Rothlauf, F. (2004): OptiNet: Ein Optimierungswerkzeug für baumförmige Netzwerkprobleme. In: Leena Suhl und Stefan Voß (Hrsg.): Quantitative Methoden in ERP und SCM (Teilkonferenz auf der MKWI 2004). S. 15-34, Universität Paderborn.

OptiNet: Ein Optimierungswerkzeug für baumförmige Netzwerkprobleme

Franz Rothlauf

Lehrstuhl für Wirtschaftsinformatik I

Universität Mannheim, Schloss, 68131 Mannheim

rothlauf@uni-mannheim.de

Abstract

OptiNet ist ein objektorientiertes Optimierungswerkzeug für die Lösung von baumförmigen Netzwerkproblemen mit Hilfe von naturanalogen Optimierungsverfahren. In OptiNet können verschiedene naturanaloge Optimierungsverfahren mit unterschiedlichen Problemrepräsentationen (charakteristische Vektoren, Network Random Keys, Link-and-Node-biased Kodierung und Prüfernummern) kombiniert werden. Der vorliegende Beitrag stellt OptiNet vor und demonstriert an drei Szenarien wie es für das Lösen von Kommunikationsnetzwerkproblemen eingesetzt werden kann. Es wird beispielhaft für Genetische Algorithmen untersucht, wie die Problemrepräsentation die Lösungsqualität der Optimierungsverfahren beeinflusst. Hierbei zeigen sich große Unterschiede in Abhängigkeit der verwendeten Problemrepräsentation.

1 Einleitung

In Enterprise Ressource Planning (ERP) Systemen vollzog sich in den letzten Jahren ein Paradigmenwandel. Standen früher die bloße Planung von Prozessen und das Finden von zulässigen Lösungen für komplexe Planungsprobleme im Vordergrund, so liegt mittlerweile das Hauptaugenmerk auf dem Ermitteln von optimalen Lösungen. Planungsverfahren, welche eine zulässige Lösung (z.B. einen gültigen Maschinenbelegungsplan) ermitteln, werden zunehmend durch Optimierungsverfahren abgelöst, welche versuchen, die optimale Lösung (z.B. den Maschinenbelegungsplan mit minimalen Durchlaufzeiten) für betriebswirtschaftliche Planungs- oder Optimierungsprobleme zu finden. Als Methoden haben sich neben klassischen Ansätzen (z.B. lineare oder ganzzahlige Programmierung, Branch&Bound Ansätze, etc.) aus dem Bereich des Operations Research auch zunehmend Heuristiken und Metaheuristiken für komplexere Optimierungsprobleme etabliert. Derartige Verfahren können durch ein besseres Methodenverständnis und einer konsequenten Weiterentwicklung bestehender Verfahren, sowie der in den letzten Jahrzehnten stark angestiegenen Rechnerleistungen, respektable Erfolge erzielen und werden mittlerweile auch standardmäßig in Advanced Planning Systemen wie z.B. von SAP (APO-Suite) oder i2 (logistics optimization) eingesetzt.

Im Rahmen der Optimierung von Supply Chains ist es für Unternehmen wichtig, ihre Prozessstrukturen sowohl innerhalb eines einzelnen Unternehmens, als auch zwischen einzelnen Unternehmen in einer Supply Chain optimal aufeinander abzustimmen. Insbesondere die Optimierung des Aufbaus und der Struktur von Logistiknetzwerken, wie z.B. Kommunikations-, Transport-,

Entsorgungs-, Prozess- oder auch Zuliefernetzwerken, ist von hoher Bedeutung für den Aufbau von effizienten Supply Chains. Hierbei zeigt sich allerdings, dass die zugrundeliegenden Netzwerkprobleme oft NP-vollständig sind und keine effizienten Lösungs- oder Approximationsverfahren dafür existieren. Als Ausweg bietet sich der Einsatz von Metaheuristiken an. Für Netzwerkprobleme wurden in den letzten Jahren unterschiedliche Ansätze vorgestellt, welche eine effiziente Lösung von kleinen bis mittleren Probleminstanzen ermöglichen.

Der vorliegende Beitrag verfolgt zwei Ziele. Zum einen soll das Optimierungswerkzeug OptiNet vorgestellt werden. Mit diesem Werkzeug können im Rahmen der Optimierung von Supply Chains optimale Strukturen (Lösungen) für baumförmige Logistiknetzwerkprobleme gefunden werden. Im Rahmen von OptiNet können Genetische Algorithmen, Bayesian Optimization Algorithmen, Evolutionsstrategien, Cooperative Simulated Annealing und Simulated Annealing eingesetzt werden. Mit diesen Verfahren können das allgemeine Netzwerkproblem (network design problem (NDP)) und das optimale baumförmige Kommunikationsnetzwerkdesignproblem (minimum communication spanning tree (MCST) Problem) gelöst werden. Bei diesen zwei Klassen von Problemen geht es um das Ermitteln eines optimalen Graphen (NDP) bzw. Baums (MCST), welcher die vorgegebenen Transport- oder Kommunikationsanforderungen erfüllt und minimale Kosten in Bezug auf einer vorgegebenen Zielfunktion aufweist. Des Weiteren soll im zweiten Teil des Beitrags OptiNet auf reale Beispielszenarien angewendet werden und der Einfluss unterschiedlicher Repräsentationen untersucht werden. Beim Einsatz von Metaheuristiken müssen mögliche Lösungen eines Optimierungsproblems (z.B. Graphen oder Bäume) mit Hilfe einer Repräsentation kodiert werden. Der Beitrag stellt die in OptiNet enthaltenen Repräsentationen für baumförmige Netzwerkstrukturen (charakteristische Vektoren, Network Random Keys, Link-and-Node-biased Kodierung und Prüfernummern) vor und untersucht deren Einfluss auf die Lösungsqualität der oben angeführten Optimierungsverfahren. Die empirischen Ergebnisse zeigen den großen Einfluss unterschiedlicher Repräsentationen auf.

Der Beitrag ist wie folgt aufgebaut. Im folgenden Abschnitt wird eine kurze Einführung in Logistik- bzw. Kommunikationsnetzwerkprobleme gegeben und das allgemeine Netzwerkproblem (NDP) sowie das optimale baumförmige Kommunikationsnetzwerkdesignproblem (MCST) näher betrachtet. Anschließend wird in Abschnitt drei OptiNet vorgestellt. Es wird auf den prinzipiellen Aufbau des Optimierungswerkzeugs eingegangen und die im Werkzeug realisierten Repräsentationen und Optimierungsmethoden für das MCST Problem näher erläutert. In Abschnitt vier wird OptiNet für konkrete Szenarien eingesetzt und ein Vergleich der Problemrepräsentationen durchgeführt. Der Beitrag endet mit einer kurzen Zusammenfassung.

2 Netzwerkprobleme

Beim Aufbau von Logistik- bzw. Kommunikationsnetzwerken[1] stehen Unternehmen vor dem Problem, die Struktur eines Netzes zu finden, welches alle Kommunikations- oder Transportanforderungen erfüllt und minimale Transport- oder Übertragungskosten aufweist.

[1] Im Folgenden wird nicht explizit zwischen Logistik- und Kommunikationsproblemen unterschieden. Beide Probleme lassen sich durch das gleiche Modell beschreiben und unterscheiden sich nur bezüglich des Charakters der über dem Netz transportierten Güter (Waren versus Informationen).

Dieses Problem kann durch die Verwendung von Graphen formalisiert werden. $G = (V, E)$ sei hierbei ein ungerichteter Graph mit $n = |V|$ Knoten und $m = |E|$ Kanten. Weiterhin existieren Kommunikations- oder Transportanforderungen zwischen den n verschiedenen Knoten des Graphen. Diese Anforderungen können durch eine $n \times n$ Bedarfsmatrix R beschrieben werden. Die einzelnen Elemente r_{ij} der Matrix geben den Kommunikationsbedarf zwischen Knoten i und j an. Weiterhin legt eine $n \times n$ Distanzmatrix W die Kantengewichte w_{ij} zwischen den einzelnen Knoten fest. Die Kantengewichte beschreiben die Kosten, die bei der Nutzung der entsprechenden Kante auftreten. Oft orientieren sich die Kantengewichte w_{ij} an dem Abstand (euklidischer Distanz) zwischen zwei Knoten i und j.

2.1 Das Network Design Problem

Beim network design problem (NDP) (Johnson, et al. (1978)) muss ein Subgraph G' gefunden werden, welcher alle n Knoten miteinander verbindet und minimale Kosten aufweist:

$$\min_{G'} \sum_{s,d} \left(\sum_{i,j \in SP_{sd}(G')} w_{sd} r_{ij} \right)$$

G' ist hierbei definiert als die Menge aller Subgraphen von G und $SP_{sd}(G')$ ist der kürzeste Weg von Knoten s nach Knoten d in dem Subgraphen G'. In der Arbeit von Johnson et al. (1978) wurde gezeigt, dass das KNAPSACK Problem in das NDP überführbar ist und damit das NDP NP-vollständig ist.

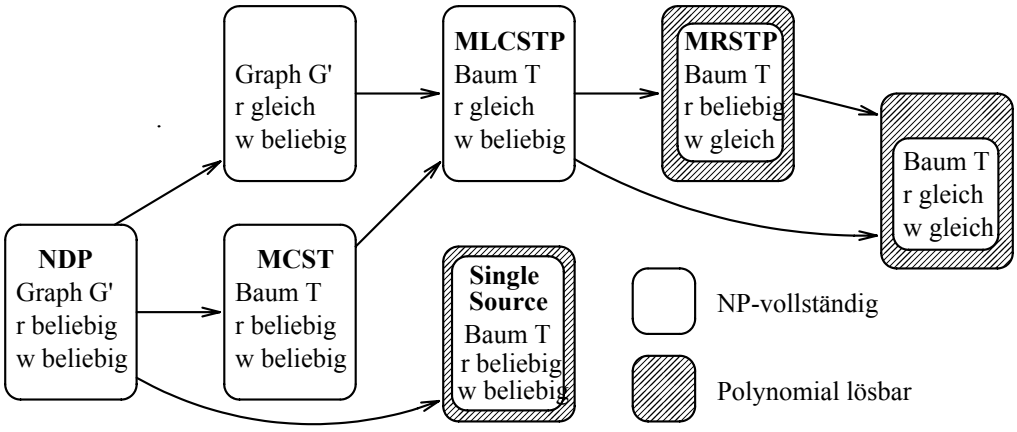


Abbildung 1: Komplexitätsklassen für aus dem NDP abgeleitete Probleme

Abbildung 1 gibt eine Übersicht über eine Reihe von weiteren Netzwerkproblemen, welche aus dem NDP abgeleitet werden können. Im NDP muss ein Subgraph G' mit minimalen Kosten und beliebig vorgegebenen Bedarfen r_{ij} und Kantengewichten w_{ij} gefunden werden. Aus dem NDP lässt sich das minimal communication spanning tree (MCST) Problem ableiten, bei welchem die

gesuchte Netzstruktur ein Baum T sein muss (genauere Beschreibung des MCST wird in Abschnitt 2.2 gegeben). Weiterhin lassen sich aus dem NDP Probleme ableiten, bei welchem die Elemente r_{ij} der Bedarfsmatrix gleich sind und nur die Kantengewichte w_{ij} unterschiedlich sind. Es wurde in Johnson et al. (1978) gezeigt, dass auch diese Probleme NP-vollständig sind. Weiterhin wurde in Hu (1974) neben dem MCST Problem auch das minimum requirement spanning tree (MRST) Problem eingeführt. Bei diesem Problem sind alle Kantengewichte w_{ij} gleich und das Problem kann mit polynomialem Aufwand mit Hilfe des Gomory-Hu Algorithmus (Gomory und Hu (1961)) gelöst werden. Das Problem des Findens eines optimalen Baumes T bei identischen Bedarfen und Kantengewichten ist trivial, da jeder beliebige Baum die gleichen (minimalen) Kosten aufweist. Auch das Single-Source Problem, bei dem ein Baum mit minimalen Kosten bei beliebigen Bedarfen und Kantengewichten zu finden ist, ist polynomial durch Standardalgorithmen (z.B. Kruskal oder Dijkstra) lösbar. Bei diesem Problem hat nur ein einziger Knoten q Bedarfe zu allen anderen Knoten ($r_{ij} = 0$ für $i \neq q$).

2.2 Das Minimal Communication Spanning Tree Problem

Wie im vorherigen Abschnitt aufgezeigt, kann das minimum communication spanning tree (MCST) Problem aus dem NDP abgeleitet werden. Das Problem wurde von Hu (1974) eingeführt und ist NP-vollständig (Johnson et al. (1978)). Beim MCST Problem wird im Gegensatz zum NDP kein beliebiger Subgraph G' gesucht, sondern die Struktur des gesuchten Graphen soll ein spannender Baum sein. Ein Baum $T = (V, F)$, wobei $F \subseteq E$ und $|F| = |V| - 1$, wird als spannender Baum von G bezeichnet, wenn er alle Knoten verbindet. Ein Baum verbindet n Knoten durch genau $n-1$ Kanten und es existieren keine Zyklen. Die Anzahl der spannenden Bäume mit n Knoten wurde in Cayley (1889) als n^{n-2} bestimmt. Beim MCST Problem muss der Baum mit den niedrigsten Kantenkosten gefunden werden:

$$\min_{T} \sum_{s,d} \left(\sum_{i,j \in P_{sd}(T)} w_{sd} r_{ij} \right)$$

P_{sd} ist hierbei der (eindeutige) Pfad im Baum T von Knoten s nach Knoten d.

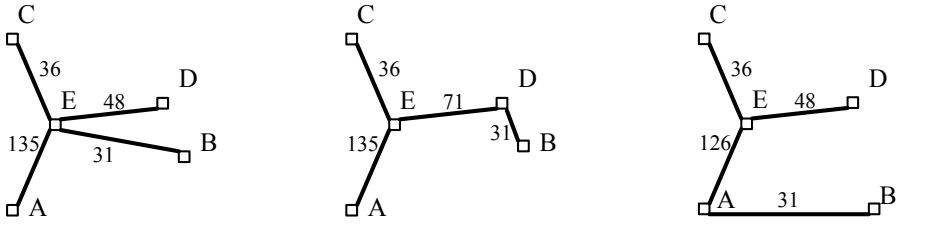
Im Folgenden soll zur Verdeutlichung ein kurzes Beispiel für das MCST Problem gegeben werden. Ein Unternehmen habe fünf verschiedene Niederlassungen mit den Koordinaten $(x_A = 0, y_A = 0)$, $(x_B = 10, y_B = 3)$, $(x_C = 0, y_C = 10)$, $(x_D = 8, y_D = 6)$ und $(x_E = 3, y_E = 5)$, welche durch ein baumförmiges Kommunikationsnetz miteinander verbunden werden sollen. Tabelle 1 gibt die Bedarfe r_{ij} zwischen den fünf Standorten an. Es wird angenommen, dass zwischen der Zentrale A und den einzelnen Niederlassungen B, C, D und E ein wesentlich höherer Kommunikationsbedarf als zwischen den einzelnen Niederlassungen besteht. Es soll eine baumförmige Netzwerkstruktur gefunden werden, welche alle Kommunika-

r_{ij}	A	B	C	D	E
A	-	20	35	40	50
B	-	-	4	4	3
C	-	-	-	2	5
D	-	-	-	-	2

Tabelle 1: Bedarfe r_{ij}

tionsanforderungen bei minimalen Kosten erfüllt. Wie für das MCST Problem, können die Kosten einer Verbindung als $w_{ij} * b_{ij}$ berechnet werden. w_{ij} ist in diesem Beispiel die euklidische Entfernung zwischen zwei Niederlassungen i und j und b_{ij} der gesamte Verkehr, welcher über die Kante von Knoten i nach Knoten j fließt.

Abbildung 2(a) zeigt die optimale Lösung inklusive des resultierenden Verkehrs b_{ij} über die einzelnen Kanten. Die Gesamtkosten des Baums belaufen sich auf 1467,5.



(a) $(x_B = 10, y_B = 3)$ *(b)* $(x_B = 10, y_B = 4)$ *(c)* $(x_B = 10, y_B = 0)$

Abbildung 2: Optimale Lösungen für das Beispielproblem

Nun sollen die Auswirkungen von kleinen Änderungen aufgezeigt werden. Abbildung 2(b), bzw. Abbildung 2(c) zeigen die optimale Lösung des Problems, wenn die Koordinaten von Niederlassung B auf $(x_B = 10, y_B = 4)$, bzw. $(x_B = 10, y_B = 0)$ geändert werden. Es ist deutlich zu erkennen, dass sich sowohl die Struktur der optimalen Lösung als auch der Verkehr über die einzelnen Leitungen ändert. Weiterhin ergeben sich als minimale Kosten 1466,8 $(x_B = 10, y_B = 4)$ bzw. 1499,37 $(x_B = 10, y_B = 0)$.

2.3 Praxistauglichkeit

Sowohl beim NDP als auch beim MCST Problem hängen die Kosten einer Kante linear von der über die Kante transportierten Menge an Daten b_{ij} ab. Der gesamte Verkehr b_{ij} über eine Kante von Knoten i nach j wird mit dem Kantengewicht w_{ij} gewichtet, womit sich als Gesamtkosten einer Kante $w_{ij} * b_{ij}$ ergibt. Bei der Verwendung eines derartigen Modells ist allerdings kritisch zu hinterfragen, ob ein linearer Zusammenhang eine praxisnahe Bewertung der beim Aufbau eines Kommunikationsnetzwerks anfallenden Kosten erlaubt.

In der Praxis hängen die Kosten einer Kante (Leitung) nichtlinear sowohl von der Länge der Kante als auch von der Menge der über die Kante transportierten Daten ab. Mit steigender Kapazität einer Leitung werden die Kosten pro Einheit Übertragungskapazität abnehmen (degressiver Kostenverlauf). Weiterhin muss beim Aufbau eines Kommunikationsnetzes berücksichtigt werden, dass die Kapazität einer Leitung nicht beliebig festgelegt werden kann, sondern im Regelfall nur diskrete Kapazitäten zur Verfügung stehen[2]. Netzwerkdesigner haben zum Beispiel beim Einsatz von Ethernet für den Aufbau eines lokalen Netzwerkes nur die Auswahl zwi-

[2] Dies gilt analog für Logistiknetzwerke.

schen Übertragungskapazitäten von 10 MBit/s (Ethernet), 100 MBit/s (Fast Ethernet) oder 1 GBit/s (Gigabit Ethernet). Zusammenfassend ist also festzustellen, dass im Gegensatz zu den Annahmen im NDP und MCST Problem

- keine beliebigen Abstufungen der Leitungskapazitäten möglich sind und auch
- kein linearer Zusammenhang zwischen den anfallenden Kosten und den über eine Leitung fließenden Verkehr existiert.

Darüber hinaus fallen beim Netzwerkdesign im Regelfall fixe Kosten für die Bereitstellung einer Leitung an, welche oft von der einzurichtenden Leitungskapazität abhängen. Diese Kosten hängen nicht von dem über die Leitung transportierten Verkehr ab, sondern von deren Länge und Kapazität. Aufgrund dieser zusätzlichen Restriktionen aus der Praxis, muss für das MCST Problem die Kostenbewertung der einzelnen Kanten angepasst werden:

$$\min_{T} \sum_{s,d} \left(\sum_{i,j \in SP_{sd}(T)} f(w_{sd}, r_{ij}) \right)$$

Es existiert kein linearer Zusammenhang zwischen dem über eine Leitung transportierten Verkehr b_{ij} und den anfallenden Kosten. Die allgemeine Definition von $f(w_{sd}, r_{ij})$ ermöglicht die Beschreibung von Kantenkosten, welche sowohl Bereitstellungskosten für eine Leitung beinhalten, als auch beliebig von dem über eine Kante transportierten Verkehr abhängen können.

3 OptiNet

OptiNet ist ein Optimierungswerkzeug für Logistik- und Kommunikationsnetzwerkprobleme. Es wurde in den vergangenen Jahren an der Universität Bayreuth (Lehrstuhl für Wirtschaftsinformatik), bzw. an der Universität Mannheim (Lehrstuhl Wirtschaftsinformatik I) kontinuierlich weiterentwickelt. Es ermöglicht das Lösen von NDP und OCST Problemen mit Hilfe von Metaheuristiken, insbesondere naturinspirierten Optimierungsansätzen. Mit Hilfe von OptiNet können Netzwerkprobleme modelliert, dargestellt, analysiert, gespeichert und optimiert werden. Als Optimierungsverfahren sind gegenwärtig Genetische Algorithmen, Bayesian Optimization Algorithmen, Evolutionsstrategien, Cooperative Simulated Annealing und Simulated Annealing implementiert. Diese Metaheuristiken können jeweils in Kombination mit unterschiedlichen Problemrepräsentationen (charakteristische Vektoren, Network Random Keys, Link-and-Node-biased Kodierung und Prüfernummern) zur Lösung des MCST Problems verwendet werden.[3] Abbildung 3 zeigt einen Screenshot von OptiNet. Es sind drei unterschiedliche Fenster sichtbar. Das rechte Fenster zeigt jeweils die Struktur eines Netzwerkes, links können Optimierungsprobleme definiert werden als auch die Eigenschaften der unterschiedlichen Lösungsverfahren eingestellt werden und links unten ist ein Fortschrittsfenster sichtbar, welches Kennzahlen über den aktuellen Verlauf der Optimierung liefert. Zusätzlich kann eine Vielzahl von statistischen Informationen zum Verlauf der Optimierung protokolliert und statistisch ausgewertet werden.

[3] Für das NDP Problem können nur charakteristische Vektoren als Problemrepräsentation eingesetzt werden, da Graphen (und keine Bäume) optimiert werden. Der vorliegende Beitrag fokussiert sich auf die Verwendung von OptiNet ausschließlich für baumförmige Probleme.

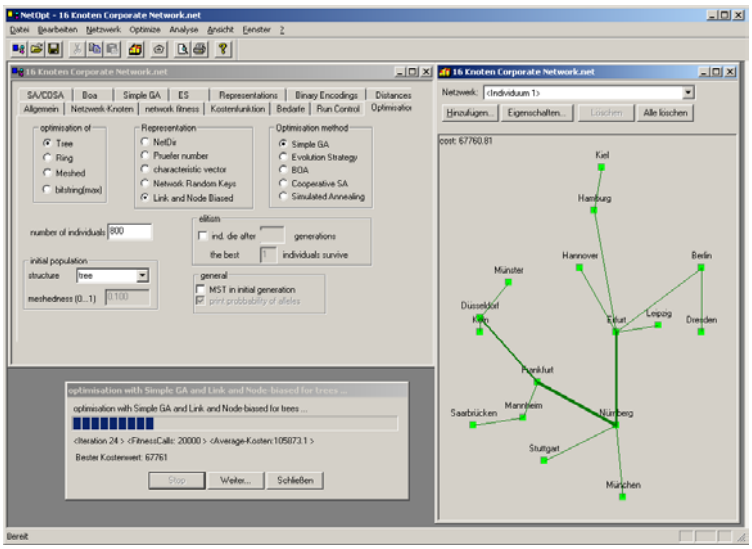


Abbildung 3: Screenshot OptiNet

3.1 Struktur

OptiNet wurde komplett objektorientiert in C++ entwickelt und hat im Moment einen Umfang von mehreren 10.000 Zeilen an Code. Beim Aufbau lässt sich unterscheiden zwischen plattformunabhängigem Code, welcher bei der Implementierung der Optimierungsmethoden und Problemrepräsentationen eingesetzt wurde und den grafischen Schnittstellen, welche mit Hilfe der Microsoft MFC Bibliotheken realisiert wurde.

Die Wahl von Visual C++ als Implementierungswerkzeug wurde ursprünglich aufgrund der Mächtigkeit der zur Verfügung stehenden grafischen Bibliotheken getroffen. Aufgrund der bisherigen Erfahrungen mit OptiNet im praktischen Einsatz erscheint aber rückblickend aus Gründen einer höheren Plattformunabhängigkeit und flexibleren Anpassungsmöglichkeiten die Verwendung von Standard C++ auf der Basis von Unix, resp. Linux vorteilhafter.

3.2 Kostenfunktionen

Wie in Abschnitt 2.3 dargestellt, erlaubt die ursprüngliche Modellierung des MCST Problems nur eine eingeschränkte Abbildung der in der Realität auftretenden Kostenfunktionen. Aus diesem Grund wurde in OptiNet neben der Möglichkeit der Definition von Kantengewichten w_{ij} auch die Möglichkeit geschaffen, wesentlich komplexere abschnittsweise definierten Kostenfunktionen $f(w_{sd}, r_{ij})$ zu verwenden. Abbildung 4 gibt ein Beispiel für die Definition einer komplexeren Kostenfunktion, welche auch in Abschnitt 4.1 wiederverwendet wird. Die Kostenfunktion basiert auf dem Tarifmodell, welches von der Deutschen Telekom im Jahr 1999 für die Anmietung von Festverbindungen zwischen Unternehmensstandorten verwendet wurde. Hierbei standen unterschiedliche Leitungstypen mit unterschiedlichen Kapazitäten (z.B. 64 kBit/s, 512 kBit/s, oder 2 Mbit/s) zur Verfügung. Für jeden dieser Leitungstypen fallen fixe Kosten für die Bereitstellung der Kapazität und längenabhängige Kosten an. Darüber hinaus wurden in diesem

Kostenmodell noch unterschiedliche Entfernungszonen (entsprechend dem Gebührenfeldverfahren) verwendet. Für jede dieser unterschiedlichen Entfernungszonen existieren für jeden Leitungstyp unterschiedliche fixe und längenabhängige Kosten. Hierbei nehmen mit zunehmender Länge der Leitungen die Kosten pro Übertragungskapazität ab.

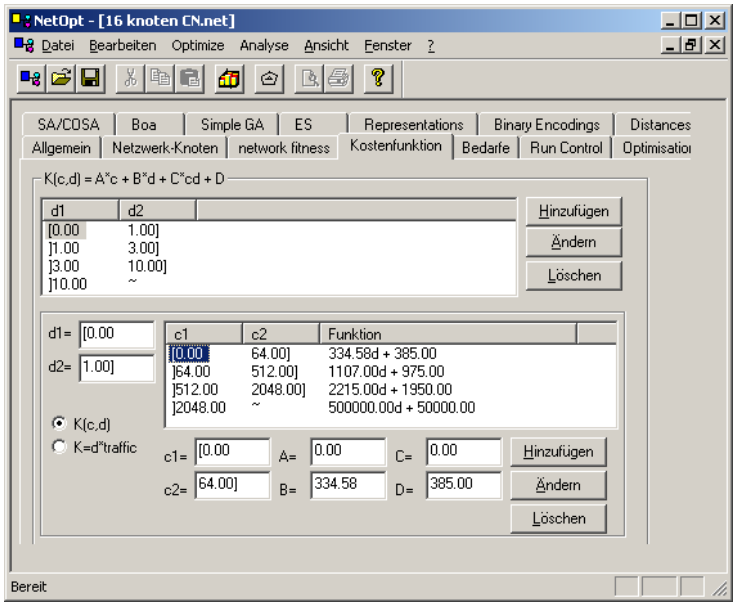


Abbildung 4: Kostenfunktionen

Gegenwärtig können in OptiNet für einzelne Entfernungsklassen neben fixen und längenabhängigen Kosten sowohl verkehrsabhängige Kosten als auch Kosten, welche vom Produkt der Länge und dem transportierten Verkehr abhängen, berücksichtigt werden.

3.3 Problemrepräsentationen

In OptiNet wurden unterschiedliche Problemrepräsentationen für Bäume implementiert. Frühere Forschungsarbeiten haben gezeigt, dass die Wahl einer geeigneten Repräsentation von großer Bedeutung für die Leistungsfähigkeit derartiger Verfahren ist.

3.3.1 Charakteristischer Vektor

Charakteristische Vektoren (CV) stellen eine der am weitesten verbreiteten Kodierungen für Graphen dar (Davis et al. (1993)). Ein repräsentatives Beispiel für die Verwendung der CV Kodierung für Netzwerkprobleme kann in Sinclair (1995) gefunden werden. Ein repräsentatives Beispiel der Verwendung von CV für die Kodierung von Bäumen ist in Berry et al. (1994) zu finden.

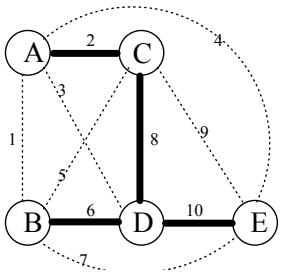


Abbildung 5: Ein Baum mit fünf Knoten

Bei der CV Kodierung wird ein Graph durch einen binären Vektor repräsentiert, welcher festlegt, welche Kanten in einem Graph ver-

wendet werden. Da ein Graph mit n Knoten maximal $n(n-1)/2$ Kanten hat, muss ein CV die Länge $l = n(n-1)/2$ besitzen. Bei der Interpretation des CV wird jeder möglichen Kante eines Graphs eine eineindeutige Zahl $i = \{0,...,l-1\}$ zugeordnet. Der Wert des CV an der i-ten Position gibt an, ob die entsprechende Kante im Graph existiert oder nicht.

Tabelle 2 gibt ein Beispiel für die Kodierung des Baums, welcher in Abbildung 5 dargestellt ist. Hierbei sind die einzelnen Knoten mit A, B, C, D und E bezeichnet und die Kante A-B wird der ersten Position des CV zugeordnet, die Kante A-C der zweiten Position, usw. Der Graph in Abbildung 5 besteht aus den Kanten A-C, B-D, C-D und D-E. Die entsprechenden Einträge des CV Vektors sind auf 1 gesetzt.

0	1	0	0	0	1	0	1	0	1
A-B	A-C	A-D	A-E	B-C	B-D	B-E	C-D	C-E	D-E

Tabelle 2: Charakteristischer Vektor für Baum in Abbildung 5

Die CV Kodierung ist sehr gut für die Kodierung von Graphen geeignet. Probleme treten allerdings auf, wenn sie für die Kodierung von Bäumen verwendet werden soll. Es können invalide Lösungen auftreten, bei denen kein korrekter Baum repräsentiert wird. So führt z.B. jede Änderung eines zulässigen CV zu einem invaliden Baum. Zufällig erzeugte CV können in zweierlei Hinsicht invalide sein:

- Es existieren Zyklen im Graph.
- Der Baum ist nicht vollständig.

Ein Zyklus entsteht z.B. wenn der erste Wert des CV in Tabelle 2 auf 1 gesetzt wird. Der CV repräsentiert keinen Baum mehr, da ein Zyklus A-C-D-B-A existiert. Wird hingegen eine der 1en im CV auf 0 gesetzt, erhält man zwei unverbundene Bäume.

Heuristische Lösungsraumsuchverfahren gehen in unterschiedlicher Weise mit invaliden Lösungen um. So werden z.B. in Davis et al. (1993) invalide Lösungen bis zu einem gewissen Maße in der Population akzeptiert. Im Regelfall werden jedoch beim Einsatz von Heuristiken invalide Lösungen repariert. Ein repräsentativer zweistufiger Ansatz ist in Berry et al. (1994) zu finden:

1. Zufälliges Entfernen von Kanten innerhalb von Zyklen.
2. Falls kein vollständiger Baum vorliegt, werden zusätzliche Kanten eingefügt, welche unverbundene Bäume miteinander verbinden.

Ein derartiger Reparaturmechanismus stellt sicher, dass durch CV keine unzulässigen Bäume repräsentiert werden können.

3.3.2 Gewichtete Repräsentationen

Bei der Verwendung von CV kann keine Information darüber gespeichert werden, wie wichtig eine spezielle Kante für die Konstruktion eines Baums ist. Dies führt dazu, dass Reparaturansätze auf das zufällige Löschen und Hinzufügen von Kanten angewiesen sind und deswegen manchmal wichtige Kanten aus einem überspezifizierten Baum gelöscht werden, bzw. qualitativ schlechte Kanten wieder hinzugefügt werden.

Diese Probleme können durch gewichtete Repräsentationen umgangen werden, da hierbei statt binärer Werte (wie beim CV) kontinuierliche Werte für die Repräsentation eines Baumes ver-

wendet werden. Die im Individuum kodierten Gewichte werden von Konstruktionsalgorithmen verarbeitet, welche aus den Gewichten einen spannenden Baum erzeugen.

3.3.2.1 Network Random Keys

Network Random Keys (Rothlauf et al. (2002)) basieren auf der Random Key (RK) Kodierung, welche von Bean (1992) für die Kodierung von Permutationsproblemen vorgestellt wurde. Später wurde die Kodierung auch für Maschinenbelegungsprobleme, Fahrzeugrouting, Kapazitätszuordnungs- und Travelling Salesperson-Probleme verwendet. Ein Überblick über die Verwendung von RK ist in Norman (1995) zu finden.

Bei der RK Kodierung wird ein Vektor r der Länge $l = n(n-1)/2$ von zufälligen reellwertigen Zahlen $r_i \in [0,1]$ für die Kodierung einer Permutation von l Zahlen verwendet. Ein mögliches Beispiel für einen RK Vektor stellt r=(0,07; 0,75; 0,56; 0,67) dar. Für die Kodierung einer Lösung ist die Position und der relative Wert der einzelnen Elemente r_i des Vektors wichtig. Bei der Erzeugung der Permutation aus dem RK Vektor werden die Positionen der einzelnen Elemente des Vektors entsprechend ihres Wertes in absteigender Ordnung geordnet. In unserem Beispiel muss also als erstes die Position mit dem höchsten Wert gefunden werden (0.75 an Position 2). Der nächsthöhere Wert in r ist an Position 4 (0.67) zu finden. Durch eine derartige Anordnung der Elemente des Vektors findet man für das Beispiel den Permutationsvektor r^s=2-4-3-1. Gemäß dieser Vorgehensweise kann aus jedem RK Vektor der Länge l eine Permutation von l Zahlen gewonnen werden. Jede Zahl zwischen 1 und l tritt in der Permutation nur ein einziges Mal auf, da die Position der einzelnen Werte im RK Vektor eineindeutig ist.

Die Network Random Key (NetKey) Kodierung verwendet RK für die Kodierung eines Baumes. Hierbei wird ein RK Vektor der Länge $l = n(n-1)/2$ für die Kodierung eines Baumes mit n Knoten verwendet. Für die Konstruktion des Baums aus r wird in einem ersten Schritt der Permutationsvektor r^s von l Zahlen erzeugt. Anschließend wird aus dem Permutationsvektor r^s der Baum gemäß folgender Regel konstruiert:

1. Sei $i = 0$, G ein leerer Graph mit n Knoten und r^s ein Permutationsvektor der Länge $l = n(n-1)/2$, welcher aus dem Vektor r erzeugt werden kann. Alle möglichen Kanten von G sind nummeriert von 1 bis l.
2. Sei j die Zahl an der i-ten Position der Permutation r^s.
3. Falls die Einfügung von Kante j keinen Zyklus erzeugt, füge Kante j in G ein.
4. Stop, sobald $n-1$ Kanten in G eingefügt worden sind.
5. Erhöhe i um 1 und fahre mit Schritt 2 fort.

Mit Hilfe dieser Konstruktionsregel kann aus jedem beliebigen Permutationsvektor r^s ein eindeutiger und zulässiger Baum erzeugt werden.

Im Folgenden soll die Funktionsweise von NetKeys demonstriert werden. Tabelle 3 stellt einen RK Vektor r für ein Netzwerk mit fünf Knoten dar. Aus r lässt sich der Permutationsvektor

$r^s = 10-8-6-9-2-7-1-5-4-3$ konstruieren. Dieser Permutationsvektor wird anschließend für die Konstruktion des Baumes verwendet. Hierbei wird zum ursprünglich leeren Graph G die Kante D-E (Position 10) eingefügt. Danach wir Kante C-D (Position 8) und Kante B-D (Position 6) eingefügt. Da das Hinzufügen der Kante C-E (Position 9) zu einem unzulässigen Graph führen würde (Zyklus C-E-D-C), wird diese Kante nicht hinzugefügt. Als letzte Kante wird A-C (Position 2) eingefügt. Damit wurden insgesamt vier Kanten eingefügt und man erhält den in Abbildung 5 dargestellten Graph.

Kante	A-B	A-C	A-D	A-E	B-C	B-D	B-E	C-D	C-E	D-E
Position	1	2	3	4	5	6	7	8	9	10
Wert	0.55	0.73	0.09	0.23	0.40	0.82	0.65	0.85	0.75	0.90

Tabelle 3: Ein RK Vektor für den Baum in Abbildung 5

3.3.2.2 Link-and-Node-biased Kodierung

Als weiterer Vertreter der Klasse von gewichteten Repräsentationen soll die Link-and-node-biased (LNB) Kodierung von Palmer (1994) vorgestellt werden. Abuali et al. (1995) verwendeten die Repräsentation mit Erfolg für das probabilistic minimum spanning tree (PMST) Problem. Später präsentierten Raidl und Julstrom (2000) und Krishnamoorthy und Ernst (2001) eine Variante dieser Repräsentation für das degree-constrained minimum spanning tree Problem. Eine Analyse der Eigenschaften der LNB-Kodierung ist in Rothlauf und Goldberg (2003) zu finden.

Bei der LNB Kodierung wird ein Baum durch reellwertige Gewichte für die Kanten und die Knoten eines Graphen repräsentiert. Ein Individuum besteht aus $n(n-1)/2$ Kantengewichten $b_{ij} \in [0,1]$ ($i, j \in \{0,...,n(n-1)/2\}$) und n Knotengewichten $b_i \in [0,1]$ ($i \in \{0,...,n\}$). Entsprechend der Gewichte werden die ursprünglichen Kantengewichte w_{ij} modifiziert:

$$w'_{ij} = w_{ij} + P_1 b_{ij} w_{\max} + P_2 (b_i + b_j) w_{\max}$$

$w_{\max} = \max(w_{ij})$ ist das maximale Kantengewicht, P_1 ein Kanten-spezifischer Bias und P_2 ein Knoten-spezifischer Bias. P_1 und P_2 sind a priori vorgegeben und steuern den Einfluss der Kanten- und Knotengewichte auf die Konstruktion des Baumes. Man erhält den Baum, welcher durch ein LNB-kodiertes Individuum repräsentiert wird, durch die Bestimmung des minimal spannenden Baums unter Verwendung der veränderten Kantengewichte w'_{ij}. In der ursprünglichen Version der LNB Kodierung wird hierfür Prim's Algorithmus verwendet. Da die Kanten- und Knoten-spezifischen Gewichte die ursprünglichen Kantengewichte verändern, sind im repräsentierten Baum überwiegend Kanten mit niedrigen Kanten- und Knotengewichten vertreten. In Palmer (1994) und Rothlauf und Goldberg (2003) wurde gezeigt, dass durch die LNB Kodierung alle möglichen Bäume repräsentiert werden können, sobald $P_1 > 1$.

Im Folgenden soll ein kurzes Beispiel für die Funktionsweise der LNB Kodierung gegeben werden. Gegeben sei $P_1 = 1$, $P_2 = 1$ und die ursprünglichen Kantengewichte w_{ij} (Tabelle 4). Das kodierte Individuum hat die Knoten-spezifischen Gewichte (0,7; 0,5; 0,2; 0,8; 0,1) und die Kan-

ten-spezifischen Gewichte b_{ij} aus Tabelle 5. Basierend auf den Knoten- und Kanten-spezifischen Gewichten müssen die modifizierten Kantengewichte w'_{ij} (Tabelle 6) bestimmt werden. So ergibt sich beispielsweise für das modifizierte Kantengewicht $w'_{AB} = 3 + 0{,}1 * 10 + (0{,}7 + 0{,}5) * 10 = 16$. Abschließend muss auf Basis der modifizierten Kantengewichte w'_{ij} der minimal spannende Baum berechnet werden. Für den minimal spannenden Baum werden die Kanten A-C, B-D, C-D und D-E verwendet und es ergibt sich der in Abbildung 5 dargestellte Baum.

w_{ij}	A	B	C	D	E
A	-	3	1	3	4
B	-	-	8	1	5
C	-	-	-	2	3
D	-	-	-	-	10

Tabelle 4: Kantengewichte w_{ij}

b_{ij}	A	B	C	D	E
A	-	0,1	0,2	0,2	0,8
B	-	-	0,1	0,1	0,5
C	-	-	-	0,3	0,2
D	-	-	-	-	0,4

Tabelle 5: Kanten-spezifische Gewichte b_{ij}

w'_{ij}	A	B	C	D	E
A	-	16	12	20	20
B	-	-	16	15	16
C	-	-	-	15	8
D	-	-	-	-	23

Tabelle 6: modifizierte Kantengewichte w'_{ij}

3.3.3 Prüfernummern

Prüfernummern sind eine eineindeutige Repräsentation von Bäumen, bei der ein Baum mit n Knoten durch eine Zeichenkette der Länge $n-2$ dargestellt wird. Jedes Element der Zeichenkette kann n verschiedene Werte annehmen. Prüfernummern wurden von Prüfer (1918) als konstruktiver Beweis von Cayley's Formel (Cayley (1889)) eingeführt. Damit konnte gezeigt werden, dass für einen Graph mit n Knoten genau n^{n-2} verschiedene Bäume existieren. Im Folgenden soll dargestellt werden, wie ein Baum mit n Knoten aus einer Prüfernummer konstruiert werden kann.

1. $k = 1$. G sei ein leerer Graph mit n Knoten. P sei eine Prüfernummer der Länge $n-2$. Alle Knoten seien nummeriert $\{0,....,n\}$. Alle Knoten, die nicht in der Prüfernummer vorkommen, können für die Konstruktion des Baumes verwendet werden.
2. Aus der Menge der verfügbaren Knoten wird der Knoten mit der kleinsten Nummer i ausgewählt. j sei die Nummer an der k-ten Position der Prüfernummer.
3. Eine Kante von Knoten i nach Knoten j wird zum Graph G hinzugefügt.
4. Knoten i kann nicht mehr verwendet werden. Falls j nicht mehr an einer höheren Position k in der Prüfernummer vorkommt, kann j verwendet werden.
5. $k = k + 1$
6. Wiederhole Schritte 2-6 solange bis $k = n - 2$.

7. Genau zwei Knoten r und s können noch verwendet werden. Füge die Kante zwischen Knoten r und Knoten s zum Graphen G hinzu und erhalte einen vollständigen Baum T.

Im Folgenden wird ein kurzes Beispiel für die Konstruktion eines Baums mit $n = 5$ Knoten aus einer Prüfernummer gegeben. Gegeben sei die Prüfernummer P=344. Zu Beginn des Konstruktionsprozesses können die Knoten 1, 2 und 5 verwendet werden (sind nicht in P enthalten). Es wird die Kante zwischen dem verfügbaren Knoten mit der niedrigsten Nummer (Knoten 1) und der Nummer des Knotens an der ersten Stelle der Prüfernummer (Knoten 3) eingefügt (Kante 1-3). Damit kann Knoten 1 nicht mehr verwendet werden und Knoten 3 wird in die Menge der verfügbaren Knoten aufgenommen (dieser kommt sonst nicht mehr in der Prüfernummer vor). Anschließend wird der verfügbare Knoten mit der niedrigsten Nummer (Knoten 2) ausgewählt und mit dem Knoten an der $k = 2$ ten Stelle der Prüfernummer verbunden. Es wird also die Kante 2-4 eingefügt. Danach wird Knoten 2 aus der Menge der verfügbaren Knoten gestrichen und es sind noch die Knoten 3 und 5 verfügbar (4 kommt nochmals in der Prüfernummer vor). Demzufolge wird im nächsten Schritt Knoten 3 mit dem Knoten an der $k = 3$ ten Stelle der Prüfernummer (Knoten 4) verbunden und die Kante 3-4 eingefügt. Knoten 4 kommt nicht mehr in der Prüfernummer vor und wird somit verfügbar. Es sind noch die Knoten 4 und 5 verfügbar. Abschließend wird die Kante 4-5 eingefügt und man erhält den Baum aus Abbildung 5.

3.4 Optimierungsverfahren

In OptiNet sind eine Reihe von unterschiedlichen naturanalogen Optimierungsverfahren implementiert. Im Moment stehen als Verfahren Genetische Algorithmen, Bayesian Optimization Algorithmen, Evolutionsstrategien, Simulated Annealing und Cooperative Simulated Annealing zur Verfügung. Die Optimierungsverfahren können beliebig mit den unterschiedlichen Problemrepräsentationen kombiniert werden. Aufgrund des modularen Aufbaus von OptiNet können zusätzliche Verfahren problemlos integriert werden. Im Folgenden soll ein kurzer Überblick über die gegenwärtig in OptiNet vorhandenen Lösungsverfahren gegeben werden.

Genetische Algorithmen: Genetische Algorithmen (Goldberg (1989)) orientieren sich an den Prinzipien der Evolution („survival of the fittest“) und wenden Suchoperatoren (Rekombination und Mutation) auf eine Menge (Population) von Problemlösungen (Individuen) über mehrere Iterationen (Generationen) an. Zusätzlich wird in jeder Generation die Qualität eines jeden Individuums bezüglich einer vorgegebenen Zielfunktion bewertet (Fitness) und durch einen Selektionsoperator schlechte Lösungen aus der Population entfernt. GA wenden als Hauptsuchoperator Rekombination an. In OptiNet können sowohl Einpunkt-, Zweipunkt- als auch Multipunkt-Crossover mit Wahrscheinlichkeit p_c eingesetzt werden. Als Mutationsoperator können einfache Veränderungen einzelner Allele mit der Wahrscheinlichkeit p_m durchgeführt werden. Als Selektionsverfahren sind fitnessproportionale Selektion, Wettkampfselektion und stochastic universal sampling verfügbar (der interessierte Leser sei für zusätzliche Informationen über GA auf entsprechende Lehrbücher verwiesen). GA können für alle in Abschnitt 3.3 vorgestellten Repräsentationen verwendet werden.

Bayesian Optimization Algorithmen: Bayesian Optimization Algorithmen (BOA) wurden erstmalig von Pelikan et al. (1999) zur Lösung von binär kodierten Problemen vorgestellt. Bei

diesem Optimierungsverfahren werden die genetischen Suchoperatoren Rekombination und Mutation durch die Konstruktion eines statistischen Modells einer Population ersetzt. Dabei werden die statistischen Eigenschaften einer Population und insbesondere die Abhängigkeiten zwischen den einzelnen binären Allele durch ein Bayes Netz modelliert. Anschließend wird basierend auf dem Bayes Netz eine neue Population erzeugt. Als Selektionsverfahren kommt rangbasierte Selektion zum Einsatz, bei der in jeder Generation 50% der Individuen entsprechend ihrer Fitness aus der Population entfernt werden. Aufgrund der Beschränkung auf binäre Problemrepräsentationen kann BOA nur für die CV und Prüfernummern verwendet werden. Die ganzzahligen Prüfernummern werden hierbei binär kodiert.

Evolutionsstrategien: Evolutionsstrategien (ES) wurden in den 60er Jahren von Rechenberg (1973) und Schwefel (1965) entwickelt. Als Hauptsuchoperator wird Mutation (üblicherweise normalverteilt) verwendet, welche auf kontinuierlichen Variablen angewendet wird. Darüber hinaus werden Strategieparameter, welche für die Richtung und Schrittweite der Mutation relevant sind, zusätzlich im Individuum kodiert. Gängige ES sind die (1+1)-ES, die $(\mu + \lambda)$-ES und die (μ, λ)-ES. Dabei wird jeweils aus 1, bzw. μ Eltern insgesamt 1, bzw. λ Nachkommen erzeugt. Bei der $(\mu + \lambda)$-Evolutionsstrategie überleben die λ Besten aus den Eltern und Nachkommen. Bei der (μ, λ)-ES überleben nur die λ Besten aus den Nachkommen.

Simulated Annealing: Simulated Annealing (SA) ist ein naturinspiriertes Optimierungsverfahren, welches den Vorgang des kontrollierten Abkühlens als Vorbild für die Entwicklung eines mutationsbasierten Optimierungsverfahren nimmt. Bei SA werden iterative Mutationen einer Lösung vorgenommen und das alte Individuum durch das neue ersetzt, wenn die Fitness $f(x)$ des neu erzeugten Individuums ansteigt. Ist die Fitness niedriger, wird das alte Individuum x_a mit der Wahrscheinlichkeit $P(T) = \exp\left(-\left(f(x_a) - f(x_n)\right)/T\right)$ durch das neue Individuum x_n ersetzt. Im Laufe des Optimierungsprozesses wird die Temperatur T verringert und die Wahrscheinlichkeit, dass schlechtere Lösungen akzeptiert werden, sinkt. Für weitere Informationen zu SA sei auf van Laarhoven et al. (1988) verwiesen.

Cooperative Simulated Annealing: Cooperative Simulated Annealing (COSA) wurde von Wendt (1995) vorgestellt. Bei diesem Verfahren werden GA mit SA-Elementen kombiniert. Es wird eine Population von Individuen verwendet und für jedes einzelne Individuum ein SA durchgeführt. Die Wahrscheinlichkeit für einen Informationsaustausch wird, wie bei SA, durch die Metropolis-Wahrscheinlichkeit gesteuert. COSA weist ähnliche Charakteristika wie SA auf.

4 Einsatz von OptiNet in der Praxis

4.1 Problembeschreibung

Im Folgenden soll OptiNet auf ein Anwendungsproblem angewendet werden, bei dem ein baumförmiges Kommunikationsnetz unter Verwendung von angemieteten Leitungen mit diskreten Leitungskapazitäten aufzubauen ist. Eine Menge von Städten in Deutschland und mögliche diskrete Leitungskapazitäten sind a priori vorgegeben. Dieses Problem kann entsprechend Abschnitt 2.2 als MCST Problem mit einer gemäß Abschnitt 2.3 modifizierten Kostenfunktion modelliert werden. Es soll dabei die Baumstruktur gefunden werden, welche alle (vorgegebe-

nen) Kommunikationsbedarfe erfüllt und minimale Kosten aufweist. Im Gegensatz zum ursprünglichen MCST Problem, bei welchem die Kosten einer Kante nur vom Produkt der Länge und dem darüberfließenden Verkehr abhängen, setzen sich die Kosten gemäß Abschnitt 2.3 (vgl. auch Abbildung 4) für unterschiedliche Entfernungsklassen aus einem fixen und einem längenabhängigen Anteil zusammen. Tabelle 7 zeigt, wie die Kosten einer Leitung von deren Länge d (gemäß dem Gebührenfeldverfahren) und Kapazität abhängen.

Kantenkosten	64 kBit/s	512 kBit/s	2048 MBit/s	>2048 Mbit/s
$d < 1$	$334{,}6\,d$ +385	$1107\,d$ +975	$2215\,d$ +1950	$50000\,d$ +50000
$1 \le d < 3$	$148{,}7\,d$ +572	$520\,d$ +1567	$1040\,d$ +3135	$50000\,d$ +50000
$3 \le d < 10$	$29{,}7\,d$ +972,5	$178\,d$ +2717	$356{,}9\,d$ +5435	$50000\,d$ +50000
$d \ge 10$	$22{,}3\,d$ +1047	$111{,}5\,d$ +3392	$223\,d$ +6785	$50000\,d$ +50000

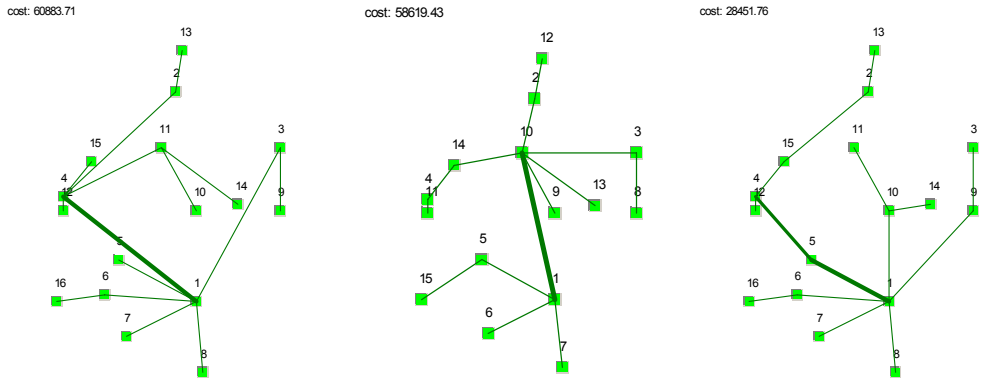
Tabelle 7: Kostenfunktion für Praxisproblem (vgl. Abschnitt 2.3)

Im Rahmen der Untersuchung sollen die folgenden drei Szenarien betrachtet werden:

Szenario 1: Bei diesem Netzwerkproblem mit 16 unterschiedlichen Städten tauscht nur Standort 1 mit den anderen Niederlassungen Daten aus. Standort 1 ist die Zentrale der Unternehmung und alle anderen Niederlassungen wickeln ihre Datenverarbeitung direkt in der Zentrale ab. Die Struktur der optimalen Lösung für dieses Problem zeigt Abbildung 6(a).

Szenario 2: Bei dieser Problemvariation existieren nur 15 unterschiedliche Standorte. Weiterhin existiert zwischen wenigen ausgewählten Standorten zusätzlicher Verkehr. Die optimale Lösung für dieses Problem zeigt Abbildung 6(b).

Szenario 3: Die Anzahl und Position der Niederlassungen, als auch die Bedarfe sind wie in Problem 1 gewählt. Allerdings ist in diesem Szenario der fixe Anteil der Kosten um 90% reduziert. Damit sind die Kosten der Kanten überwiegend von der Länge der Leitungen abhängig. Abbildung 6(c) zeigt die optimale Lösung für dieses Szenario.



(a) Szenario 1 *(b) Szenario 2* *(c) Szenario 3*

Abbildung 6: Optimale Lösungen für die unterschiedlichen Beispielszenarien

4.2 Ergebnisse

Im Rahmen einer Simulationsstudie wurden die unterschiedlichen Szenarien mit Hilfe von OptiNet gelöst. Insbesondere wurde dabei der Einfluss der in Abschnitt 3.3 vorgestellten unterschiedlichen Repräsentationen auf die Lösungsqualität von GA untersucht. Da ein umfassender Vergleich aller Verfahren den Rahmen dieses Beitrags sprengen würde, wurden GA als ein repräsentativer Vertreter der in OptiNet implementierten naturanalogen Optimierungsverfahren verwendet.

Da bei GA Rekombination als Hauptsuchoperator fungiert (Goldberg (1989)), wurde in den Experimenten vollständig auf Mutation verzichtet und nur Crossover eingesetzt ($p_m = 0$, $p_c = 1$). Die zusätzliche Verwendung von Mutation führt zu keiner prinzipiellen Veränderung des Suchverfahrens, sondern verlangsamt den Konvergenzprozess etwas. Als Rekombinationsoperator wurde uniformer Crossover und als Selektionsverfahren Wettkampfselektion mit der Wettkampfgröße zwei verwendet. Der GA wurde entweder nach 200 Generationen oder nachdem die Population vollständig konvergiert ist (alle Individuen repräsentieren die gleiche Lösung) abgebrochen. Für jede Parameterkonstellation sind 100 Läufe durchgeführt worden. Abbildung 7 (Szenario 1), Abbildung 8 (Szenario 2) und Abbildung 9 (Szenario 3) zeigen den Einfluss unterschiedlicher Repräsentationen (CV, NetKeys, Prüfernummern und LNB) auf die Lösungsqualität von GA. Es wird dargestellt, jeweils gemittelt über 100 Durchläufe, wie die Kosten der besten Lösung in der Population am Ende des Simulationslaufs, als auch dessen Dauer, von der verwendeten Populationsgröße N abhängen. Aus Gründen der Übersichtlichkeit wurde auf die Darstellung der Standardabweichungen verzichtet. Es ergibt sich für die einzelnen Repräsentationen ein einheitliches Bild für alle drei unterschiedlichen Szenarien.

Die schlechtesten Ergebnisse liefert die CV Kodierung. GA liefern qualitativ sehr schlechte Lösungen und alle Simulationsläufe mussten nach jeweils 200 Generationen abgebrochen werden.[4] Die schlechten Ergebnisse lassen sich dadurch erklären, dass durch den notwendigen Reparaturprozess von invaliden Lösungen zufällig gute Kanten aus einer Lösung entfernt, bzw. schlechte Kanten eingefügt werden (vgl. auch Rothlauf (2002), Kapitel 6.4). Prüfernummern liefern durchweg die zweitschlechtesten Ergebnisse. Der Grund für die schlechte Leitungsfähigkeit liegt in der niedrigen Lokalität der Repräsentation (Gottlieb et al. (2001)). Geringe Lokalität bedeutet, dass kleine Veränderungen einer Prüfernummer zu großen Veränderungen im repräsentierten Baum führen. Dadurch können sich GA nicht mehr zielgerichtet durch den Lösungsraum bewegen und die Suche wird „randomisiert". NetKeys führen in allen drei Szenarien zu guten Ergebnissen und ermöglichen es GA die in Abbildung 6 dargestellten optimalen Lösungen in nur wenigen Generationen zu finden. Diese Repräsentation hat eine hohe Lokalität (kleine Änderungen der RK Vektors führen zu kleinen Änderungen der repräsentierten Lösung) und es sind auch keine zusätzlichen Reparaturmechanismen notwendig. In Rothlauf und Goldberg (2003) wurde gezeigt, dass sich die LNB Kodierung bei der Verwendung eines großen Kanten-spezifischen Bias P_1 genauso wie NetKeys verhält. Dieses Verhalten (LNB mit $P_1 = 20$ und NetKeys führen zu sehr ähnlichen Ergebnissen) wird auch durch die Ergebnisse bestätigt. Weiterhin wurde gezeigt, dass die LNB Kodierung für kleine P_1 Lösungen ähnlich dem minimal

[4] Aus illustrativen Gründen wurden die Werte aus den entsprechenden Abbildungen entfernt.

spannenden Baum und für kleine P_2 ähnlich einem Stern überrepräsentiert. Da gleichzeitig auch oft die optimale Lösung für MCST Probleme Ähnlichkeiten zum minimal spannenden Baum aufweist (Rothlauf et al. (2003)), führt die Verwendung der LNB Kodierung mit einem geringen Kanten- oder Knoten-spezifischen Gewicht P_1 , bzw. P_2 dann zu sehr guten Ergebnissen, wenn die optimale Lösung entweder ähnlich dem MST oder einem Stern ist.

Zusammenfassend lässt sich feststellen, dass CV und Prüfernummern für die Lösung des MCST Problems ungeeignet sind. In Abhängigkeit von der Struktur der optimalen Lösung können durch die Verwendung der LNB Kodierung mit einem niedrigen $P_1 \approx 1$ oder $P_2 \approx 1$ oft gute Ergebnisse erzielt werden. NetKeys und LNB mit großem P_1 liefern allerdings gleichbleibend gute Ergebnisse und sollten dann eingesetzt werden, wenn keine Informationen über die Struktur der optimalen Lösung vorliegen oder die optimale Lösung keine Ähnlichkeit mit dem MST aufweist.

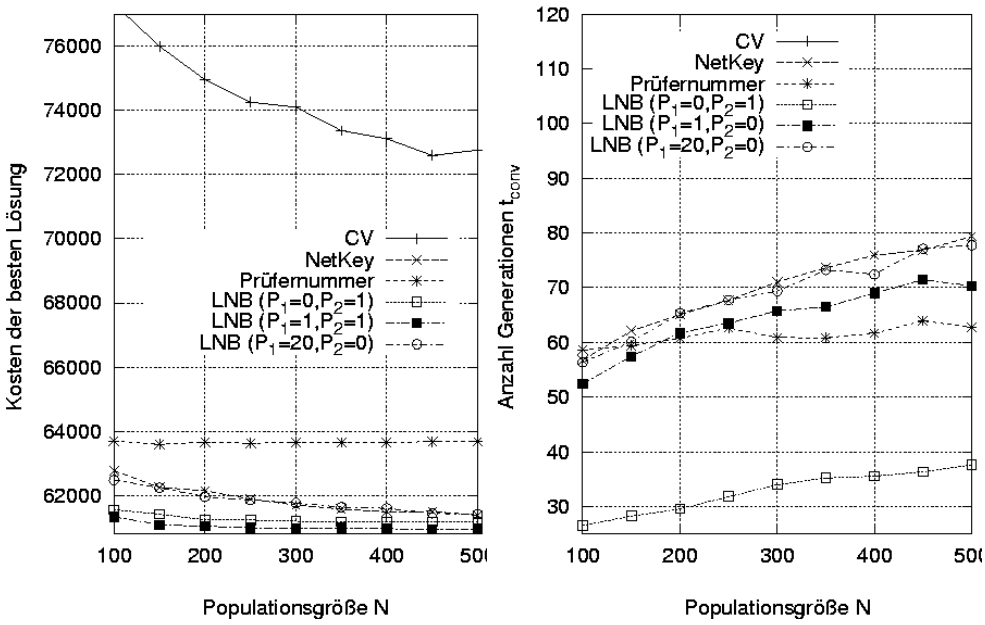


Abbildung 7: Einfluss unterschiedlicher Repräsentationen auf die Lösungsqualität von GA für Szenario 1

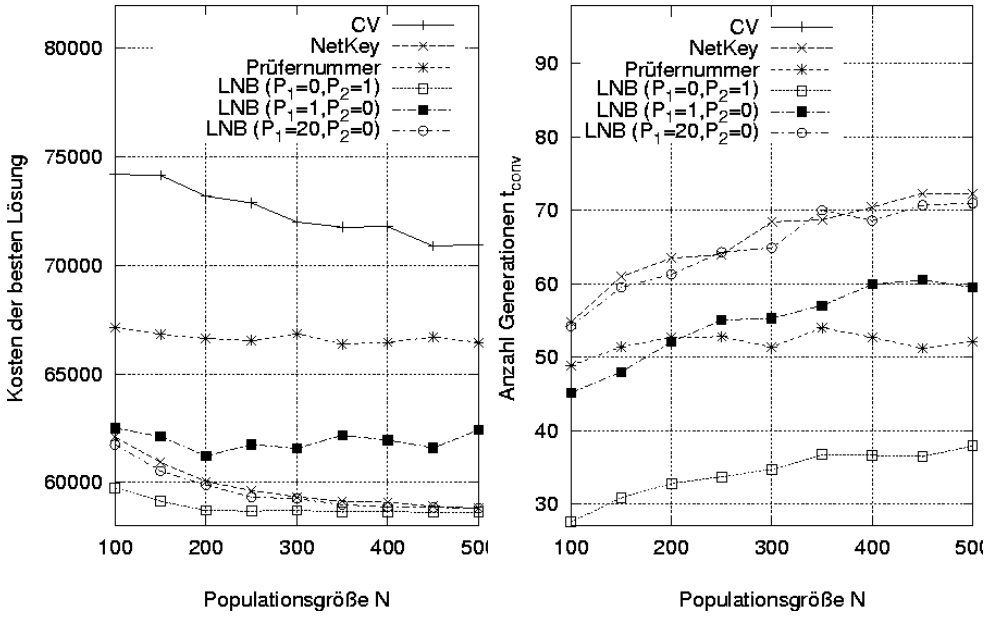


Abbildung 8: Einfluss unterschiedlicher Repräsentationen auf die Lösungsqualität von GA für Szenario 2

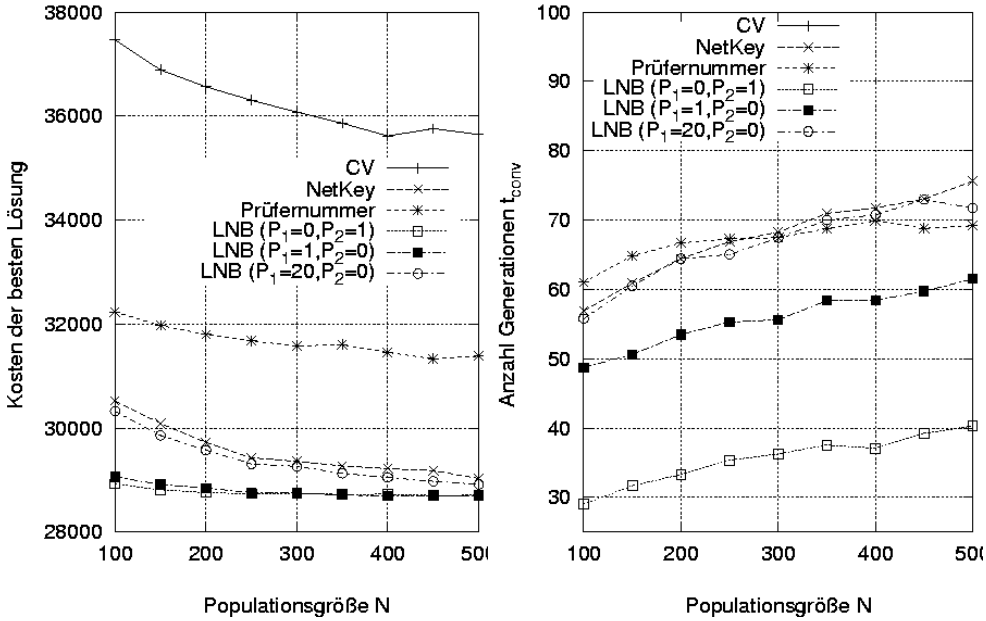


Abbildung 9: Einfluss unterschiedlicher Repräsentationen auf die Lösungsqualität von GA für Szenario 3

5 Zusammenfassung

Der vorliegende Beitrag stellt das Optimierungswerkzeug OptiNet vor und demonstriert, wie es zur Untersuchung und zur Lösung von baumförmigen Netzwerkproblemen eingesetzt werden kann. In OptiNet können naturanaloge Optimierungsverfahren wie z.B. Genetische Algorithmen, Bayesian Optimization Algorithmen, Evolutionsstrategien, Cooperative Simulated Annealing und Simulated Annealing mit unterschiedlichen Problemrepräsentationen (charakteristische Vektoren, Network Random Keys, Link-and-Node-biased Kodierung und Prüfernummern) kombiniert werden. Der praktische Einsatz von OptiNet für ein reales Netzwerkoptimierungsproblem mit entsprechenden, an die realen Gegebenheiten angepassten, Kostenfunktionen zeigt, dass große Unterschiede in der Leistungsfähigkeit von Repräsentationen existieren und charakteristische Vektoren und Prüfernummern ungeeignet für die Kodierung von baumförmigen Netzwerken sind. Verlässlich gute Ergebnisse können nur durch die Verwendung von Network Random Keys oder speziellen Varianten der Link-and-Node-biased Kodierung sichergestellt werden.

Literatur

Abuali, F. N.; Wainwright, R. L.; Schoenefeld, D. A. (1995) Determinant factorization: A new encoding scheme for spanning trees applied to the probabilistic minimum spanning tree problem. In Proceedings of the Sixth International Conference on Genetic Algorithms (S. 470–477). San Mateo, Morgan Kaufmann.

Bean, J. C. (1992) Genetics and random keys for sequencing and optimization (Technical Report 92-43). Ann Arbor, MI: Department of Industrial and Operations Engineering, University of Michigan.

Berry, L. T. M.; Murtagh, B. A.; Sugden, S. J. (1994) A genetic-based approach to tree network synthesis with cost constraints. In Second European Congress on Intelligent Techniques and Soft Computing - EUFIT' 94, Volume 2 (S. 626–629). Aachen, Verlag der Augustinus Buchhandlung.

Cayley, A. (1889) A theorem on trees. Quarterly Journal of Mathematics, 23, 376–378.

Davis, L.; Orvosh, D.; Cox, A.; Qiu, Y. (1993) A genetic algorithm for survivable network design. In Proceedings of the Fifth International Conference on Genetic Algorithms (S. 408–415). San Mateo, Morgan Kaufmann.

Goldberg, D. E. (1989) Genetic algorithms in search, optimization, and machine learning. Reading, Addison-Wesley.

Gottlieb, J.; Julstrom, B. A.; Raidl, G. R.; Rothlauf, F. (2001) Prüfer numbers: A poor representation of spanning trees for evolutionary search. In Proceedings of the Genetic and Evolutionary Computation Conference 2001 (S. 343-350). San Mateo, Morgan Kaufmann Publishers.

Hu, T. C. (1974) Optimum communication spanning trees. SIAM Journal on Computing, 3 (3), S. 188–195.

Johnson, D. S.; Lenstra, J. K.; Kan, A. H. G. R. (1978) The complexity of the network design problem. Networks, 8, S. 279–285.

Krishnamoorthy, M.; Ernst, A. T. (2001) Comparison of algorithms for the degree constrained minimum spanning tree. Journal of Heuristics, 7, S. 587–611.

Norman, B. A. (1995) Scheduling using the random keys genetic algorithm. Unveröffentlichte Dissertation, University of Michigan. Ann Arbor, Michigan.

Palmer, C. C. (1994) An approach to a problem in network design using genetic algorithms. Unveröffentlichte Dissertation. Polytechnic University, Troy, NY.

Pelikan, M.; Goldberg, D. E.; Cantu-Paz, E. (1999) BOA: The Bayesian optimization algorithm (IlliGAL Report No. 99003). Urbana, IL: University of Illinois at Urbana-Champaign.

Prüfer, H. (1918) Neuer Beweis eines Satzes über Permutationen. Archiv für Mathematik und Physik , 27, S. 742–744.

Raidl, G. R.; Julstrom, B. A. (2000) A weighted coding in a genetic algorithm for the degree-constrained minimum spanning tree problem. In Proceedings of the 2000 ACM Symposium on Applied Computing (S. 440–445). ACM Press.

Rechenberg, I. (1973) Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Stuttgart-Bad Cannstatt, Friedrich Frommann Verlag.

Rothlauf, F. (2002) Representations for genetic and evolutionary algorithms. Heidelberg, Springer.

Rothlauf, F.; Goldberg, D. E.; Heinzl, A. (2002) Network random keys – A tree network representation scheme for genetic and evolutionary algorithms. Evolutionary Computation, 10 (1),S. 75–97.

Rothlauf, F.; Goldberg, D. E. (2003). Redundant representations in evolutionary computation. Evolutionary Computation, 11(4).

Rothlauf, F.; Gerstacker, J.; Heinzl, A. (2003). On the optimal communication spanning tree problem. Working Paper 10/2003. Working Paper in Information Sytems. Lehrstuhl Wirtschaftsinformatik 1, Universität Mannheim.

Schwefel, H.-P. (1965) Kybernetische Evolution als Strategie der experimentellen Forschung in der Strömungstechnik. Diplomarbeit, Technische Universität Berlin.

Sinclair, M. C. (1995) Minimum cost topology optimisation of the COST 239 European optical network. In Proceedings of the 1995 International Conference on Artificial Neural Nets and Genetic Algorithms (S. 26–29). New York, Springer.

van Laarhoven, P. J. M.; Aarts, E. H. L. (1988) Simulated Annealing: Theory and Applications. Dordrecht, Kluwer.

Wendt, O. (1995) Tourenplanung durch Einsatz naturanaloger Verfahren. Promotion. Wiesbaden, Gabler.