine geometrische Aktion auszuführen zum Beispiel ein Rechteck aufzuspannen. Wenn dies zu den definierten Bedingungen möglich ist, wird diese Aktion ausgeführt (Rechteck platziert), wenn nicht wird der Vorgang wiederholt, bis die Aufgabe erfüllt ist. Danach wird mit der nächsten Aufgabe begonnen und erneut wiederholt, bis alle Aufgaben erledigt sind (Abbildung 26).

Abbildung 26: Einfacher Softwareagent, Eigene Darstellung

Der hier kurz abstrakt beschriebene Agent erfüllt die minimalen Bedingungen für einen Softwareagenten. Er handelt autonom und reagiert auf seine Umgebung. Weiter fortgeschrittene Implementierungen von Agenten sind auch lernfähig und können gemachte Erfahrungen in den Entscheidungsprozess einfließen lassen bzw. können diese in einem Multiagentensystem auch miteinander kommunizieren.

2.2 Such- und Optimierungsalgorithmen

2.2.1 Suchverfahren allgemein

Algorithmische Suchverfahren sind ein Teilgebiet der Informatik, bei denen ein definierter Suchraum mit bestimmten Strukturen nach bestimmten Kriterien durchsucht wird. Folgend wird ein Überblick über die die wichtigsten Begriffe und Verfahren gegeben.

Zeitkomplexität

Die Zeitkomplexität (Knuth, 1976) ist ein Teilgebiet der Informatik, mit der die Laufzeit von Algorithmen, unabhängig von der Rechenleistung eines Computersystems, klassifiziert werden kann. Ziel ist es, den Rechenaufwand zur Lösung eines Problems in Relation zur Schwierigkeit bzw. Größe des Problems zu ermitteln. Folgend eine Aufzählung der wichtigsten Zeitkomplexitätsklassen dargestellt mittels O-Notation (Edmund Landau, 1909):

Klasse	Einfluss der Problemgröße
O(1)	konstant - Laufzeit unabhängig von Problemgröße
O(log n)	logarithmisch - Laufzeit wächst weniger stark als die Problemgröße
O(n)	linear - Laufzeit wächst gleich mit der Problemgröße
$O(n^2)$	quadratisch - Laufzeit wächst um die Quadratfunktion schneller als die Problemgröße
$O(n^k)$ für k >= 1	polynomial - Laufzeit wächst mit der k-ten Potenz schneller als die Problemgröße

NP-Schwere

Die NP-Schwere (Garey, 1979) oder NP-Vollständigkeit (Cook, 1971) beschreibt in der Informatik ein Problem, bei dem sich zwar effizient, also in polynomialer Zeit, die Lösung des Problems überprüfen lässt, es jedoch keinen Algorithmus gibt, der eine Lösung in polynomialer Zeit findet.

2.2.2 Suchen in Listen

Lineare Suche

Die einfachste Form eines Suchverfahren ist die Lineare Suche (Knuth, 1998). Bei der Linearen Suche wird jedes Element einer Liste, beginnend bei Index Null, mit dem gesuchten Element verglichen, bis eine Übereinstimmung zutrifft.

Soll zum Beispiel die Zahl 10 in einer Liste, die in aufsteigender Reihenfolge sortiert ist, gefunden werden, so sind 11 Iteration notwendig (Abbildung 27).

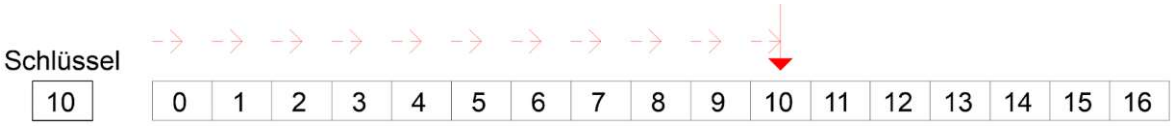

Abbildung 27: Lineare Suche in einer Liste, Eigene Darstellung

Komplexität: O(n)

Binäre Suche

Die Binäre Suche (Knuth, 1998) ist ein rekursives Suchverfahren, bei dem in einer geordneten Liste in jeder Iteration das in der Mitte (bei einer geraden Anzahl an Elementen in einer Liste wird auf oder abgerundet) liegende Element geprüft wird, ob es mit dem gesuchten Element (Schlüssel) übereinstimmt. Ist dies der Fall, ist das gesuchte Element gefunden und die Suche ist beendet. Ist dies nicht der Fall, wird geprüft, ob der Wert des Schlüssels größer oder kleiner als der des überprüften Elementes ist. Ist der Wert zum Beispiel größer, so wird der Teil der Liste, der kleiner als das Element in der Mitte ist, nicht mehr weiter durchsucht.

Das Verfahren wird solange wiederholt, bis der Wert gefunden ist (Abbildung 28).

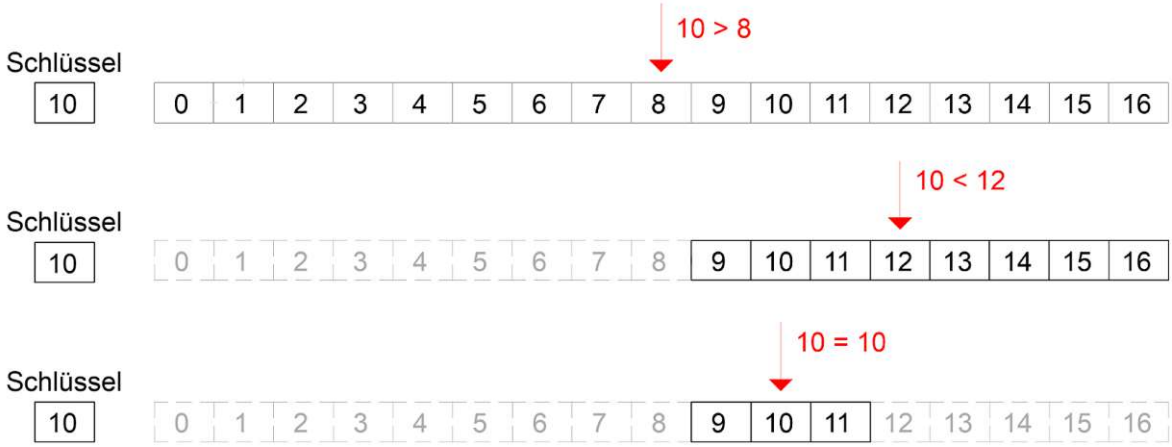

Abbildung 28: Binäre Suche in einer Liste, Eigene Darstellung

In unserem Beispiel wird der Wert 10 in der 3ten Iteration gefunden.

Komplexität: O(log n)

2.2.3 Suchen in Datenbäumen

Datenbäume (Knuth, 1997) werden in der Informatik als abstrakte Darstellungsmethode von Datenstrukturen verwendet. Ein Datenbaum ist hierarchisch strukturiert und besteht zumindest aus Knoten und gerichteten Kanten, wobei sowohl Knoten als auch Kanten einen Wert aufweisen können. (Abbildung 29). Datenbäume eignen sich sehr gut für algorithmische Suchverfahren, da sie eine logarithmische (O(log n) Laufzeit haben.

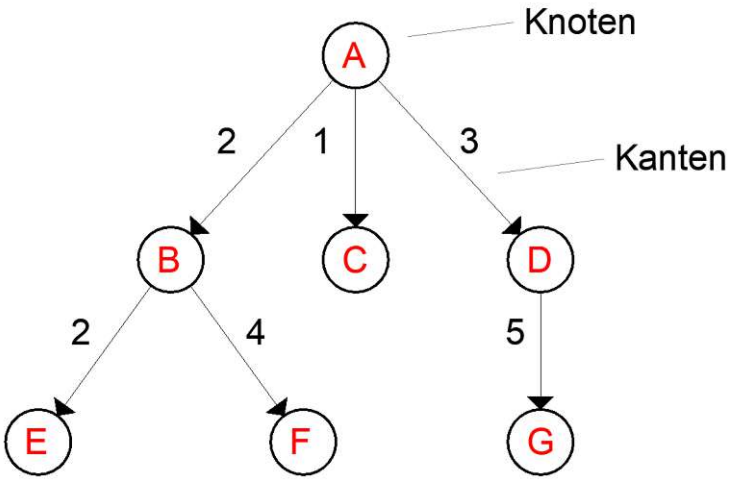

Abbildung 29: Datenbaum allgemein, Eigene Darstellung

Binärbaum

Die Binäre Suche in Listen kann auch als Binärbaum dargestellt werden. Wie bei der binären Suche muss der Suchraum geordnet sein. Bei einem Binärbaum hat jeder Knoten genau zwei Blätter, wobei das linke Blatt immer den im Wert kleineren Folgeknoten und das rechte Blatt den im Wert größeren Folgeknoten hält. Jeder Knoten bekommt dann einen Schlüssel zugewiesen. Der Suchbaum "wächst" solange weiter, bis der gesuchte Schlüssel gefunden ist (Abbildung 30).

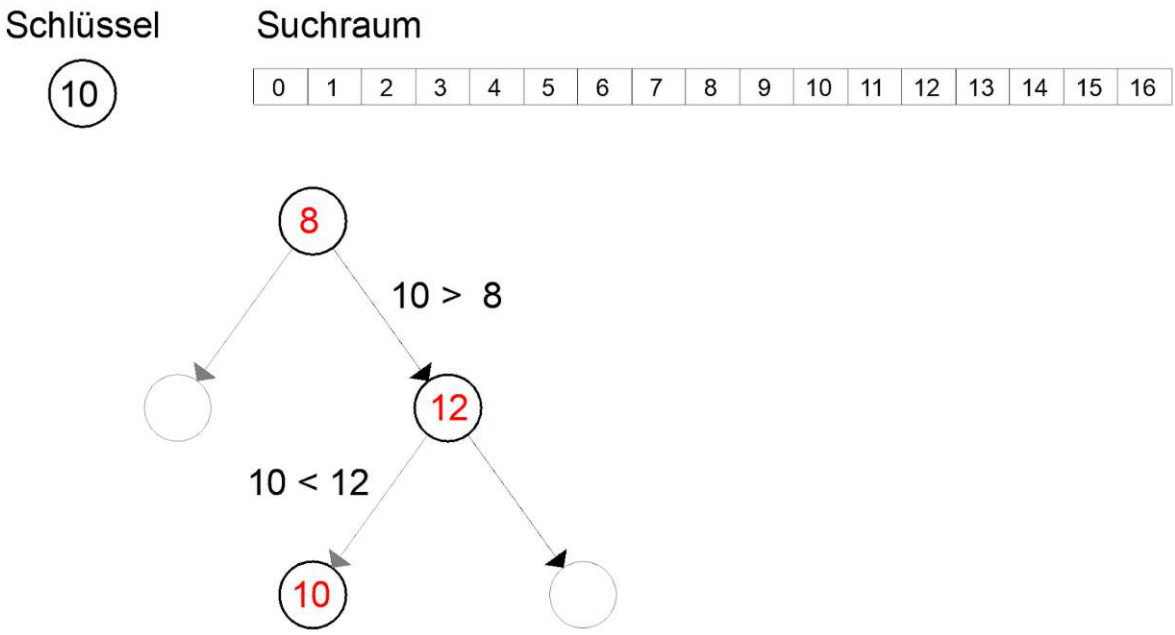

Abbildung 30: Binärer Suchbaum, Eigene Darstellung

Komplexität: O(log n)

Kd-Baum

Der in Kapitel 2.1.1 als generatives Teilungsverfahren beschriebene Kd-Baum (Bentley, 1975) oder k-dimensionaler Baum ist ein spezieller Binärbaum zur räumlichen Unterteilung, bei dem jeder Knoten k-dimensionale Werte halten kann. Die Unterteilungsrichtung wird in jeder Iteration um den k-ten Wert geändert. In einem 2-dimensionalen Raum hält jeder Knoten des kd-Baums 2 Werte (x,y) und die Unterteilungsrichtung wechselt immer zwischen x und y (Abbildung 31).

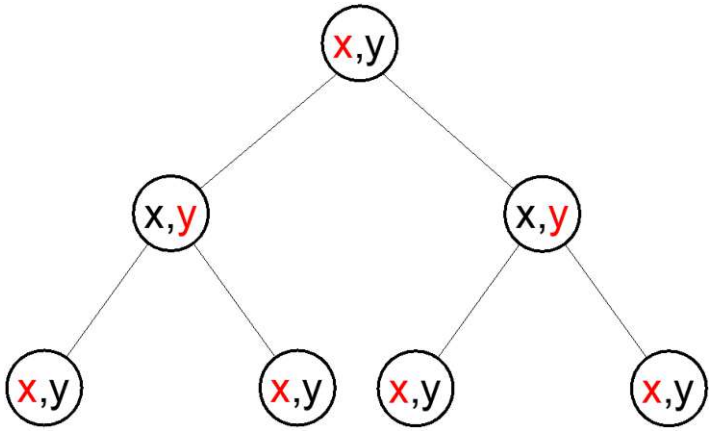

Abbildung 31: 2-dimensionaler kd-Baum, Eigene Darstellung

2.2.4 Exakte Verfahren

Die bisher beschriebenen Suchverfahren sind alle exakte Verfahren, das heißt, sie finden auf jeden Fall die beste Lösung. In der Informatik gibt es jedoch oft Problemstellungen mit einem so großen Lösungsraum, der mittels eines exakten bzw. vollständigen Suchverfahrens nicht in angemessener Zeit durchsucht werden kann. Man spricht in diesem Zusammenhang von der NP-Schwere (Kapitel 2.2.1).

2.2.5 Greedy-Algorithmus

Um NP-Schwere Probleme trotzdem effizient lösen zu können, gibt es Greedy- bzw. Gierige-Algorithmen (Knuth, 1998), die nur einen Teil des Suchraumes absuchen. Um zu entscheiden, in welche “Richtung” ein Suchraum durchsucht wird, gibt es eine Bewertungsfunktion mit der der vielversprechendste Weg gefunden werden soll. Nachteil von reinen Greedy-Algorithmen ist, dass immer nur die nächste Iteration durch die Bewertungsfunktion bewertet wird und nicht unbedingt eine annähernd optimale Lösung gefunden werden kann.

Nearest-Neighbour-Verfahren

Der bekannteste Greedy-Algorithmus ist das Nearest-Neighbour-Verfahren, bei dem immer der Folgeknoten mit der geringsten Kantenlänge durch die Bewertungsfunktion ausgewählt wird. Da jedoch nicht die gesamte Lösung einer Validierung unterzogen wird, findet das Nearest-Neighbour Verfahren meist nicht die optimale Lösung (Abbildung 32).

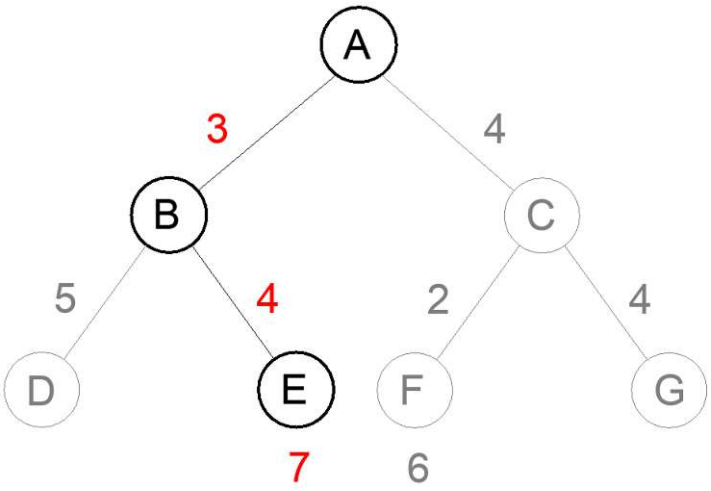

Abbildung 32: Nearest-Neighbour-Verfahren, Eigene Darstellung

In unserem Beispiel hat die gefundene Lösung den Wert 7. Die optimale Lösung wäre jedoch die mit dem Gesamtwert 6 gewesen.

2.2.6 Heuristische Optimierung

In der Informatik bezeichnet man mit heuristischer Optimierung (Pearl, 1984) Verfahren, bei denen man trotz unvollständigem Wissen eine optimale (vollständige Verfahren) oder nahezu optimale (unvollständige Verfahren) Lösung findet. Im Gegensatz zu reinen Greedy-Algorithmen, die bei der Abschätzung der Erfolgsaussichten immer nur die nächste Iteration bewerten und daher oft in einem lokalen Optimum enden, soll mit heuristischen Methoden das globale Optimum gefunden werden (Abbildung 33). Das No-Free-Lunch-Theorem (Wolpert et al, 1995) besagt, dass alle Optimierungsalgorithmen gemessen an der Gesamtheit der Optimierungsprobleme gleich gut bzw. gleich schlecht funktionieren. Daraus folgt, dass jedes Optimierungsverfahren problemspezifisch eingesetzt bzw. entwickelt werden muss.

Globales vs lokales Optimum

Ein lokales Optimum beschreibt einen Wert einer Funktion, dessen direkt benachbarte Werte immer niedriger sind als der betrachtete Wert. Bei einem globalen Optimum sind alle Werte der gesamten Funktion niedriger als der betrachtete Wert (Abbildung 33).

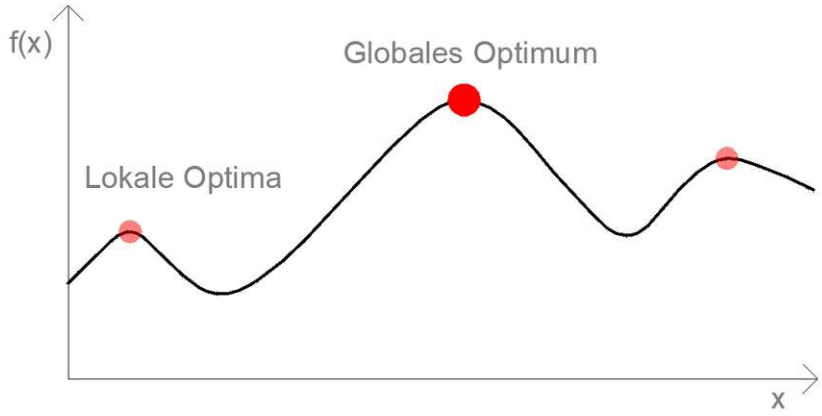

Abbildung 33: Globales und lokales Optimum, Eigene Darstellung

Zielfunktion

Mit einer Zielfunktion (bei evolutionärer Optimierung Fitnessfunktion) wird beschrieben, wann ein Optimum erreicht worden ist. Die Wahl der Zielfunktion bzw. deren sinnvolle Definition für ein spezifisches Optimierungsproblem ist entscheidend, ob ein Optimierungsverfahren die gewünschten Ergebnisse liefert.

Konstruktionsheuristik

Unter einer Konstruktionsheuristik versteht man Verfahren, mit denen die erste Lösung bzw. die Startlösung in einem heuristischen Optimierungsverfahren definiert wird. Diese kann entweder per Zufall bestimmt werden oder problemspezifisch definiert sein. Die Wahl einer passenden Konstruktionsheuristik hat großen Einfluss auf die erfolgreiche Implementierung eines heuristischen Optimierungsverfahren.

2.2.7 Metaheuristiken

Unter Metaheuristiken versteht man eine Abfolge von Schritten, die problemunabhängig definiert sind, die jedoch in der tatsächlichen Implementierung problemspezifisch angewandt werden müssen (No-Free-Lunch-Theorem).

Greedy Randomized Adaptive Search Procedure (GRASP)

GRASP (Feo et al, 1989) ist ein Nachbarschaftssuchverfahren, bei dem zuerst mittels eines Greedy-Algorithmus eine Lösung generiert wird und diese Lösung anschließend mittels einer lokalen Suche verbessert werden soll. Um nicht in einem lokalen Optimum stecken zu bleiben, wird als neue Basislösung nicht immer die beste vorangegangene Lösung gewählt, sondern eine Restricted-Candidates-List (RCL) mit beliebiger Länge aus den besten Lösungen gebildet und aus dieser zufällig die Ausgangslösung gewählt.

Variable Neighbourhood Search

Die Variable Neighborhood Search (Mladenović et al, 1997, S. 1097-1100) ist ein Nachbarschaftssuchverfahren, bei dem mittels lokaler Suche eine Nachbarschaft strukturiert abgesucht wird. Wege zur strukturierten Nachbarschaftssuche sind vor allem Einfügen (Insert) oder Tauschen (Swap) von Elementen. Um nicht in einem lokalen Optimum stecken zu bleiben, wird die Lösung immer wieder durch einen Zufallsgenerator geschüttelt (shaking). Neben der Standard Variable Neighbourhood Search (VNS) gibt es auch Erweiterungen wie Variable Neighbourhood Descent (VND), Reduced Variable Neighbourhood Search (RVNS) oder Skewed Variable Neighbourhood Search (SVNS) (Hansen et al, 2000).

Simulated Annealing

Simulated Annealing (Kirkpatrick et al, 1983) ist ein Nachbarschaftssuchverfahren, das abgeleitet von der Metallbearbeitung einen Temperaturfaktor verwendet, um die Wahrscheinlichkeit zu steuern, inwieweit eine schlechtere Lösung als neue Ausgangslösung akzeptiert wird, um nicht in einem lokalen Optimum stecken zu bleiben. Je niedriger die Temperatur desto niedriger ist die Wahrscheinlichkeit, dass eine schlechtere Lösung als neue Ausgangslösung akzeptiert wird. Alleinstellungsmerkmal des Simulated Annealing Verfahrens ist, dass durch die Rate, mit der die Temperatur abnimmt, auch die Suchdauer bestimmt werden kann.

Evolutionäre Optimierung

Evolutionäre bzw. genetische Optimierungsverfahren (De Jong, 1975) sind Optimierungsalgorithmen, die die natürliche Selektion (Darwin, 1859) nachahmen. Sie gehören zu den ersten und am weitesten verbreiteten Optimierungsalgorithmen. Sie sind, im Gegensatz zu den bisher erwähnten Metaheuristiken, keine Nachbarschaftssuchverfahren, sondern basieren auf Populationen mit Individuen, die ihre Eigenschaften an die nächste Generation vererben.

Konkret wird dabei eine Startpopulation mit einem gewissen Umfang initiiert und für jedes Individuum ein Fitnesswert mittels einer Fitnessfunktion errechnet. Danach wird eine Selektion (selection) der sich später kreuzenden (crossover) Individuen durchgeführt, wobei jene Individuen, die einen höheren Fitnesswert haben, eine höhere Chance haben, ausgewählt zu werden. In gewissen Abständen wird, um nicht in einem lokalen Optimum stecken zu bleiben, eine Mutation (mutation) mittels eines Mutationsfaktors errechnet. Die Idee dabei ist, dass bei jeder Iteration neue, fittere Individuen entstehen und weniger fitte Individuen aus der Population ausscheiden und sich so die Fitness der gesamten Population erhöht und daraus folgend optimiert.

2.3 Beispiele

2.3.1 Traveling Salesman Problem

Das Traveling Salesman Problem (Beardwood, 1956) oder das Problem des Handlungsreisenden, ist ein NP-Schweres kombinatorisches Optimierungsproblem aus der Graphentheorie und eines der bekanntesten Beispiele für heuristische Optimierungsverfahren in der Informatik. Es kann durch unterschiedliche Metaheuristiken gelöst werden.

Konkret geht es dabei darum, für einen Handlungsreisenden, der eine Vielzahl von verschiedenen Städten besuchen will, jene Abfolge von Punkten bzw. Städten zu finden, durch die der kürzeste Weg entsteht (Abbildung 34). Es wird in der

Informatik und anderen Disziplinen als Standardverfahren zum Testen neu erforschter Optimierungsalgorithmen verwendet.

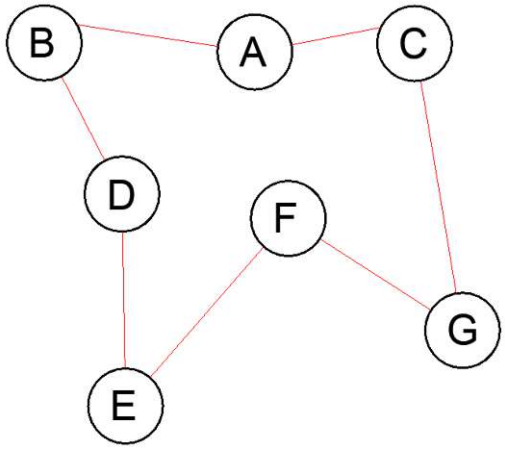

Abbildung 34: Städte mit Route, Eigene Darstellung

2.3.2 Pin Packing

Pin Packing (Korte, 2006) oder Behälterproblem ist ein NP-Schweres kombinatorisches Optimierungsproblem, bei dem es darum geht, in eine bestimmte Anzahl von Behältern eine bestimmte Anzahl an Objekten mit einer definierten Größe so einzuordnen, dass dabei der Raum der Behälter möglichst optimal genutzt wird (Abbildung 35).

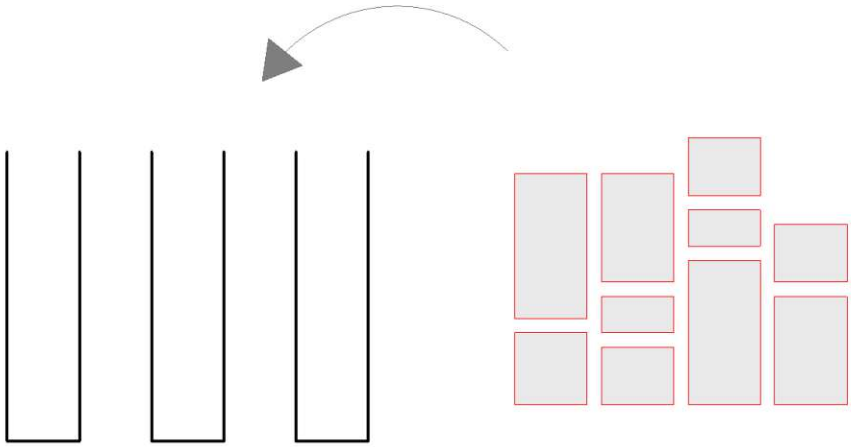

Abbildung 35: Bin Packing Problem, Eigene Darstellung

Als Einordnungstrategie werden unter anderem unterschieden:

First-Fit

First-Fit (Ullman, 1971) ist eine Vorgehensweise, bei der beginnend beim ersten Behälter das erste Objekt in der Liste in jenem Behälter platziert wird, in den es als erstes hineinpasst. Von den drei hier genannten Einordnungstrategien ist sie die schnellste, hat jedoch den Nachteil, dass wahrscheinlich kein globales Optimum gefunden wird.

Best-Fit

Bei der Best-Fit (Ullman, 1971) Vorgehensweise wird jeder Behälter überprüft und immer das erste Objekt in der Liste in jenem Behälter platziert, bei dem nach der Platzierung die geringste Restfläche im Behälter bleibt.

Worst-Fit

Worst-Fit ist das genau Gegenteil von Best-Fit. Es wird jeder Behälter ausprobiert und das Objekt dort platziert, wo die größte Restfläche bleibt

Next-Fit

Next-Fit (Johnson, 1973) ist eine modifizierte Variante des First-Fit-Verfahrens. Das erste Objekt in der Liste wird immer im ersten passenden Behälter platziert. Im Unterschied zu First-Fit wird dabei aber nicht immer beim ersten Behälter zu suchen begonnen, sondern als erstes jener Behälter gewählt, der auf den Behälter folgt, in dem in der vorhergehenden Iteration das Objekt platziert wurde.

3 Automatisches Grundrissgenerierungsverfahren

3.1 Konzept

Die Problemstellung beim Entwerfen von Grundrisslösungen und somit auch bei der automatischen bzw. semiautomatischen Generierung von Grundrissvarianten ist, ein vorhandenes Raumprogramm innerhalb einer Gebäudeform bzw. Morphologie unterzubringen bzw. zu organisieren. Durch die Wahl der Anordnung der Funktionen/Räume, deren Lage im Gebäude und deren Lage bzw. Beziehungen zueinander ergeben sich verschiedene Grundrisstopologien für das gleiche Raumprogramm in der gleichen Gebäudemorphologie.

Eine Möglichkeit die Topologie zu steuern, ist die hierarchische Zuweisung von Funktionen zu Abteilungen bzw. Departments. Das heißt, dass Funktionen/Räume zu Funktionseinheiten zusammengefasst werden und daraus ihr geometrisches Näheverhältnis abgeleitet wird.

Neben der geometrischen Nähe von Funktionen ist auch die Lage im Gebäude ein entscheidender topologischer Parameter. Neben der Überlegung, ob ein Raum natürliches Licht benötigt, daher an einer Aussenkante der Gebäudeform angeordnetn sein muss, sind auch Parameter wie Ecklage vs. Nicht-Ecklage oder Lage direkt am Erschließungskern oder nicht von Bedeutung.

Folgend wird eine neue Methodik vorgestellt, wie ein Raumprogramm durch ein algorithmisches Verfahren innerhalb einer Gebäudemorphologie unter Berücksichtigung der oben genannten topologischen Einstellungsmöglichkeiten verteilt werden kann.

3.2 Methodik

3.2.1 Dateneingabe

Das Raumprogramm

Um die Räume bzw. Funktionen je nach Raumeigenschaften in der Gebäudegeometrie verteilen zu können, müssen jedem Raum folgende Eigenschaften zugewiesen werden:

- Bezeichnung
- Department-Index
- Quadratmeter
- Natürliches Licht
- Raumtyp
    - undefiniert
    - direkt an der Erschließung
    - an einem Gebäudeeck

Die Departements

Die Räume werden dann den einzelnen Departements zugewiesen (Abbildung 36). Später werden die Departments über Attraktorenlinien in der Gebäudegeometrie platziert.

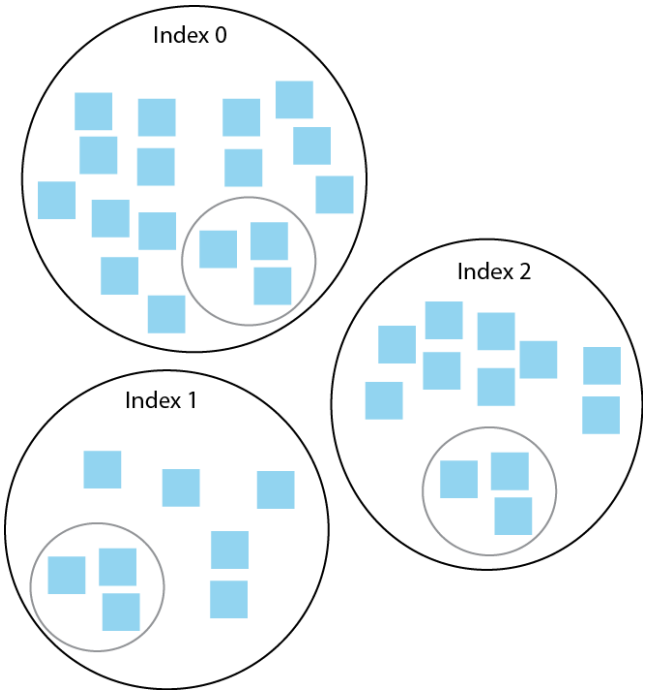

Abbildung 36: Departements mit Räumen, Eigene Darstellung

Die Gebäudegeometrie

Die Gebäudegeometrie wird durch mindestens 2 geschlossene Polylinien definiert. Eine für die Gebäudeaußenkante (Abbildung 37.a) und eine für den Erschließungskern (Abbildung 37.b) wobei die Polylinie für die Erschließung vollständig innerhalb der Gebäudekanten-Polylinie liegen muss.

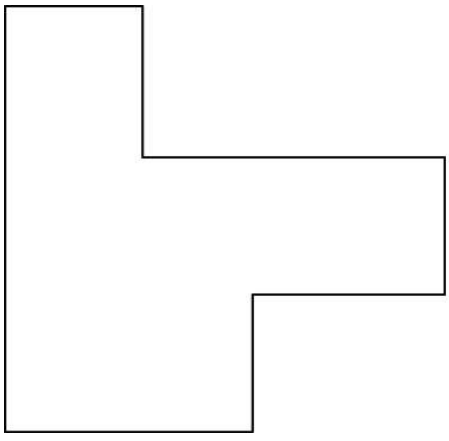

Abbildung 37.a: Gebäudeoutline, Eigene Darstellung

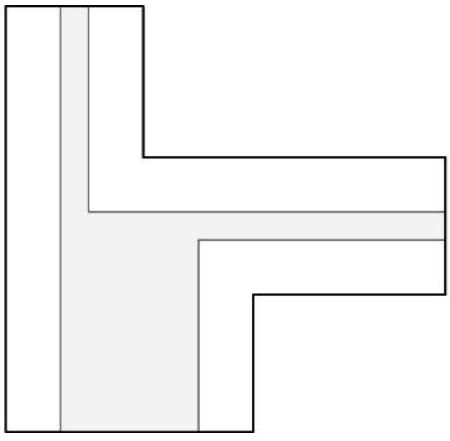

Abbildung 37.b: Erschließungsoutline, Eigene Darstellung

3.2.2 Datentransformation

Punktmatrix

Um aus der so erstellten Geometrie mehr Information herauslesen zu können, wird zwischen der Gebäudekanten-Polylinie und der Erschließungs-Polylinie eine 2d-Punktmatrix aufgespannt (Abbildung 38).

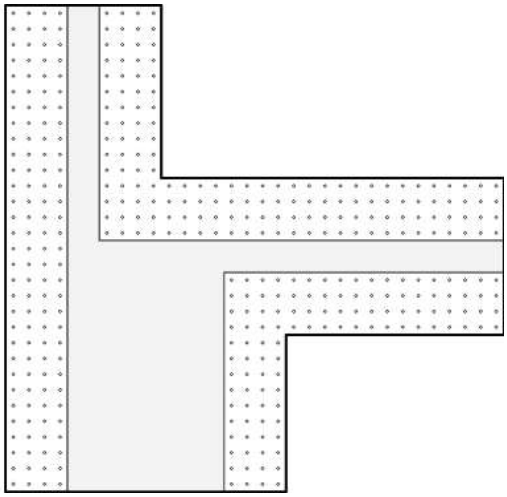

Abbildung 38: Punktmatrix, Eigene Darstellung

Moore-Nachbarschaft

Abgeleitet von der Moore-Nachbarschaft können die Punkte in Gruppen bzw. Typen eingeteilt werden.

Jeder Punkt hat folgende mögliche Nachbarn:

Punkt X/Y: X+1/Y, X+1/Y+1, X/Y-1, X-1/Y-1, X-1/Y, X-1/Y+1, X/Y+1 (Abbildung 39)

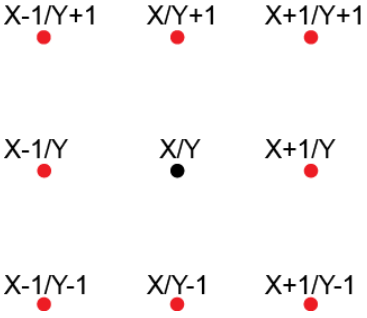

Abbildung 39: Moore-Nachbarschaft, Eigene Darstellung

Punkttypen

Daraus ergeben sich folgende Punkttypen

- Rot - Eckpunkt mit 4 oder weniger Nachbarpunkten
- Orange - Eckpunkt mit 6 oder mehr Nachbarpunkten
- Violet - Eckpunkt zur Erschließungslinie mit 4 oder weniger Nachbarpunkten
- Blau - Kantenpunkt zur Gebäudekanten-Polylinie mit 5 Nachbarpunkten
- Grün - Kantenpunkt zur Erschließungs-Polyline mit 5 Nachbarpunkten
- Grau - Kernpunkt mit 8 Nachbarpunkten

In der gewählten Beispiel-Geometrie ergibt sich folgendes Bild (Abbildung 40).

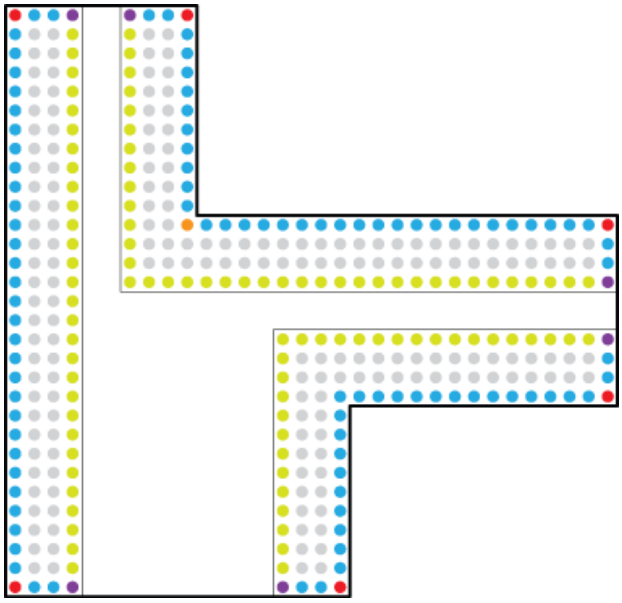

Abbildung 40: Punkttypen, Eigene Darstellung

Mit den daraus gewonnenen Informationen können später im Verteilungs- und Optimierungsprozess Aussagen darüber getroffen werden, ob ein Raumplatzierungsversuch positiv bewertet wird und der Raum somit platziert werden darf.

Attraktor

Um die Departements innerhalb der Gebäudeform zuweisen zu können, wird vom Anwender eine Attratorenlinie innerhalb der Gebäudegeometrie gesetzt. Von

dieser Attraktorenlinien wird für jeden Punkt ein Distanzwert berechnet (Abbildung 41).

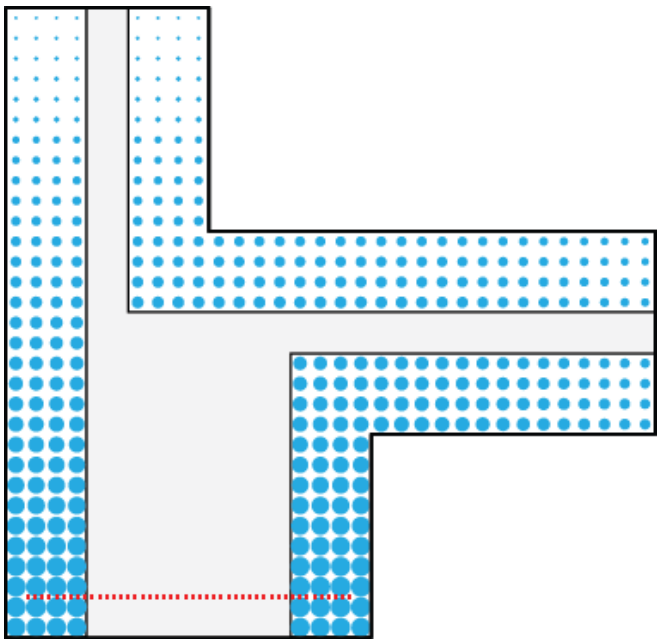

Abbildung 41: Attraktor, Eigene Darstellung

3.2.3 Verteilungsalgorithmus

Liste von möglichen Startpunkten

Um eine Liste von möglichen Startpunkten zu definieren, wird die Länge der Attraktorlinie in Meter [n] ermittelt und dann die nähesten [n] Kantenpunkte zur Gebäudeumrisslinie ausgewählt (Abbildung 42).

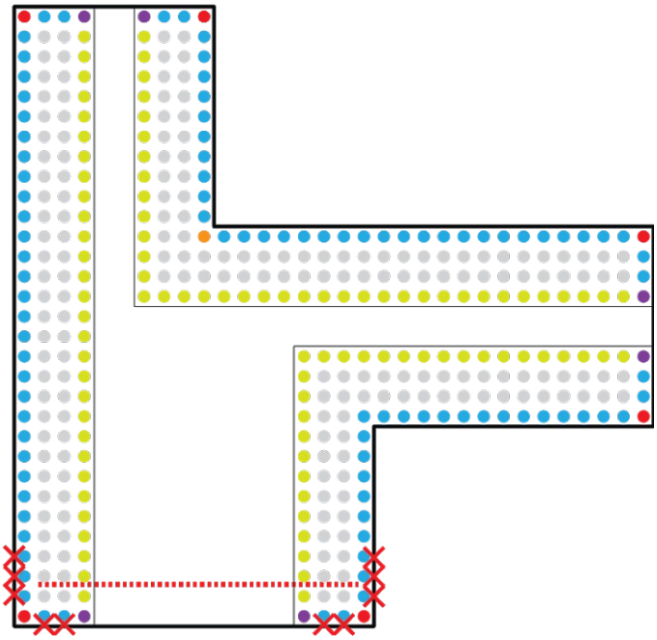

Abbildung 42: Mögliche Startpunkte, Eigene Darstellung

Auswahl des Startpunktes

Aus der so erstellten Punktliste wird nun per Zufall der tatsächliche Startpunkt ausgewählt (Abbildung 43).

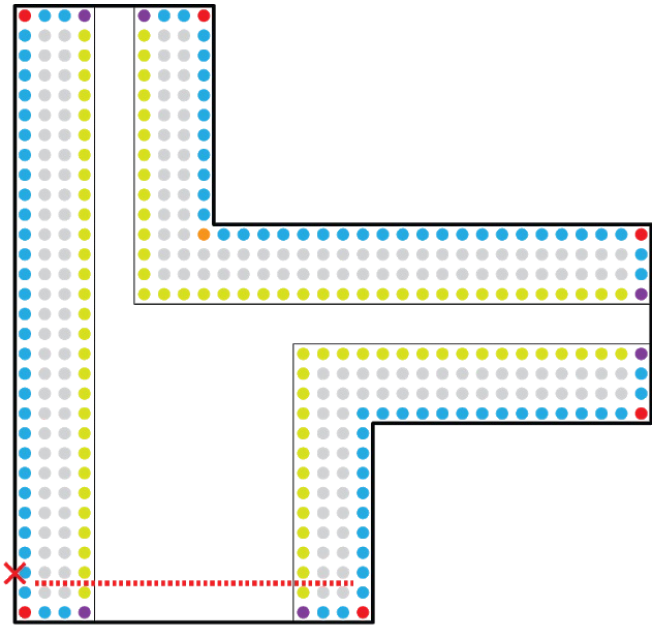

Abbildung 43: Gewählter Startpunkt, Eigene Darstellung

Am Startpunkt wird auf der Gebäudeumrisslinie eine neue Konstruktionsebene, deren x-Achse parallel zur Gebäudeumrisslinie ist, gesetzt.

Platzieren der Räume

Beginnend mit dem Departement [0] und der Liste von Räumen, die diesem Departement zugewiesen wurden, beginnt nun der Softwareagent mit der Anordnung der Räume (Rechtecke) entlang der Gebäudeumrisslinie.

Dabei wird beginnend Richtung y-Achse der neu gesetzten Konstruktionsebene eine Linie generiert, bis entweder ein gewisses Verhältnis im Bezug zur Fläche des Raumes, oder eine Kollision mit der Erschließungslinie erreicht ist.

Danach wird beginnend von der soeben generierten Linie in die positive und negative Richtung der x-Achse jeweils ein Rechteck aufgespannt, bis der addierte Flächeninhalt des Rechteckes die Fläche des zu platzierenden Raumes erreicht hat (Abbildung 44).

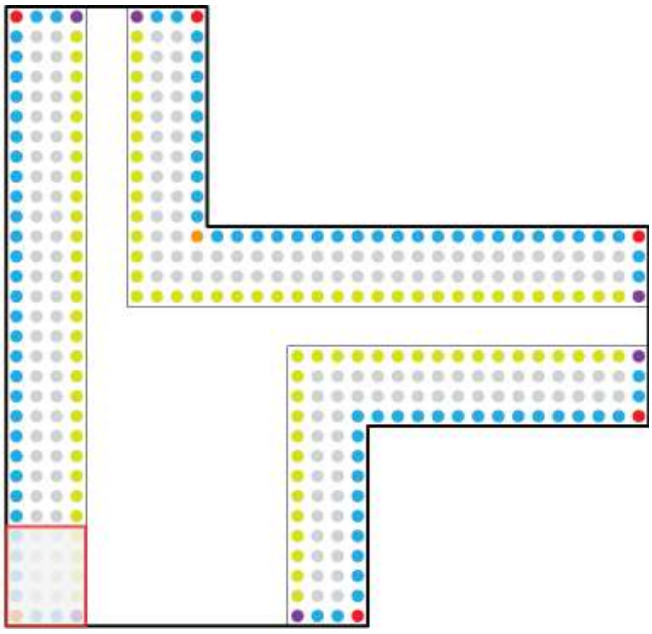

Abbildung 44: Aufspannen der Räume, Eigene Darstellung

Bedingungen Raumtyp

Neben der Raum-Soll-Fläche müssen auch zum Raumtyp spezifische Erfordernisse erfüllt sein (Abbildung 45). Dazu zählen:

- Raumtyp **Undefiniert** - keine zusätzlichen Bedingungen

- Raumtyp **Standard-Raum** - Raum muss mindestens einen Punkt vom Typ Kantenpunkt zur Erschließungs-Polyline enthalten (Grün)
- Raumtyp **Eck-Raum** - Raum muss mindestens einen Punkt vom Typ Kantenpunkt zu Erschließungs-Polyline enthalten (Grün) und einen Eckpunkt mit maximal 4 Nachbarpunkten enthalten (rot)

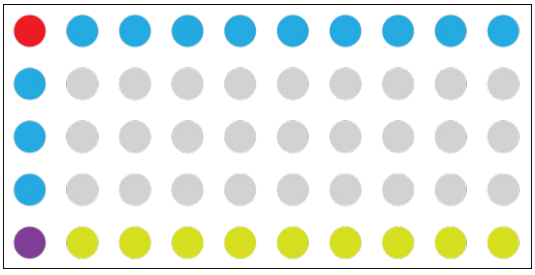

Abbildung 45: Raumtypen, Eigene Darstellung

Lokale Suche - First-Fit-Evaluierung

Um die Räume möglichst effizient zu platzieren, wird nach der First-Fit-Methode vorgegangen. Das heißt, es wir aus der Raumliste [0] des Departments [0] der Raum mit dem Index 0 einem Platzierungsversuch unterzogen. Sind die jeweils zutreffenden Bedingungen erfüllt, wird der Raum platziert und der Raum wird aus der Raumliste [n] entfernt ohne alle anderen Räume in der Raumliste zu prüfen.

Sind die Parameter nicht erfüllt, wird der Raum mit dem Index 1 einem Platzierungsversuch unterzogen. Dieser Prozess wird solange wiederholt, bis ein Raum platziert werden kann.

Wenn kein Raum so platziert werden kann, werden die betreffenden Punkte für die Dauer des Raumplatzierungsversuches auf “nicht verfügbar” gesetzt und der Prozess beginnt beim nächsten verfügbaren Punkt.

Dieser Prozess wird für jeden Raum des aktuellen Departments und für jedes existierende Departement wiederholt, bis alle Räume platziert sind (Abbildung 46) oder keine Startpunkte mehr vorhanden sind.

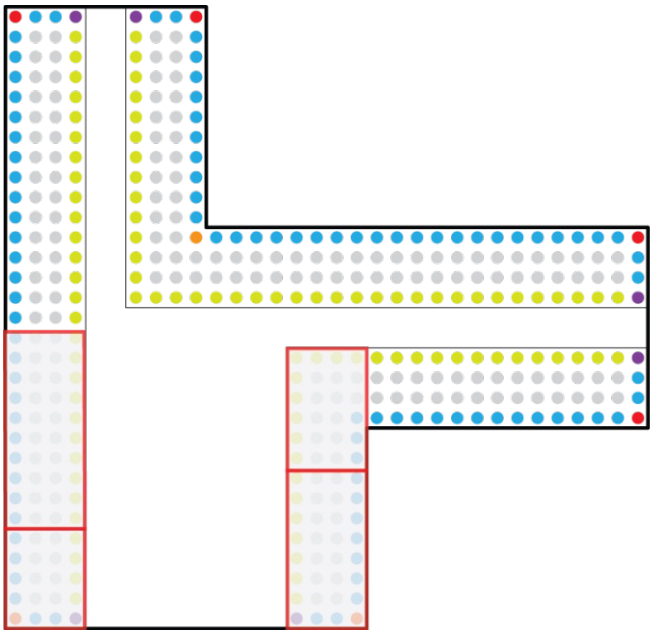

Abbildung 46: First-Fit, Eigene Darstellung

Danach startet der Prozess mit der Raumliste [1] für das Department [1] bis alle Departments [n] diesen Prozess durchlaufen haben.

3.2.4 Optimierungsverfahren

Wahl des Optimierungsverfahrens

Für das oben beschriebene Verteilungsverfahren wird ein Nachbarschaftssuchverfahren als Optimierungsstrategie angewendet. Der Hauptgrund ist, dass bei diesem Verfahren, im Gegensatz zu genetischen Verfahren, keine Start-Population gebildet werden muss. Da bei der Berechung jeder neuen Lösung wieder die Raumverteilung neu berechnet werden müssen, ist dafür ein hoher Rechenaufwand nötig.

Bei der Wahl eines genetischen Optimierungsverfahrens müsste, um brauchbare Lösungen zu bekommen, eine Startpopulation von 75 - 100 Lösungen berechnet werden, nur um dann mit dem eigentliche Optimierungsprozess zu starten.

Bei einem Nachbarschaftssuchverfahren wird mit wenig Rechenaufwand eine den Erfordernissen entsprechende Konstruktionsheuristik erstellt. Von dieser Startlösung wird dann durch strukturierte Nachbarschaftsoperationen eine neue Lösung berechnet und durch eine Zielfunktion evaluiert. Falls die neu berechnete

Lösung besser ist als die vorhande, wird diese durch die neue Lösung ersetzt und bildet die Grundlage für den nächsten Verbesserungsversuch.

Das Optimierungsverfahren setzt sich aus einer globalen Suche über die Zielfunktion und eine lokalen Suche im Generierungverfahren zusammen.

Zielfunktion

Die Zielfunktion für das Nachbarschaftssuchverfahren setzt sich aus einem Wert, der Auskunft darüber gibt, ob alle Räume platziert werden konnten, und einem Wert wieviel Fläche im Vergleich zur verfügbaren Fläche verbraucht wurde, zusammen. Aus den beiden Werten wird der Mittelwert errechnet, wodurch sich der Zielwert ergibt.

Ziel 1:

- Wenn die Anzahl der zu platzierenden Raume = Anzahl der platzierten Räume, dann Wert = 1, sonst Wert = 0,

Ziel 2:

- Anzahl der nicht verwendeten Matrix-Punkte / Summe der Soll-Flächen der Räume

Zielfunktion:

- (Ziel 1 + Ziel 2) / 2

Konstruktionsheuristiken

Neben der Wahl des Startpunktes des Raumverteilungsalgorithmus und der Reihenfolge der Räume ist der zweite wichtige Parameter für das Platzieren der Räume die Raumabfolge je Department. Hierfür bieten sich folgende Möglichkeiten an:

- Räume gleichmäßig nach Raumgrößen in Raumliste sortiert
- Räume absteigend nach Raumgröße in Raumliste sortiert
- Räume nach Raumtyp in Raumliste sortiert

Nachbarschaftsstruktur

- Kombination der Startpunkte

Zufällige Wahl der Startpunkte je Department

Bei 3 Departments ist das eine Kombination aus 3 Startpunkten, die zufällig aus der Liste der möglichen Startpunkte je Department ausgewählt werden. Dazu folgendes Beispiel:

_ Department (D_n) = 3

_ Mögliche Startpunkte (p) je Department (Abbildung 47)

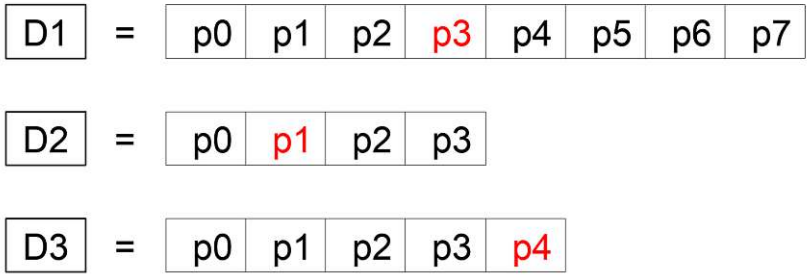

Abbildung 47: Listen möglicher Startpunkte je Department, Eigene Darstellung

Zufällige Auswahl zur Startpunkt-Liste (P) (Abbildung 48)

Abbildung 48: Ausgewählte Startpunkte, Eigene Darstellung

- Reihenfolge der zu platzierenden Räume in der Raumliste R_D je Department (Abbildung 49)

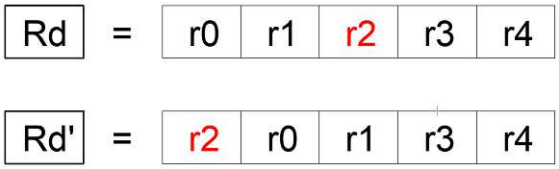

Abbildung 49: Raumlisten, Eigene Darstellung

Pseudocode

- Initialisiere
  _ Zufällige Auswahl von einem Startpunkt (P) je Department - Liste der Startpunkte (P)
  _ Konstruktionsheuristik - Festlegen der Startreihenfolge der Raumliste (R) je Department
- Wiederhole - bis Abbruchkriterium erreicht ist
  _ Startlösung (S)
  _ Zufällige Wahl der Startpunkte (P')
  _ Nehme zufälligen Raum in der Raumliste und füge den Raum an zufälligen Index in der Raumliste (R') ein
  _ Generiere Lösung (S')
    - Lokale Suche - Platziere Räume je Department
      _ Erstelle Testraum-Liste (TR) aus Raumliste (R')
      _ Wähle ersten Raum (r0) aus (TR) für Platzierungsversuch
      _ Wenn Kriterien erfüllt, dann platziere Raum (r0) - lösche (r0) aus (TR) und nehme nächsten Raum (r1)
      _ Wenn nicht - lösche (r0) nicht aus (TR) und nehme (r1)
      _ Wiederhole bis TR = 0
    - Wiederhole für alle Departments
- Evaluiere Lösung laut Zielfunktion - Wenn S' ist besser als S dann wird S' zu S, wenn nicht S = S

3.3 Implementierung

Softwareumgebung:	Rhinoceros3D, McNeel Associates
Programmiersprache	C# .NET
Betriebssystem:	Microsoft Windows 10 Home
Prozessor:	Intel64 Family 6 Model 158 Stepping 9 GenuineIntel ~4200 Mhz
BIOS:	LENOVO O2YKT45A

Bedingungen:

- 10 Versuche pro Fall
- 13 - 22 Räume / Funktionen je nach Typologie
- Immer 3 Departments
- Sollflächen aller Räume = Nettogeschossfläche
- Beginnend mit 95 % minimaler Flächenerfüllungsquote. Ist es nicht möglich mit 3 Versuchen alle Räume zu platzieren, wird die Flächenerfüllungsquote jeweils um 5 % gesenkt.
- Wenn innerhalb von 30 Iterationen keine bessere Lösung generiert werden kann, wird der Algorithmus abgebrochen.
- Raumtyp immer “Default Type”, das heißt, der Raum muss sowohl einen Punkt vom Type “Kantenpunkt zu Gebäudeumrisslinie” als auch einen Punkt vom Type “Kantenpunkt zu Erschließungsumrisslinei” enthalten.

3.4 Fallstudien

3.4.1 Büro als Block

10 Versuche, besten ausgewählt

Flächenerfüllungsquote: 95 %

Raumtiefe: 4 m

Nettogeschossfläche: 488m²

Fläche aller Räume: 485 m²

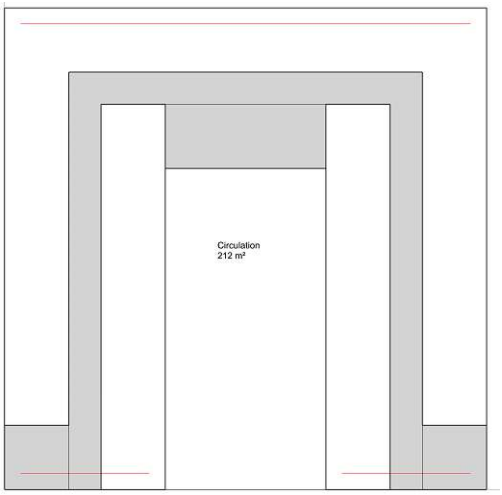

Abbildung 50: Gebäudegeometrie mit Attraktorlinien, Eigene Darstellung

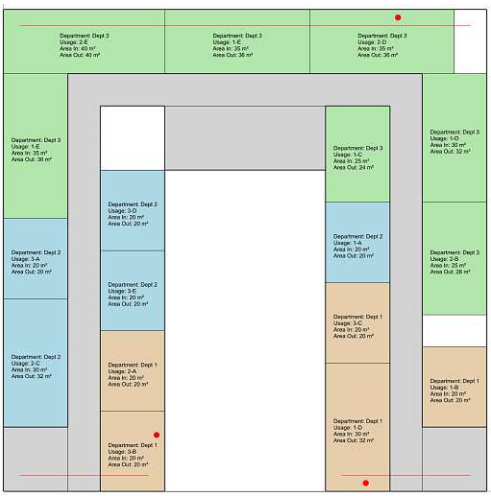

Abbildung 51: Iteration 1, Wert: 0,40

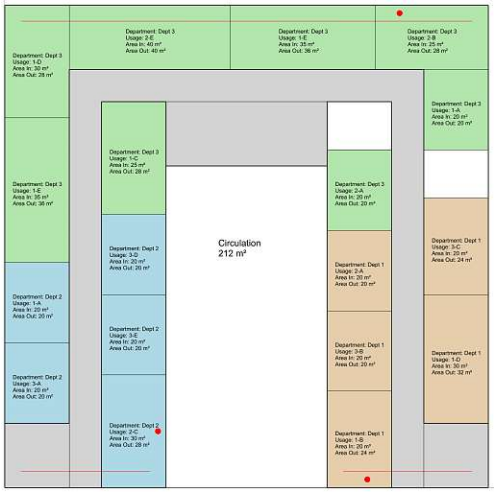

Abbildung 52: Iteration 1, Wert: 0,41

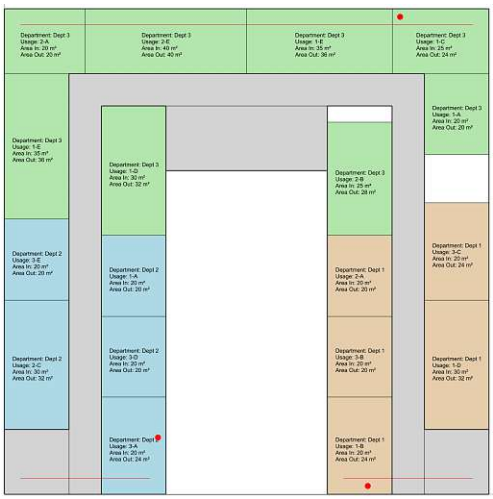

Abbildung 53: Iteration 1, Wert: 0,43

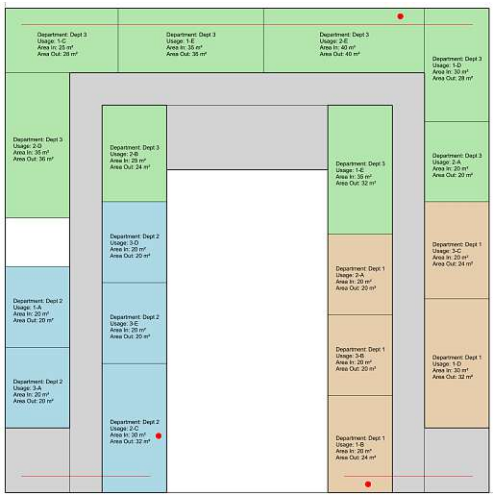

Abbildung 54: Iteration 3, Wert: 0,45

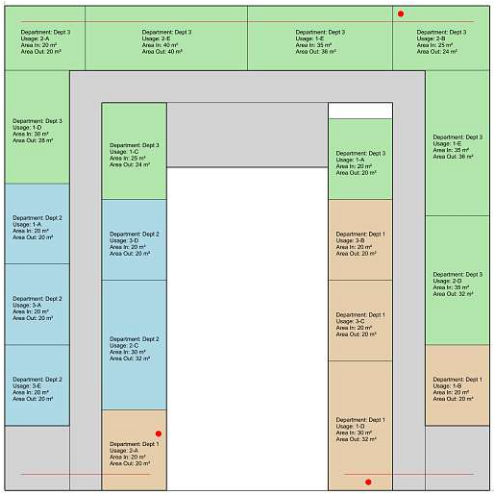

Abbildung 55: Iteration 16, Wert: 0,95

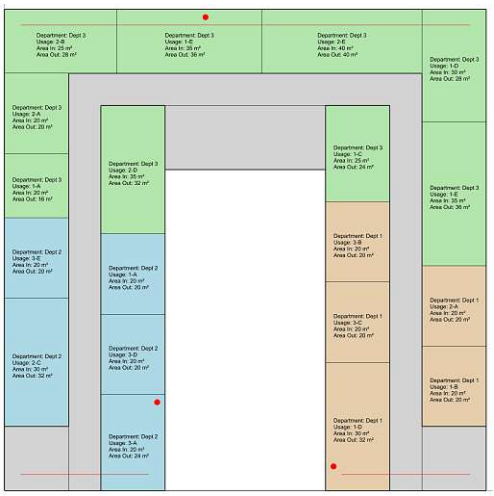

Abbildung 56: Iteration 34, Wert: 1,0

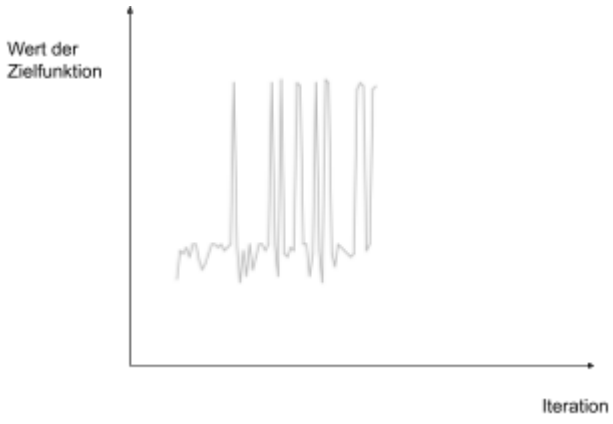

Abbildung 57, Diagramm aller 86 Iterationen, Eigene Darstellung

3.4.2 Hochhaus als Solitär

10 Versuche, besten ausgewählt

Flächenerfüllungsquote: 90 %

Raumtiefe: 6m

Nettogeschossfläche: 405m²

Fläche aller Räume: 405 m²

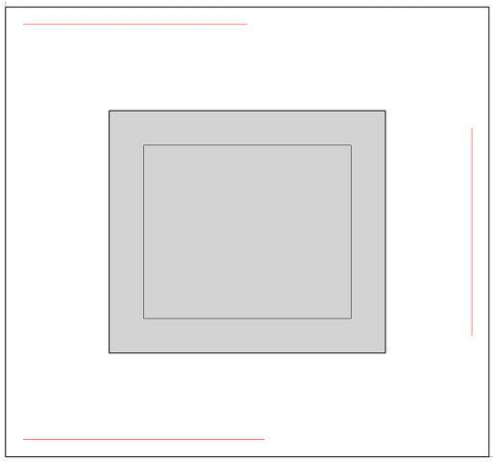

Abbildung 58: Gebäudegeometrie mit Attraktorlinien, Eigene Darstellung

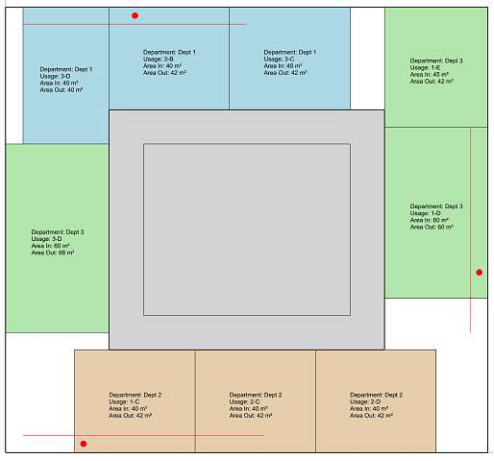

Abbildung 59: Iteration 1, Wert: 0,35

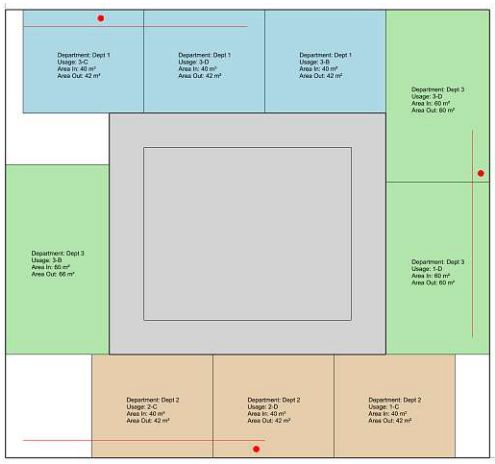

Abbildung 60: Iteration 2, Wert: 0,4

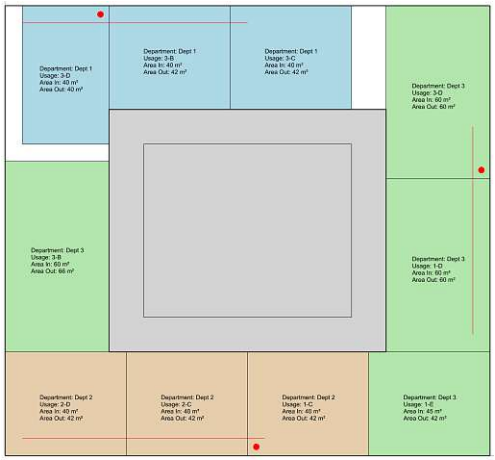

Abbildung 61: Iteration 17, Wert: 0,45

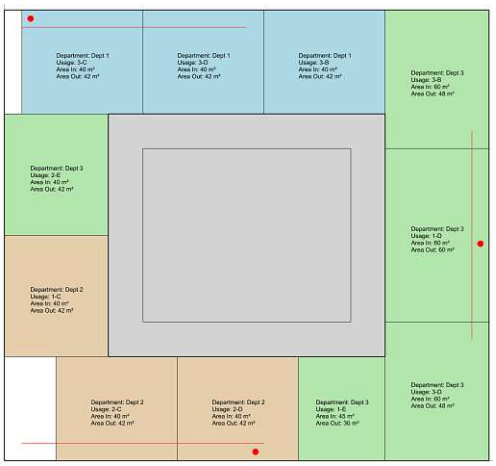

Abbildung 62: Iteration 32, Wert: 0,95

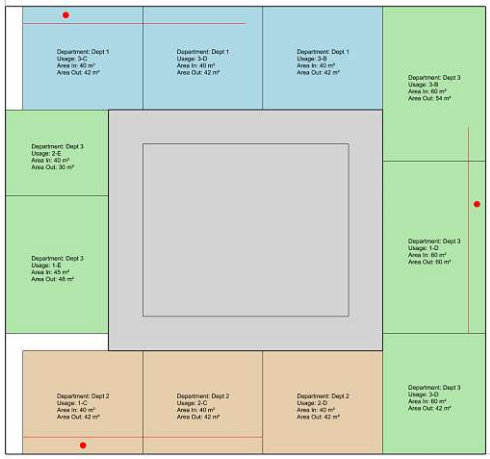

Abbildung 63: Iteration 60, Wert: 0,95

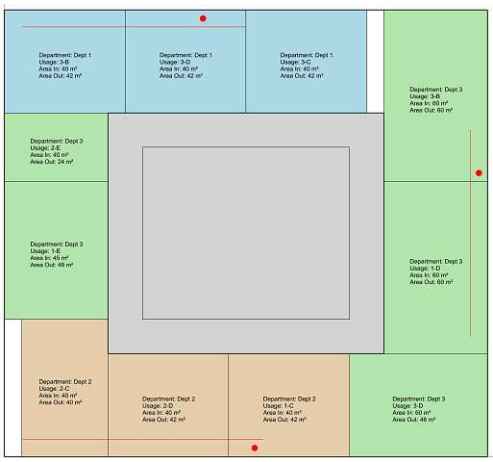

Abbildung 64: Iteration 89, Wert: 0,97

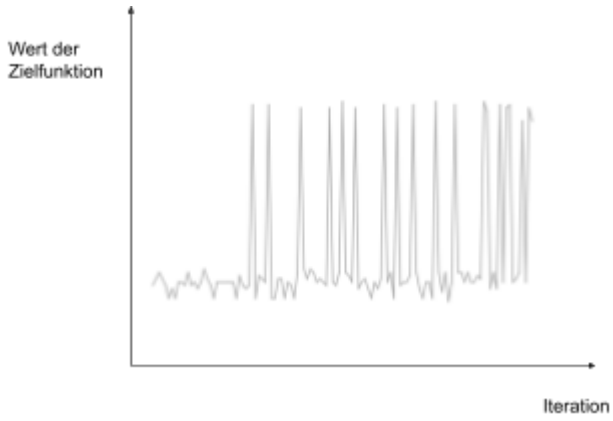

Abbildung 65: Diagramm aller 94 Iterationen, Eigene Darstellung

3.4.3 Altenheim als Block

10 Versuche, besten ausgewählt

Flächenerfüllungsquote: 95 %

Raumtiefe: 4m

Nettogeschossfläche: 405m²

Fläche aller Räume: 405 m²

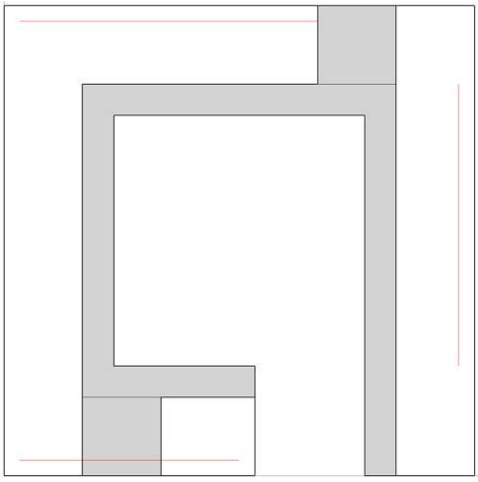

Abbildung 66: Gebäudegeometrie mit Attraktorlinien, Eigene Darstellung

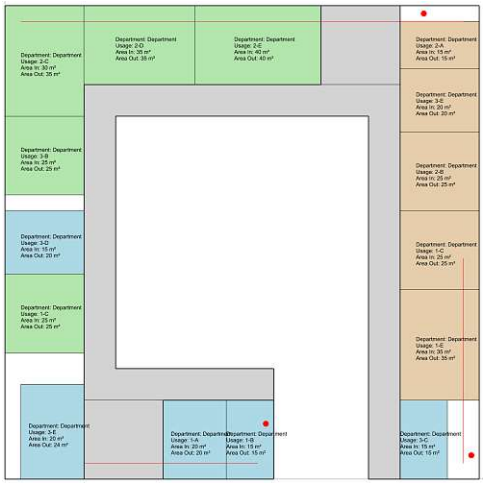

Abbildung 67: Iteration 1, Wert: 0,39

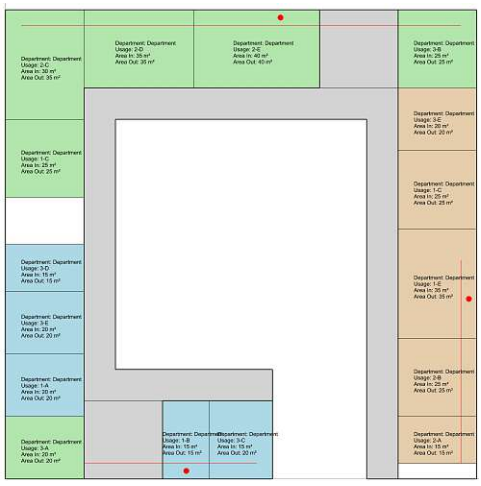

Abbildung 68: Iteration 2, Wert: 0,40

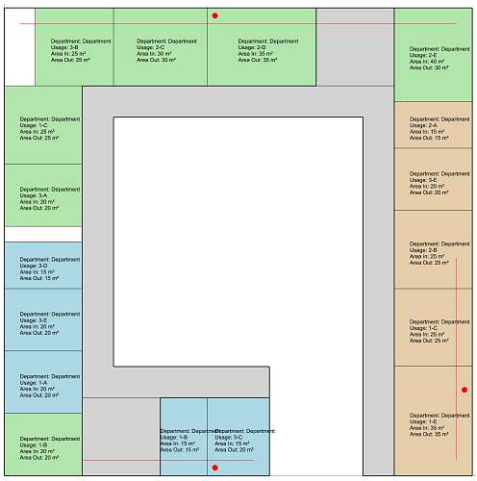

Abbildung 69: Iteration 22, Wert: 0,90

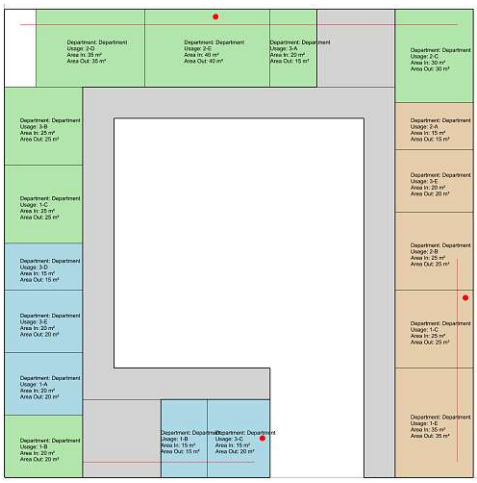

Abbildung 70: Iteration 29, Wert: 0,95

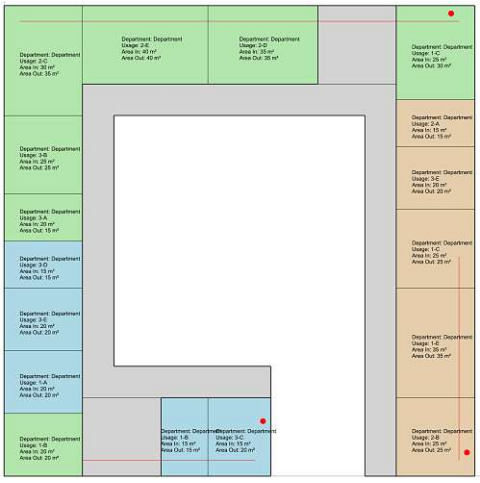

Abbildung 71:Iteration 49, Wert: 1,0

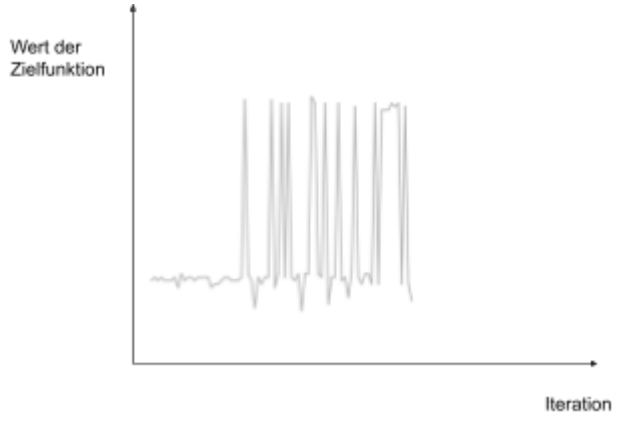

Abbildung 72, Diagram aller 79 Iterationen, Eigene Darstellung

3.4.4 Krankenhaus als Solitär

10 Versuche, besten ausgewählt

Flächenerfüllungsquote: 95 %

Raumtiefe: 4 m

Nettogeschossfläche: 352 m²

Fläche aller Räume: 350 m²

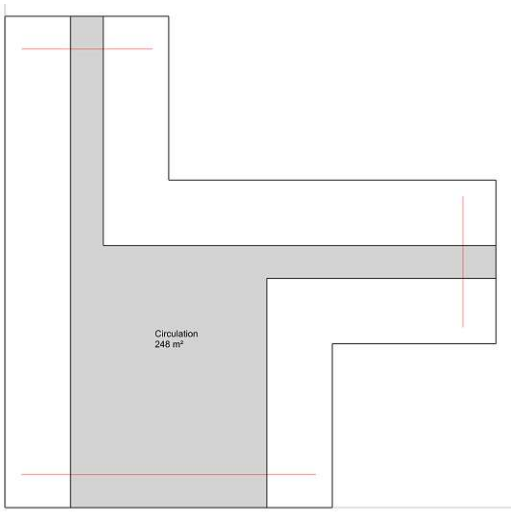

Abbildung 73: Gebäudegeometrie mit Attraktorlinien, Eigene Darstellung

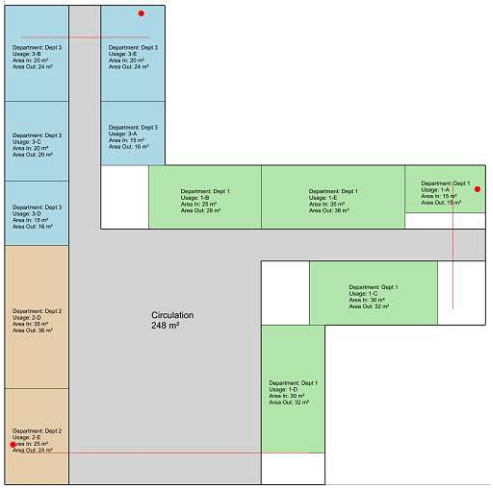

Abbildung 74:, Iteration 1, Wert: 0,40

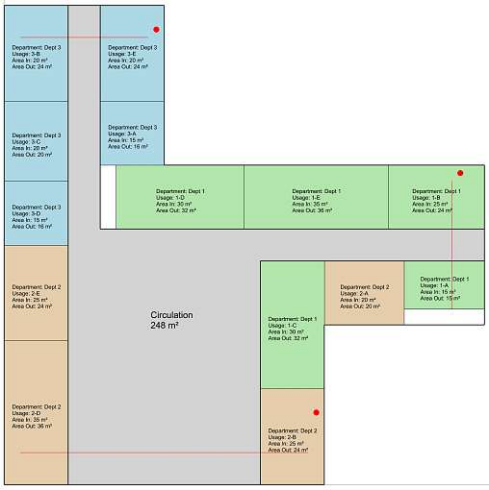

Abbildung 75: Iteration 2, Wert: 0,46

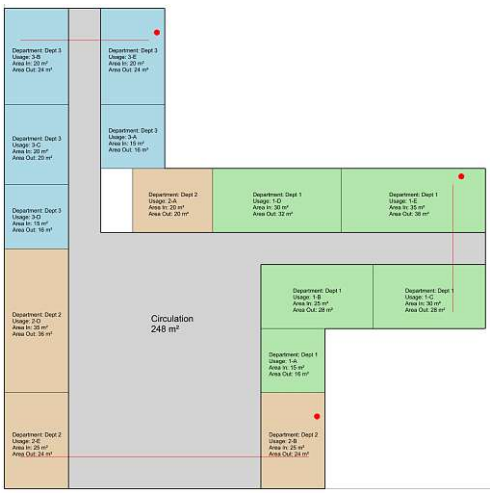

Abbildung 76: Iteration 6, Wert: 0,47

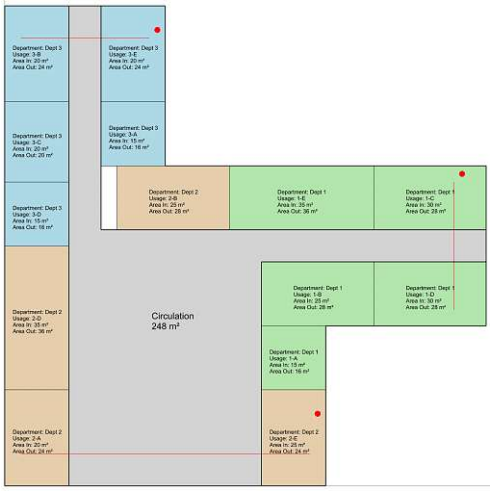

Abbildung 77: Iteration 9, Wert: 0,48

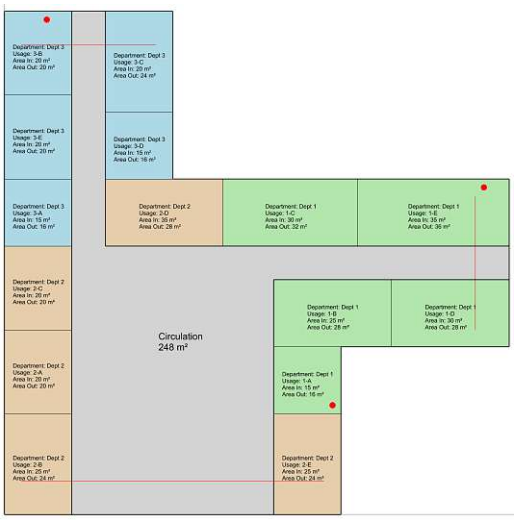

Abbildung 78: Iteration 27, Wert: 1,0

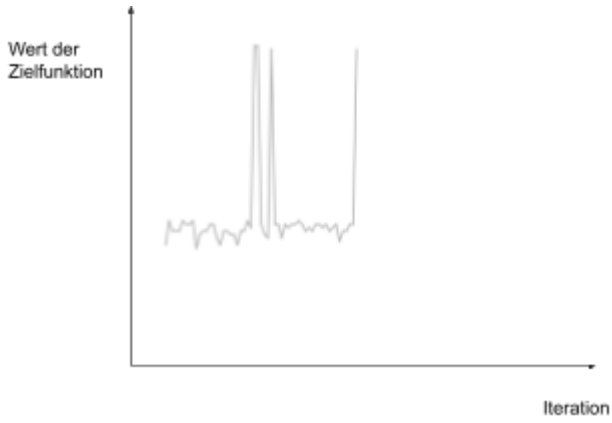

Abbildung 79: Diagramm aller 73 Iterationen, Eigene Darstellung

4 Diskussion

Problem

Bei der automatischen Generierung und Optimierung von Grundrissen stellt sich das Problem, ein vorhandenes Raumprogramm, das oft von Dritten entwickelt wurde, in einem Gebäude bzw. innerhalb einer Gebäudegeometrie zu platzieren. Näher betrachtet, geht es dabei darum, eine Ausgangsgeometrie möglichst optimal in Subgeometrien zu unterteilen. Ein Problem gängiger generativer Verfahren ist dabei, das Fehlen von interaktiven Steuerungsmöglichkeiten des räumlichen Allokationprozesses.

Konzept

In dieser Arbeit wurde ein automatisches Grundrissgenerierungsverfahren vorgestellt, das im Gegensatz zu oft vorgeschlagenen vollautomatischen Verfahren (Das et al, 2016) das Finden der Gebäudeform den ArchitektInnen überlässt und diese bei der Verteilung der Räume bzw. Funktionen innerhalb der Gebäudegeometrie unterstützt.

Konkret wird dabei das Raumprogramm in Funktionseinheiten bzw. Departments gruppiert und die jeweiligen Gruppen über Attraktoren innerhalb der Gebäudegeometrie zugewiesen. Diese Zuweisung kann leicht sowohl geometrisch, also über die Lage der Attraktoren, als auch Funktionell, also über das dem jeweiligen Attraktor zugewiesene Department, geändert werden.

Lösung

Wie in Kapitel 3.4 vorgestellt, wurde die angedachte Lösung einer Versuchsreihe mit verschiedenen Gebäudetypologien und -morphologien unterzogen. Grundsätzlich gilt wie für alle computergestützten Verfahren, dass der Output stark vom durch die Benutzenden erstellten Input abhängig ist. So bleibt es auch hier so, dass die grundlegenden Überlegungen zu Gebäudeform, Belichtung oder sinnvolle Gruppierung der Funktionen von den AnwenderInnen zu erbringen sind. Hier kommen neben rationalen Kriterien auch persönliche Vorlieben, kulturelle Usancen oder einfach nur der Zeitgeist als mathematisch nicht quantifizierbare Ziele bzw. Vorstellungen hinzu.

Auch ist zu beachten, dass automatische Grundrissgenerierungsverfahren nicht für jede Gebäudetypologie (Kapitel 1) sinnvoll eingesetzt werden kann. Unter sinnvoll wird in diesem Zusammenhang verstanden, dass sich daraus auch ein konkreter Vorteil für die Anwendenden ergibt. Meist meint man dabei eine signifikante Zeitersparnis, wodurch es möglich werden soll, in derselben Zeit wesentlich mehr Varianten prüfen zu können. Als Daumenregel kann dafür die Bruttogeschossfläche und die Anzahl der Räume angesehen werden. Je höher beide Parameter, desto wahrscheinlicher sind automatisierte Grundrissgenerierungsverfahren sinnvoll einsetzbar.

Sind diese Voraussetzungen jedoch erfüllt, so kann das vorgeschlagene Verfahren eine sinnvolle Ergänzung der Werkzeugpalette für ArchitektInnen sein. Wie eingangs erwähnt, ist jedoch auf einen validen Input zu achten. Die Stärke des vorgeschlagenen Verfahrens, nämlich die Departmentallokation mittels Attraktoren, ist gleichzeitig, falsch angewandt, die größte Schwäche. Es ist sowohl auf eine gewisse Distanz zwischen Attraktoren zu achten, als auch auf die Reihenfolge der Departments. In der Versuchsreihe hat sich gezeigt, dass es von Vorteil ist, jene Departments vorzureihen, die eine kleinere Raumanzahl aufweisen. So können die Räume der kleineren Departments relativ treffsicher in der Gebäudegeometrie platziert werden und die größeren Departments danach die Restflächen ausfüllen.

Ein weiteres Problem ergibt sich bei Gebäuden mit einer hohen Anzahl an Geschoßen durch den Plazierungsaufwand der Attraktoren. So sind zum Beispiel bei einem 20-stöckigen Gebäude mit jeweils 3 Departments pro Geschoß, 60 Attraktoren zu platzieren. Das selbe Problem tritt auf, wenn die Gebäudeform geändert wird und so die Lage der Attraktoren angepasst werden muss. Ein weiteres Problem in diesem Zusammenhang ist, dass in der momentanen Implementierung jedes Geschoß separat optimiert wird.

In zukünftigen Weiterentwicklungen ist daher auf die eben genannten Problematiken einzugehen. So ist anzudenken, die Attraktorenzuweisung direkt mit der Gebäudegeometrie zu verbinden und automatisiert auf andere Geschoße zu übertragen. Auch eine vertikale Verbindung von Departments sollte berücksichtigt werden.

Das vorgeschlagene Optimierungsverfahren, bestehend aus den Nachbarschaften zur Wahl des Startpunktes je Department und der Reihenfolge der zu plazierenden Räume je Department als global bewertete Parameter sowie der lokalen Suche bei der Raumplatzierung, zeigte in der Versuchsreihe zufriedenstellende Ergebnisse hinsichtlich Platzierungserfolg und die dafür benötigte Zeit. Im Schnitt konnte dies nach ca. 30 Iterationen erreicht werden, wobei eine Iteration ca. 0,25 Sekunden dauert und daher ein valides Ergebnis in unter 10 Sekunden zu erwarten ist. Der Minimalparameter für einen erfolgreichen Platzierungsversuch ist, dass alle Räume platziert werden konnten. Durch die Festlegung des Raumtyps kann der Platzierungsversuch direkt während der lokalen Suche evaluiert und so effizient nicht zulässige Raumplatzierungen sofort ausgeschossen werden.

Ein weiterer Entwicklungsschritt könnte sein, auch eine separate Departmentevaluierung durchzuführen. So zeigte sich während der Versuchsreihe, dass teilweise Lösungen, bei denen die Räume eines Departments durch Räume eines anderen Departements unterbrochen wurden, einen höheren Zielfunktionswert aufwiesen, als Lösungen, bei denen dies nicht der Fall war. Durch eine Departmentevaluierung können hier noch Verbesserungen erzielt werden.

Zum validen Input gehört auch, dass die Gesamtfläche der zu platzierenden Räume pro Geschoß ähnlich der jeweiligen verfügbaren Nettogeschossfläche ist. Ist dies nicht der Fall, wird der Algorithmus zwar ausgeführt, jedoch kein valides Ergebnis erzielt und nach dem festgelegten Terminierungkriterien, in unserem Beispiel wenn innerhalb von 30 Iterationen keine bessere Lösung zustande kommt, abgebrochen.

5 Schlussbetrachtung

In dieser Arbeit wurde ein Verfahren zur automatischen Generierung und Optimierung von Grundrissen vorgestellt, bei dem das Problem der mangelhaften Steuerbarkeit des räumlichen Allokationsprozesses adressiert wird.

Eingangs wurde dabei erläutert, für welche Entwurfsaufgaben sich automatische Grundrissgenerierungsverfahren eignen und gängige generative wie optimierende Methoden vorgestellt. Überblicksmäßig wird auch auf grundlegende Logiken der Informatik im Allgemeinen eingegangen.

Die vorgestellte Grundrissgenerierungsmethode überlässt das Finden der Gebäudeform den Architekturschaffenden, ermöglicht es jedoch, die zu platzierenden Räume bzw. Funktionen in Funktionsgruppen einzuteilen und diese dann mittels korrespondierenden Attraktoren innerhalb der Gebäudegeometrie zuzuweisen.

Über einen agentenbasierten Verteilungsalgorithmus werden anschließend die Räume innerhalb der Gebäudegeometrie verteilt, wobei schon im generativen Prozess durch lokale Suche nicht valide Raumplatzierungen ausgeschlossen werden. Darauf folgend wird durch Optimierungsverfahren, bei denen die Kombination der Startpunkte je Department und die Abfolge der zu platzierenden Räume in jeder Iteration variieren, nach einer möglichst guten Lösung gesucht.

In den Fallstudien wurde das Verfahren an verschiedenen Gebäudetypologien und -morphologien ausprobiert und grundsätzlich zufriedenstellende Ergebnisse erzielt. So war es möglich, in sehr kurzer Zeit von im Schnitt 30 Iterationen einen validen Grundrissvorschlag zu erhalten und daher alle zu platzierenden Räume mit den definierten Raumparametern zu platzieren.

In der Diskussion wurde auf die Stärken und Schwächen des hier behandelten Verfahrens eingegangen. Generell lässt sich feststellen, dass die richtige Anwendung des Werkzeugs hauptausschlaggebend für einen sinnvollen Output ist. Weiters wurden einige Verbesserungsmöglichkeiten der Attraktorenzuweisung und des Optimierungsverfahrens diskutiert.

Allgemein kann man festhalten, dass es weder Ziel des vorgestellten Werkzeuges noch praktischen möglich ist, die Architekturschaffenden in der Entwurfsphase zu ersetzen. Intention ist, eine Möglichkeit aufzuzeigen, wie automatische Grundrissgenerierungsverfahren die Arbeit von Planenden unterstützen und sie von repetitiven Aufgaben entlasten kann.

So beinhaltet der architektonische Entwurf mehr als die Optimierung eines Raumprogramms innerhalb einer Gebäudegeometrie, sondern spannt einen Bogen über viele Fachdisziplinen vom Ingenieurwesen über Kunst, Ökonomie, Philosophie und Soziologie, die mit deterministischen Verfahren alleine nicht zu fassen sind.

6 Literaturverzeichnis

Beardwood, J.; Halton, J.H.; Hammersley, J.M.; 1959: *The Shortest Path Through Many Points*, Proceedings of the Cambridge Philosophical Society, 55 (4): 299–327.

Bentley, John; 1975: *Multidimensional binary search trees used for associative searching*. In: Communications of the ACM, New York, NY, USA, S. 509–517.

Coates, P. et al.; 2005: Generating architectural spatial configurations. Two approaches using Voronoi tessellations and particle systems, in Proceedings of the VIII Generative Art International Conference - GA2005. Milan, Italy.

Cook, Stefan A.; 1971: *The Complexity of Theorem Proving Procedures*. Proceedings Third Annual ACM Symposium on Theory of Computing, S. 151 - 158.

Das Subhajit, Day Colin, Hauck Anthony, Haymaker John, Davis Diana, 2016: *Space Plan Generator - Rapid Generation and Evaluation of Floor Plan Design Options to Inform Decision Making*, Acadia Conference, Ann Arbor, Michigan, October 2016.

Darwin, Charles; 1859: *On the origin of species by means of natural selection*, or the preservation of favoured races in the struggle for life. London: Murray. [1st ed.]

De Jong, Kenneth; 1975: *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems*, PhD thesis, University of Michigan.

Feo, Thomas A.; Resende, Mauricio G. C.; 1989: *A probabilistic heuristic for a computationally difficult set covering problem*. Operations Research Letters. 8 (2): S. 67–71.

Garey, Michael R.; Johnson, David S.; 1979: *Computers and Intractability. A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York, Kapitel 5: NP-Hardness.

Gips, James; Stiny George; 1971: *Shape grammars and the generative specification of painting and sculpture*. In: Information Processing, S. 1460–1465.

Hansen, P.; Jaumard, B.; Mladenović, N.; Parreira, A.; 2000: *Variable neighborhood search for weighted maximum satisfiability problem*. Les Cahiers du GERAD G–2000–62, HEC Montréal, Canada.

Johnson, David S.; 1973: *Near-optimal bin packing algorithms*, Massachusetts Institute of Technology.

Kirkpatrick, S.; Gelatt, Jr C. D.; Vecchi, M. P.; 1983: *Optimization by Simulated Annealing. Science*. 220 (4598): 671–680.

Knecht Katja, König, Reinhard, 2010: *Generating Floor Plan Layouts with K-d Trees and Evolutionary Algorithms*, GA2010 – XIII Generative Art Conference – Milano, Italy, S. 238-253.

Knuth Donald E., 1976: *Big Omicron and big Omega and big Theta*. In: SIGACT News. Band 8, Nr. 2. ACM, S. 18–24.

Knuth Donald E., 1997: The Art of Computer Programming, Volume 1: Fundamental Algorithms, Third Edition. Addison-Wesley.

Knuth Donald E., 1998: The Art of Computer Programming, Volume 3: Sorting and Searching. The Art of Computer Programming. 3 (2nd ed.). Reading, MA: Addison-Wesley Professional.

Korte, Bernhard; Vygen, Jens 2006: *Bin-Packing*. Combinatorial Optimization: Theory and Algorithms. Algorithms and Combinatorics 21. Springer. S. 426–441.

Landau Edmund, 1909: Handbuch der Lehre von der Verteilung der Primzahlen, S. 31 und S.61, Berlin

Neufert, Ernst, 2012: *Bauentwurfslehre - Grundlagen, Normen, Vorschriften über Anlage, Bau, Gestaltung, Raumbedarf, Raumbeziehungen, Maße für Gebäude, Räume, Einrichtungen, Geräte mit dem Mensch als Maß und Ziel*. 40. Auflage Wiesbaden, Springer Vieweg 2012

Mladenović, Nenad; Hansen, Pierre; 1997: *Variable neighborhood search*. Computers and Operations Research 24, Nr. 11, S. 1097-1100.

Pearl, Judea; 1984: *Heuristics: intelligent search strategies for computer problem solving.* United States: Addison-Wesley Pub. Co., Inc., Reading, MA. S. 3.

Schneider, Klaus Jürgen; 2020: Bautabellen für Architekten, Entwurf - Planung - Ausführung, Reguvis Verlag, Köln 2020.

Ullman, J. D. 1971: *The performance of a memory allocation algorithm*. Technical Report 100 Princeton Univ.

Von Neumann John, 1963: The general and logical theory of automata. In A. Taub(ed.) J. von Neumann Collected Works S. 288 – 328.

Wooldridge Michael, 2002: *Intelligent Agents: The Key Concepts*. Multi-Agent-Systems and Applications II, 9th ECCAI-ACAI/EASSS 2001, AEMAS 2001, HoloMAS 2001, Selected Revised Papers, Liverpool, S. 3 - 43.

Wolpert David H., Macready William G., 1995: *No free lunch theorems for search*. Vol. 10, Technical Report SFI-TR-95-02-010, Santa Fe Institute.

7 Abbildungsverzeichnis