UNIVERSITÄT OSNABRÜCK

Institut für Informatik

Bachelorarbeit

Implementation und Evaluation eines genetischen Algorithmus für Node-Placement Strategien in Multi-Hop Netzwerken

Joris Jan Clement

13. August 2016

Erstgutachter: Prof. Dr. N. Aschenbruck
Zweitgutachter: Prof. Dr. S. Knust

Kurzfassung & Abstract

Deutsch Kabellose Netzwerke haben eine wichtige Rolle in unserer Gesellschaft. Sie kommen in vielen verschiedenen Situationen zum Einsatz, hauptsächlich bei WLAN, Sensoranwendungen oder dem Mobilfunk. In bestimmten Szenarien, z. B. einem Katastrophenfall, ist es wichtig, auf eine infrastrukturlose Kommunikation zurückgreifen zu können. Dafür können drahtlose Multi-Hop Netzwerke genutzt werden. In dieser Bachelorarbeit wird ein genetischer Algorithmus entworfen, dessen Ziel es ist, eine gute Platzierung für Multi-Hop Netzwerke im stätdtischen Gebiet zu finden. Dieses heuristische Verfahren wird verwendet, weil die Berechnung einer guten Platzierung für realistischere, größere Szenarien sehr zeitaufwendig ist. Dabei wird einerseits als Grundlage eine simple kreisförmige Signalausbreitung genutzt, aber andererseits realistischere Daten, die durch einen Ray Launcher, der basierend auf Kartenmaterial die Signalausbreitungsdaten berechnet, verwendet. Zudem wird der entwickelte genetische Algorithmus mit anderen Ansätzen verglichen und dabei verdeutlicht worin die Vor- und Nachteile bestehen. Es wird ebenfalls gezeigt, dass die Wahl des Ausbreitungsmodells einen deutlichen Unterschied auf die Lösung hat.

English Wireless networks play an important role in our society. These come into play in different situations, mainly with WLAN, Sensor applications or mobile transmission. In certain situations, f.e. a disaster, it is of importance to have access to non-infrastructural communication, to support quick and effective actions. Wireless multi-hop networks can be used for this. This barchelor dissertation shows the design of a genetic algorithm with the goal to optimize the positioning of multi-hop networks in urban areas. This heuristic approach is used because calculating the optimal positioning is very time consuming, esspecially in case of large and realistic scenarios. On the one hand a simple circle shaped signal distribution basis is used, on the other hand realistic data via a Ray Launcher, which is based on map and signal distribution data calculations. The genetic algorithm compares this with other approaches and identifies the pro's and con's, and that choosing the propagation model will lead to a significant impact on the solution.

Inhaltsverzeichnis

1 Einleitung

In unserer Gesellschaft nehmen drahtlose Netzwerke eine immer bedeutendere Rolle ein. Die gebräuchlichste Anwendung dieser Netze findet sich im heimischen WLAN Netzwerk, mit dem man eine Verbindung zum Internet herstellen kann. Es gibt aber auch ganz andere Szenarien, in denen drahtlose Netzwerke benötigt werden. Ein Beispiel dafür ist ein Art von Katastrophenszenario, in dem die reguläre Kommunikation nicht mehr möglich ist. Weil auch wichtige Infrastruktur heutzutage auf digitaler Kommunikation beruht, ist es in solchen Situationen wichtig den Datenaustausch aufrecht zu erhalten. Eine Methode um sich auf solch einen Fall vorzubereiten sieht so aus, dass man mehrere Knoten, die miteinander über die Fläche kommunizieren, so verteilt, dass die kritischen Systeme, z.B. die Ambulanzen der Rettungsdienste, miteinander kommunizieren können. Das Wichtige dabei ist, dass die Knoten autark arbeiten, so dass sie nicht vom regulären Netz abhängig sind. So ein Knoten könnte z.B. eine simple Funkstation oder auch ein Wi-Fi Router sein, der in der Lage ist Signale zu senden und zu empfangen. Die nächste Frage ist nun wie man die Knoten auf einer Fläche, z.B. einer Stadt, verteilt, um z.B. eine möglichst gute Abdeckung der Fläche unter Berücksichtigung der Anzahl der Knoten des Netzwerkes zu gewährleisten.

Um dieses exemplarische Beispiel lösen zu können, werden in Kapitel 2 zuerst die Grundlagen zu dieser Art von drahtlosen Netzwerken, der Signalausbreitung und ihrer Berechnung und genetischen Algorithmen besprochen. Genetische Algorithmen dienen in dieser Arbeit als Ansatz zur Lösung dieses Optimierungsproblems. Desweiteren werden die Tools RaLaNS (Abschnitt 2.2.2), welches für die Berechnung der Signalausbreitung auf der Fläche benötigt wird, und DEAP (Abschnitt 2.4.1), ein Framework für evolutionäre Algorithmen, vorgestellt.

RaLaNS ist dabei von entscheidender Bedeutung für die Qualität der Lösungen für das genannte Node-Placement Problem, weil viele der in Kapitel 3 vorgestellten verwandten Verfahren zum Lösen von Node-Placement Problemen auf einer kreisförmigen Signalausbreitung beruhen. RaLaNS hingegen berechnet die Signalausbreitung deutlich realistischer (Abschnitt 2.2).

Danach wird die eigentliche Umsetzung gezeigt (Kapitel 4). Darin wird vor allem auf konzeptueller Ebene erklärt wie der genetische Algorithmus, insbesondere die Berechnung der Fitness, die Multi-Hop Fähigkeit des Netzes und die Bereitstellung der Signalausbreitungsdaten funktionieren. Es wird ebenfalls die Umsetzung der Algorithmen, die zum Vergleich dienen, vorgestellt.

In der Evaluation (Kapitel 5) wird der entstandene genetische Algorithmus Szenarien mit anderen Algorithmen verglichen und die Unterschiede zwischen der Berechnung mit kreisförmiger Signalausbreitung und RaLaNS gezeigt.

Abschließend werden die Erkenntnisse zusammengefasst und es wird ein Ausblick auf weiterführende Arbeiten gegeben (Kapitel 6).

2 Grundlagen

Dieses Kapitel gibt einen Einblick in die für diese Arbeit relevanten Themengebiete. Dazu wird zunächst eine Einführung in die Kommunikation in drahtlosen Multi-Hop Netzwerken gegeben (Abschnitt 2.1). Damit die Berechnung für das Node-Placement hier durchgeführt werden kann, wird die Signalausbreitung näher erläutert (Abschnitt 2.2), die von wichtiger Bedeutung ist, um zu ermitteln, ob es eine Verbindung zwischen den Knoten geben kann. Dabei wird auch die Simulation der Signalausbreitung mit Hilfe von RaLaNS vorgestellt. Danach folgt eine Einführung zu Node-Placement, in der das Problem und dessen Komplexität näher beschrieben wird (Abschnitt 2.3). Abschließend wird in das Thema der genetischen Algorithmen eingeführt, welches in dieser Arbeit als Methode dienen soll, um das Node-Placement Problem zu lösen (Abschnitt 2.4). Dabei wird auch das Framework, das in dieser Arbeit dafür verwendet wird, vorgestellt.

2.1 Kommunikation in drahtlosen Multi-Hop Netzwerken

Bei Netzwerken unterscheiden man generell zwischen drahtlosen und drahtgebundenen Netzen. Ein bekannter Vertreter der drahtgebundenen Netze ist das Local Area Network (LAN). Die drahtlose Variante dazu nennt sich Wireless Local Area Network (WLAN). Die Topologie, also die Struktur der Netzwerkteilnehmer untereinander, eines typischen WLAN Netzwerkes ist sternenförmig [1] (Abbildung 2.1).

Bei so einem Netzwerk ist der Router die Basisstation, auch Access Point (AP) genannt, weil über ihn alle Daten geleitet werden müssen. Durch den Router ist dann die Verbindung ins Internet möglich, die dann meist kabelgebunden weiterläuft. Das Routing, also das Bestimmen des Weges für ein Datenpaket von Sender zu Empfänger, ist hier sehr einfach, weil es nur eine Möglichkeit für den Weg gibt, der über den Link vom Sender zum Router und dann über den Link vom Router zum Empfänger führt. Das bedeutet ebenfalls,

Abbildung 2.1: Netzwerk mit Stern-Topologie, erstellt mit [0]

dass WLAN für sich alleine betrachtet ein Single-Hop Netzwerk ist, weil das Paket nur über ein Gerät, nämlich den Router, läuft [2, S. 37].

In dieser Arbeit werden aber drahtlose Multi-Hop Netzwerke (Abbildung 2.2) behandelt. Diese unterscheiden sich darin, dass in Ihnen ein Datenpaket über mehrere Router, allgemein auch Stations (STAs) oder Knoten genannt, wandert bis es am Empfänger angekommen ist. Die Anzahl der Hops für ein Paket erhöht sich also mit jedem Gerät, das auf dem Weg passiert werden muss, was der Grund ist, warum sie Multi-Hop Netzwerke heißen [2, S. 37].

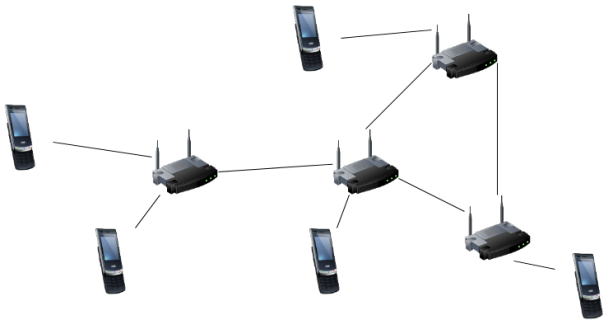


Abbildung 2.2: beispielhaftes Multi-Hop Netzwerk, erstellt mit [0]

Alle hier behandelten Multi-Hop Netzwerke sind Ad-Hoc Netze. Das bedeutet, dass sie nicht auf schon bestehender Infrastruktur wie drahtgebundene Leitungen, etc. beruhen und so weitgehend unabhängig sind [2, S. 36] [3, S. 19]. Sie werden auch als Wireless Ad Hoc Network (WANET) bezeichnet. Die Multi-Hop Netzwerke lassen sich grob in stationäre und mobile Netze einteilen. Der Unterschied besteht darin, dass bei mobilen Netzwerken die Knoten, also die Netzwerkteilnehmer, beweglich sind, z. B. fahrende Funkstation. Diese Netze werden Mobile Ad Hoc Network (MANET) genannt [4]. Andere spezielle

[0] www.draw.io

Netze, die eine Form von drahtlosen Multi-Hop Netzwerken darstellen, sind Wireless Mesh Networks (WMNs), Wireless Sensor Networks (WSNs) und Vehicular Ad Hoc Networks (VANETs).

Ein WMN ist ein stationäres drahtloses Ad-Hoc Netzwerk, das sich für den Datentransport selbst konfiguriert und organisiert [5]. Sie sind also eine Unterklasse der drahtlosen Multi-Hop Netzwerke, also der WANETs.

Die VANETs sind eine Spezialisierung der MANETs, bei der die Teilnehmer des Netzes hauptsächlich Fahrzeuge sind [6].

WSNs sind eine Erweiterung der WANETs, bei der jeder Knoten seine Umgebung untersucht. Die Daten der Untersuchung werden direkt oder über Multi-Hop an eine Basis-Station gesendet und dort gesammelt [7].

Bei Multi-Hop Netzwerken allgemein ist das Routing nun auch nicht mehr trivial, wie beim sternenförmigen Netz, weil es mehrere mögliche Transportwege gibt. Es gibt dafür einige verschiedene Routing-Protokolle, um ein Datenpaket an sein Ziel zu befördern, z. B. die in [8] untersuchten. Darauf wird in dieser Arbeit aber nicht näher eingegangen.

2.2 Signalausbreitung

Essentiell wichtig für das Node-Placement ist die Signalausbreitung zwischen den Knoten. Damit eine Verbindung zwischen zwei Knoten eines Netzwerkes zustande kommen kann, muss das Signal des Senders beim Empfänger ankommen. Dabei ist wichtig, dass dieses Signal noch eine ausreichende Signalstärke hat, wenn es am Empfänger ankommt. Es muss also eine gewisse Grenzsignalstärke überschreiten (Abschnitt 2.2.3).

2.2.1 physikalische Grundlagen

Die Signale zwischen den Knoten sind elektromagnetische Wellen mit einer bestimmten Wellenlänge. Die Ausbreitung dieser Wellen unterliegt gewissen physikalischen Gesetzen. Allgemein ist zu beachten, dass die Stärke, also die Amplitude, einer Welle mit steigender Entfernung abnimmt. Wenn man nun annimmt, dass es keine Hindernisse wie Gebäude und sonstigen abweichenden Einflüsse z. B. durch die Atmosphäre gibt, dann kann man der Ausbreitung das Modell von Friis zu Grunde legen [9]. Mit der Friis-Übertragungsgleichung (Gleichung 2.1) lässt sich der Übertragungsverlust ausrechnen.

$$\frac{P_r}{P_t} = \frac{A_r \cdot A_t}{r^2 \cdot \lambda^2} \tag{2.1}$$

In der Gleichung stehen A_r bzw. A_t für die Fläche der Antenne, die zum Senden bzw. Empfangen, genutzt werden kann. Der Abstand zwischen Sender

und Empfänger r und die Wellenlänge λ des Signals tragen ebenfalls zur Signalstärke bei. Die Signalstärke ist der Quotient aus der Leistung des Signals am Empfänger P_r und der Leistung am Sender P_t. Sie ist also eine relative Einheit und wird in dB ausgedrückt [3, S. 278]. Gleichung 2.2 zeigt wie man sie aus dem Quotient der Leistungen berechnen kann. Weil die Signalstärke am Empfänger geringer ist als am Sender, ist der Wert immer negativ.

$$signal[dB] = 10 \cdot \log_{10}(\frac{P_r}{P_t}) \tag{2.2}$$

Das Ergebnis ist also abhängig von den Eigenschaften der Antennen, der gesendeten Wellenlänge und natürlich dem Abstand zwischen den beiden Antennen. Wenn man also nur die gleichen Antennen und die gleiche Frequenz in einem Netzwerk einsetzt, hängt der Signalverlust laut der Formel von Friis nur von der Distanz ab. Selbst das ist noch vereinfacht, weil die Ausrichtung der Antennen nicht beachtet wird. Das Problem mit diesem Modell ist aber, wie schon erwähnt, dass es sich nicht auf Städte und andere ähnliche Gebiete anwenden lässt.

Wenn nämlich eine Welle z. B. auf ein Gebäude trifft, dann treten ggf. physikalische Effekte wie Reflexion, Brechung, Absorption und Beugung auf [10, S. 7ff]. In Folge davon kann es auch noch zur Interferenz kommen.

Unter **Reflexion** versteht man das Zurückwerfen einer Welle durch das Auftreffen der Welle an einem anderen Medium, z. B. an einem Gebäude. Die Welle wird abhängig von den Eigenschaften der beiden Medien zu einem bestimmten Teil reflektiert und zu einem anderen Teil gebrochen. Unter **Brechung** versteht sich der Übergang der Welle inkl. einer Richtungsänderung in ein anderes Medium, in Städten also den Übergang in der Regel in ein Gebäude. Die **Beugung** kann in diesem Szenario an Kanten von Gebäuden auftreten. **Interferenz** kann bei mehreren elektromagnetischen Wellen auftreten. Sie sorgt dafür, dass sich Wellen entweder gegenseitig abhängig von ihrer Schwingung verstärken oder abschwächen. Damit eine möglichst genaue Berechnung der Signalstärke erfolgen kann, müssen all diese genannten Faktoren mit einbezogen werden.

Ein Tool um diese ganzen Verhaltensweisen von Wellen zu berücksichtigen, ist RaLaNS.

2.2.2 Simulation mit RaLaNS

Ray Launching Based Progagation Loss Model for ns-3 (RaLaNS) [10, 11] ist ein Simulator, der auf Ray Launching basiert.

Beim Ray Launching oder Ray Tracing werden Strahlen in einer Welt von bestimmten Orten abgeschickt oder auch zurückverfolgt. Meist wird dabei

berechnet an welchen Orten die Strahlen landen bzw. wo sie herkommen. Eingesetzt werden Ray Launcher vor allem in der Computergrafik z. B. als globale Beleuchtungsmodelle [12] so wie in der Berechnung der Signalausbreitung in Netzwerken [10, S. 16].

Für diese Software bedeutet das einfach ausgedrückt, dass vom Sender bzw. den Sendern Strahlen ausgesendet werden. Von den ausgesendeten Signalwellen wird ermittelt welche beim Empfänger bzw. den Empfängern ankommen. Aus der Stärke der Strahlen am Empfänger wird dann die endgültige Signalstärke berechnet.

Bei RaLaNS ist es so, dass jeder Sender als punktförmige Antenne betrachtet wird, die in keine Richtung ausgerichtet ist. Das ist eine vereinfachende Annahme. Diese Sender senden von sich aus Strahlen gleichverteilt in alle Richtungen aus. Wenn so ein Strahl nun z. B. auf ein Gebäude trifft, dann müssen die Folge-Strahlen dieses Strahles berechnet werden. RaLaNS berücksichtigt dabei die Reflexion, Streuung und Beugung von Wellen. In der Regel entstehen aus einem auftretenden Strahl mehrere Strahlen, die sich in unterschiedliche Richtungen weiterbewegen. Die Brechung wird nicht berücksichtigt, weil angenommen wird, dass die Wände der Gebäude dick und zumindest in Europa in der Regel aus Backstein sind, so dass der durchgelassene Teil keine Relevanz für das Ergebnis hat. Dadurch, dass mehrere Strahlen von den Antennen ausgesandt werden, kann auch Interferenz auftreten, die in der Simulation beachtet wird.

Damit RaLaNS überhaupt arbeiten kann, benötigt es Kartenmaterial, das die geographischen Eigenschaften des Gebietes gut abbildet. Ein geeigneter Typ dafür ist Open Street Maps (OSM) [1] [10, S. 66]. In diesen Karten sind die Grundflächen von Gebäuden enthalten. Wenn man nun noch eine Höhe für die Gebäude festlegt, kann man die Simulation für das entsprechende Gebiet durchführen.

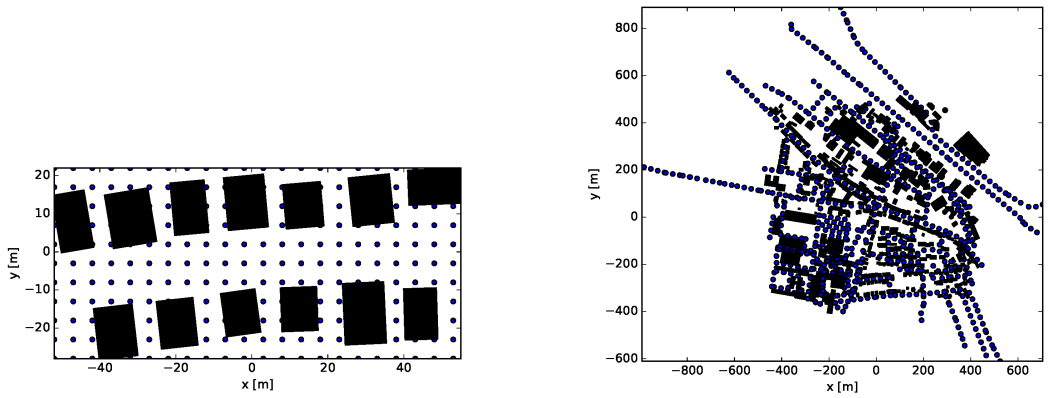
RaLaNS kann unterschiedliche Typen von Positionen für die Datenauswertung bereitstellen. Die beiden Wichtigsten sind eine Fläche, also eines Grids von Positionen, und eine Liste von Positionen (Abbildung 2.3).

Die Signalausbreitung für jeweils eine Position in den beiden Typen (Abbildung 2.3) sieht dann so aus wie in Abbildung 2.4 zu sehen ist.

2.2.3 Grenzsignalstärke

Für das Node-Placement ist es wichtig zu wissen bis zu welcher Signalstärke eine Kommunikation noch möglich ist. Dazu gibt es in Bezug auf RaLaNS zwei

[1] www.openstreetmap.org



(a) RaLaNS Grid-Positionstyp **(b)** RaLaNS Liste-Positionstyp

Abbildung 2.3: 2 Abbildungen verschiedener RaLaNS Datentypen, die blauen Punkte stehen für die Positionen zwischen denen die Signalausbreitung berechnet wird

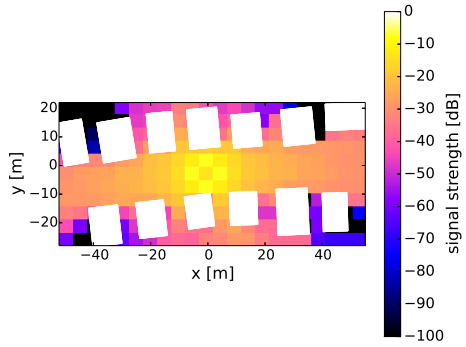
verschiedene Verfahren [10, S. 44]. Durch die Analyse von normalerweise empfangenen Leistungen z. B. bei Smartphones kann eine untere Grenzsignalstärke ermittelt werden, hier $-80dB$. Diese Methode ist nicht so gut geeignet, da die Antennen in der Simulation besser modelliert werden müssten.

Die andere Variante benutzt die übliche Reichweite der eingesetzten Geräte. Die übliche Reichweite von WLAN Geräten beträgt auf freiem Gebiet ca. 250 Meter. In der Simulation mit RaLaNS kann man nun den Wert ermitteln, der bei diesem Abstand auftritt, und als Grenze benutzen. Dieser Wert muss für eine genaue Analyse für jedes Szenario einzeln ermittelt werden.

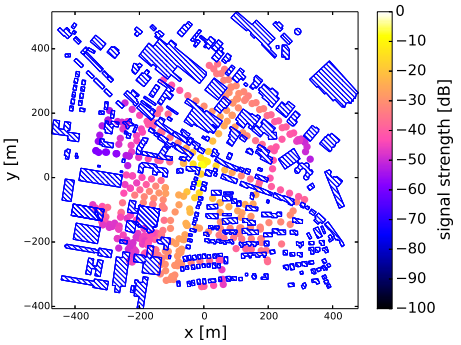
Wenn eine Kommunikation zwischen zwei Knoten noch möglich ist, weil die Grenzsignalstärke überschritten ist, dann wird hier im Weiteren auch davon gesprochen, dass diese zwei Knoten eine Verbindung haben oder verbunden sind.

2.3 Node-Placement

Beim Node-Placement geht es darum für die Knoten eines Netzwerks gute Position meist auf einer geographischen Karte zu finden, um bestimmte Eigenschaften dieses Netzes zu optimieren [10, S. 26]. Weil das Node-Placement meist auf einer Karte stattfindet, damit es einen gewissen Realitätsbezug hat, gibt es zwei verschiedene Ansätze hinsichtlich der zulässigen Positionen. Es können entweder alle Orte auf der Karte zugelassen sein, was bedeutet, dass es theoretische unendlich viele Positionen gibt, oder man bestimmt eine Auswahl an Positionen als mögliche Orte für die Platzierung der Knoten. In der Regel



(a) RaLaNS Grid-Positionstyp: die Signalausbreitung von einem Punkt in der Mitte aus, Koordinaten des Punktes: $x = -2$,$y = -3$



(b) RaLaNS Liste-Positionstyp: die Signalausbreitung vom Punkt mit Koordinaten $(x = -8$,$y = 11)$ aus

Abbildung 2.4: die Signalausbreitungen für die Typen aus Abbildung 2.3 von jeweils einem Punkt aus

wird die letztere Methode bevorzugt, wie auch bei einem Vergleich mit den Arbeiten im Kapitel 3 zu sehen ist. Bei der Vorauswahl der Positionen gibt es ebenfalls grobe Einteilungen. Man kann eine Liste von Positionen, die auf irgendeine Art und Weise ausgewählt wurden, als Vorauswahl treffen oder man diskreditiert die Karte nach einem bestimmten Schema, in der Regel mit Hilfe eines Grid. In RaLaNS ist wie in Abschnitt 2.2.2 beschrieben beides möglich.

Hier bezieht sich das Node-Placement natürlich auf drahtlose Netzwerke. Durch die geographischen Eigenschaften einer Gegend, welche die Signalstärke zwischen den Knoten stark beeinflussen kann, kann die Wahl der Position der Knoten nämlich großen Einfluss auf die Güte einer Platzierung haben.

2.3.1 Metriken

Nun gibt es in einem Netzwerk einige Eigenschaften, die beim Design optimiert werden sollten [13]. Um diese Eigenschaften zu bewerten werden Metriken benutzt. Eine Metrik beim Node-Placement ist ein quantitatives Maß für dessen Güte, so dass ein Vergleich zwischen verschiedenen Platzierung erst wirklich stattfinden kann. Einige dieser Kriterien ergeben natürlich nur im Zusammenhang mit anderen wirklich Sinn. Eine Auswahl sinnvoller Metriken präsentiert die folgende Aufzählung.

Die **Abdeckung** bezieht sich darauf welche Anteile der Nutzer in Verbindung mit dem Netzwerk stehen. Die Nutzer können sich dabei entweder auf einer

meist rechteckigen Fläche oder einer Liste von Positionen aufhalten. Die Abdeckung ist von wichtiger Bedeutung, weil man in der Regel die Möglichkeit haben möchte, an allen Positionen eine Verbindung zum Netz zu haben.

Die **Konnektivität** bezieht sich selbsterklärend auf die Verbundenheit des Netzwerkes. Es wird je nach Implementation auf eine bestimmte Art die Anzahl der verbundenen Knoten gezählt. Bei Multi-Hop Netzen ist die Verbundenheit wichtig, damit ein Nutzer in Verbindung mit dem gesamten Netz und nicht nur einem Teil davon stehen kann.

Die **Anzahl der Knoten** zu optimieren ist selbstverständlich, weil weniger Knoten in vielen Fällen weniger Kosten verursachen.

Die **Kapazität** steht für die durchschnittliche Informationsmenge, meist Bits, die in einem bestimmten Zeitraum, meist 1 Sekunde, zwischen den Knoten des Netzwerkes übertragen werden kann. In der Regel ist es das Ziel diese auch zu maximieren.

Wenn der **Energieverbrauch** als Optimierungskriterium benutzt wird, muss wirklich eine Auswahl der verwendeten Geräte getroffen werden. So kann dann auf den Energieverbrauch geschlossen werden, der auch von der bereitgestellten Übertragungsleistung abhängt, die somit bestimmt werden muss. Bei einer höheren Übertragungsleistung ist es nämlich so, dass die Sende- und auch Empfangsreichweite der Geräte zunimmt (Abschnitt 2.2.1).

Ebenfalls kann es ein Ziel sein, die **Kosten** der Geräte inkl. der Kosten der Platzierung des Netzwerkes zu optimieren.

Als Kriterien für eine gutes Placement werden hier die Abdeckung, die Konnektivität des Netzes und die Anzahl der Knoten betrachtet. Die quantitative Bewertung dieser drei Kriterien wird im folgenden Abschnitt genauer beschrieben.

Damit diese Metriken genutzt werden können, müssen die Signalstärken für das geographische Gebiet vorliegen. Das ist hier die Aufgabe von RaLaNS.

Die folgende vorgestellte Metrik bezieht sich auf die Abdeckung. Es existiert also eine gerasterte Fläche bzw. gewisse Punkte, welche die relevanten Positionen der Nutzer repräsentieren sollen, zu denen das Netzwerk eine Verbindung haben soll. Diese Punkte sollen also im besten Fall alle abgedeckt sein. Mathematisch kann dies so ausgedrückt werden, dass der Anteil der abgedeckten Positionen P ins Verhältnis zur Anzahl der verwendeten Knoten N gesetzt wird (Abschnitt 2.3). Das Ziel ist es nämlich bei möglichst geringer Anzahl von Knoten eine möglichst große Fläche bzw. Anzahl an Punkten abzudecken. Je höher der Wert der Gleichung ist, desto besser ist die Abdeckung A.

$$A = \frac{P}{N} \tag{2.3}$$

Das Problem bei dieser Metrik ist, dass die Verbundenheit des Netzwerkes nicht beachtet wird. Die Knoten können also so ggf. nicht untereinander kommunizieren.

Man kann die Verbundenheit nun so betrachten, dass nur beobachtet wird, welcher Knoten zu welchem Knoten basierend auf den Signalstärken zwischen den Knoten Pakete senden kann. Bezogen auf ein Graphenproblem, bei dem die Knoten des Graphen den Knoten beim Node-Placement entsprechen und zwei Knoten eine Kante besitzen, wenn sie verbunden sind, kann die Verbundenheit durch die Untersuchung der Komponenten in einem Graph gemessen werden [14]. Näheres zur Berechnung der Komponenten eines Graphen findet sich in Kapitel 4.2. Eine Möglichkeit besteht darin die Anzahl der Komponenten zu minimieren. Dies sorgt zwar nicht immer dafür, dass eine Lösung mit weniger Komponenten direkt die bessere ist, weil es Graphen gibt, die mehr Komponenten haben, aber auch mehr Knoten zueinander Pfade besitzen. Ein Vorteil ist aber, dass bei weniger Komponenten es wahrscheinlicher wird, dass sich in einer der folgenden Lösungen das Netzwerk zu einer Komponente zusammenschließt. Wie schon beschrieben können Knoten hier nämlich Pakete zueinander schicken, wenn in dem Graph ein Pfad zwischen diesen existiert. Das bedeutet also, dass es das Ziel der Metrik ist ein verbundenes Netzwerk zu generieren.

Die Verbundenheit kann auch mit einer Metrik bewertet werden, die Größe des *Giant Component* misst [15]. Das ist die Größe der größten Komponente des Graphen. Wenn diese maximal ist, also alle Knoten des Netzwerkes umfasst, dann besteht das Netzwerk aus nur noch einer Komponente.

Eine andere Herangehensweise besteht darin, den Paketverlust p in einem Netzwerk zu messen. Eine passende Metrik dafür ist das Verhältnis der Anzahl der gesendeten Pakete n_t eines Knoten zu der Anzahl der ankommenden Pakete n_r an einem anderen Knoten [10, S. 28]:

$$p = \frac{n_r}{n_t} \tag{2.4}$$

Umso näher der Wert dieser Metrik an 1 ist, desto besser ist die Verbundenheit zwischen den beiden Knoten in dem Netzwerk. Damit ein guter Wert für den Paketverlust im gesamten Netzwerk berechnet werden kann, besteht die Möglichkeit entweder die Verbindungen zwischen allen Knoten oder nur einem Teil der Knoten zu untersuchen und die resultierenden Werte zu mitteln. Um überhaupt den Paketverlust in einem Netzwerk gut ermitteln zu können, muss man Netzwerksimulatoren wie z. B. den Network Simulator 3 (ns-3) [16] verwenden, die den Datentransport eines Netzwerkes abbilden. Da kann es nämlich passieren, dass nicht nur durch das Signalausbreitungsmodell, sondern auch durch das verwendete Routing und einen höheren Traffic ein Verlust von

Paketen auftreten kann. Das wird in dieser Arbeit nicht behandelt. Es gibt hier also keinen Paketverlust, wenn eine Verbindung zwischen zwei Knoten existiert.

Weil für dieses Node-Placement Problem die Abdeckung und Verbundenheit relevant ist, ist es sinnvoll diese in einer Metrik zu vereinen. Die Scanning Probe Network Efficiency (SPNE) [11] ist eine passende Metrik für dieses Problem. Die Metrik wird in Gleichung 2.5 verdeutlicht. Um diese Gleichung gut zu verstehen, stellt man sich folgendes Verfahren vor. Es gibt einen Probe-Knoten, der auf jedem Empfängerfeld oder Nutzerfeld der Karte platziert wird. Auf einer gerasterten Karte iteriert dieses Verfahren also über alle Reihen, bezeichnet mit R (Row), und alle Spalten, bezeichnet mit C (Column). Für jedes Feld wird berechnet, ob ein Paket von jedem Knoten des Netzes bei diesem Probe-Knoten ankommt. Die Anzahl der ankommenden Pakete wird mit $p_{r,c}$ bezeichnet. Diese Paketenanzahlen werden nun aufsummiert und ins Verhältnis zur Anzahl der Abdeckungspositionen, also $C \cdot R$, und der Anzahl der Knoten des Netzwerkes N gesetzt, so dass das Ergebnis zwischen 0 und 1 liegen muss. In der Implementation von SPNE in dieser Arbeit kommt also kein Paketverlust vor, wenn die Knoten zueinander in Verbindung stehen, weil der Paketverlust nicht simuliert wird, wie weiter oben erklärt wurde.

$$SPNE = \frac{1}{C \cdot R \cdot N} \sum_{r=0}^{R-1} \sum_{c=0}^{C-1} p_{r,c} \tag{2.5}$$

2.3.2 Komplexität

Für die Berechnung einer guten Lösung ist die Komplexität des Problems sehr wichtig, weil davon abhängt wie lange ein Algorithmus braucht, um die beste Lösung zu finden.

Bei diesem Node-Placement Problem ist es ähnlich wie in der Begründung bei [17] so, dass es zu viele Möglichkeiten gibt, um diese alle zu testen. Wenn man annimmt, dass durch RaLaNS die Signalstärken von 100 Positionen bereitgestellt werden, was noch keine so große Zahl ist, und man die Positionierung von 10 Knoten optimieren möchte, dann ist die Anzahl der Möglichkeiten zu groß, um die Lösung exakt zu berechnen wie die folgende Rechnung zeigt:

$$\binom{100}{10} \approx 1,731 \cdot 10^{13} \tag{2.6}$$

Wenn man nun 100000 Lösungen pro Sekunde berechnen könnte, würde ein einfaches Testen aller Lösungen immer noch ca. $5,5$ Jahre dauern.

Man müsste nun Strukturen in diesem Problem, also in den Daten von RaLaNS, finden, die es einem Algorithmus ermöglichen viele mögliche Lösungen zu verwerfen, weil sie aus bestimmten Gründen nie die beste Lösung sein können. Wenn das gelingt, kann es möglich sein für realistische Szenarien exakte Berechnungen durchzuführen.

Komplexere Probleme wie die die Planung eines WMN, in der Node-Placement nur ein Teil ist, sind wohl NP-hart und somit nicht für größere Instanzen mit exakter Berechnung wie linearer Programmierung lösbar [5].

Ein heuristisches Verfahren, das einem häufig eine recht gute Lösung für Probleme verschafft, die nicht praktisch exakt zu lösen sind, besteht in der Nutzung von genetischen Algorithmen.

2.4 Genetische Algorithmen

Genetische Algorithmen (G.A.) sind eine Klasse von Algorithmen, die durch Simulation der Evolution versuchen gute Lösungen für Optimierungsprobleme zu erzielen. Als Informationsgrundlage für den G.A. diente in dieser Arbeit [18]. Unter den Optimierungsverfahren gehören sie zu den heuristischen, weil sie nicht garantiert das globale Optimum, also die beste Lösung, liefern, sondern nur je nach Einstellung und Effektivität der Implementation möglichst gute Lösungen.

Abbildung 2.5 zeigt die grundlegende Struktur jedes G.A..

Von essentieller Bedeutung ist die Population. Die Population besteht aus Individuen wie in der Natur, welche die Lösungen für das jeweilige Problem darstellen. Eine Lösung für das Problem in dieser Arbeit muss also die Positionen der Knoten passend kodieren. Das Ziel ist es, dass mindestens eine dieser Lösungen nach Ablauf des Algorithmus eine sehr gute Fitness hat. Die Fitness ist ein quantitativer Wert dafür wie gut die Lösung ist. Bei einem Minimierungs- bzw. Maximierungsproblem ist es das Ziel die Fitness der Population zu minimieren bzw. zu maximieren. Damit man mit dieser Population nun arbeiten kann, muss man sie erst initialisieren. Einfache Verfahren für diesen Schritt basieren auf dem Zufall.

Um die Population nun zu optimieren, müssen wiederholend mehrere Schritte ausgeführt werden bis ein Kriterium zum Beenden des G.A. eintritt.

Zuerst wird im Schritt der **Selektion** ausgewählt welche Individuen für den Schritt der Rekombination zugelassen sind. Bezogen auf die Evolution ist das die Auswahl der Lebewesen, die sich paaren werden. Meist ist die Auswahl der Individuen von ihrer Fitness abhängig. In der Regel werden dabei Lösungen mit einer besseren Fitness bevorzugt.

In der **Rekombination** pflanzen sich die Lösungen fort, so dass aus Ihnen

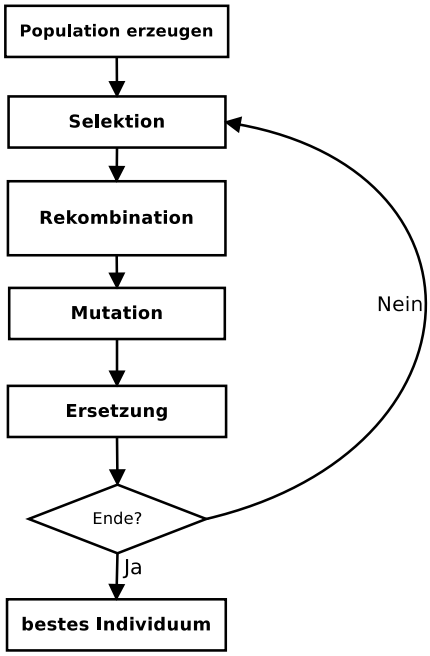


Abbildung 2.5: grundlegende Struktur eines genetischen Algorithmus, ähnlich wie in [17]

Kind-Lösungen entstehen. In der Regel generieren dabei zwei Individuen eine oder zwei Kind-Lösungen. Die Annahme bei diesem Schritt ist, dass gute Eltern-Lösungen häufig auch gute oder sogar bessere Kind-Lösungen erzeugen.

Im nächsten Schritt, der **Mutation**, werden Individuen einzeln verändert, es wird also z. B. eine Knotenposition des Netzwerkes zufällig geändert.

Schlussendlich wird in der **Ersetzung** ein Teil oder auch die ganze Population durch die neu entstandenen Individuen ersetzt. Häufig ist dabei die Regel, dass der Teil mit der schlechteren Fitness ersetzt wird.

Grundsätzlich gibt es bei heuristischen Optimierungsverfahren das Problem, dass man in lokalen Optima hängen bleibt. In vielen Verläufen von genetischen Algorithmen ist es nämlich so, dass durch den Schritt der Rekombination die Lösungen oft ähnlicher werden. Hier ist nun die Mutation sehr wichtig, weil diese die Funktion hat die Lösungen aus lokalen Optima auszubrechen oder anders ausgedrückt die genetische Vielfalt aufrecht zu erhalten.

Für die Schritte der Selektion und der Ersetzung gibt es einige verschiedene Grund-Strategien. Die für diese Arbeit verwendeten Strategien werden in der Umsetzung vorgestellt (Kapitel 4.3).

Eine weitere wichtige Einstellung ist die Wahl des Ende bzw. Stop-Kriteriums. Meist ist es so, dass man entweder einfach nach einer bestimmten Anzahl von Iterationen, beim G.A. auch Generationen genannt, aufhört oder nachdem sich

für eine bestimmte Anzahl von Generationen die beste Lösung nicht weiter verbessert hat.

Nach dem der G.A. den Schleifendurchlauf durch Erfüllen des Stop-Kriteriums beendet hat, wird meist die beste Lösung der finalen Population als Lösung für das Optimierungsproblem genommen.

2.4.1 DEAP

Das Distributed Evolutionary Algorithm Package (DEAP) ist ein Paket, um schnell evolutionäre Algorithmen in Python zu programmieren [19]. Dieses Framework unterstützt den Anwender unter anderem darin einfach Individuen anzulegen, seinen genetischen Algorithmus zu strukturieren und die Fitness zu berechnen. Die Parallelisierung der eigenen Anwendung wird ebenfalls sehr einfach gemacht.

2.5 Zusammenfassung

In diesem Kapitel wurde das Node-Placement Problem für Multi-Hop Netzwerke detailliert vorgestellt. Dabei wurde insbesondere beschrieben was Node-Placement und Multi-Hop Netzwerke überhaupt sind und Kriterien vorgestellt, um die Güte einer Platzierung zu messen. Damit man diese Aufgabe lösen kann, benötigt man Daten für die Signalstärken auf der Karte. Dafür wird RaLaNS verwendet. Es wurden genetische Algorithmen als ein Verfahren zur Lösung von Node-Placement Problemen vorgestellt. Dafür wird in dieser Arbeit das Framework DEAP als Unterstützung genutzt.

3 Verwandte Verfahren

In diesem Kapitel werden verwandte Verfahren für Node-Placement vorgestellt. Es gibt einige andere Arbeiten, die heuristische Ansätze benutzen, um die Konfiguration oder die Platzierung eines Netzwerkes zu bestimmen, weil in den meisten Szenarien das Problem nicht mit exakten Algorithmen in akzeptabler Zeit zu lösen ist. Bei Vielen wird dazu ein genetischer Algorithmus als Verfahren zur Lösung benutzt. Hier werden nun erst drei verschiedene Arbeiten vorgestellt, die Node-Placement Probleme ähnlich zu dem Problem in dieser Arbeit mit Hilfe von G.A. lösen (Abschnitt 3.1). Danach werden andere speziellere Verfahren vorgestellt, die auch zum Lösen von Node-Placement Problemen angewendet werden (Abschnitt 3.2). Abschließend werden allgemeine Verfahren vorgestellt, die für die Optimierung dieses Node-Placement Problem genutzt werden können (Abschnitt 3.3).

3.1 Verwandte Arbeiten

In der Arbeit von [17] wird ein genetischer Algorithmus benutzt, um ein Node-Placement Problem für die Positionierung von Supporting Units (SUs) zur Optimierung des Verkehrsflusses zu lösen. Es handelt sich also um ein Optimierungsproblem in einem VANET.

Das System soll dem Fahrer eines Fahrzeugs dabei helfen, die richtige Route basierend auf dem jetzigen Verkehrsaufkommen zu wählen, wenn es mehrere Möglichkeiten gibt. Dazu wird zuerst eine hierarchische Aufteilung der Verkehrsknotenpunkte in einem Gebiet vorgenommen, um die Menge der Daten für Berechnung der Fahrzeiten zu verringern. Auf einer der höchsten Ebenen sind z. B. die Fahrzeiten zwischen Städten gespeichert, aber noch nicht die Fahrzeiten zu bestimmten Orten in der Stadt. Ein großes Problem ist nun die Verbreitung der Informationen über die voraussichtlichen Fahrzeiten. Generell läuft es in diesem Szenario so, dass Fahrzeuge selbst die Dauer für eine Strecke messen und diese Erkenntnisse anderen Fahrzeugen und den genannten SUs mitteilen. Diese Units kann man sich so vorstellen, dass sie an bestimmten, wichtigen Verkehrsknotenpunkten stehen. Die vorbeifahrenden Fahrzeuge teilen

die Information mit der jeweiligen SU. Weil alle SUs miteinander verbunden sind, kann diese Information mit dem gesamten Netzwerk an Knoten geteilt werden, so dass Informationen deutlich schneller in entfernte Regionen gelangen als es mit einer Kommunikation von Fahrzeug zu Fahrzeug alleine möglich wäre. Weil man natürlich aus Kostengründen nur eine bestimmte Anzahl dieser Knoten platzieren möchte, ist die entscheidende Frage an welchen Standorten diese aufgestellt werden sollten, so dass sie mit möglichst vielen Fahrzeugen kommunizieren werden. Eine Kommunikation findet statt basierend auf einer kreisförmigen Signalausbreitung. Weil es ähnlich wie in dieser Arbeit zu viele Möglichkeiten für eine Enumeration gibt, wird ein G.A. dafür eingesetzt.

In der Arbeit von [20] wird ein genetischer Algorithmus benutzt, um die Platzierung der Router-Knoten eines WMN zu optimieren. Die Kriterien, die für dieses Netz optimiert werden sollen, sind die Größe der Giant Component und die Nutzerabdeckung. Der Giant Component ist wie in Abschnitt 2.3.1 beschrieben die größte Komponente eines Graphen, in diesem Fall also eines WMN. Diese soll maximiert werden. Dabei soll ebenso auf die Abdeckung der Nutzer geachtet werden, die auch maximiert werden soll. Eine Verbindung zwischen zwei Knoten des Mesh Netzwerkes oder zwischen einem Mesh-Knoten und einem Nutzerknoten kommt zustande, wenn sich diese in einem bestimmten Abstand zu einander befinden. Es wird also eine kreisförmige Signalausbreitung zu Grunde gelegt. Das Problem ist so spezifiziert, dass auf einem Grid verteilt sich Client Nodes, also die Nutzer nach einer bestimmten Verteilung, z. B. zufällig, befinden. Diese sollen am besten alle mit dem Netz verbunden sein. Die Aufgabe des G.A. ist es gute Positionen für die STAs des WMN zu finden. Weil die Konnektivität in solchen Netzwerken wichtiger ist, wird diese als Hauptkriterium bei der Fitnessberechnung betrachtet. Die Nutzerabdeckung ist also nur sekundär zu optimieren. In der Arbeit werden auf Grund der Grid-Struktur die Lösungen auch als Grid gespeichert, so dass sich damit speziellere Operatoren bilden können.

Die Arbeit von [13] handelt von der Optimierung des Node-Placement und der Konfiguration eines WLAN auf einem städtischen Gebiet. Als Ausbreitung wird hier ebenfalls eine kreisförmige Signalausbreitung zu Grund gelegt. Diese kann aber natürlich gegen andere Modelle ausgetauscht werden. In der Arbeit ist das Netzwerk, das optimiert werden soll, kein Multi-Hop Netzwerk, sondern nur Single-Hop. Es soll also im Grunde nur die Platzierung der Knoten auf einem Gebiet hinsichtlich einiger Kriterien optimiert werden. Die Kommunikation zwischen den Knoten kann so in einem drahtgebundenen Backbone-Netz

stattfinden.

Ähnlich zu der vorigen Arbeit wurde es so umgesetzt, dass es eine Vorauswahl an möglichen Standorten für die Access Points (APs) gibt. Die Geräte können also nicht an irgendwelchen Orten auf der Karte platziert werden. Bei der Optimierung des Node-Placement wird aber nicht nur auf die Abdeckung der Fläche geachtet, sondern auch auf das verwendete Gerät, z. B. dessen mögliche Übertragungsenergie oder Energieverbrauch, die verwendeten Antennen, die tatsächlich zu nutzende Übertragungsenergie und die Zuweisung der Channel für die einzelnen AP, damit die Interferenz zwischen den Geräten möglichst gering ist. Um diese ganzen Bedingungen zu optimieren wird ein genetischer Algorithmus benutzt, der es schafft eine Lösung zu erzeugen, die besser ist als das manuelle Planen eines solchen Netzwerkes.

3.2 Andere Verfahren

Abgesehen von G.A. gibt es auch andere Verfahren, mit denen man Node-Placement Probleme lösen kann.

Um akzeptable Lösungen zu erhalten, kann man für solche Probleme so genannte Eröffnungsverfahren [21], auch Greedy Algorithmen genannt, nutzen [22, S. 87]. Das sind Verfahren, die normalerweise in kurzer Zeit nach bestimmten Regeln Schritt für Schritt eine Lösung konstruieren. Dabei wird häufig das Wissen über die Struktur des Problems genutzt, um ein Resultat zu erhalten.

Eines davon ist ein Grid-basiertes Verfahren [7, S. 628]. Bei diesem muss man natürlich nicht unbedingt die Knoten nacheinander setzen, weshalb es ein sehr schneller Verfahren ist. Eine optimale Strategie besteht dort darin, entweder horizontal bzw. vertikal die Knoten auf einer Geraden so zu platzieren, dass es immer eine Verbindung zum Nachbar gibt. Zwischen diesen horizontalen bzw. vertikalen Geraden werden Knoten platziert, die diese Geraden miteinander verbinden, so dass ein Netzwerk entsteht, das einer rechteckigen Fläche gleicht. Diese Anordnung ist aber offensichtlich nicht so gut für städtische Gebiete geeignet, weil dort die Ausbreitung nicht kreisförmig möglich ist. Diese Anordnung kann also eher in ländlichen Gebieten eingesetzt werden.

Ein iteratives Verfahren, das aber auch den Zufall nutzt, für städtische Gebiete besteht darin, die Straßen zu nutzen. Es werden Knoten auf den Kreuzungen von Straßen platziert und dann zwischen diesen Kreuzungen passende Knoten auf den Verbindungsstraßen ausgewählt, so dass das Netzwerk verbunden ist [10, S. 27]. Der Vorteil an dieser Variante ist, dass man die

Struktur der Daten gut ausnutzt, weil die Signalausbreitung auf Straßen in der Regel gut sein sollte. Man reduziert ebenfalls die möglichen Positionen der STAs gegenüber einer Instanz, welche die Positionen auf einer ganzen Fläche betrachtet, so dass die Probleminstanz kleiner ist.

3.3 Allgemeine Verfahren

Abgesehen von iterativen Verfahren, die versuchen durch das Ausnutzen der Struktur der Daten eine möglichst gute Lösung zu berechnen, gibt es allgemeine heuristische Optimierungsalgorithmen. Dazu zählen unter anderem allgemeine, andere Eröffnungsverfahren, die lokale Suche, die Tabusuche, Simulated Annealing und genetische Algorithmen.

Die simple **lokale Suche** funktioniert so, dass man eine zufällig oder nach einer bestimmten Heuristik generierte Lösung iterativ verbessert [23, Kapitel 19] [24, Kapitel 1]. Dazu wird eine Nachbarschaftsstruktur für die Lösungen definiert. Die jetzige Lösung kann sich verbessern, indem sie ihre Nachbarn untersucht und dann einen Nachbarn auswählt, meist den, der überlegen ist. In der einfachen Variante wird immer die beste oder ein besserer Nachbar ausgewählt. Das passiert so lange bis es keine bessere Lösung mehr in der Nachbarschaft gibt, es wurde also ein lokales Optimum oder vielleicht sogar das globale Optimum erreicht. Ein grundsätzliches Problem bei den simplen Varianten der lokalen Suche ist somit das Enden in lokalen Optima. Dafür gibt es mehrere Verbesserungen, zwei davon sind die **Tabusuche** und **Simulated Annealing** [22, Kapitel 5].

G.A. sind eng mit lokalen Suchverfahren verwandt, sind aber trotzdem klar von diesen zu unterscheiden, weil sie nicht die Begriffe der Nachbarschaft kennen und mehrere Lösungen zusammen und nicht nur eine optimiert werden.

3.4 Zusammenfassung

In diesem Kapitel wurden einige Verfahren zum Lösen von Node-Placement Problemen vorgestellt. Die Entscheidung welches Verfahren man nutzt, hängt stark vom Problem ab und ist meist nicht einfach zu entscheiden [22, S. 55]. In dieser Arbeit wird, wie in den ersten drei vorgestellten Artikeln, ein G.A. genutzt, um dieses Problem zu lösen. Da in ähnlichen Problemstellungen ebenfalls genetische Algorithmen benutzt wurden, um das Problem zu lösen, wird das in dieser Arbeit auch getan.

Von den drei vorgestellenten Artikeln unterscheidet sich diese Arbeit darin, dass sie als Grundlage für die Signalausbreitung einen Ray Launcher nutzt, der deutlich realistischere Daten liefert als die kreisförmige Signalausbreitung.

4 Umsetzung

In diesem Kapitel wird die Umsetzung der Implementation beschrieben. Zuerst wird beschrieben wie die Daten von RaLaNS genutzt wurden (Abschnitt 4.1). Im zweiten Abschnitt wird ausführlich die Implementation des genetischen Algorithmus und dessen Teile beschrieben (Abschnitt 4.3). Ein G.A. benötigt für jede Lösung einen Fitnesswert. Die Berechnung davon wird im folgenden Abschnitt erläutert (Abschnitt 4.2). Das Kapitel schließt mit der Beschreibung der Implementation der Algorithmen, die zum Vergleich zum G.A. genutzt werden, ab (Abschnitt 4.4).

4.1 RaLaNS Datenverwertung

Die Daten von RaLaNS liegen als Datei vor. Das bedeutet, dass diese als erstes in den Arbeitsspeicher geladen werden müssen, weil die Signalausbreitungsdaten während der gesamten Berechnung zur Verfügung stehen sollten. Die Daten werden dazu in einem 2-dimensionalen Array abgespeichert, das die Signalstärke von jedem Sender zu jedem Empfänger enthält. Diese macht es während der Berechnung einfach auf die für die gerade zu berechnende Instanz benötigten Verbindungen zuzugreifen.

Ein Nachteil an diesem Ansatz ist nur, dass bei einem recht großen Datensatz der Arbeitsspeicher überfüllt ist, so dass das Betriebssystem Teile auf die Festplatte auslagern muss, was die Berechnung deutlich verlangsamt.

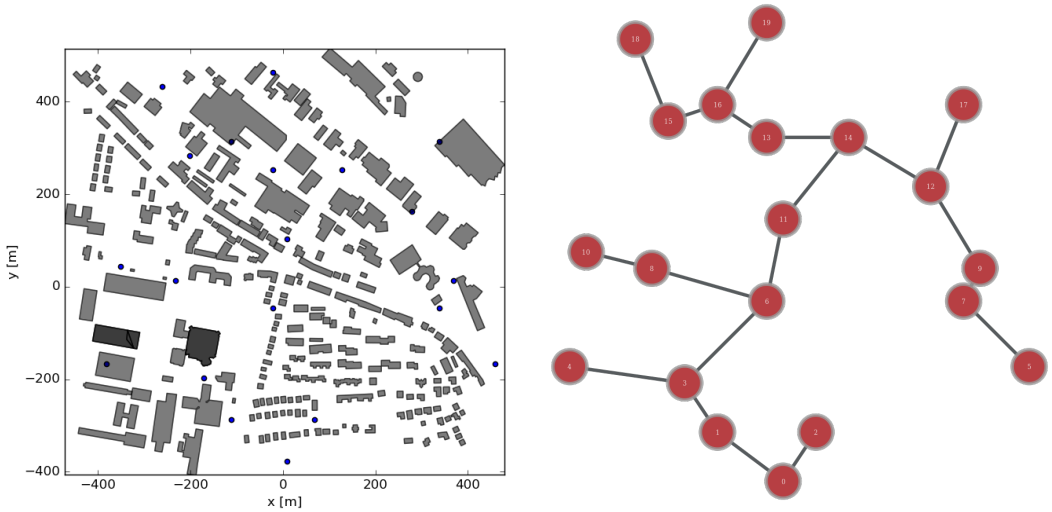
4.2 Fitnessberechnung

Um den Wert für eine Metrik zu bestimmen, müssen die Nutzer, die abgedeckt werden sollen, erstmal festgelegt werden, weil diese Bestandteil der Berechnung sind. Hier wird es so modelliert, dass die Nutzer alle Positionen, zu denen RaLaNS Signalausbreitungsdaten ausgerechnet hat, sein können. Die Nutzer sind also alle Positionen, auf denen auch ein Knoten platziert werden kann.

In den Grundlagen wurde die SPNE Metrik vorgestellt (Abschnitt 2.3.1). Diese wird hauptsächlich zur Bestimmung der Fitness eingesetzt.

Für Multi-Hop Netzwerke muss zur Berechnung, ob ein Paket über mehrere Links beim Empfänger ankommen kann, bestimmt werden, ob passende Links existieren. Hier wird ein Verfahren beschrieben, um das zu berechnen.

Um das Netzwerk gut abzubilden, wird aus der Lösung ein Graph konstruiert (Algorithmus 1). Die Knoten des Netzwerkes sind auch die Knoten des Graphen und es gibt eine Kante im Graph, wenn eine Verbindung zwischen den Netzwerkknoten vorherrscht (Abbildung 4.1). Die Positionen der Knoten im Graph (Abbildung 4.1b) entsprechen den Positionen der Platzierung (Abbildung 4.1a). Algorithmus 1 prüft beim Hinzufügen der Kante um eine Verbindung darzustellen, ob die Signalstärke in beide Richtungen in Ordnung ist, weil es erhebliche Unterschiede in der Signalstärke zwischen einer Richtung und ihrer Gegenrichtung in der Simulation mit RaLaNS geben kann.



(a) Eine Platzierung der Knoten **(b)** der korrespondierende Graph

Abbildung 4.1: Ansicht eines Node-Placement und des korrespondierenden Graphen, Anzahl der Knoten: 20, SPNE-Fitness: 0.79

Nach der Konstruktion des Graphen, beginnt die eigentliche SPNE Berechnung. Dazu wird für jedes relevante Feld der Probeknoten zum Graphen hinzugefügt und wieder entfernt bevor der Nächste hinzugefügt wird. Man kann sich das so vorstellen, dass der Graph in Abbildung 4.1b um einen Knoten erweitert wird. Die Kanten zum Probeknoten werden genauso wie bei der Konstruktion des Graphen erzeugt. Für diesen neuen Graphen wird nun bestimmt mit wie vielen Knoten der Probeknoten in Verbindung steht. Es wird also nur die Komponente gesucht, zu der der Probe-Knoten gehört. Zu

Algorithmus 1 : Konstruktion des Graphen zu einer Lösung

```
input  : nodes: Indices der Knoten, n: Anzahl der Konten
output : Graph g passend zu nodes

g ← ungerichtete, leere Graphinstanz;
Erzeuge n Knoten für g;
for i ← 0 to n-1 do
    node1 ← nodes at position i;
    for j ← i + 1 to n-1 do
        node2 ← nodes at position j;
        if hat-gutes-Signal(node1, node2) == True und
        hat-gutes-Signal(node2, node1) == True then
            erzeuge Kante zwischen Knoten i und j in g
        end
    end
end
return g
```

einer Komponente gehören alle Knoten, die über irgendeinen Pfad von Kanten eine Verbindung zu einander haben [25, S. 13]. In der Graphentheorie ist das die Aufgabe des Labeln von Komponenten, das hier beschrieben wird [14, S. 373]. Mit einer Tiefen- oder auch Breitensuche kann man dies berechnen. Leicht abgewandelt von [14, S. 373] wird dazu die Suche beim Probeknoten gestartet. Es werden zu Beginn alle Knoten außer der Probeknoten als unmarkiert gelabelt. Wenn die Suche keinen Knoten mehr über Kanten von bereits markierten Knoten finden kann, dann gehören alle markierten Knoten zur gleichen Komponente. Weil der Algorithmus hier nur sagen muss, ob ein Knoten zur Komponente gehört, in der sich der Probeknoten befindet, ist die Berechnung dann abgeschlossen. Die Größe der Komponente des Probeknoten, also die Anzahl der markierten Knoten, ist so die Anzahl der Pakete, die dieser Knoten an der Position empfängt.

Diese Umsetzung der SPNE ist anders als die in [11]. Da wird nämlich der Paketverlust wirklich simuliert. Hier wird ein Graph benutzt, um das zu berechnen. Gleichung 4.1 formalisiert die hier in dieser Arbeit umgesetzte Implementation. Dabei steht P für die Menge an Probeknoten, K repräsentiert die Menge an Knoten im Graphen und K_p ist die Menge an Knoten, die zusammen mit Knoten p in einer Komponente sind.

$$SPNE = \frac{1}{|P| \cdot |K|} \cdot \sum_{p \in P} (|K_p|) \tag{4.1}$$

In der jetzigen Implementation ist es wie eben beschrieben so, dass es bei einer Verbindung zwischen mehreren Knoten, keinen Paketverlust zwischen diesen gibt. Das hat zur Folge, dass sich die Metrik auf keinen Fall verschlechtert, wenn ein Knoten hinzugefügt wird, der sich in genau einer Komponente mit allen anderen Knoten befindet. Weil es natürlich nicht gewünscht ist, dass unnötige Knoten zur Lösung hinzugefügt werden, wird bei der flexiblen Variante eine mehrdimensionale Fitnessberechnung durchgeführt. Um mehrdimensionale Zielfunktionen zu optimieren, gibt es mehrere Ansätze [26]. Einer davon besteht darin die gewichtete Summe s der einzelnen Zielfunktionen zu optimieren. In diesem Fall bedeutet das die SPNE und die Anzahl der Knoten k zu summieren (Gleichung 4.2). Dabei muss darauf geachtet, dass die Gewichte richtig eingestellt sind. k muss negativ gewichtet sein, weil die Anzahl der Knoten minimiert werden soll. Je nachdem wie wichtig SPNE im Verhältnis zur Anzahl der Knoten ist, muss ein großes positives Gewicht p für SPNE benutzt werden. Wenn das Gewicht für SPNE größer ist als die Anzahl der möglichen Positionen, auf welche die Knoten des Netzwerkes gesetzt werden können, dann ist die Funktionsweise so, dass erst primär SPNE und danach sekundär die Anzahl der Knoten optimiert wird, was einer hierarchischen Optimierung einer mehrdimensionalen Zielfunktion ähnelt [26]. Durch die flexible Anzahl an Knoten kann es passieren, dass Kinder von Eltern, die eine SPNE von 1 haben, das nicht mehr haben, weil z. B. durch das Entfernen eines Knoten ihre SPNE Fitness nicht mehr gut genug ist. Diese Lösungen sind aber immer noch valide, so dass sie auch in zukünftigen Generationen vorkommen können. Das ist für die flexible Variante recht sinnvoll, weil so mehr Variabilität für die Anzahl der Knoten herscht.

$$s = p \cdot SPNE + (-1) \cdot k \tag{4.2}$$

4.3 Genetischer Algorithmus

Bei der Umsetzung des G.A. wurde so vorgegangen, dass zuerst ein Prototyp entwickelt wurde. Dieser ist der hier implementierte G.A., bei dem eine Verbindung zwischen den Knoten des Netzwerkes zustande kommt, wenn eine bestimmte Distanz unterschritten ist, was einer vereinfachte Variante der Signalausbreitung von Friis entspricht (Abschnitt 2.2.1). In [11] wird diese als *Range* bezeichnet. Es wurden also nicht die Daten von RaLaNS genutzt. Um eine Fläche oder auch eine Liste von Punkten zu kodieren, wurde das gleiche Format wie in RaLaNS benutzt.

Abgesehen vom Prototypen gibt es bei der Umsetzung des G.A. zwei grobe Einteilungen, die von dem Optimierungsziel abhängen. Einerseits soll bei einer

konstanten oder auch maximalen Anzahl von Knoten die Platzierung optimiert werden, andererseits soll diese Anzahl flexibel sein und dabei eine bestimmte Fitness als Endergebnis garantiert werden. Die Variante mit der konstanten Anzahl an Knoten wird hier auch als feste Variante bezeichnet, weil die Anzahl der Knoten festgeschrieben ist. Die andere Variante wird hier auch flexible Variante genannt.

Diese beiden Einteilungen werden in den folgenden Unterkapiteln zusammen behandelt. Zuert wird dazu noch hier genauer die Implementation des G.A. vorgestellt (Algorithmus 2). In der wird eine Hall of Fame benutzt, die im folgenden Abschnitt beschrieben wird (Abschnitt 4.3.1). Danach wird die Umsetzung der Kodierung diskutiert (Abschnitt 4.3.2). Darauf aufbauend werden Möglichkeiten gezeigt, um die Population zu initialisieren (Abschnitt 4.3.3). Danach werden wie beim Ablauf des G.A. nacheinander die Implementation der Selektion (Abschnitt 4.3.4), die Rekombination (Abschnitt 4.3.5), die Muation (Abschnitt 4.3.6) und die Ersetzung (Abschnitt 4.3.7) vorgestellt.

Algorithmus 2 : Genetischer Algorithmus

input : Pop: initialisierte Population, sprob: Selektionswahrscheinlichkeit, maprob: Vermehrungswahrscheinlichkeit, muprob: Muationswahrscheinlichkeit

output : finale Population Pop, HallofFame

berechne Fitness parallel für Pop;
initialisiere HallofFame *mit* Pop;
for *gen* ← 0 **to** ngen **do**
  Nachfahren ← *selektiere mit Wahrscheinlichkeit* sprob *Nachfahren mit bestimmten Verfahren von* Pop;
  führe Vermehrung mit Wahrscheinlichkeit maprob *auf* Nachfahren *durch*;
  führe Mutation mit Wahrscheinlichkeit muprob *auf* Nachfahren *aus*;
  berechne Fitness parallel für veränderte Nachfahren;
  aktualisiere HallofFame *mit veränderten* Nachfahren;
  ersetze je nach Verfahren Teil von Pop *durch* Nachfahren;
end
return Pop, HallofFame

4.3.1 Hall of Fame

Bei G.A. kann es passieren, dass sich auch die beste Lösung einer Population verschlechtert, z. B. durch eine Mutation des besten Individuums. Damit die

beste bis jetzt gefundene Lösung nicht verloren geht, wird in dieser Arbeit eine so genannte Hall of Fame eingesetzt [27]. In dieser werden die besten Individuen gespeichert, damit sie auch dann nicht verloren gehen, wenn sie aus der Population aus irgendwelchen Gründen verschwinden.

4.3.2 Kodierung

Für beide Varianten ist die Kodierung der Individuen die selbe. Die Kodierung der Chromosomen eines G.A. ist von kritischer Bedeutung, weil davon die Umsetzung der Rekombination und der Mutation entscheidend abhängt.

Eine sehr übliche Variante für die Kodierung ist der Bit-String [18, S. 43f]. Das bedeutet, dass man eine Sequenz von 1 und 0 benutzt, um jedes Individuum zu kodieren. In diesem Fall ist die Länge einer solchen Sequenz die Anzahl der möglichen Knotenpositionen. Die 0 an einer Stelle in dem String steht dafür, dass sich hier kein Sender befindet, bei einer 1 hingegen schon [17]. Der Vorteil dieser Variante ist, dass jede Lösung die gleiche Länge hat, und es so auch einfach ist, den Prozess der Vermehrung und Mutation zu implementieren. Der Nachteil ist aber, dass in der Regel nur sehr wenige Felder eine 1 enthalten, weil bei einer genauen Abtastung der Karte es viele mögliche Knotenpositionen gibt, aber nur wenige Knoten benötigt werden, um den Bereich gut abzudecken, wie in der folgenden Abbildung gezeigt wird. Die Instanz von Abbildung 2.3b hat z. B. 771 mögliche Positionen und es kann eine komplette Abdeckung dieser Positionen basierend auf der SPNE Metrik mit unter 40 Knoten erreicht werden (Abschnitt 5.3.2).

Eine andere Möglichkeit die Lösungen für dieses Problem zu repräsentieren besteht darin, die echten Index-Werte zu nutzen [18, S. 98f]. Das bedeutet, dass die Länge des Werte-Strings der Anzahl der gesetzten Sender entspricht und jeder Wert für den Index der Knotenposition steht. Ein Vorteil dieser Variante ist, dass so aus dem selben Grund wie in der ersten Variante die Länge recht kurz ist. Es ist aber für diesen Fall schwieriger die Rekombination und Mutation umzusetzen, weil man leicht unzulässige Lösungen erzeugt. Hier können ungültige Lösungen dadurch erzeugt werden, dass zwei Stellen in der Sequenz den gleichen Wert bekommen. Das Ganze kann natürlich durch einen Mehraufwand in der Umsetzung und auch Berechnung verhindern werden.

In dieser Arbeit wird wegen der besseren Umsetzbarkeit bzgl. der Rekombination und Mutation der Bit-String gewählt.

4.3.3 Initialisierung

Die Aufgabe der Initialisierung ist es die erste Generation der Population zu kreieren. Die Art und Weise der Umsetzung hängt von der Variante ab.

Bei der festen Variante wird die Positionen für die zu setzende Anzahl Knoten bestimmt. Diese Positionen bekommen im Bit-String dann den Wert 1. Die feste Variante wäre auch mittels der Kodierung der Indices gut möglich bei der Initialisierung, weil direkt die zufälligen Indices gespeichert werden.

Bei der flexiblen Variante ist der Bit-String hingegen von Vorteil, weil dieser erlaubt jedes Gen, also jedes Bit, mit einer bestimmten Wahrscheinlichkeit p mit einem Knoten zu füllen. Das hat zur Folge, dass sich in der Regel die Anzahl der Knoten in der Population um den Wert $n \cdot p$ befinden, wobei n für die Länge des Bit-Strings steht.

Eine andere Methode besteht darin die Ergebnisse eines anderen Algorithmus für die Initialisierung des G.A. zu nutzen. Es werden gerne einfache Heuristiken benutzt, die einem besser Lösungen als der simple Zufall verschaffen, so dass der G.A. mit besseren Individuen beginnt [18, S. 42]. Hier werden die besten k Ergebnisse einer größeren Zufallssuche als Initialisierung für einen G.A. mit einer Population der Größe k benutzt.

4.3.4 Selektion

Bei der Selektion werden zwei verschiedene Methoden von einigen möglichen Selektionsverfahren getestet [18, S. 46ff]. Sie ist selbstverständlich für die feste und flexible Variante die gleiche.

Eine sehr einfache Methode ist es die Individuen zufällig auszuwählen. Die Fitness wird bei der Selektion also überhaupt nicht beachtet. Diese Methode ist häufig nicht die beste, weil sie keinen Trend hat gute Lösungen zu bevorzugen. Sie wird aber als ein guter Vergleich zur zweiten Methode dienen.

Die andere Methode ist die Turnierauswahl. Bei dieser Methode treten eine bestimmte Anzahl von zufällig aus der Population ausgewählten Individuen gegeneinander an. Das Individuum mit der besten Fitness ist der Sieger des Turniers. Dies ist laut [18, S. 49] eine effiziente und optimale Methode.

Alternative Methoden, die ebenfalls die Fitness berücksichtigen, sind unter anderem die Roulette- und Rang-Auswahl. Die Rouletteauswahl funktioniert so, dass es eine Rouletteschеibe gibt, an der jedes Individuum der Population seinen Anteil am Rad so bekommt wie es der Division seiner Fitness und der Gesamtfitness der Population, also der Summer über alle Individuen, entspricht. Dieses Rad wird dann gedreht und das gewinnende Individuum wird ausgewählt. Das bedeutet natürlich, dass die Fitnesswerte positiv, auch

nicht 0, sein müssen, um einen Anteil am Rad berechnen zu können und es kann so nur die Selektion für eine Maximierung durchgeführt werden [18, S. 48]. Deshalb ist es hier ungeeignet, weil durch die Kombination mit der Anzahl der Knoten die resultierende Fitness oft auch negativ ist (Abschnitt 4.2).

Die Rang-Auswahl ist eine abgewandelte Form der Rouletteauswahl. Dabei bekommen die Individuen einen Fitnesswert gemäß ihres Rangs in der Population mit einer Größe N. Das beste Individuum hat so eine Fitness von N und das schlechteste eine Fitness von 1. Auf diesen Fitnesswerten wird so die Rouletteauswahl ausgeführt.

Weil man die Turnierauswahl deutlich besser mit der Wahl der Anzahl der Turnierteilnehmer einstellen kann, wird die gewählt.

Bei allen Verfahren wird hier eine Selektionswahrscheinlichkeit benutzt. Diese gibt an mit welcher Wahrscheinlichkeit die Selektion ausgeführt wird. So kann man einstellen welcher Anteil der Population für die Rekombination und Mutation bereitgestellt werden soll. Die Selektionswahrscheinlichkeit wird hier zusätzlich noch zu der Rekombinationswahrscheinlichkeit und der Mutationswahrscheinlichkeit benutzt, weil sich herausgestellt hat, dass eine häufige Mutation sinnvoll ist (Abschnitt 5.2). Damit kann das feiner eingestellt werden.

4.3.5 Rekombination

Bei der Vermehrung muss entschieden werden wie viele Nachfahren aus einem Vermehrungsvorgang hervorgehen und wie viele Eltern daran beteiligt sind. Die Anzahl der Eltern beträgt in der Regel 2 [18, S. 50ff] und die Anzahl der Kinder ist normalerweise 1 oder 2. In dem Framework DEAP wird häufig 2 verwendet, wie z. B. beim Two-Point Crossover [28]. Deswegen wird das in dieser Arbeit auch so umgesetzt. Das ist auch aus Performance-Gründen ein Vorteil, weil so nicht so oft eine Rekombination durchgeführt werden muss, um eine bestimmte Anzahl Nachfahren zu produzieren. Die Methode zur Vermehrung hängt hier hauptsächlich davon ab, ob die Knotenanzahl fest oder variabel ist.

Bei der festen Variante muss die Anzahl der Knoten natürlich nach dem Rekombinationsschritt die gleiche sein wie vorher. Dies muss also bei der hier gewählten Methode, die von [17] abgeleitet wurde, beachtet werden. Das Verfahren (Bsp. in Gleichung 4.3) geht so, dass beide Kinder von beiden Eltern die Schnittmenge der Positionen der Eltern übernehmen. Das sind in der Regel natürlich weniger als die Gesamtzahl der Knoten, welche die Kinder haben müssen. Von den Knoten der Eltern, die nicht in der Schnittmenge enthalten sind, wird die passende Anzahl zufällig ausgewählt, so dass die Kinder die gleiche Anzahl an STAs haben wie die Eltern. In der Regel sind die beiden Kinder so auch unterschiedlich, weil für jedes Kind einzeln zufällig die übrigen

Knoten gezogen werden.

$$\begin{aligned} Eltern1 &: 1001011000001 \\ Eltern2 &: 0101001001100 \\ (Schnittmenge &: 0001001000000) \\ &\downarrow \\ Kind1 &: 1101001000001 \\ Kind2 &: 1001011000100 \end{aligned} \tag{4.3}$$

Populäre Varianten wie das Single-Point, Two-Point oder auch Uniform Crossover sind nicht ohne weiteres auf die feste Variante anwendbar, weil die fixe Anzahl der Knoten nicht garantiert werden kann. Dazu müsste man diese bei jedem Schritt überprüfen und anpassen. Deshalb werden diese Methoden nicht für die feste Variante benutzt.

Die beiden Methoden Two-Point und Single-Point Crossover werden für die flexible Variante eingesetzt. Informationen zu deren Funktionsweise finden sich in [18, S. 51].

4.3.6 Mutation

Die Mutation ist bei beiden Varianten recht einfach. Bei der festen Variante werden eine 1 und eine 0 getauscht [18, S. 57][17]. Das bedeutet, dass die Position eines Knoten geändert wird. Daraus folgt natürlich, dass die Anzahl der Knoten nach der Mutation noch die gleiche ist wie vorher.

Bei der flexiblen Variante wird zufällig genau ein Bit oder jedes Bit mit einer bestimmten Wahrscheinlichkeit geflippt [18, S. 57]. Das sorgt dafür, dass die Lösung beim flippen genau eines Bits entweder einen Knoten mehr oder weniger hat als vorher. Bei der Variante mit der Wahrscheinlichkeit ist das Festlegen von dieser von zentraler Bedeutung.

Eine übliche Herangehensweise ist es den Wert der Wahrscheinlichkeit so zu setzen, dass er ähnlich zu dem reziproken der Länge des Bit-Strings ist [18, S. 56].

4.3.7 Ersetzung

Die Ersetzung bestimmt nach welchem Verfahren die Nachfahren einer Generation in die Population aufgenommen werden. In dieser Arbeit ist die Populationsgröße konstant. Es gibt Alternativen, bei denen die Populationsgröße sich verändert, also meist erst größer wird und danach wieder schrumpft [18, S. 58]. Hier wird der Einfachheit halber aber die Größe konstant gelassen.

Das bedeutet, dass die Nachfahren einen Teil der älteren Individuen ersetzen müssen. Weil man die Fitness der Individuen natürlich verbessern möchte, ist es in der Regel ratsam eher die schlechten Lösungen zu ersetzen. Es werden dazu in dieser Arbeit drei verschiedene Varianten untersucht.

Im vorigen Abschnitt wurde die Turnierauswahl vorgestellt (Kapitel 4.3.4). Als Gegenstück zur Selektion gibt es bei der Ersetzung das Turnier, bei dem das schlechteste Individuum gewinnt und so ersetzt wird. Diese beiden Methoden arbeiten häufig gut zusammen [18, S. 58].

Daneben wird die zufällige Ersetzung implementiert. Alle Nachfahren ersetzen also zufällig einen ihrer Vorfahren. Diese Variante erfordert wenig Rechenaufwand, aber hat keine Tendenz dazu die Individuen mit einer höheren Fitness zu bevorzugen.

Eine weitere bekannte Variante ist, dass die Kinder die Eltern ersetzen. Diese Methode ist ebenfalls sehr effizient bzgl. des Rechenaufwands. Hier passt es sehr gut, dass beim Schritt der Rekombination (Abschnitt 4.3.5) immer zwei Kinder erzeugt werden, so dass diese auch beide Eltern ersetzen können.

4.3.8 Stop-Kriterium

Beim Stop-Kriterium gibt es wie bei den anderen Schritten viele Möglichkeiten [18, S. 59] Die simpelste Variante besteht darin nach einer gegebenen Anzahl von Generation aufzuhören. Diese Methode wird hier auch eingesetzt. Das Problem ist nur, dass dabei in keinster Weise beachtet wird, ob eine optimale Lösung erreicht wurde. Bei G.A. ist es nie möglich zu wissen, dass man eine optimale Lösung erreicht hat. Es gibt aber Indizien dafür, dass sich keine bessere Lösung mehr finden lässt. Ein Indiz besteht darin, dass sich das beste Individuum nicht für eine festgelegte Generationenanzahl verbessert hat. Diese Variante wird ebenfalls implementiert.

4.4 Vergleichsalgorithmen

Damit die Wirksamkeit des G.A. besser überprüft werden kann, werden drei der schon beschriebenen Verfahren ausgewählt (Kapitel 3). Es sind die Zufallssuche (Abschnitt 4.4.1), die lokale Suche (Abschnitt 4.4.2 und ein von der lokalen Suche abgewandeltes Eröffnungsverfahren (Abschnitt 4.4.3).

4.4.1 Zufallssuche

Die **Zufallssuche** dient nur als simpler Vergleich, um zu bestimmen, ob der G.A. überhaupt besser ist, weil es wohl ab und zu vorkommt, dass dem nicht

so ist [18, S. 26f]. Bei der Variante, in der eine variable Anzahl an Knoten sich in einer Lösung befinden darf, beginnt die Zufallssuche bei einer Knotenanzahl, bei der es offensichtlich ist, dass damit ein bestimmter Fitnesswert erreicht werden kann, also im schlimmstem Fall die Anzahl der möglichen Positionen für die Platzierung der Knoten. Wenn eine Lösung gefunden wurde, die die Fitnesskriterien erfüllt, dann wird die Anzahl der zu setzenden Knoten um 1 verringert. Bei der festen Variante werden zufällig Lösungen mit genau der gewünschten Anzahl Knoten generiert. Es wird nicht darauf geachtet, das schon generierte Lösungen nicht erneut generiert werden, was in Richtung einer Tabusuche gehen würde.

4.4.2 Lokale Suche

Für die **lokale Suche** ist das Entscheidende die Nachbarschaft zu definieren. Hier wird eine variable Nachbarschaftsstruktur benutzt, die sich nach jeder Iteration ändert [29]. Diese wird hier so umgesetzt, dass das Verschieben eines bestimmten Knoten der Übergang zu einem Nachbarn bedeutet. Die Verschiebung der verschiedenen Knoten der Lösung passiert so nacheinander. Als Erstes werden für den ersten Knoten alle Nachbarn bestimmt. Von diesen wird die Fitness berechnet und der beste ausgewählt. Wenn der beste Nachbar besser als die jetzige Lösung ist, dann ersetzt dieser die jetzige. Dann wird von dem ggf. neuen Ergebnis aus der nächste Knoten verschoben, um den besten Nachbarn davon zu bestimmen und ggf. als neue Lösung zu benutzen. Das Ganze passiert bis der letzte Knoten verschoben wurde. Danach geht es wieder beim ersten Knoten los solange bis in einem ganzen Durchgang vom ersten bis letzten Knoten keine Verbesserung vorkam. Es werden also in vielen Zyklen nacheinander die Positionen der Knoten einzeln verbessert. Die Nachbarschaft ist also variabel darin, welchen Teil der Lösung sie optimiert, aber nicht variabel in der Nachbarschaftsgröße. Diese beträgt nämlich konstant $n - k$, wobei n der Anzahl möglicher Knotenpositionen und k der Anzahl der Knoten entspricht. Diese Vorgehensweise hat zur Folge, dass die lokale Suche so nur für die feste Variante eingesetzt werden kann.

4.4.3 Eröffnungsverfahren

Für Greedy-Algorithmen muss man sich deterministische Regeln überlegen, um eine Lösung zu generieren. In dieser Arbeit ist abgewandelt von der lokalen Suche so dieses Verfahren entstanden. Das Eröffnungsverfahren wird so konstruiert, dass es mit einer Lösung beginnt, in der kein Knoten gesetzt ist. Dann werden alle möglichen Platzierungen dieses Knotens getestet und die Platzierung gewählt, welche die höchste Fitness hat. Danach wird ein weiterer

Knoten hinzugefügt, indem alle Möglichkeiten getestet werden. Das ganze wird bei der festen Variante so lange fortgesetzt, bis die Anzahl der zu setzenden Knoten erreicht ist. Bei der flexiblen Variante wird das so lange wiederholt, bis die Platzierung der Knoten eine bestimmte Fitness hat. Ein Ziel könnte z. B. sein, dass die SPNE Metrik einen Wert von 1 hat.

4.5 Zusammenfassung

In diesem Kapitel wurde detailliert die eigene Implementation des G.A. mit samt der Verarbeitung der Daten von RaLaNS vorgestellt. Dabei wurden nacheinander die einzelnen Komponenten beschrieben und Alternativen zu den eingesetzten oder mehrere mögliche Implementationen, die in der Evaluation untersucht werden, genannt. Zum Schluss wurde noch die Umsetzung der Algorithmen, die als Vergleich zum G.A. dienen, erläutert.

5 Evaluation

Dieses Kapitel beinhaltet die Parametrisierung und Auswertung der Algorithmen, die in der Umsetzung (Kapitel 4) beschrieben wurden. Dazu werden zuerst die Szenarien beschrieben, auf denen die Untersuchung stattfindet (Abschnitt 5.1). Danach werden in Kapitel 5.2 die Parameter des G.A. analysiert und eingestellt. Im folgenden Abschnitt wird der G.A. mit den anderen Methoden hinsichtlich seiner Wirksamkeit verglichen (Abschnitt 5.3). Im letzten Abschnitt werden die Unterschiede der Ergebnisse bzgl. der Wahl des Signalausbreitungsmodells gezeigt (Abschnitt 5.4).

5.1 Szenarien

Es werden zwei verschiedene Szenarien untersucht. Beide beziehen sich auf das selbe Gebiet, nämlich das Gebiet um den Sedanplatz (Osnabrück) herum, für den die OSM Karte in Abbildung 5.1 zu sehen ist. Dieses Gebiet ist hauptsächlich eine Wohngegend in der Stadt, so dass auf ihr viele Einfamilienhäuser und kleinere Straßen sind. Ebenfalls befindet sich ein Teil des Westerberg Campus der Uni Osnabrück auf der Karte, so dass auch einige deutlich größere Gebäude sich dort befinden. Diese Karte stellt also eine gute Repräsentation für ein städtisches Gebiet dar.

Ein Szenario setzt sich aus den Punkten zusammen, die in Abbildung 2.3b zu sehen sind. Es besteht aus einer Liste von Positionen. Die Positionen befinden sich in dem Szenario zum Teil deutlich um das Gebiet mit den Gebäuden herum, weil die Positionen sich auf den Straßen befinden und das Platzieren dieser Positionen so passiert ist, dass Positionen gesetzt wurden, die über das Gebiet mit den Gebäuden hinausgehen. Das hat für die Berechnung hier aber nur den Nachteil, dass das Szenario aus mehr Punkten besteht, also umfangreicher zu berechnen ist. Das andere Szenario ist ein Grid von Positionen (Abbildung 5.2). Die Schrittweite (Distanz) zwischen den Positionen in den beiden Dimensionen beträgt 30 Meter.

Für beide Szenarien muss noch eine untere Grenzsignalstärke bestimmt werden, so dass gesagt werden kann, ob eine Verbindung zustande kommen

Abbildung 5.1: OSM Karte für das quadratische Gebiet um den Sedanplatz (Osnabrück), Screenshot des Programms JOSM [1] mit geladener Sedanplatz OSM Karte

kann oder nicht. Dafür wird die zweite Methode benutzt, die in Abschnitt 2.2.3 beschrieben wurde. Es werden für jedes Szenario drei Punkte bestimmt, die ungefähr einen Abstand von 250 Meter zueinander haben und nicht per Luftlinie durch ein Gebäude behindert werden. Die Werte werden gemittelt und auf einen passenden Wert gerundet. In Tabelle 5.1 werden die Werte und die Anzahl der Positionen, für die RaLaNS Daten berechnet hat, aufgezählt.

Szenario	**Anzahl Positionen**	**Grenzsignal-stärke**
Liste Sedanplatz	771	-28
Fläche Sedanplatz, Schrittweite 30	992	-29

Table 5.1: Beschreibung der Größe und Grenzsignalstärke der Szenarien

5.2 Parametrisierung

Die Parametereinstellung ist von wichtiger Bedeutung für den G.A., weil von ihr sehr stark die Effektivität des Algorithmus abhängt. Es gibt bei G.A.

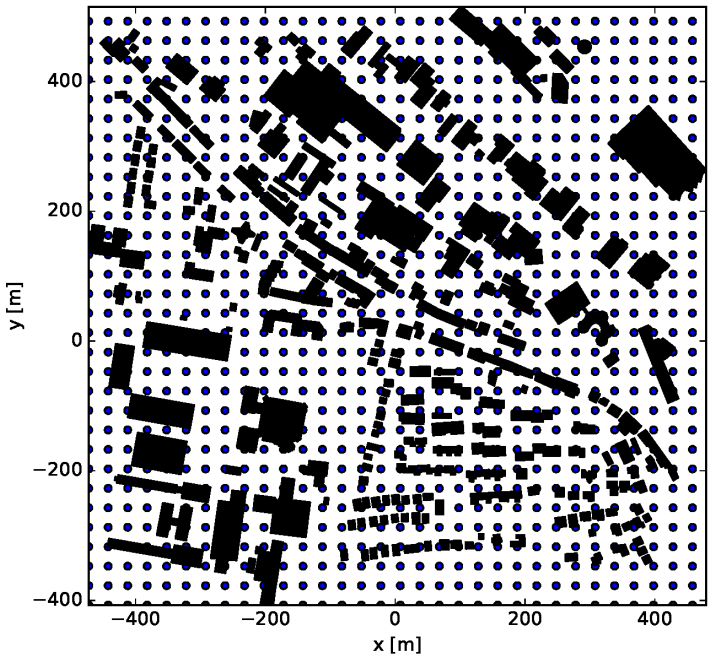


Abbildung 5.2: Grid-Szenario des Sedanplatzes mit einer Schrittweite von 30

dabei mehrere Arten, um die Einstellung vorzunehmen. Einerseits können die Parameter die ganze Zeit fixiert sein, so wie hier, oder aber über die Zeit verändert werden, je nachdem in welcher Phase sich der G.A. gerade befindet [30]. Die Variante mit den fixen Parametern wird hier gewählt, weil sie einfacher zu überblicken und einzustellen ist.

Zur Optimierung der Parameter wird hier ein einfaches Verfahren gewählt, bei dem die Parameter nacheinander bzgl. selbst gewählter sinnvoll erscheinender Werte optimiert werden. Dazu müssen alle Parameter erst eine Voreinstellung haben (Kapitel 5.2.1). Ausgehend von diesen Standardwerten werden die Parameter in Abschnitt 5.2.2 nacheinander eingestellt. Das wird nacheinander für einige ausgewählte Parameter getan.

Das ist wie oben erwähnt nicht die beste Variante, um die Parameter zu optimieren, aber sie wird auf Grund der Komplexität der Parameter-Optimierung und des Zeitaufwandes gewählt.

Ebenfalls aus Zeitgründen wird die Parametrisierung auf die SPNE-Fitness und das Szenario auf dem Sedanplatz mit der Liste von Positionen beschränkt. Es wäre natürlich besser diese auf mehreren Szenarien durchzuführen, um zu sehen, ob es Unterschiede in der Parametereinstellung gibt oder nicht, insbesondere weil in dem Szenario mit der Liste die Positionen nur auf der Straße angeordnet sind und bei dem anderen die Positionen ein Grid ergeben.

5.2.1 Voreinstellung

Hier wird den Parametern Standardwerte gegeben. Auf Grund der großen Parameterzahl wird eine Vorauswahl an Parameterern für die Optimierung getroffen, die besonders kritisch für die Gesamt-Wirksamkeit des G.A. erscheint. Die anderen Parameter behalten folglich nur ihren Standardwert. Ein weiterer Grund für die Reduktion der Parameterzahl liegt in der Berechnungszeit der Durchläufe. Diese liegt je nach Rechner, der eingesetzt wurde, bei einigen Stunden pro Durchlauf.

In dieser Arbeit umschließt der G.A. einige Parameter mit ihren Standardwerten. Diese sind aufgeteilt nach qualitativen Parameter (Tabelle 5.2) und quantitativen Parameter (Tabelle 5.2) jeweils nach der festen und flexiblen Variante. Die möglichen Verfahren und deren Wahrscheinlichkeiten, die hier die Parameter des G.A. sind, wurden in Abschnitt 4.3 beschrieben. Qualitative Parameter entscheiden darüber welches Verfahren benutzt wird, quantitative darüber in welchem Umfang das passiert.

	Variante	
Qualitative Parameter	fixe Anzahl	flexible Anzahl
Vermehrung	nach Lochert	Two-Point Crossover
Mutation	nach Lochert	Random-FlipBit
Initialisierung	feste Anzahl zufällig	Zufall mit Wahrscheinlichkeit
Selektion	Turnierauswahl, 3 Teilnehmer	
Ersetzung	Turnierauswahl, 3 Teilnehmer	
Stop-Kriterium	keine Verbesserung der besten Lösung	

Table 5.2: Voreinstellung der qualitativen Parameter

Bei der Initialisierung wird hier als Voreinstellung die Methode mit zufällig gesetzter fester Anzahl für die fixe Variante und ein Setzen jedes Bits mit einer gegebenen Wahrscheinlichkeit für die flexible Variante.

Bei der fixen Variante wird für die Vermehrung und Mutation das Verfahren nach Lochert als Voreinstellung gewählt, weil hier keine Alternativen implementiert wurden, so dass es gar nicht anders möglich ist. Bei der flexiblen Variante ist es bei der Vermehrung so, dass das Two-Point Crossover und Single-Point Crossover sehr ähnlich sind. Hier wird als Voreinstellung das Two-Point benutzt. Für die Mutation wird das zufällige Flippen als Voreinstellung gebraucht, weil man so mehr Variabilität in den Mutationen hat und es auch eine Veränderung um mehr als einen Knoten durch eine Mutation geben kann, als wenn man nur das OneFlipBit benutzt. Als Voreinstellung für die Wahrscheinlichkeit ein Bit

beim Random-FlipBit zu flippen, wird das Reziproke der Länge des Bit-Strings benutzt, also z. B. für das Szenario für die Liste von Positionen wäre das 1/771.

Für die Selektion und Ersetzung wird bei beiden die Turnierauswahl mit drei Teilnehmern als Standard benutzt, weil sich die beiden als gute Kombination erwiesen haben, wie in Abschnitt 4.3.7 erläutert wurde.

Für die Selektion, Vermehrung und Mutation müssen Wahrscheinlichkeiten festgelegt werden. Diese Werte bedingen sich gegenseitig, weil z. B. eine höhere Selektion bei gleicher Vermehrungswahrscheinlichkeit natürlich dazu führt, dass mehr Individuen gepaart werden. Für die Selektion wird hier $0,7$ gesetzt, weil es ratsam ist einen Teil der alten Population beizubehalten, aber natürlich nicht zu wenig ausgewählt werden sollen, weil ansonsten der G.A. deutlich mehr Generationen benötigt, um die gleiche Anzahl an veränderten Individuen zu erzielen [18, S. 56]. Die Mutation soll typischerweise sehr niedrig sein. Deshalb wird hier erst $0,05$ gewählt [18, S. 57]. Für die Vermehrung wird ein hoher Wert, nämlich $0,8$, genommen, weil man hofft, dass sich durch die Paarung die Fitness der Nachfahren besser ist [18, S. 50].

Für den Vergleich mit den anderen Algorithmen wird als Stop-Kriterium die Variante benutzt, bei der nach einer bestimmten Anzahl von Generationen ohne Verbesserung gestoppt wird. Für die Parametrisierung eignet sich das jedoch nicht so gut. Da ist es deutlich einfacher die Generationenanzahl zu fixieren, um die Parameter zu optimieren. Weil sich in einigen Läufen dabei ergeben hat, dass nach ungefähr 300 Generationen sich für das Listenszenario nicht mehr viel ändert, ist die Voreinstellung dafür 330 (Abbildung 5.3). In der Abbildung ist zu sehen, dass die Fitnesssteigerung vor Generation 100 deutlich größer ist als in den Generationen danach von 100 bis 500.

Die Standard-Populationsgröße wird auf 40 gesetzt. Ein praktikabler Wert für die Populationsgröße ist 100 [18, S. 42]. Der Wert wird hier auf Grund der Abwägung zwischen der Lösungsqualität und der Rechnezeit niedriger gewählt.

Die Größe der Hall of Fame ist 1, weil in dieser Auswertung nur die beste Lösung berücksichtigt wird und nicht mehrere.

5.2.2 Parameter Tuning

Die Einstellung der Parameter wird wie weiter oben erklärt nur für bestimmte Parameter durchgeführt. Es wird hier so vorgegangen, dass zuerst beschrieben wird welche Parameter eingestellt werden und welche nicht. Danach wird die Einstellung erst für die feste Variante und danach für die flexible Variante durchgeführt. Als Grundlage für die Ergebnisse der Parametereinstellung dienen in beiden Varianten 7 oder etwas mehr Durchläufe pro Einstellung als Datengrundlage. Es wäre natürlich besser mehr Durchläufe zu machen. Dies war aber aus Zeitgründen nicht möglich.

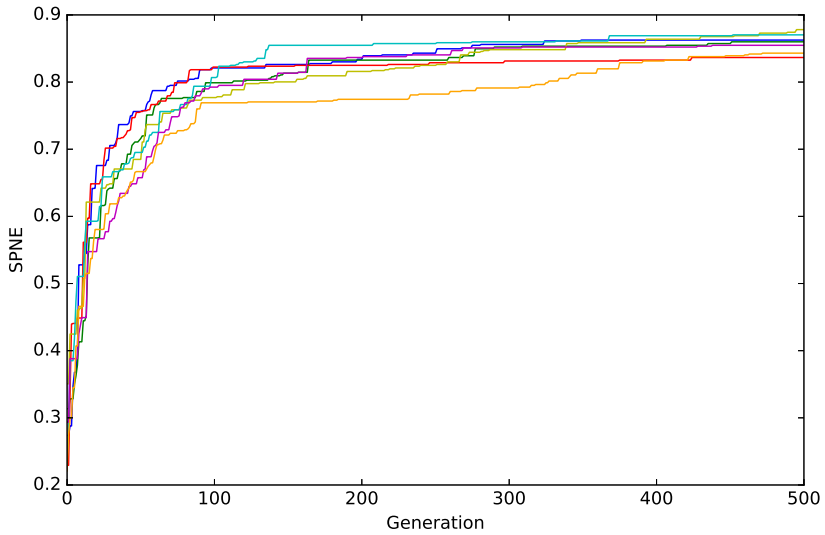


Abbildung 5.3: typische Verläufe des G.A. mit fester Anzahl Knoten für das Listenszenario

Hier werden von den quantitativen Parameter die Mutationswahrscheinlichkeit, Crossover-Wahrscheinlichkeit und Populationsgröße optimiert. Die Selektionswahrscheinlichkeit wird ausgelassen, weil dieser Parameter schon leicht über die Einstellung der Mutations- und Vermehrungswahrscheinlichkeit beeinflusst wird.

Damit bei der fixen Variante die Resultate überhaupt vergleichbar sind, ist die Anzahl der Knoten fest auf 20 gesetzt. Bei der flexiblen Variante wäre eine Anpassung möglich, wird aber auch aus Zeitgründen weggelassen.

Von den qualitativen Parametern wird die für beide Varianten die Selektion und Ersetzung untersucht. Bei der flexiblen kommt noch die Auswahl für die Mutation und Vermehrung dazu. Die Einstellung des Initialisierungsverfahrens wird hier nicht durchgeführt. Es gäbe, wie in Abschnitt 4.3.3 beschrieben, Alternativen. Diese werden hier aber ebenfalls nicht untersucht.

Die Ergebnisse der Berechnungsdurchläufe werden in Boxplots dargestellt, z. B. Abbildung 5.4. In diesen Abbildungen steht die rote Linie für den Median der Werte und das rote Quadrat für den Durchschnitt.

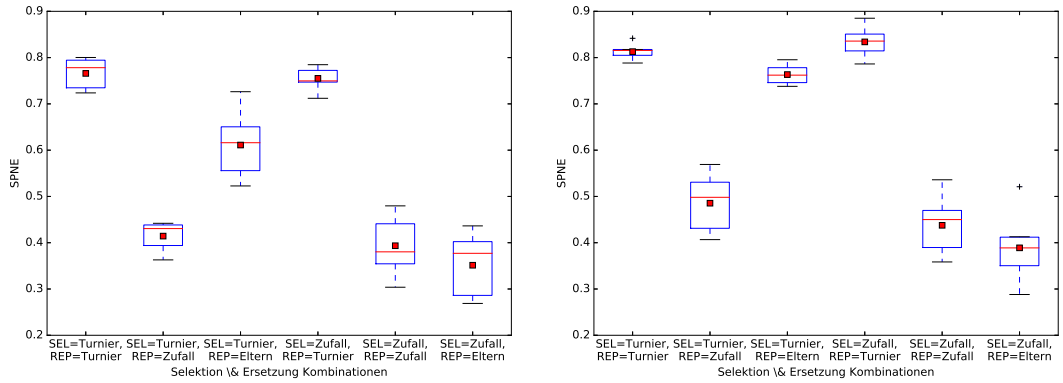
Parametrisierung feste Variante

	Variante	
Quantitative Parameter	fixe Anzahl	flexible Anzahl
Vermehrung Wahrscheinlichkeit	$0,8$	
Mutation Wahrscheinlichkeit	$0,05$	
Random-FlipBit Wahrscheinlichkeit		Reziproke der Länge des Bit-Strings
Initialisierungsparameter	20	$0,1$
Selektion Wahrscheinlichkeit	$0,7$	
Populationsgröße	40	
Größe der Hall of Fame	1	
Anzahl Generationen keine Verbesserung	150	
Anzahl Generationen für Parametrisierung	200	

Table 5.3: Voreinstellung der quantitativen Parameter

Für diese Variante wird so vorgegangen, dass nacheinander die Parameter Selektion zusammen mit Ersetzung, Crossover-Wahrscheinlichkeit, Mutationswahrscheinlichkeit und Populationsgröße eingestellt werden.

Für die Selektion gibt es zwei Varianten, nämlich die Turnierauswahl und zufällige Auswahl (Abschnitt 4.3.4). Für die Ersetzung gibt es drei Methoden, die umgekehrte Turnierauswahl, die zufällige Auswahl und die Ersetzung der Eltern (Abschnitt 4.3.7). Diese werden miteinander kombiniert, so dass sich sechs Optionen ergeben. In Abbildung 5.4a sind diese Optionen für die Liste von Positionen und in Abbildung 5.4b für die Fläche von Positionen zu sehen. Sie werden nach der SPNE-Fitness des besten Individuums in der Hall of Fame am Ende des Durchlaufs verglichen. Es fällt auf, dass für ein gutes Ergebnis es hauptsächlich darauf ankommt, dass das Verfahren der Ersetzung die Turnierauswahl ist. Dies sorgt nämlich dafür, dass auf jeden Fall nie die besten Lösungen ersetzt werden und vor allem schlechte Lösungen ausgewählt werden. Leicht besser scheint dabei für das Listenszenario die Kombination, bei welcher auch bei der Selektion die Turnierauswahl benutzt wird. Bei dem Flächenszenario ist hingegen die Kombination aus Zufall und Turnier leicht besser. Es kann also hier nicht gesagt werden welche Kombination der beiden die bessere ist. Schlussfolgernd bedeutet dies, dass für die Selektion und Ersetzung die Turnierauswahl die gewählte Methode bleibt.



(a) Vergleich für das Liste-Szenario **(b)** Vergleich für das Grid-Szenario

Abbildung 5.4: Vergleich der 6 Kombinationen von Selektion und Ersetzung für G.A. mit fester Anzahl Knoten für beide Szenarien, (SEL steht für Selektion, REP steht für Ersetzung)

Als Nächstes wird die Crossover-Wahrscheinlichkeit untersucht. Es wird ein Vergleich zwischen den Werten $0,6$; $0,7$; $0,8$; $0,9$; 1.0 durchgeführt (Abbildung 5.5). Bei dem Vergleich wird darauf geachtet, dass die Anzahl der Evaluationen ungefähr gleich ist. Eine höhere Crossover-Wahrscheinlichkeit führt natürlich dazu, dass mehr Individuen gepaart und somit auch öfter die Fitness neu berechnet werden muss, d.h. es gibt eine höhere Anzahl an Evaluationen von Lösungen. Ein Vergleich macht natürlich nur Sinn, wenn diese Anzahl ungefähr gleich ist. Sie ist nicht ganz gleich, weil immer die ganze Population ausgewertet wird und somit Unterschiede im Größenbereich kleiner der Populationsgröße auftreten können. Die Anzahl der Evaluationen liegt für alle Berechnungen ungefähr bei 5840, was ungefähr bei $330 \cdot 40 \cdot 0,7 \cdot 0,6 = 5544$ liegt, was der Anzahl der Evaluationen der Instanz mit der kleinsten Wahrscheinlichkeit von $0,6$ entspricht. 330 für die Anzahl der Generationen, 40 für die Populationsgröße, $0,7$ für die Selektionswahrscheinlichkeit und $0,6$ die Crossover-Wahrscheinlichkeit. Der Unterschied von ungefähr 300 zu 5840 kommt daher, dass es auch noch die Mutation von $0,05$ gibt, die diesen Wert auch noch erhöht.

Das Ergebnis dieses Vergleichs ist, dass $0,7$ die beste Wahl zu sein scheint. Der Unterschied ist aber auf jeden Fall nicht groß. Deswegen wird die Vermehrungswahrscheinlichkeit auf $0,7$ geändert.

Nun wird die Einstellung der Mutationswahrscheinlichkeit vorgenommen. Dafür wird ein Vergleich zwischen den Werten $0,01$; $0,05$; $0,1$; $0,2$ und $0,3$ durchgeführt. Bei diesem wird wie bei der Vermehrung darauf geachtet, dass die Anzahl der Fitnessberechnungen ähnlich ist. Generell ist es so, dass eine

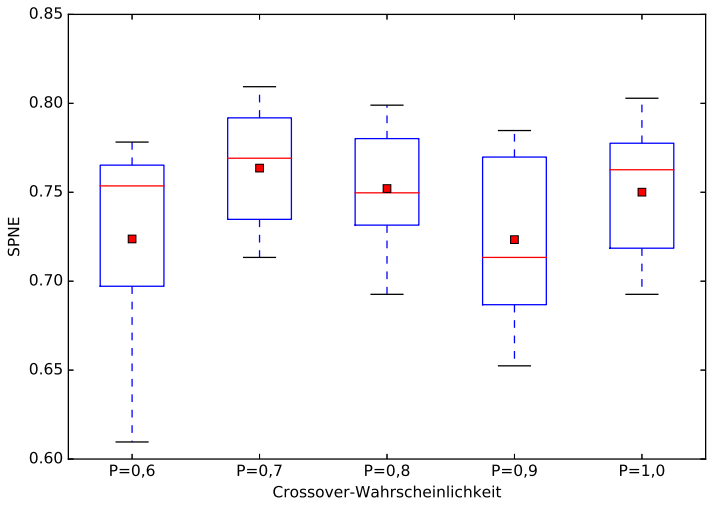


Abbildung 5.5: Vergleich verschiedener Crossover-Wahrscheinlichkeiten für die Parametrisierung des G.A. mit fester Anzahl Knoten und ausgeglichener Anzahl Evaluationen

niedrige Mutation sinnvoll ist. Hier stellt sich aber heraus, dass für dieses Problem eine höhere Mutationsrate Vorteile bietet. Die beste Fitness wird nämlich mit einem Wert von $0,3$ erzielt. Es existieren aber keine Ergebnisse für Durchläufe mit einer Mutationsrate, die größer als $0,3$ ist.

Die Gründe für diesen merkwürdigen Parameterwert werden nicht genauer untersucht, es könnte aber sein, dass es unter anderem daran liegt, dass die Mutation nach Lochert nicht viel ändert und die Mutation Vielfalt in die Population bringt, die für eine Fitnesswertsteigerung mancher Individuen benötigt wird, weil beim Crossover nach Lochert keine neuen Positionen genutzt werden, bei der Mutation aber natürlich schon, weil die Position eines Knoten zufällig geändert wird.

Zum Schluss dieser Parametereinstellung wird noch die Populationsgröße optimiert (Abbildung 5.7). Dazu werden die Größen 20, 40, 60 und 80 miteinander verglichen. Wichtig ist, dass hier auch die Anzahl der Fitnessberechnungen ähnlich ist für die Vergleichbarkeit. Dazu wird die Anzahl der Generationen abhängig von den Populationsgrößen gesetzt. Diese betragen also 660, 330, 220 und 165. Die Unterschiede zwischen den Ergebnisse sind hier sehr gering und auf jeden Fall nicht statistisch signifikant. Weil die Werte sich nicht wirklich unterscheiden, wird die höchste Populationsgröße gewählt, also 80, da das den Vorteil hat, dass die Evaluation der Nachfahren wegen der Parallelisierung relativ gesehen schneller funktionieren sollte, wenn die Population größer ist. Das liegt daran, dass das Starten der Parallelisierung an sich Rechenleistung

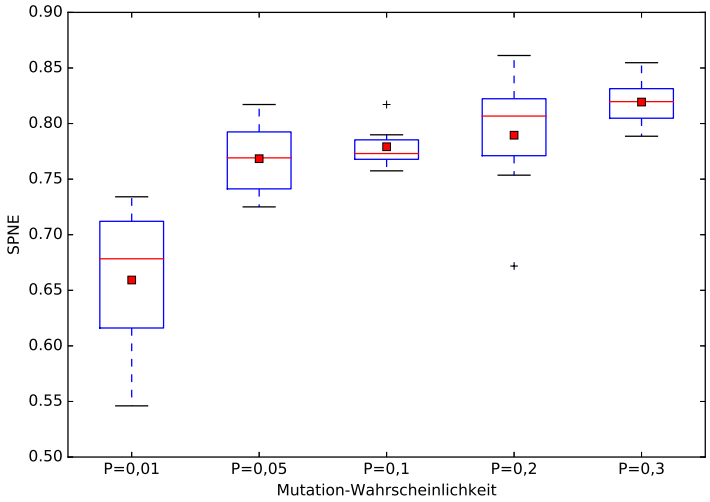


Abbildung 5.6: Vergleich verschiedener Mutationswahrscheinlichkeiten für die Parametrisierung des G.A. mit fester Anzahl Knoten und ausgeglichener Anzahl Evaluationen

kostet, so dass vereinzelte, länger andauernde Parallelisierungen mehreren, kurzen vorzuziehen ist. Bei kleineren Populationen wäre das Einleiten der Parallelisierung aber häufiger der Fall.

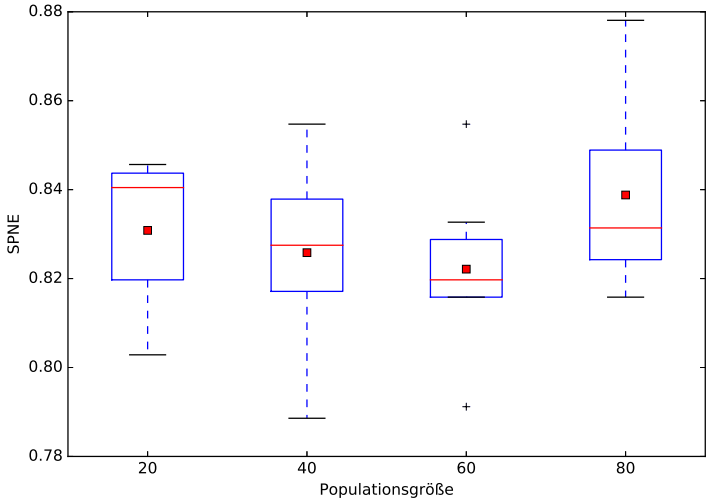


Abbildung 5.7: Vergleich verschiedener Populationsgrößen für die Parametrisierung des G.A. mit fester Anzahl Knoten und ausgeglichener Anzahl Evaluationen

Parametrisierung flexible Variante

Für die Parametrisierung bei der flexiblen Variante müssen die Gewichte der beiden Metriken eingestellt werden, wie es Gleichung 4.2 ausdrückt. Das Gewicht für die Anzahl der Knoten ist -1 und das Gewicht für SPNE muss so hoch liegen, dass der Wert von SPNE primär optimiert wird. Ein Wert von 200 reicht für das Listenszenario aus, weil schon Lösungen mit einem SPNE Wert von 1 mit weniger als 40 Knoten erreicht werden können (Abschnitt 5.3.2). Die feine Einstellung dieser Gewichte und welche Folgen das auf eine gute Platzierung hat, wird hier nicht behandelt.

Bei der Parametrisierung für die flexible Variante wird so vorgegangen, dass zuerst das Vermehrungsverfahren und dann das Mutationsverfahren eingestellt wird. Die übrigen Parameter werden aus Zeitgründen nicht eingestellt. Sinnvoll wäre hier aber auch wohl das Selektions- und Ersetzungsverfahren, so wie die Wahrscheinlichkeiten für die Rekombination, Mutation und Selektion einzustellen.

Die Boxplots messen hier die Anzahl der platzierten Knoten, weil diese minimiert werden sollen unter der Bedingung, dass der SPNE-Wert 1 ist. Dazu wurde das mit der Summe der beiden Metriken umgesetzt (Abschnitt 4.2). Abbildung 5.8 zeigt den durchschnittlichen Wert der SPNE-Metrik der finalen Population für die Crossover-Verfahren. Die Werte sind für alle Durchläufe 1. Das bestätigt, dass diese Vorgehensweise funktioniert.

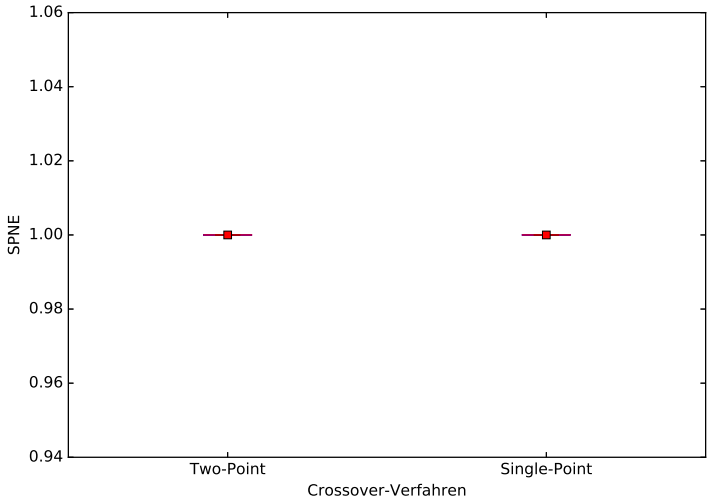


Abbildung 5.8: Verdeutlichung des SPNE-Wertes von 1 für die flexible Variante des G.A.

In Abbildung 5.9 werden die beiden Crossover-Verfahren dargestellt. Die

Wahl des Verfahren scheint aber nicht wirklich einen Unterschied auf die Anzahl der Knoten zu machen. Ein eher unwahrscheinlicher Grund dafür könnte aber auch in der geringen Anzahl von Durchläufen liegen.

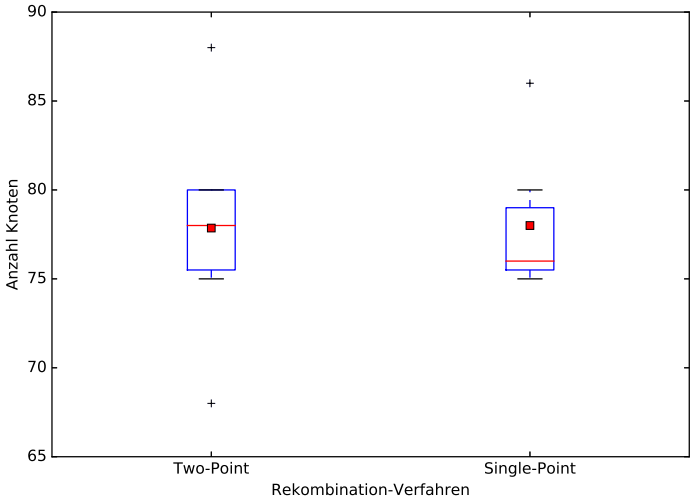


Abbildung 5.9: Vergleich der beiden Crossover-Verfahren für die flexible Variante des G.A.

In Abbildung 5.10 werden die Mutationsverfahren verglichen. Dabei wird das Random-FlipBit mit den Wahrscheinlichkeiten 1/771, 2/771 und 4/771 und das OneFlipBit miteinander verglichen. Hier ist wiederum so, dass die Unterschiede recht klein sind und kein statistisch relevanter Unterschied zu sehen ist. Es wird trotzdem das Mutationsverfahren gewählt, das die besten Ergebnisse erzielt, nämlich das Random-FlipBit mit der Wahrscheinlichkeit von 2/771.

5.3 Auswertung

In diesem Abschnitt werden die Ergebnisse der Vergleichsalgorithmen aus Abschnitt 4.4 vorgestellt. Zuerst werden die Ergebnisse der Zufallssuche präsentiert (Abschnitt 5.3.1), danach die Ergebnisse des Eröffnungsverfahrens (Abschnitt 5.3.2) und als drittes noch die lokale Suche (Abschnitt 5.3.3). Abschließend werden in diesem Kapitel die Ergebnisse mit den Ergebnissen des G.A. verglichen (Abschnitt 5.3.4).

Zum besseren Vergleich der Wirksamkeit der Algorithmen wird zusätzlich zur erreichten Fitness die Anzahl der Fitnessberechnungen genannt. Es wäre vielleicht sogar sinnvoller statt der Anzahl Fitnessberechnungen die Dauer der

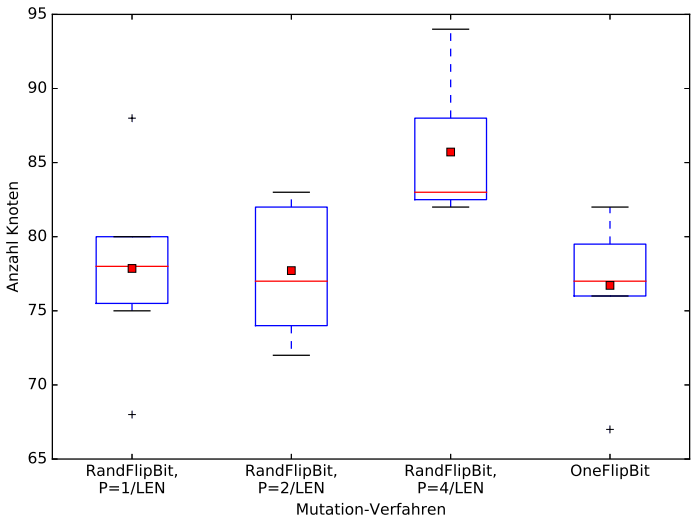


Abbildung 5.10: Vergleich der Mutationsverfahren für die flexible Variante des G.A.

Berechnungen zu messen. Das ist aber hier nicht möglich, weil die Berechnungen auf mehreren Rechnern durchgeführt wurden. Die Fitnessberechnung ist sowieso für den Großteil der Berechnungszeit verantwortlich, so dass das auch eine sinnvolle Bewertung des Berechnungsaufwandes sein kann.

5.3.1 Zufallssuche

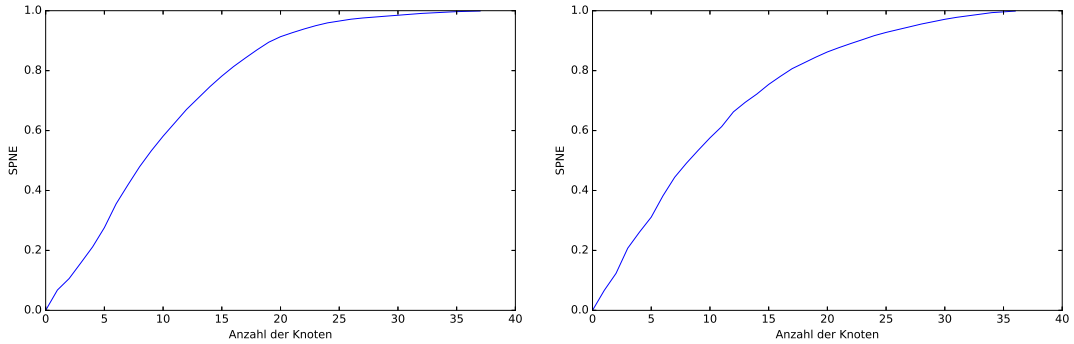
Die Zufallssuche für die feste Variante wird natürlich mit der gleichen Anzahl Knoten, also 20, initialisiert. Bei der flexiblen Anzahl Knoten wird der Initialwert auf 110 gesetzt, weil bei dieser Zahl für beide Szenarien gut eine Lösung zu finden ist und so in der Statistik besser geschaut werden kann, wie sich die Zufallssuche verbessert. Bis zum Abbruch werden zufällig 14000 Lösungen erzeugt, jeweils 80 parallel. Es werden einige Lösungen parallel erzeugt, damit die Berechnung schneller abläuft.

Die Ergebnisse werden im Vergleich zu den anderen Algorithmen in den Abbildungen 5.13 und 5.14 gezeigt.

5.3.2 Eröffnungsverfahren

Das Eröffnungsverfahren wird für beide Szenarien genau ein Mal berechnet, weil es deterministisch ist. Das Ergebnis für die Liste ist eine Platzierung mit 36 Knoten (Abbildung 5.11b). Auf den Flächenszenario des Sedanplatzes sind es 37 Knoten (Abbildung 5.11a). Die beiden Abbildungen zeigen den

Verlauf des SPNE-Wertes in Abhängigkeit der Anzahl der platzierten Knoten. Man sieht, dass am Anfang die Fitness schnell steigt und gegen Ende es noch einige Knoten benötigt, bis der gewünschte Zielwert von 1 erreicht ist. Für die Ergebnisse der festen Variante werden die Werte benutzt, die die Durchläufe bei 20 Knoten erzielt haben. Das ist ein SPNE Wert von $0,93$ für das Grid und $0,88$ für die Liste von Positionen.



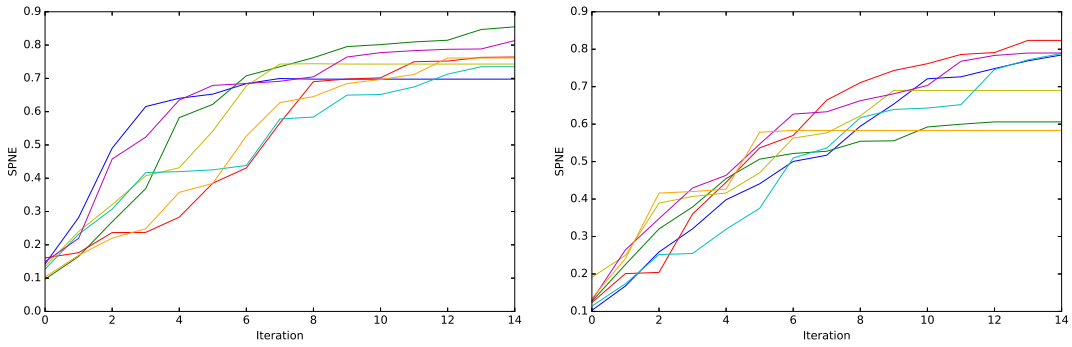
(a) Verlauf für Grid-Szenario **(b)** Verlauf für Listenszenario

Abbildung 5.11: Verlauf des SPNE-Wertes für beide Szenarien für das Eröffnungsverfahren bis der Maximalwert der SPNE Metrik von 1 erreicht ist

Die beiden Durchläufe benötigen 36038 Evaluationen für das Grid-Szenario und 27126 für das Liste-Szenario. Die beiden Berechnungen mit 20 Knoten erfordern einen Teil davon, nämlich 19650 für das Grid und 15230 für die Liste.

5.3.3 Lokale Suche

Die lokale Suche wird, wie in Abschnitt 4.4.2 erklärt, nur für die feste Variante durchgeführt. Die beiden Abbildungen 5.12b und 5.12a zeigen die Ergebnisse für mehrere Durchläufe für das Liste- bzw. Grid-Szenario mit jeweils $(771 - 20) \cdot 14 + 1 = 10515$ Evaluationen für die Liste bzw. $(992 - 20) \cdot 14 + 1 = 13899$ für das Grid. Es ist zu sehen, dass es bei der lokalen Suche so öfter vorkommt, dass ein Durchlauf in einem lokalen Optimum hängen bleibt. Das ist bei diesen Ergebnissen nicht nur nicht ganz sicher zu sagen, weil sich die Nachbarschaft in jeder Iteration ändert. Es müssten also min. 20 Iterationen passieren, bis das mit Sicherheit gesagt werden kann.



(a) Verlauf für Grid-Szenario **(b)** Verlauf für Listenszenario

Abbildung 5.12: Verlauf des SPNE-Wertes für beide Szenarien für die lokale Suche

5.3.4 Vergleich

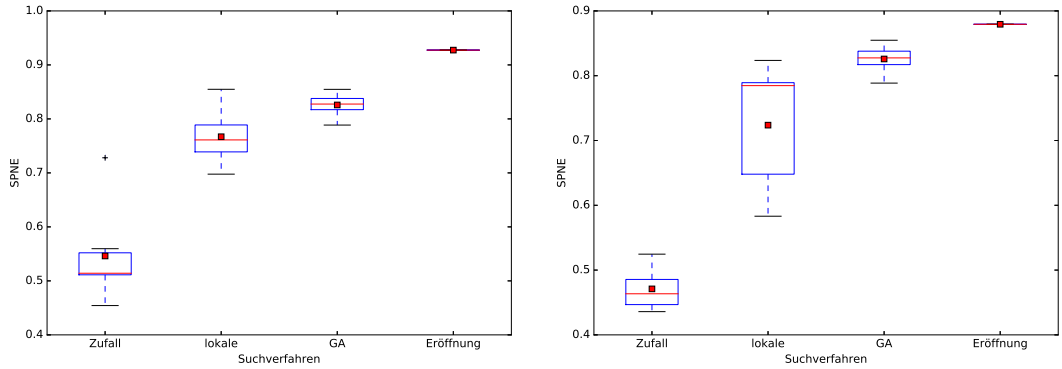
Hier werden die Algorithmen zuerst mit einer festen Anzahl von Knoten verglichen. Danach wird der Vergleich noch für die flexible Anzahl Knoten vollführt.

Vergleich feste Variante

Abbildung 5.13 zeigt den Vergleich der verschiedenen Verfahren mit fester Anzahl Knoten für beide Szenarien (Abbildung 5.13a für Grid, Abbildung 5.13b für Liste). Die Ergebnisse beider Szenarien sind recht ähnlich. Es ist klar zu sehen, dass die Zufallssuche erwartend deutlich schlechter ist. Das beste Ergebnis erzielt das Eröffnungsverfahren, danach kommt der G.A., gefolgt von der lokalen Suche. Bei diesen Ergebnissen muss berücksichtigt werden, dass die Anzahl der Evaluationen nicht ähnlich ist. Die Werte für die Evaluationsanzahlen für die Vergleichsalgorithmen wurden schon in den vorigen Abschnitten genannt. Für die Ergebnisse G.A. in diesen Durchläufen sind es um die 8000 Evaluationen für beide Szenarien. Es wurde aus zeitlichen Gründen keine weiterer Durchlauf, der ähnlicher zu dem Wert der lokalen Suche von 10515 bzw. 13899 ist, durchgeführt. Das bedeutet aber, dass der G.A. sehr wahrscheinlich besser ist, weil er bei einer geringeren Anzahl Evaluationen ein leicht besseres Ergebnis erzielt. In Abbildung 5.15 (RaLaNS Ausbreitungsmodell) sieht man Ergebnisse von Durchläufen, die eine deutlich höhere Fitness haben, dabei aber auch eine Evaluationsanzahl zwischen ca. 2000 und 50000.

Das Eröffnungsverfahren ist wie schon erwähnt das beste. Ein wichtiger Grund dafür ist, dass es auch deutlich mehr Evaluationen benötigt (Ab-

schnitt 5.3.2). Es ist für dieses Verfahren auch so, dass es auf größeren Szenarien nicht wirklich anwendbar ist, so dass es nicht richtig mit der lokalen Suche und dem G.A. konkurriert. In der bereits erwähnten Abbildung 5.15 wird gezeigt, dass bei längeren Durchläufen der genetische Algorithmus ähnliche Fitnesswerte zum Eröffnungsverfahren erzielt. Der G.A. erreicht dort Werte um $0,87$, also ähnlich zu $0,88$ beim Eröffnungsverfahren, für die Liste und Werte um $0,91$, ähnlich zum Wert von $0,93$ des Eröffnungsverfahrens, für das Grid Szenario.



(a) Vergleich für Grid-Szenario **(b)** Vergleich für Liste-Szenario

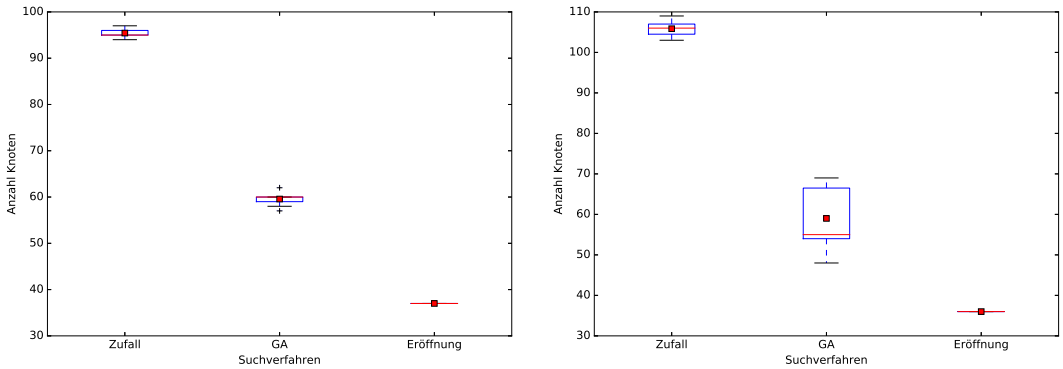
Abbildung 5.13: Vergleich der Algorithmen für beide Szenarien mit fester Anzahl Knoten

Vergleich flexible Variante

Abbildung 5.14 zeigt den Vergleich der verschiedenen Verfahren mit flexibler Anzahl Knoten für beide Szenarien. Das Eröffnungsverfahren ist hier deutlich besser als die beiden anderen. Die Zufallssuche ist selbstverständlich deutlich schlechter als der G.A.. Man kann aber aus diesen Ergebnissen nicht schließen, dass das Eröffnungsverfahren deswegen besser ist als die anderen. Ein großer Nachteil ist nämlich, dass es bei größeren Szenarien sehr lange benötigt, um berechnet zu werden, so dass es dann wie schon weiter oben erwähnt als Option ausbleibt.

Man sieht aber deutlich, dass noch ein großes Potential für den G.A. besteht, weil das Eröffnungsverfahren ca. die Hälfte an Knoten benötigt, um einen SPNE-Wert von 1 zu erhalten. Die Anzahl der Evaluationen für die Berechnungen hier sind sogar deutlich höher als bei der festen Variante. Es wird nämlich die Variante mit dem Stop-Kriterium ohne Verbesserung eingesetzt. Die Anzahl der Evaluationen liegt zwischen 15958 und 36720 für das Liste-Szenario bzw. 12943

und 24688 für die Grid-Szenario. Sie ist also sogar teilweise höher als die Anzahl der Evaluationen des Eröffnungsverfahrens. Die Gründe für das eher schlechte Abschneiden des G.A. können vielfältig sein. Ein Grund mag sein, dass für die flexible Variante die Parametrisierung nicht so detailliert durchgeführt wurde oder aber auch andere Methoden für die Rekombination, Mutation oder Initialisierung hätten verwendet werden müssen. Die Initialisierung wird wohl einen wichtigen Faktor spielen, weil sie entscheidet wie viele Knoten die Individuen der Start-Population enthalten.



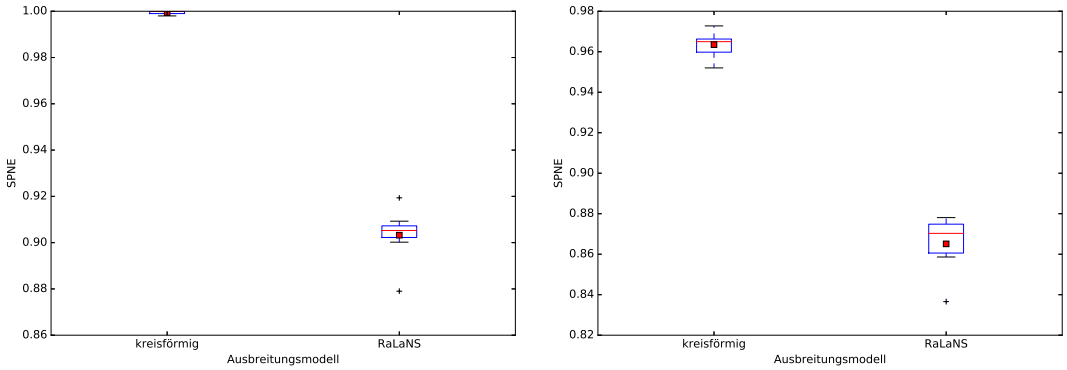
(a) Vergleich für Grid-Szenario **(b)** Vergleich für Liste-Szenario

Abbildung 5.14: Vergleich der Algorithmen für beide Szenarien mit flexibler Anzahl Knoten

5.4 RaLaNS vs. kreisförmig

Hier wird untersucht welchen Einfluss das Ausbreitungsmodell auf das Ergebnis hat. Wenn die Fitnesswerte der Platzierungen für die verschiedenen Ausbreitungsmodelle unterschiedlich sind, dann heißt das, dass die Platzierungen des Netzwerkes auf jeden Fall unterschiedlich sind. Dies bestätigt Abbildung 5.15, in welcher der G.A. einerseits auf Grundlage einer kreisförmigen Ausbreitung und andererseits auf Basis von RaLaNS mit fester Anzahl Knoten durchgeführt wird. Es zeigt sich, dass für beide Szenarien die Fitness bei RaLaNS für jeden Durchlauf schlechter ist als bei der kreisförmigen Signalausbreitung. Für das Grid-Szenario liegt der Wert der festen Signalausbreitung bei 1 oder fast 1 für das Grid-Szenario bzw. ca. $0,96$ für das Liste-Szenario. Die Werte mit RaLaNS sind geringer, nämlich um die $0,91$ bzw. $0,87$. Daraus kann man folgern, dass die Wahl des Ausbreitungsmodells einen großen Einfluss auf die Qualität einer Platzierung hat. Dies bestätigt die Erkenntnis aus [11]. Dort ist der Unterschied

zwischen den Modellen jedoch deutlich größer als hier. Der Grund dafür wird hier aus zeitlichen Gründen nicht weiter untersucht. Ein simpler Grund könnte sein, dass durch die ohnehin hohen Werte bei der RaLaNS Ausbreitung die Werte bei der kreisförmigen Ausbreitung nicht viel höher sein können.



(a) Vergleich für Grid-Szenario **(b)** Vergleich für Liste-Szenario

Abbildung 5.15: Vergleich der Ergebnisse der kreisförmigen Signalausbreitung mit der Ausbreitung nach RaLaNS für beide Szenarien für eine feste Anzahl Knoten

5.5 Zusammenfassung

In diesem Kapitel wurde die Parametrisierung des G.A. für die flexible Variante und die fixe Variante durchgeführt. Dabei wurde die Parametrisierung hauptsächlich auf die fixe Variante konzentriert. Das lag hauptsächlich an zeitlichen Gründen und dem Umfang den die Parametrisierung ausgemacht hat.

In der Auswertung mit den anderen Algorithmen wurde festgestellt, dass für die feste Anzahl von Knoten der G.A. im Vergleich zu den anderen Verfahren gute Ergebnisse liefert. Das konnte für die flexible Anzahl von Knoten so nicht gezeigt werden. Da ist das Ergebnis nämlich deutlich von einem lokalen Optimum, das vom Eröffnungsverfahren erreicht wird, entfernt.

Beim Vergleich der Ausbreitungsmodelle konnte gezeigt werden, dass die Ergebnisse sich deutlich unterscheiden. Das bekräftigt die Relevanz von RaLaNS als realistischeres Ausbreitungsmodell für die Berechnung beim Node-Placement.

6 Zusammenfassung & Ausblick

In dieser Bachelorarbeit wurde ein genetischer Algorithmus für ein Node-Placement Problem entwickelt. Dieser Algorithmus kann basierend auf einem kreisförmigen Signalausbreitungsmodell und dem Ausbreitungsmodell nach RaLaNS eine Platzierung von Knoten eines Multi-Hop Netzwerkes auf einer Karte berechnen. Die Karte wird dabei als Grid oder Liste von Positionen umgesetzt. Für das RaLaNS Ausbreitungsmodell stehen für diese Positionen Signalstärken zwischen den Positionen zur Verfügung, die basierend auf OSM Karten berechnet werden.

Es wurde ein genetischer Algorithmus als Ansatz zur Lösung für dieses Problem auf Grund von dessen Komplexität gewählt. Eine exakte Berechnung realistischer Szenarien ist praktisch unmöglich. Ein genetischer Algorithmus hat den Vorteil, dass er basierend auf Heuristiken und dem Zufall eine Lösung berechnet und so nicht so viele Lösungen erzeugen muss, um eine einigermaßen gute Lösungen zu generieren.

Bei der Umsetzung des G.A. wurden zwei Varianten implementiert. Es wurde einerseits eine Variante zur Maximierung der Fitness des Netzwerkes bei fester Anzahl Knoten und andererseits eine Minimierung der Anzahl der Knoten unter Einhaltung einer gewissen Fitness des Netzwerkes umgesetzt. Diese Varianten ergeben sich durch eine passende Kombination der Mutations- und Rekombinationsverfahren und einer Festlegung der Gewichte der Fitnessfunktion. Für die Umsetzung der Varianten wurden mehrere Methoden für die einzelnen Schritte des G.A. getestet, z.B. die verschiedenen Selektions- und Ersetzungsverfahren. Dann wurde noch teilweise eine Parametrisierung des G.A. umgesetzt. Dies konnte aus zeitlichen Gründen nicht komplett umgesetzt werden.

Passend zur Implementation des G.A. wurde eine Fitnessberechnung abgewandelt von SPNE auf einem Graphen umgesetzt. Diese wurde benutzt, um die Güte der Platzierungen zu bewerten.

Zusätzlich zur Implementation des G.A. wurden Algorithmen zum Vergleich erarbeitet und implementiert. Das waren die Zufallssuche, die lokale Suche und ein selbst erarbeitetes Eröffnungsverfahren. Diese wurden genutzt, um die

Effektivität des G.A. zu untersuchen. Dabei hat sich ergeben, dass der G.A. deutlich besser ist als die Zufallssuche. Er ist bei den untersuchten Szenarien besser als die lokale Suche, aber fast immer schlechter als das Eröffnungsverfahren. Dies bedeutet, dass er basierend auf dieser Evaluation eine gute Wahl als Verfahren zur Lösung dieses Node-Placement Problem fűr die Optimierung mit einer festen Anzahl Knoten scheint, weil das Eröffnungsverfahren nicht auf deutlich größere Szenarien anwendbar ist. Für die Variante mit flexibler Anzahl Knoten war das Ergebnis nicht so überzeugend wie für die feste. Dort könnte eine Verbesserung dadurch erreicht werden, dass die Parametrisierung intensiver durchgeführt wird. Möglichwerweise könnten auch andere Methoden bei der Rekombination, Mutation oder Initialisierung zu einer Verbesserung beitragen.

Abgesehen von einer Verbesserung der Parametrisierung können andere weiterführende Arbeiten zu dieser Bachelorarbeit aus einer Evaluation bestehen, die generell auf deutlich mehr Berechnungen beruht, so dass die Schlussfolgerungen statistisch stärker belegt sind.

Es können zudem mehr Szenarien, vor allem größere untersucht werden, um genauere und bessere Ergebnisse für die Effektivität der Algorithmen zu erhalten, weil sich dann erst klar zeigt, welche Algorithmen praktisch eingestzt werden können. Hier wurden nur Vermutungen geäußert, weil die Szenarien beide recht überschaubar waren.

Ein weiterer guter Kandidat als Vergleichsalgorithmus ist das in Kapitel 3.2 beschriebene Vorgehen, welches die Knoten auf die Kreuzungen platziert. Der G.A. sollte nämliche bessere Ergebnisse erzielen als das Verfahren, um eine legitime Alternative zu sein, weil die Berechnung mit dem G.A. wahrscheinlich länger dauert.

Diese Arbeit könnte ansonsten durch eine andere Form der Fitnessberechnung erweitert werden. In den Grundlagen wurde der Netzwerksimulator ns-3 erwähnt. Dieser kann statt der Fitnessberechnung mit Hilfe eines Graphen genutzt werden, um die Qualität der Platzierung zu messen. Dadurch wird die Berechnung noch realistischer, aber auch berechnungsaufwendiger. Durch diese Erweiterung würde die Bedeutung des Berechnungsaufwandes dieses Node-Placement noch deutlicher, weil die Evaluation einer Lösung noch länger dauern würde.

Andere weiterführende Arbeiten können in der Veränderung des G.A. bestehen. Es kann z. B. verändert werden, dass die Größe der Population konstant sein muss oder es können mehere Populationen benutzt werden. Ebenfalls kann die andere Form der Kodierung (Abschnitt 4.3.2) ausprobiert werden oder die Implementation in einer anderen Sprache als Python stattfinden, um die Performance zu erhöhen.

Statt einem genetischen Algorithmus ist es auch möglich andere Opti-

mierungsverfahren zu verwenden, hier wurde selbst eine einfache lokale Suche zum Vergleich genutzt. Weiterführende Arbeiten können sich mit einer detaillierteren Implementation von lokalen Suchverfahren wie der Tabusuche oder auch Simulated Annealing beschäftigen.

Abbildungsverzeichnis

[1] www.josm.openstreetmap.de

Liste der Algorithmen

Literaturverzeichnis

[1] M. Rouse, "star network." `http://searchnetworking.techtarget.com/definition/star-network`, 2006. [Online; zuletzt aufgerufen 8.8.16].

[2] N. Aschenbruck, "Einführung und Überblick zu Rechnernetzen (Kapitel 4)." Folien zur Vorlesung 'Betriebssysteme und Rechnernetze', 2014.

[3] "Ieee 100 the authoritative dictionary of ieee standards terms seventh edition," 2000.

[4] Wikipedia, "Wireless ad hoc network — wikipedia, the free encyclopedia." `https://en.wikipedia.org/wiki/Wireless_ad_hoc_network`, 2016. [Online; zuletzt aufgerufen 10.8.16].

[5] R. Pries, B. Staehle, D. Staehle, and V. Wendel, "Genetic algorithms for wireless mesh network planning," in *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, pp. 226–234, ACM, 2010.

[6] M. Fiore, J. Harri, F. Filali, and C. Bonnet, "Vehicular mobility simulation for vanets," in *Simulation Symposium, 2007. ANSS'07. 40th Annual*, pp. 301–309, IEEE, 2007.

[7] M. Younis and K. Akkaya, "Strategies and techniques for node placement in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 6, no. 4, pp. 621–655, 2008.

[8] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking*, pp. 85–97, ACM, 1998.

[9] H. T. Friis, "A note on a simple transmission formula," *Proceedings of the IRE*, vol. 34, no. 5, pp. 254–256, 1946.

[10] R. T. Hänel, "Simulative evaluation of multi hop node placement strategies," Master's thesis, University of Osnabrück, Osnabrück, 2014.

[11] T. Hänel, A. Bothe, and N. Aschenbruck, "RaLaNS - a ray launching based propagation loss model for ns-3," in *Networked Systems (NetSys), 2015 International Conference and Workshops on*, pp. 1–7, March 2015.

[12] O. Vornberger, "Grundlagen ray tracing." `http://media2mult.uni-osnabrueck.de/pmwiki/fields/cg/index.php?n=Main.Grundlagen2`, 2016. [Online; zuletzt aufgerufen 8.8.16].

[13] T. Vanhatupa, M. Hannikainen, and T. D. Hamalainen, "Genetic algorithm to optimize node placement and configuration for wlan planning," in *2007 4th International Symposium on Wireless Communication Systems*, pp. 612–616, IEEE, 2007.

[14] J. Hopcroft and R. Tarjan, "Algorithm 447: efficient algorithms for graph manipulation," *Communications of the ACM*, vol. 16, no. 6, pp. 372–378, 1973.

[15] X. Ta, G. Mao, and B. D. Anderson, "On the properties of giant component in wireless multi-hop networks," in *INFOCOM 2009, IEEE*, pp. 2556–2560, IEEE, 2009.

[16] "What is ns-3." `https://www.nsnam.org/overview/what-is-ns-3/`, 2015. [Online; zuletzt aufgerufen 13.8.16].

[17] C. Lochert, B. Scheuermann, C. Wewetzer, A. Luebke, and M. Mauve, "Data aggregation and roadside unit placement for a VANET traffic information system," in *Proceedings of the Fifth ACM International Workshop on VehiculAr Inter-NETworking*, VANET '08, (New York, NY, USA), pp. 58–65, ACM, 2008.

[18] S. N. Sivanandam and S. N. Deepa, *Introduction to genetic algorithms*. Springer, 2008.

[19] F.-A. Fortin, F.-M. D. Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "Deap: Evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, no. Jul, pp. 2171–2175, 2012.

[20] F. Xhafa, C. Sánchez, and L. Barolli, "Genetic algorithms for efficient placement of router nodes in wireless mesh networks," in *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pp. 465–472, IEEE, 2010.

[21] S. Knust, "Einführung in die Kombinatorische Optimierung." Skript zur Vorlesung 'Einführung in die Kombinatorische Optimierung', 2013.

[22] Z. Michalewicz and D. B. Fogel, *How to solve it: modern heuristics*. Springer Science & Business Media, 2013.

[23] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1982.

[24] J. K. Lenstra, *Local search in combinatorial optimization*. Princeton University Press, 1997.

[25] J. A. Bondy and U. S. R. Murty, *Graph theory with applications*, vol. 290. Citeseer, 1976.

[26] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[27] "Hall-of-fame." `http://deap.gel.ulaval.ca/doc/dev/api/tools.html#deap.tools.HallOfFame`, 2014. [Online; zuletzt aufgerufen 12.8.16].

[28] "Two-point crossover." `http://deap.gel.ulaval.ca/doc/dev/api/tools.html#deap.tools.cxTwoPoint`, 2014. [Online; zuletzt aufgerufen 12.8.16].

[29] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097–1100, 1997.

[30] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on evolutionary computation*, vol. 3, no. 2, pp. 124–141, 1999.

Akronyme

G.A. genetischer Algorithmus

STA Station

AP Access Point

LAN Local Area Network

WLAN Wireless Local Area Network

DEAP Distributed Evolutionary Algorithm Package

SPNE Scanning Probe Network Efficiency

MANET Mobile Ad Hoc Network

WANET Wireless Ad Hoc Network

VANET Vehicular Ad Hoc Network

OSM Open Street Maps

RaLaNS Ray Launching Based Progagation Loss Model for ns-3

WMN Wireless Mesh Network

WSN Wireless Sensor Network

SU Supporting Unit

ns-3 Network Simulator 3

Erklärung der Selbstständigkeit

Ich versichere, dass ich die eingereichte Bachelor-Arbeit selbstständig und ohne unerlaubte Hilfe verfasst habe. Anderer als der von mir angegebenen Hilfsmittel und Schriften habe ich mich nicht bedient. Alle wörtlich oder sinngemäß den Schriften anderer Autoren entnommenen Stellen habe ich kenntlich gemacht.

Osnabrück, 13. August 2016