NETZWERK-STANDORTPROBLEME MIT OD-PAAREN

Diplomarbeit

vorgelegt von

Marie Schmidt

aus

Uelzen

angefertigt

im Institut für Numerische und Angewandte Mathematik

der Georg-August-Universität zu Göttingen

2009

Inhaltsverzeichnis

1 Einleitung

Diese Arbeit beschäftigt sich mit dem Problem der Platzierung von Teilnetzwerken in Netzwerken, wobei verschiedene Netzwerktypen betrachtet werden. Ziel ist dabei eine optimale Lage des Teilnetzwerks bezüglich einer Menge von Knotenpaaren, zwischen denen eine Nachfrage definiert ist, zu bestimmen.

Während in Standortproblemen die Standorte ursprünglich als punktförmig modelliert werden, gibt es auch Anwendungen, bei denen es sinnvoll erscheint, ganze Netzwerke als zu platzierende Standorte anzunehmen, zum Beispiel bei der Einrichtung von Transportnetzwerken.

In einigen Fällen erscheint es zudem angebracht, das einzurichtende Netzwerk nicht frei in der Ebene zu platzieren, sondern in einem vorgegebenen unterliegenden Netzwerk. Betrachtet man zum Beispiel die Einrichtung eines Busnetzwerks, so ist zu berücksichtigen, dass die Buslinien nur auf bereits bestehenden Straßen verlaufen können. Das einzurichtende Netzwerk ist also Teilnetzwerk eines vorgegebenen größeren Netzwerks.

Es wird angenommen, dass im gegebenen Netzwerk eine Nachfrage zwischen Knotenpaaren besteht, die durch so genannte Origin-Destination-Paare (kurz: OD-Paare) gegeben ist. Diese repräsentieren im Modell des Stadtnetzwerks mit einzurichtender Buslinie Start- und Zielorte von potenziellen Kunden. Auch die Höhe der Nachfrage eines OD-Paares, das heißt die Anzahl der Reisenden zwischen einem Start- und Zielpunkt, wird als bekannt vorausgesetzt.

Unter der Annahme, dass sich die Reisezeit pro zurückgelegter Strecke bei Benutzung des einzurichtenden Netzwerks um einen konstanten Faktor verringert, soll die Reisezeit der Nutzer minimiert werden. Dazu werden zwei unterschiedliche Kriterien herangezogen: die Minimierung der maximalen Reisezeit und die Maximierung der gesamten Reisezeitersparnis. Weitere mögliche Zielsetzungen, wie etwa eine Maximierung der Abdeckung durch das eingerichtete Netzwerk, werden nicht behandelt.

Da sich beide Probleme in ihrer allgemeinen Formulierung als wenig zugänglich erweisen, wird eine Unterteilung in Teilprobleme durch Spezifizierung des gegebenen und des gesuchten Netzwerks vorgenommen.

Ziel dieser Arbeit ist erstens, durch Untersuchung der Komplexität der Teilprobleme herauszufinden, wie die Wahl des unterliegenden und des gesuchten Graphen die Lösbarkeit der beiden Optimierungsprobleme beeinflussen und somit Einsicht in die Problemstruktur zu erlangen. Es soll hierbei zwischen polynomiell lösbaren und NP-schweren Problemen (die unter der Annahme P$\neq$NP nicht polynomiell lösbar sind) unterschieden werden. Weiterhin sollen Verfahren zur Lösung einiger Teilprobleme entwickelt werden. Für die in Polynomialzeit lösbaren Probleme

ergeben diese sich direkt durch den Beweis der polynomiellen Lösbarkeit. Im Falle der NP-schweren Probleme werden viele Probleme weiter als stark NP-schwer charakterisiert, während für einige wenige pseudopolynomielle Lösungsalgorithmen angegeben werden können.
Zusätzlich soll eines der stark NP-schweren Optimierungsprobleme, die Minimierung der maximalen Distanz durch einen Teilbaum im Baum, eingehend betrachtet werden.
Nach einem kurzen Überblick über Arbeiten im Bereich der Netzwerkoptimierung werden einige Begriffe definiert, die zum Verständnis der Problemstellung notwendig sind. Grundlegende Definitionen der Graphen- und Komplexitätstheorie werden allerdings als bekannt vorausgesetzt (siehe zum Beispiel [Die00] zur Graphentheorie und [GJ79] zur Komplexitätstheorie). Es folgen einige in dieser Arbeit verwendete, nicht problemspezifische Aussagen mit Beweisen. Im Anschluss daran wird die Einteilung in Teilprobleme anhand der Struktur von gegebenem und gesuchtem Graph dargestellt.
Kapitel 4 widmet sich dem Problem, einen Teilgraph eines gegebenen Graphen zu finden, der die maximale Distanz zwischen den Knoten eines OD-Paares minimiert. Nach der mathematischen Formulierung dieses Optimierungs- und des dazugehörigen Entscheidungsproblems wird eine Einordnung der Teilprobleme in die Komplexitätsklassen der polynomiell lösbaren, NP-vollständigen und stark NP-vollständigen Probleme vorgenommen. Lösungsalgorithmen für die Teilprobleme aus P werden angegeben.
Im fünften Kapitel wird das Problem, die gesamte Reisezeitersparnis zu minimieren, betrachtet. Das Vorgehen ist analog zu Kapitel 4, zusätzlich werden drei der NP-vollständigen Probleme auf pseudopolynomiell lösbare Probleme zurückgeführt.
Das sechste Kapitel beschäftigt sich mit dem Finden von Lösungen für eines der Teilprobleme aus dem vierten Kapitel, der Suche nach einem bezüglich der maximalen Distanz optimalen Unterbaum eines Baumes. Aufbauend auf einem Algorithmus zur Konstruktion von sicher in der Optimallösung enthaltenen Unterbäumen wird das Entscheidungsproblem in Spezialfällen polynomiell gelöst. Die sukzessive Lösung von Entscheidungsproblemen kann dann zu einer Lösung des Optimierungsproblems führen. Das Prinzip der in der Optimallösung sicher enthaltenen Unterbäume wird weiterhin dazu genutzt, heuristische Verfahren und letztendlich einen Branch-and-Bound-Algorithmus für das betrachtete Teilproblem zu entwickeln.
Abgeschlossen wird die Arbeit mit einer Zusammenfassung der Ergebnisse.

2 Literaturüberblick

Im Bereich der Standortplanung gibt es zahllose Arbeiten über die optimale Platzierung von Teilnetzwerken in Netzwerken. Eine erste Einteilung der Probleme kann anhand der Struktur des gegebenen Netzwerks sowie den Anforderungen an den Aufbau des gesuchten Netzwerks vorgenommen werden. Da Lösungsverfahren oft auf diesen für die Netzwerke geforderten Eigenschaften beruhen, haben die Arten von gegebenen und gesuchten Netzwerken häufig entscheidenden Einfluss auf die Komplexität des entsprechenden Problems. Während viele Arbeiten sich auf eine bestimmte Art von Netzwerk und Teilnetzwerk beschränken, beschäftigen sich Hakimi, Schmeichel und Labbé gezielt mit der Untersuchung der Komplexität der durch Auswahl verschiedener Graphentypen entstehenden Teilprobleme [HSL93], wie es auch in vorliegender Arbeit der Fall sein wird.

Ziel der Netzwerk-Standortplanung ist das Finden eines optimalen Teilnetzwerks von einem gegebenen Netzwerk, wobei die Definition von Optimalität problemspezifisch ist.

Wie auch in dieser Arbeit spielt dabei oft die Erreichbarkeit eine Rolle, wobei diese sich meist nicht auf die Distanz zwischen Knotenpaaren, sondern auf die Entfernung jedes Knoten zum Teilnetzwerk bezieht. Die Erreichbarkeit kann über die Abdeckung durch das Netzwerk (siehe [CRC85], [CS89], [KLWF90], [KLTW96], [CS94], [GLS97]) definiert werden, wobei üblicherweise die Knoten als abgedeckt gelten, deren Distanzen zum Teilnetzwerk kleiner sind als ein vorgegebener Wert. Der in dieser Arbeit verwendeten Zielsetzung entsprechen jedoch eher die Betrachtung der maximalen Distanz eines Knotens zum Teilnetzwerk (siehe [MCH81], [Sla82], [MP83], [Min85], [Ric90], [AN92], [HSL93]) oder der Summe der Distanzen zum Netzwerk (siehe [Sla82], [MP83], [Min85], [Ric90], [HSL93], [CS94], [GR03]).

Mitchell Hedetniemi, Cockayne und Hedetniemi weiten die Definition des Zentrums eines Baumes von einem zentralen Punkt auf zentrale Teilpfade aus. Hier ist der eindeutig bestimmte zentrale Pfad P eines Baumes derjenige, von welchem aus die Distanz zu dem am weitesten von P entfernten Knoten minimal ist [MCH81]. Neben der Distanz zum maximalen Punkt betrachtet Slater in [Sla82] auch die Summe der Distanzen zu allen Punkten.

Während in [MCH81] und [Sla82] die Länge des Teilnetzwerks noch keine Rolle spielte, wird sie zunehmend selbst als Zielfunktion angesetzt (siehe [CS89], [CS89], [KLWF90], [KLTW96], [CS94], [GLS97]). In anderen Fällen tritt sie als Beschränkung bei der Konstruktion des Teilnetzwerks auf. Beispielsweise beschreiben Minieka und Patel in [MP83] zwei Probleme: Unter den Teilpfaden einer fest vorgegebenen Länge l_0 eines Netzwerks soll erstens der Pfad gefunden werden, dessen Distanz zum am weitesten entfernten Knoten minimal ist; zweitens wird ein

Pfad gesucht, für den die Summe der Distanzen zu allen Knoten am geringsten ist. In [Min85] werden diese Probleme statt für eine vorgegebene genaue Pfadlänge l_0 für alle Pfade P, deren Länge kleiner als l_0 ist, betrachtet. Im Folgenden werden die beschriebenen Probleme wie in [Min85] mit *Minimum Eccentricity* beziehungsweise *Minimum Distancesum* bezeichnet. In [Min85] wird unter anderem für diese beiden Probleme für Teilpfade oder Teilbäume in Bäumen ein polynomieller Optimierungsalgorithmus angegeben.

Aneja und Nair beweisen die NP-Vollständigkeit von *Minimum Eccentricity* in allgemeinen Graphen durch Reduktion auf das Steinerbaumproblem [AN92]. Komplexitätsbetrachtungen für die zwei genannten Probleme für einen oder mehrere Pfade oder Bäume in Bäumen oder allgemeinen Netzwerken als unterliegende Graphen findet man in [HSL93].

Während zuvor bei der Platzierung des Netzwerkes alle Knoten als gleich angesehen wurden, wird in [CRC85] jedem Knoten v des Netzwerks eine Nachfrage, gedeutet als Population in diesem Knoten, zugeordnet. Diese Nachfrage gilt als befriedigt, wenn der gesuchte Pfad den Knoten v selbst oder einen Knoten innerhalb einer festgelegten Distanz zu v enthält. Es wird ein bikriterielles Problem betrachtet, das darin besteht, einen Pfad P ausfindig zu machen, so dass einerseits möglichst niedrige Kosten entstehen und andererseits die nachfragegewichtete Summe der abgedeckten Punkte maximal ist.

Auch in [GR03] werden den Knoten unterschiedliche Nachfragen zugeordnet. Hier wird das bikriterielle Problem betrachtet, einen Unterbaum eines Baumes zu finden, der die Einrichtungskosten und die Summe der populationsgewichteten Reisedistanzen minimiert. Das Problem wird durch Ausweitung auf einen vollständigen Graph verallgemeinert. Der gesuchte Teilbaum darf dabei weiterhin nur im ursprünglichen Baum liegen.

Ein Modell, das Knotenpaare statt einzelner Knoten betrachtet, findet man in [FM03]. Dort wird das Problem behandelt, eine Buslinie zwischen zwei gegebenen Punkten so einzurichten, dass die Haltestellen in Bezug auf das Knotenpaar Start-/Zielort vorteilhaft liegen. Modelliert wird das Problem als Suche nach einem Pfad in einem gerichteten Graph und einer ausgezeichneten Teilmenge der auf dem Pfad liegenden Knoten (Haltestellen). Als Zielfunktion werden die Kosten abzüglich des Nutzens der Buslinie minimiert. Die Kosten sind in diesem Modell gegeben durch die Kantengewichte der im Pfad enthaltenen Knoten und die Kosten für das Einrichten von Haltestellen. Für eine Menge von Reisenden werden die für einen Reisenden vorteilhaften Paare von Haltestellen und ihr Nutzen festgelegt. Der Nutzen eines Reisenden ist dann das Maximum seines Nutzens aus allen eingerichteten Haltestellen.

In [GDOM07] wird die Zielfunktion ebenfalls über Knotenpaare definiert. Es wird ein möglichst

kurzer Pfad gesucht, der eine maximale Anzahl an Knotenpaaren enthält.
Körner et al. wenden in [KMP+08] das auch in dieser Arbeit benutzte Konzept der OD-Paare und der Distanzverringerung um einen festen Faktor im eingerichteten Netzwerk an. Im Weiteren unterscheidet sich das Problem aber von den hier betrachteten. Statt in einem unterliegenden Netzwerk werden Distanzen in der Ebene gemessen. Auf einer bereits vorhandenen Bus- oder Zuglinie, auf der die Distanz um den Faktor α verkleinert ist, sollen Standorte für Haltestellen gefunden werden, so dass die mit den Nutzerzahlen gewichtete Summe der abgedeckten OD-Paare maximal ist. Dabei gilt hier ein OD-Paar als abgedeckt, wenn die Distanz zwischen den beiden Punkten des OD-Paares unter Nutzung der Buslinie nicht größer ist als die Distanz ohne Benutzung der Buslinie.

3 Voraussetzungen

3.1 Einführung

Sei $G = (V, E)$ ein ungerichteter Graph, wobei mit V die Menge der Ecken oder Knoten und mit E die Menge der Kanten bzeichnet seien. Besteht keine Verwechslungsgefahr, so wird im Folgenden für einen Knoten $v \in V$ und eine Kante $e = [i, j] \in E$ auch manchmal $v \in G$ und $[i, j] \in G$ geschrieben. $n := |V|$ bezeichne die Anzahl der Knoten. Es seien für $[i, j] \in E$ Kantenlängen $c_{ij} > 0$ gegeben. Die Länge eines Teilgraphen $g = (V_g, E_g)$ sei durch $l(g) := \sum_{[i,j] \in E_g} c_{ij}$ definiert.
Es sei OD eine Untermenge von $\{\{u, v\} : u, v \in V, u \neq v\}$. Jedem $\{u, v\}$ sei ein $w_{uv} \in \mathbb{Z}^+$ zugeordnet. OD wird Menge der Origin-Destination-Paare (OD-Paare) genannt und repräsentiert die Punkte im Netzwerk, zwischen denen eine Nachfrage besteht. w_{uv} gibt die Höhe der Nachfrage zwischen den Knoten des Knotenpaares $\{u, v\}$ an. OD-Paare und Nachfrage seien symmetrisch, das heißt $\{u, v\} = \{v, u\}$ und $w_{uv} = w_{vu}$ für alle $u, v \in K$.
Es sei ein Beschleunigungsfaktor $\alpha \in [0, 1)$ gegeben.

Definition 1 *Distanz*

1. *Sei $W_{kl} = (V_{W_{kl}}, E_{W_{kl}})$ ein kürzester Weg zwischen $k, l \in V$. Die Distanz in G zwischen k und l sei definiert durch*

$$d(k, l) = d^G(k, l) := \sum_{[i,j] \in E_{W_{kl}}} c_{ij}. \tag{1}$$

2. *Sei $g \subset G$ ein Teilgraph von G. Definiere $G_g = (V_g, E_g)$ durch $V_g = V$, $E_g = E$ mit Kantenlängen:*

$$l_{ij} := \begin{cases} \alpha * c_{ij} & \text{falls } [i, j] \in E_g \\ c_{ij} & \text{sonst.} \end{cases} \tag{2}$$

Die Distanz in G_g zwischen zwei beliebigen $k, l \in G_g$ sei nun gegeben durch

$$d_g(k, l) = d_g^G(k, l) := \sum_{[i,j] \in E_{W_{kl}^g}} l_{ij}, \tag{3}$$

wobei $W_{kl}^g = (V_{W_{kl}^g}, E_{W_{kl}^g})$ ein kürzester Weg zwischen k und l in G_g ist.

Sieht man die Distanz in G als räumliche Entfernung zweier Punkte in einem Straßennetzwerk, so lässt sich die Distanz in G_g unter der Annahme, dass sich die Reisezeit proportional

zur zurückgelegten Entfernung verhält, als Maßgröße für die Reisezeit unter Benutzung einer in g eingesetzten Beschleunigung verstehen. Beispielsweise kann die Einrichtung von g für die Einrichtung einer besseren Infrastruktur, wie etwa eines Busnetzwerks, auf den zu g gehörigen Straßen stehen. Ein OD-Paar $\{u, v\}$ repräsentiert dann den Start- und Zielort eines oder mehrerer potenzieller Kunden, w_{uk} gibt ihre Anzahl an. Die vorausgesetzte Symmetrie der OD-Paare entsteht aus der vereinfachenden Annahme, dass die Kunden auf dem Hin-und Rückweg die gleiche Strecke nutzen, also wieder zu ihrem Ausgangspunkt zurückkehren und keine „Rundreisen" unternehmen.

Bemerkung 2 *Bei der Definition der Optimierungsprobleme A und B in Kapitel 4 und 5 werden die Kantenlängen c_{ij}, die Maximallänge l_0, die maximale Distanz (in Kapitel 4) und die Nachfrage zwischen den OD-Paaren und die minimale Reisezeitersparnis (in Kapitel 5) als ganzzahlig definiert. Im Gegensatz zu der schon in obigen Definitionen geforderten Positivität der Größen, die eine wichtige Voraussetzung für die Korrektheit der angegebenen Sätze und Algorithmen ist, wird die Bedingung der Ganzzahligkeit der numerischen Größen (außer α) ausschließlich für die Gültigkeit der Sätze über pseudopolynomielle Lösbarkeit benötigt. Die angegebenen polynomiellen Algorithmen funktionieren in gleicher Weise in $\mathbb{Q}$ oder $\mathbb{R}$.*
Es sei hinzugefügt, dass in den folgenden Beweisen der NP-Vollständigkeit nur Reduktionen auf Instanzen der jeweiligen Entscheidungsprobleme mit $\alpha = 0$ durchgeführt werden. In diesen Fällen können also tatsächlich alle Zahlenwerte als ganzzahlig betrachtet werden.

3.2 Struktur

Diese Arbeit beschäftigt sich mit dem Problem für gegebene Netzwerke optimale Teilnetzwerke zu finden. In Problem A soll das Teilnetzwerk so eingerichtet werden, dass die Distanz, die ein Reisender maximal zurücklegt, minimiert wird. In Problem B wird dagegen ein Netzwerk gesucht, das die Summe der durch das Netzwerk entstehenden gesamten Reisezeitersparnis maximiert.
Beide Probleme sollen für unterschiedliche Typen von gegebenen Graphen G und gesuchten Teilgraphen g betrachtet werden. Die in folgender Tabelle unter „gegeben" aufgeführten Graphentypen stehen dabei für den unterliegenden Graph, hier werden neben allgemeinen zusammenhängenden Graphen die Vereinfachungen „Pfad", „Baum" und „Kreis" betrachtet. Dabei ist mit „Pfad" ein Kantenzug mit paarweise verschiedenen Knoten gemeint, auch bei der Darstellung des „Kreises" als Kantenzug dürfen nur Start- und Zielknoten übereinstimmen. Senkrecht sind die Arten der gesuchten Graphen aufgetragen, neben den oben genannten werden hier

als letztes auch nicht zusammenhängende Graphen betrachtet. Einige Teilprobleme, wie etwa das Finden eines optimalen Baumes in einem Pfad, sind identisch mit bereits vorher genannten Teilproblemen und müssen deshalb nicht gesondert betrachtet werden. Sie sind mit einem Gleichheitszeichen und einem Hinweis auf das schon aufgeführte Teilproblem gekennzeichnet.
Die Bezeichnung der jeweiligen Probleme erfolgt nach dem Schema:

Problem - Gegebener Graph - Gesuchter Graph.

Das Problem, den Teilpfad eines Baumes zu finden, der die maximale Distanz zwischen zwei Knoten eines Knotenpaares aus der Menge der OD-Paare minimiert, wird also mit Problem *A-Baum-Pfad* bezeichnet.
Dabei steht die Abkürzung *zGraph* für einen zusammenhängenden Graph und *Graph* für einen allgemeinen (nicht unbedingt zusammenhängenden) Graph.

g		g e g e b e n			
e		Pfad	Baum	Kreis	zGraph
s	Pfad				
u	Baum	= Pfad-Pfad		= Kreis-Pfad	
c	Kreis	nicht möglich	nicht möglich	trivial	
h	zGraph	= Pfad-Pfad	= Baum-Baum	= Kreis-Pfad	
t	Graph				

3.3 Allgemeine Aussagen

Im Folgenden wird die so genannte rekursive Darstellung von Bäumen eingeführt, die Operationen auf Bäumen, wie zum Beispiel das Finden von Wegen in Bäumen, erleichtert.

Definition 3 *[MCH81]* **Rekursive Darstellung von Bäumen**
Sei $T = (V, E)$ ein Baum mit $V = \{1, 2, \ldots, n\}$. T heißt rekursiv dargestellt, wenn jeder Knoten außer 1 adjazent zu genau einem kleineren Knoten ist. Eine kanonische rekursive Darstellung eines Baumes T ist eine lineare Sequenz C(1),C(2),...,C(n), wobei

$$C(i) := \textit{einziger zu i adjazenter Knoten, der kleiner als i ist,}$$
$$C(1) := 0.$$

Jeder rekursiv dargestellte Baum lässt sich eindeutig in der kanonischen rekursiven Darstellung darstellen.

Es folgt ein Beispiel zur kanonischen rekursiven Darstellung:

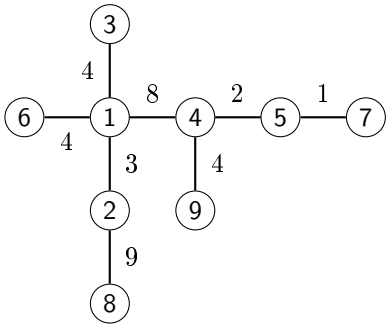

Abbildung 1: Beispiel für einen rekursiv dargestellten Baum. Die Zahlen an den Kanten repräsentieren die Kantenlängen.

Die kanonische rekursive Darstellung des in Abbildung 1 dargestellten Baumes:

Knoten i	1	2	3	4	5	6	7	8	9
Vorgängerknoten $C(i)$	0	1	1	1	4	1	5	2	4
Kantenlänge $c_{iC(i)}$	-	3	4	8	2	4	1	9	4

Lemma 4 *Sei $\tilde{T} = (\tilde{V}, \tilde{E})$ ein Baum. Die Konstruktion eines isomorphen rekursiv dargestellten Baumes T in kanonischer Darstellung aus $\tilde{T}$ kann in $O(n^2)$ vorgenommen werden.*

Beweis. Die Konstruktion von einer bijektiven Abbildung A von $\tilde{T} = (\tilde{V}, \tilde{E})$ auf einen isomorphen rekursiv dargestellten Baum $T = (V, E)$ durch Umbenennung der Knoten von $\tilde{T}$ kann folgendermaßen durchgeführt werden: Wähle im ersten Schritt ein $i_1 \in \tilde{V}$, setze $A(i_1) = 1$. Im l-ten Schritt suche (ein noch nicht betrachtetes) i_l, so dass es ein $k < l$ gibt mit $[i_k, i_l] \in \tilde{E}$. Setze $A(i_l) = i_l$. Die Kanten und Kantenlängen werden entsprechend umbenannt.
Dann gibt es für jedes $l \in E$ ein $k < l$ mit $[l, k] \in E$. Daraus folgt, dass der Baum T so beschriftet werden kann, dass für alle $m \leq n$ der von $\{1, 2, \ldots, m\}$ induzierte Teilgraph zusammenhängend, also ein Baum, ist. Gäbe es für ein l zwei Knoten $k_1, k_2 < l$ mit $[k_1, l]$, $[k_2, l] \in E$, so wäre der von $\{1, 2, \ldots, l-1\}$ erzeugte Teilgraph nicht mehr zusammenhängend, da der (im Baum eindeutige) Weg von k_1 nach k_2 nicht enthalten wäre. Also ist jeder Knoten $l \in V$ adjazent zu genau einem kleineren Knoten und es liegt eine rekursive Darstellung vor.
Die Suche nach den Knoten i_l ist in $O(n)$, da die Kantenmenge mit $n-1$ Kanten durchsucht wird. Der Vorgang wird n-mal durchgeführt. Die Konstruktion des Baumes ist also von Komplexität $O(n^2)$. ■

Beobachtung 5 *Ist T ein Pfad, so erhält man durch Auswahl eines Blattes als i_1 beim Erstellen des rekursiv dargestellten Baumes im Beweis zu Lemma 4 in $O(n^2)$ einen Pfad, dessen Knoten in aufsteigender Reihenfolge entlang des Pfades nummeriert sind.*
Für einen Kreis $K = (V, E)$ erhält man ebenfalls in $O(n^2)$ eine Nummerierung der Knoten von 1 bis n im Uhrzeigersinn, wenn man eine Kante entfernt und den entstandenen Pfad wie oben nummeriert.

Bemerkung 6 *Sei $\tilde{G} = (\tilde{V}, \tilde{E})$ mit $V = \{\tilde{1}, \tilde{2}, \ldots, \tilde{n}\}$ ein Pfad, Baum oder Kreis und $\tilde{OD}$ eine Menge von OD-Paaren in $\tilde{G}$. Sei G der durch Anwendung einer Abbildung A wie im Beweis zu Lemma 4 und Beobachtung 5 erstellte rekursiv dargestellte Baum, beziehungsweise in aufsteigender Reihenfolge nummerierte Pfad oder Kreis. Speichert man die Abbildung A als Liste mit Einträgen $\{\tilde{i}, A(\tilde{i})\}$ in aufsteigender Reihenfolge der Einträge $\tilde{i}$, so kann die Menge von OD-Paaren in G, $OD = \left\{\{A(\tilde{u}), A(\tilde{v})\} : \{\tilde{u}, \tilde{v}\} \in \tilde{OD}\right\}$, in $O(n^2)$ erstellt werden, indem für die Einträge $\{\tilde{u}, \tilde{v}\}$ von $O(n^2)$ OD-Paaren die Werte $A(\tilde{u})$ und $A(\tilde{v})$ aus der Liste abgelesen werden.*

Lemma 7 *Sei $l_0 \in \mathbb{R}$. Die Konstruktion aller bezüglich Inklusion maximalen Teilpfade mit Länge $\leq l_0$ eines Pfades oder Kreises mit aufsteigend sortierten Knoten kann in $O(n)$ durchgeführt werden.*

Beweis. Gegeben sei ein Pfad $P = (V, E)$, dessen Knoten aufsteigend von 1 bis n nummeriert sind. Sei für $i = 1, 2, \ldots, n$ $P_i = (V_i, E_i)$ der in $i \in V$ startende längstmögliche Pfad, für den noch $l(P_i) \leq l_0$ gilt. Sei $\tilde{P} = (\tilde{V}, \tilde{E})$ ein beliebiger Pfad in P. Dann startet $\tilde{P}$ ebenfalls in einem i und da P_i unter den dort startenden zulässigen Pfaden maximal war, ist $\tilde{P}$ nicht zulässig oder $\tilde{E} \subset E_i$. Damit müssen nur die Pfade P_i berechnet werden.
Bei der Konstruktion der Pfade kann man folgendermaßen vorgehen: Zur Konstruktion von P_1, setze $E_{P_1} = \emptyset$, $S = 0$. Füge zu E_{P_1} in aufsteigender Reihenfolge j_1 Kanten hinzu, bis $S = \sum_{k=1}^{j_1+1} c_k > l_0$ oder $j_1 + 1 > n$. P_1 ist dann gegeben durch $V_{P_1} = \{e_1, e_2, \ldots, e_{j_1}\}$.
Um P_i für $1 < i \leq n$ aus P_{i-1} zu konstruieren, setze $E_{P_i} = E_{P_{i-1}}$. Entferne e_{i-1} aus E_{P_i} und setze $S = S - c_{i-1}$. Füge dann zu P_i in aufsteigender Reihenfolge j_i Kanten hinzu, bis $S = \sum_{k=i}^{i+j_i+1} c_k > l_0$ oder $i + j_i + 1 > n$. Nun ist P_i gegeben durch $E_{P_i} := \{e_i, e_{i+1}, \ldots, e_{i+j_i}\}$. Ist $i + j_i = n$, so muss P_{i+l} für alle $l \geq 1$ nicht mehr konstruiert werden, da $P_{i+l} \subset P_i$. Das Verfahren bricht ab.
Sei k die Anzahl der benötigten Schritte bis $i + j_i = n$. S muss nicht in jedem Schritt $i > 1$ komplett neu berechnet werden, sondern kann durch Subtraktion von c_{i-1} und Addition von

$c_{j_{i-1}}, c_{j_{i-1}+1}, \ldots, c_{j_i+1}$ erzeugt werden. In jedem Schritt werden $j_i + 1$ Kanten hinzugefügt. Insgesamt sind also $\sum_{i=1}^{k}(j_i + 1) - 1$ Additionen nötig. Dabei ist $\sum_{i=1}^{k} j_i = n$ und $k \leq n$. Also ist

$$\sum_{i=1}^{k}(j_i + 1) - 1 < 2n. \tag{4}$$

Insgesamt werden offensichtlich $k - 1 < n$ Kanten entfernt, also $k - 1 < n$ Subtraktionen durchgeführt. Die Konstruktion der Pfade kann also in $O(n)$ durchgeführt werden, wenn die Knoten aufsteigend von 1 bis n nummeriert sind.
Zur Konstruktion aller bezüglich Inklusion maximalen Teilpfade der Länge höchstens l_0 eines Kreises $K = (V, E)$ kann man das gleiche Verfahren verwenden, indem man die Indizes der Knoten *mod* n betrachtet und die Abbruchbedingung $i + j_i = n$ durch

$$i + j_i \equiv i \ (mod\ n) \tag{5}$$

ersetzt. Wegen (5) wird der Kreis bei der Aufnahme der Kanten höchstens zweimal umlaufen, die benötigte Zeit ist also ebenfalls in $O(n)$. ■

Bemerkung 8 *Für Pfade oder Kreise mit nicht sortierten Knoten ist die Berechnung der maximalen Teilpfade mit vorgegebener Höchstlänge nach Beobachtung 5 und Lemma 7 in* $O(n^2)$.

Die rekursive Darstellung von Bäumen erlaubt eine einfache algorithmische Bestimmung von Wegen zwischen Kanten in Bäumen:

Algorithmus 1 Wege in Bäumen

Eingabe: Rekursiv dargestellter Baum $T = (V, E)$ mit Kantenlängen $c_{ij}\ \forall [i, j] \in E$; $k_1, l_1 \in V$.

Ausgabe: Weg W zwischen k_1 und l_1 durch Angabe einer Knotensequenz.

1: **for** $i = 1, \ldots, n$ **do**
2: **if** $k_i = l_i$ **then**
3: Setze $j = i$. Gehe zu Schritt 11.
4: **end if**
5: **if** $k_i > l_i$ **then**
6: $k_{i+1} = C(k_i);\ l_{i+1} = l_i$
7: **else**
8: $k_{i+1} = k_i;\ l_{i+1} = C(l_i)$
9: **end if**
10: **end for**
11: $W = \{k_1, k_2, \ldots, k_j, l_j, l_{j-1}, \ldots, l_1\}$
12: Entferne doppelte Knoten aus W.
13: Gebe W aus.

Lemma 9 *Algorithmus „Wege in Bäumen“ bestimmt in rekursiv dargestellten Bäumen in $O(n)$ einen Weg zwischen zwei Knoten.*

Beweis. Für jeden Knoten v eines rekursiv dargestellten Baumes ist $C(v)$ adjazent zu v. Da die Knotenfolge $v, C(v), C(C(v)), \ldots$ für alle $v \in V$ nach spätestens n Schritten mit $C(C(\ldots C(v)\ldots)) = 1$ endet, gibt es ein $j \leq n$ so dass $k_j = l_j$. Also stellt die ausgegebene Knotensequenz die Knotenmenge eines Weges dar. Wegen der absteigenden Aufzählung und der Überprüfung in Schritt 2 sind für alle $g, h < j$ die k_g und l_h paarweise verschieden. In W sind die k_g's absteigend, die l_h's aufsteigend angeordnet. Beim Entfernen der Knoten in Schritt 12 muss also nur überprüft werden, ob in $W = \{v_1, v_2, \ldots, v_{2j}\}$ $v_i = v_{i+1}$ und in dem Fall v_i entfernt werden. Diese Überprüfung ist also auch in $O(n)$. ■

4 Problem A: Minimiere maximale Reisezeit

Sei $G = (V, E)$ ein Graph mit Kantenlängen $c_{ij} \in \mathbb{Z}^+$ für alle $[i, j] \in E$, OD eine Menge von OD-Paaren in G und l_0 eine positive ganze Zahl.
Gesucht ist ein Teilgraph g der Länge höchstens l_0, der die maximale Distanz in G_g minimiert.

$$\min_{g, l(g) \leq l_0} \max_{\{u,v\} \in OD} d_g(u, v) \tag{6}$$

Interpretiert man G als Straßennetzwerk, OD als die Menge von Start- und Zielorten möglicher Nutzer sowie α als Beschleunigung, die sich durch Benutzen eines Busses gegenüber der vorherigen Bewegungsgeschwindigkeit ergibt, so lässt sich die Suche nach g als die Suche nach einem Busnetzwerk sehen, das die maximale Reisezeit eines Nutzers minimiert.

Definition 10 Entscheidungsproblem zu Problem A

- *Instanz* $(G, OD, \alpha, l_0, d_0)$*:*
  Ungerichteter Graph $G = (V, E)$ *mit Kantenlängen* $c_{ij} \in \mathbb{Z}^+$ *für* $[i, j] \in E$*,*
  Menge von OD-Paaren OD*,*
  Beschleunigungsfaktor $\alpha \in [0, 1)$*,*
  Maximallänge des gesuchten Teilgraphen $l_0 \in \mathbb{Z}^+$*,*
  Maximale Distanz in G_g *zwischen zwei OD-Paaren* $d_0 \in \mathbb{Z}_0^+$
- *Frage:*
  Gibt es einen Teilgraph $g \subset G$*, so dass:*

$$l(g) \leq l_0 \tag{7}$$

$$d_g(u, v) \leq d_0 \qquad \forall\ \{u, v\} \in OD\ ? \tag{8}$$

Satz 11 *Das Entscheidungsproblem A ist in NP. Dies gilt unabhängig vom gegebenen Graph G und der gewünschten Struktur des gesuchten Graphen g.*

Beweis. Sei $(G, OD, \alpha, l_0, d_0)$ eine Instanz von Entscheidungsproblem A. Für einen gegebenen Teilgraph $g \subset G$ ist es möglich in Polynomialzeit zu überprüfen, ob g die gewünschte Struktur hat (zusammenhängend, Pfad, Kreis oder Baum), ob $l(g) \leq l_0$ (durch Addition der Kantenlängen der enthaltenen Kanten) und ob $d_g(u, v) \leq d_0\ \forall\ \{u, v\} \in OD$ (durch Bestimmung der kürzesten Wege zwischen den OD-Paaren). Also lässt sich in Polynomialzeit entscheiden, ob g eine Lösung des Entscheidungsproblems ist.
$\Rightarrow$ Problem A ist in NP. ■

Lemma 12 *Sei G ein Graph und α ein Beschleunigungsfaktor. g und g' seien zwei Teilgraphen von G. Gilt $g' = \emptyset$ oder $g' \subset g$, dann ist $\forall u, v \in V$*

$$d_g(u,v) \leq d_{g'}(u,v) \tag{9}$$

Beweis. Seien für $[i,j] \in E$ die Kantenlängen in G_g gegeben durch l_{ij}, die Kantenlängen in $G_{g'}$ durch l'_{ij}. Sei $W = (V_W, E_W)$ ein kürzester Weg zwischen u und v in G_g und $W' = (V_{W'}, E_{W'})$ ein kürzester Weg zwischen u und v in $G_{g'}$. Da W ein kürzester Weg in G_g ist, ist

$$\sum_{[i,j] \in E_{W'}} l_{ij} \geq \sum_{[i,j]) \in E_W} l_{ij}. \tag{10}$$

Außerdem gilt für alle $[i,j] \in E$ $l'_{ij} \geq l_{ij}$, da $g' \subset g$. Daraus folgt:

$$\begin{aligned} d_{g'}(u,v) &= \sum_{[i,j] \in W'} l'_{ij} \\ &\geq \sum_{[i,j] \in W} l_{ij} \\ &= d_g(u,v). \end{aligned} \tag{11}$$

■

Definition 13 Menge der maximalen OD-Paare
Sei $T = (V,E)$ ein Baum. Sei für $i,j \in V$ (i,j) der eindeutig bestimmte Weg zwischen i und j in T. Dann ist die Menge der maximalen OD-Paare OD_{max} definiert durch

$$OD_{max} := \{\{u,v\} \in OD : (u,v) \text{ ist maximal bezüglich Inklusion}\}.$$

Lemma 14 *Beim Lösen von Problem A in Bäumen als zugrunde liegende Graphen müssen nur die maximalen OD-Paare betrachtet werden.*

Beweis. Seien $\{u_1,v_1\}, \{u_2,v_2\} \in OD$ mit $(u_1,v_1) \subset (u_2,v_2)$, dann gilt für alle Teilgraphen $g \subset T$:

$$d_g(u_1,v_1) \leq d_g(u_2,v_2) \tag{12}$$

$\Rightarrow$

$$\max_{\{u,v\} \in OD} d_g(u,v) = \max_{\{u,v\} \in OD_{max}} d_g(u,v) \tag{13}$$

$\Rightarrow$

$$\min_{g, l(g) \leq l_0} \max_{\{u,v\} \in OD} d_g(u,v) = \min_{g, l(g) \leq l_0} \max_{\{u,v\} \in OD_{max}} d_g(u,v) \tag{14}$$

■

4.1 Bestimmung optimaler Teilpfade

4.1.1 A-Pfad-Pfad

Lemma 15 *Das Problem A-Pfad-Pfad ist in $O(n^2)$.*

Beweis. Gegeben sei ein Pfad $P = (V, E)$. Unter den Teilpfaden von P der Länge $\leq l_0$ reicht es, die bezüglich Inklusion maximalen Teilpfade P' auf Optimalität zu überprüfen (Lemma 12). Laut Lemma 14 ist

$$\max_{\{u,v\} \in OD} d_{P'}(u, v) = \max_{\{u,v\} \in OD_{max}} d_{P'}(u, v). \tag{15}$$

Durch Berechnung von $\max_{\{u,v\} \in OD_{max}} d_{P'}(u, v)$ für alle bezüglich Inklusion maximalen Pfade der Länge $\leq l_0$ erhält man also den gesuchten Pfad $\hat{P}$.
Der Aufwand für die Bestimmung des optimalen Pfades $\hat{P}$ setzt sich nun folgendermaßen zusammen: Nach Beobachtung 5 und Bemerkung 6 erhält man aus jedem Pfad P und jeder Menge von OD-Paaren in P in $O(n^2)$ einen Pfad mit Knoten in aufsteigender Reihenfolge und die dazugehörigen OD-Paare. Ohne Beschränkung der Allgemeinheit sei V schon in aufsteigender Reihenfolge nummeriert. Die Berechnung der Pfade P' der Länge $\leq l_0$, die bezüglich Inklusion maximal sind, ist laut Lemma 7 in $O(n)$. Für jedes $u \in V$ ist das OD-Paar $\{u, v\}$ mit $v := \max\{w : \{u, w\} \in OD\}$ maximal bezüglich Inklusion. Die Berechnung von $|OD_{max}|$ ist also in $O(n^2)$, $|OD_{max}|$ selbst aber nur in $O(n)$. Um für festes P' $\max_{\{u,v\} \in OD_{max}} d_{P'}(u, v)$ zu berechnen, kann man ähnlich wie im Beweis zu Lemma 7 vorgehen:
Sei anfänglich $S_1 = 0$. Gibt es ein $v_1 \in E$ so dass $\{1, v_1\} \in OD_{max}$, so setze man

$$S_1 = d_{P'}(1, v_1) = \sum_{j=1}^{v_1 - 1} l_{jj+1}. \tag{16}$$

Für $2 \leq i < n$ sei im i-ten Schritt zunächst

$$S_i = \max\{S_{i-1} - l_{i-1i}, 0\}. \tag{17}$$

Gibt es ein $v_i \in V$ mit $\{i, v_i\} \in OD_{max}$, so setze man

$$S_i = S_i + \sum_{j=v_{i-1}}^{v_i - 1} l_{jj+1}. \tag{18}$$

Die Berechnung von $\max_{\{u,v\} \in OD_{max}} d_{P'}(u, v)$ ist für festes P' also in $O(n)$. Da es $O(n)$ maximale Pfade P' gibt, ist die Bestimmung von $\hat{P}$ in $O(n^2)$. ■

4.1.2 A-Baum-Pfad

Satz 16 *Das Problem A-Baum-Pfad ist in* $O(n^5)$.

Beweis. In $O(n^2)$ lässt sich aus jeder Instanz $(\tilde{T}, \tilde{OD}, \alpha, l_0, d_0)$ von *A-Baum-Pfad* eine Instanz $(T, OD, \alpha, l_0, d_0)$ mit rekursiv dargestelltem Baum T und dazugehörigen OD-Paaren OD erstellen (Lemma 4 und Bemerkung 6). Sei $T = (V, E)$ ein rekursiv dargestellter Baum. Da es zwischen zwei unterschiedlichen Knoten genau einen Pfad gibt, gibt es $O(n^2)$ Pfade in T. Berechnet man für jeden Pfad P' und jedes der $O(n^2)$ OD-Paare $\{u, v\}$ in $O(n)$ die Distanz $d_{P'}(u, v)$ (Lemma 9), so erhält man durch anschließenden Vergleich von $\max_{\{u,v\} \in OD} d_{P'}(u, v)$ den optimalen Teilpfad in $O(n^5)$. ■

4.1.3 A-Kreis-Pfad

Satz 17 *Das Problem A-Kreis-Pfad ist in* $O(n^3)$.

Beweis. Sei $K = (V, E)$ ein Kreis, dessen Knoten ohne Beschränkung der Allgemeinheit im Uhrzeigersinn von 1 bis n nummeriert seien und OD eine Menge von OD-Paaren in K (vergleiche Beobachtung 5 und Bemerkung 6). Für jeden Knoten $i \in V$ sei P_i der in i startende längstmögliche Pfad (so dass $l(P_i) \leq l_0$) im Uhrzeigersinn. Es reicht nach Lemma 12, nur diese Pfade P_i zu betrachten, da jeder andere Pfad P in einem dieser Pfade enthalten ist. Nach Lemma 7 ist die Konstruktion dieser Pfade in $O(n)$. Auch hier kann man ähnlich wie im Beweis zu Satz 15 maximale OD-Paare definieren, diese sind wegen der Nichteindeutigkeit der Wege aber vom jeweils betrachteten Pfad P_i abhängig. Für festes P_i sei $x_i = \frac{l(K_{P_i})}{2}$. Man bemerke, dass der kürzeste Weg zwischen zwei Knoten in K_{P_i} immer $\leq x_i$ ist.
Wähle für jeden Knoten h einen Knoten k_h, so dass:

$$\sum_{j=h}^{k_h-1} l_{jj+1}^{P_i} \leq x_i,\ \sum_{j=h}^{k_h} l_{jj+1}^{P_i} > x_i, \qquad falls\ \sum_{j=h}^{n} l_{jj+1}^{P_i} > x_i,$$

$$\sum_{j=h}^{n} l_{jj+1}^{P_i} + \sum_{j=1}^{k_h-1} l_{jj+1}^{P_i} \leq x_i,\ \sum_{j=h}^{n} l_{jj+1}^{P_i} + \sum_{j=1}^{k_h} l_{jj+1}^{P_i} > x_i, \qquad falls\ \sum_{j=h}^{n} l_{jj+1}^{P_i} + l_{12} > x_i$$

und

$$k_h = n, \qquad falls\ \sum_{j=h}^{n} l_{jj+1}^{P_i} \leq x_i,\ \sum_{j=h}^{n} l_{jj+1}^{P_i} + l_{12} > x_i.$$

Dann ist das im Uhrzeigersinn laufende, in h startende, maximale OD-Paar gegeben durch $\{h, v_h\}$, wobei

$$v_h = \begin{cases} \max\{w : \{h,w\} \in OD;\ w \le k_h\}, falls\ k_h \ge h \\ \max\big(\{w : \{h,w\} \in OD;\ h < w \le n\} \cup \{w+n : \{h,w\} \in OD;\ w < k_h\}\big)\ sonst. \end{cases}$$

Die Bestimmung der im Uhrzeigersinn verlaufenden maximalen OD-Paare und ihrer Länge kann also in $O(n^2)$ erfolgen. Analog berechne man in $O(n^2)$ die gegen den Uhrzeigersinn verlaufenden maximalen OD-Paare und ihre Länge. So kann man für gegebenen Pfad P_i in $O(n^2)$ $\max_{\{u,v\} \in OD} d_{P_i}(u,v)$ bestimmen. Da es $O(n)$ Pfade P_i gibt, ist die Berechnung einer optimalen Lösung von *A-Kreis-Pfad* also in $O(n^3)$. ■

4.1.4 A-zGraph-Pfad

Satz 18 *Das Entscheidungsproblem A-zGraph-Pfad ist stark NP-vollständig.*

Die starke NP-Vollständigkeit des Entscheidungsproblems *A-zGraph-Baum* wird durch Reduktion des Problems HAMILTONIAN PATH gezeigt. Zuerst werden dazu die folgenden Begriffe eingeführt:

Definition 19 *[GJ79, 60] Hamiltonscher Pfad*
Sei $G = (V,E)$ ein zusammenhängender, ungerichteter Graph. Ein Pfad $P = (V_P, E_P)$, der jeden Knoten genau einmal besucht, wird Hamiltonscher Pfad genannt.

Definition 20 *[GJ79, 199f.]* **HAMILTONIAN PATH**

- *Instanz ($\tilde{G}$):*
  Ungerichteter Graph $\tilde{G} = (\tilde{V}, \tilde{E})$
- *Frage:*
  Gibt es einen Hamiltonschen Pfad in $\tilde{G}$?

Satz 21 *[GJ79, 199f.] HAMILTONIAN PATH ist NP-vollständig.*

Beweis. (von Satz 18)

Sei ($\tilde{G}$) mit $\tilde{G} = (\tilde{V}, \tilde{E})$ eine Instanz von HAMILTONIAN PATH.
Dazu definiere man den Graph $G = (V,E)$. Dabei seien mit $\tilde{V} = \{1, 2, \ldots, n\}$ V und E gegeben durch $V := \tilde{V} \cup \{1', 2', \ldots, n'\}$ und $E := \tilde{E} \cup \bigcup_{i \in V} \{[i,i']\}$. Die Länge der Kanten sei $c_{ij} = 1$ für alle $[i,j] \in \tilde{E}$ und $c_{ii'} = n^2 + 1$ für alle $i \in \tilde{V}$ und $i' \in V'$. Sei $OD := \{\{i', j'\} : i', j' \in V'\}$.
Sei $\alpha := 0$, $l_0 := n^2$ und $d_0 := 2(n^2+1)$.
Keiner der numerischen Einträge in $I = (T, OD, 0, n^2, 2(n^2+1))$ ist also größer als $2(n^2+1)$.

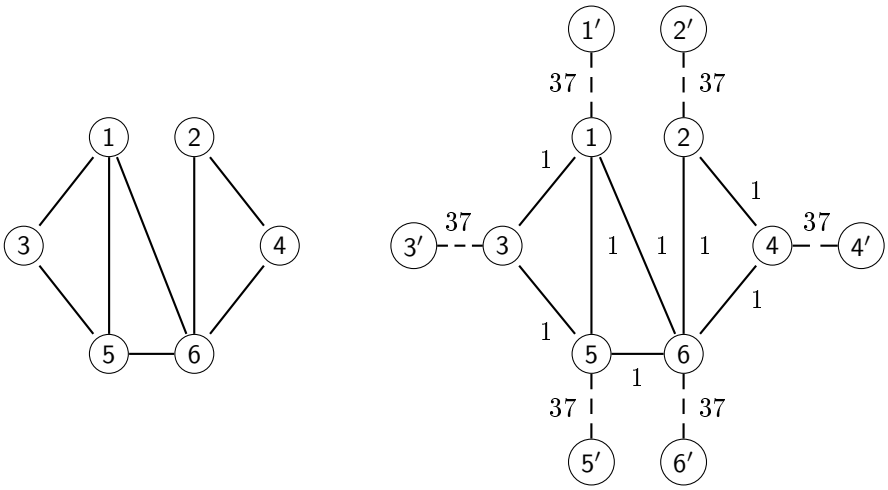

Abbildung 2: Beispiel für die Konstruktion von G aus $\tilde{G}$

Behauptung 22 *Es gibt genau dann einen Hamiltonschen Pfad in $\tilde{G}$, wenn es eine Lösung der Instanz $(G, OD, 0, n^2, 2(n^2+1))$ des Entscheidungsproblems A-zGraph-Pfad gibt.*

Sei $P = (V_P, E_P)$ eine Lösung von $(G, OD, 0, n^2, 2(n^2+1))$. Dann ist P nach Definition ein Pfad und $l(P) \leq n^2$. Außerdem gilt für alle $\{i', j'\} \in OD$

$$d_P(i', j') \leq 2(n^2+1). \tag{19}$$

Angenommen P ist kein Hamiltonscher Pfad in $\tilde{G}$. Dann gibt es ein $i \in \tilde{V}$, für das gilt $i \notin V_P$. In diesem Fall betrachte man das OD-Paar $\{i', j'\}$ für beliebiges $j' \in V'$, $j' \neq i'$. Es ist $[j, j'] \notin E_P$, da sonst $l(P) \geq c_{jj'} = n^2 + 1 > n^2$.
Aus $i \notin V_P$ und $c_{ab} > 0 \; \forall \; [a, b] \in \tilde{E}$ folgt dann $d_P(i, j) > 0$ und

$$\begin{aligned} d_P(i', j') &= d_P(i', i) + d_P(i, j) + d_P(j, j') \\ &> d(i', i) + d(j, j') \\ &= n^2 + 1 + 0 + n^2 + 1 \\ &= 2(n^2+1). \end{aligned} \tag{20}$$

Das ist ein Widerspruch zu (19). Also ist P ein Hamiltonscher Pfad der Länge $l(P) \leq n^2$ in $\tilde{G}$. Sei nun P eine Lösung der Instanz $(\tilde{G})$ von HAMILTONIAN PATH. Dann ist P ein Pfad, $P \subset G$ und wenn man P in G betrachtet, gilt $l(P) \leq \sum_{[i,j] \in \tilde{E}} c_{ij} = \sum_{[i,j] \in \tilde{E}} 1 \leq n^2$. Außerdem ist $i \in E_P \; \forall i \in \tilde{V}$.

Betrachtet man P in G, so gilt für alle OD-Paare $\{i', j'\}$

$$\begin{aligned} d_P(i', j') &= d_P(i', i) + d_P(i, j) + d_P(i, j') \\ &= n^2 + 1 + 0 + n^2 + 1 \\ &= 2(n^2 + 1). \end{aligned} \tag{21}$$

Daher ist P eine Lösung für $(G, OD, 0, n^2, 2(n^2 + 1))$.
Der Pfad P ist also genau dann Lösung der Instanz $(\tilde{G})$ von HAMILTONIAN PATH, wenn er Lösung der Instanz $(G, OD, 0, n^2, 2(n^2 + 1))$ von *A-zGraph-Pfad* ist.
Damit lässt sich das Problem HAMILTONIAN PATH auf ein Teilproblem von *A-zGraph-Pfad* mit Instanzen I reduzieren, deren numerischen Einträge durch $2(n^2 + 1)$ beschränkt sind.
$\Rightarrow$ *A-zGraph-Pfad* ist stark NP-vollständig. ■

4.2 Bestimmung optimaler Teilbäume

4.2.1 A-Baum-Baum

Satz 23 *Das Entscheidungsproblem A-Baum-Baum ist stark NP-vollständig.*

Der Beweis erfolgt durch Reduktion eines Spezialfalls des Entscheidungsproblems HITTING SET.

Definition 24 *[GJ79, 222]* **HITTING SET**

- *Instanz* (S, C, K)*:*
  Endliche Menge S*,*
  Menge von Teilmengen C*,*
  Natürliche Zahl $K \leq |S|$
- *Frage:*
  Gibt es eine Teilmenge $S' \subset S$*, so dass* $|S'| \leq K$ *und* $\forall c \in C$ *gilt* $c \cap S' \neq \emptyset$ *?*

Satz 25 *[GJ79, 222] Das Entscheidungsproblem HITTING SET ist NP-vollständig. Dies gilt auch, falls* $|c| \leq 2\ \forall c \in C$.

Beweis. (von Satz 23)
Ist $\emptyset \in C$, so gibt es offensichtlich keine Teilmenge D' mit den geforderten Eigenschaften, da $\emptyset \cap S' = \emptyset$. Im Folgenden wird deshalb dieser Fall ausgeschlossen.
Sei (S, C, K) eine Instanz von HITTING SET mit $S = \{s_1, s_2, \ldots, s_n\}$ und $|c| \leq 2\ \forall c \in C$.

Sei $S_1 := \{s_i \in S : \{s_i\} \in C\}$ und $S_2 := S \setminus S_1$. Der zugehörige Baum $T = (V, E)$ sei gegeben durch $V := \{i : s_i \in S_2\} \cup \{0\}$ und $E := \{[0, i] : s_i \in S_2\}$ mit Kantenlängen $c_{0i} := 1 \; \forall \; [0, i] \in E$. Sei $OD := \{\{i, j\} : s_i, s_j \in S_2; \{s_i, s_j\} \in C\}$. Sei $\alpha := 0$, $l_0 := K - |S_1|$ und $d_0 := 1$.
Da für $K \geq |S|$ HITTING SET die triviale Lösung $S' = S$ hat, kann man $K < |S|$ annehmen. Damit ist $l_0 = K - |S_1| < K < |S|$ und keiner der numerischen Einträge in der Instanz $(T, OD, 0, K - |S_1|, 1)$ von *A-Baum-Baum* ist größer als $|S|$.

$$S = \{s_1, s_2, s_3, s_4, s_5, s_6, s_6, s_7, s_8, s_9, s_{10}\}$$

$$C = \{\{s_1\}\{s_5\}, \{s_6\}, \{s_{10}\}, \{s_1, s_3\}, \{s_1, s_{10}\}, \{s_2, s_3\}, \{s_4, s_5\}\{s_4, s_8\}, \{s_7, s_8\}, \{s_8, s_9\}\{s_8, s_{10}\}\}$$

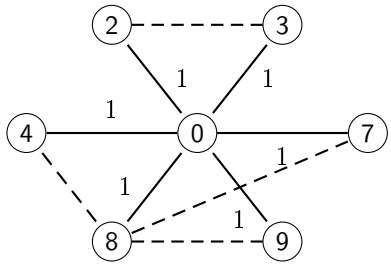

Abbildung 3: Beispiel zur Erstellung von T und OD aus (S, C, K). Die durchgezogenen Linien bilden T, die gestrichelten Linien verbinden die OD-Paare.

Behauptung 26 *Die Instanz (S, C, K) von HITTING SET hat genau dann eine Lösung, wenn die Instanz $(T, OD, 0, K - |S_1|, 1)$ von Problem A-Baum-Baum eine Lösung hat.*

Sei S' eine Lösung von (S, C, K). Dann gilt $\forall c \in C$:

$$S' \cap c \neq \emptyset \tag{22}$$

und

$$|S'| \leq K. \tag{23}$$

Es ist $S_1 \subset S'$, denn falls $\exists s_i \in S_1, \; s_i \notin S'$, so folgt $S' \cap \{s_i\} = \emptyset$. Dies ist ein Widerspruch zur Definition von S_1 und der Bedingung $S' \cap c \neq \emptyset \; \forall c \in C$.
Sei $T' := (V_{T'}, E_{T'})$ gegeben durch $V_{T'} := \{i : s_i \in S' \setminus S_1\} \cup \{0\}$, $E_{T'} := \{[0, i] : s_i \in S' \setminus S_1\}$.

Dann ist T' offensichtlich zusammenhängend und somit ein Baum. Des Weiteren gilt:

$$\begin{aligned} l(T') = & \sum_{[v_0,v_i]\in E_{T'}} 1 \\ =& |S' \setminus S_1| \\ =& |S'| - |S' \cap S_1| \\ =& |S'| - |S_1| \\ \leq& K - |S_1|. \end{aligned} \tag{24}$$

Angenommen $\exists \{u, v\} \in OD$ mit $d_{T'}(u, v) > 1$, also

$$1 < d_{T'}(u, v) = l_{0u} + l_{0v} \tag{25}$$

mit

$$l_{0k} = \begin{cases} \alpha c_{0k} & \text{falls } [0, v_k] \in E_{T'} \\ c_{0k} & \text{sonst} \end{cases} = \begin{cases} 0 & \text{falls } [0, v_k] \in E_{T'} \\ 1 & \text{sonst.} \end{cases} \tag{26}$$

Dann folgt daraus $[0, u]$, $[0, v] \notin E_{T'}$.

$\Rightarrow s_u, s_v \notin S'$

$\Rightarrow \{s_u, s_v\} \notin C$, da sonst $\{s_u, s_v\} \cap S' = \emptyset$ im Widerspruch zu (22) steht.

$\Rightarrow \{u, v\} \notin OD$ (nach Konstruktion von OD).

Dies ist ein Widerspruch zur Annahme.

Also ist $d_{T'}(u, v) \leq 1 \; \forall \; \{u, v\} \in OD$. Daraus folgt, dass T' Lösung von $(T, OD, 0, K - |S_1|, 1)$ ist.

Sei nun T' eine Lösung von $(T, OD, 0, K - |S_1|, 1)$. Dann gilt:

$$l(T') \leq K - |S_1| \tag{27}$$

und $\forall \{u, v\} \in OD$

$$1 \leq d_{T'}(u, v) = l_{0u} + l_{0v}. \tag{28}$$

Die dazugehörige Menge S' sei gegeben durch $S' := S_1 \cup \{s_i \in S : [0,i] \in E_{T'}\}$. Dann ist

$$\begin{aligned}
|S'| =& |S_1| + |\{s_i \in S : [0,i] \in E_{T'}\}| - |S_1 \cap \{s_i \in S : [0,] \in E_{T'}\}| \\
\leq& |S_1| + |\{s_i \in S : [0,i] \in E_{T'}\}| \\
=& |S_1| + \sum_{e \in E_{T'}} 1 \\
=& |S_1| + l(T') \\
\leq& |S_1| + K - |S_1| \\
=& K
\end{aligned} \tag{29}$$

Da $S_1 \subset S'$, gilt für alle $c \in C$ mit $|c| = 1$, dass $c \cap S' \neq \emptyset$. Angenommen es gibt nun ein $c = \{s_i, s_j\} \in C$ mit $|c| = 2$ und $c \cap S' = \emptyset$.
$\Rightarrow s_i \notin S',\ s_j \notin S'$.
Nach Konstruktion von S' aus T' folgt $[0,i] \notin E_T,\ [0,j] \notin E_T$.
$\Rightarrow l_{0i} = c_{0i} = 1$ und $l_{0j} = c_{0j} = 1$.
$\Rightarrow d_T(i,j) = l_{0i} + l_{0j} = 2 > 1$.
Das steht im Widerspruch zu (28), denn nach Konstruktion von OD ist $\{i,j\} \in OD$. Also ist $c \cap S' \neq \emptyset\ \forall c \in C$. Daraus folgt, dass S' Lösung von (S,C,K) ist.
Die Instanz (S,C,K) von HITTING SET hat also genau dann eine Lösung, wenn die (durch $|S|$ in der Größe der numerischen Einträge beschränkte) Instanz $(T, OD, 0, K-|S_1|, 1)$ von Problem *A-Baum-Baum* eine Lösung hat.
$\Rightarrow$ Das Entscheidungsproblem *A-Baum-Baum* ist stark NP-vollständig. ■

4.2.2 A-zGraph-Baum

Satz 27 *Das Entscheidungsproblem A-zGraph-Baum ist stark NP-vollständig.*

Dieser Satz folgt unmittelbar aus Lemma 23, da schon das Unterproblem *A-Baum-Baum* stark NP-vollständig ist. Alternativ ist auch eine Reduktion zum Steinerbaumproblem möglich.

4.3 Bestimmung optimaler Kreise

Wie man auch der Tabelle in Abschnitt 3.2 entnehmen kann, ist das einzige betrachtenswerte Unterproblem von Problem A, bei dem ein Kreis gesucht wird, das Problem *A-zGraph-Kreis*.

Satz 28 *Das Entscheidungsproblem A-zGraph-Kreis ist stark NP-vollständig.*

Der Beweis erfolgt analog zum Beweis der NP-Vollständigkeit von *A-zGraph-Pfad*. Statt des Problems HAMILTONIAN PATH betrachte man hier HAMILTONIAN CIRCUIT:

Definition 29 *[GJ79, 35]* **Hamiltonscher Kreis**
Sei $G = (V, E)$ ein zusammenhängender Graph. Ein Kreis $K = (V_K, E_K)$, der jeden Knoten genau einmal besucht, wird Hamiltonscher Kreis genannt.

Definition 30 *[GJ79, 199]* **HAMILTONIAN CIRCUIT**

- *Instanz (G):*
  Ungerichteter Graph $\tilde{G} = (\tilde{V}, \tilde{E})$
- *Frage:*
  Gibt es einen Hamiltonschen Kreis in $\tilde{G}$?

Satz 31 *[GJ79, 199]*
Das Entscheidungsproblem HAMILTONIAN CIRCUIT ist NP-vollständig.

Beweis. (von Satz 28)
Sei $(\tilde{G})$ eine Instanz von HAMILTONIAN CIRCUIT mit $\tilde{G} = (\tilde{V}, \tilde{E})$, wobei $\tilde{V} = \{1, 2, \ldots, n\}$. Mit $V_{T'} := \{1', 2', \ldots, n'\}$ seien $V := \tilde{V} \cup V_{T'}$ und $E := \tilde{E} \cup \left\{[i, i'] : i \in \tilde{V}\right\}$ mit $c_{ij} := 1$ für $[i, j] \in \tilde{E}$ und $c_{ii'} = n^2 + 1$ für $i \in V,\ i' \in \tilde{V}$ die Knoten- und Kantenmengen von G.
Die Menge der OD-Paare sei gegeben durch $OD := \{\{i', j'\} : i', j' \in V'\}$.
Sei $\alpha := 0$, $l_0 := n^2$ und $d_0 := 2(n^2 + 1)$. Man erhält also eine Instanz $I = (T, OD, 0, l_0, d_0)$ von *A-zGraph-Kreis*, deren Eingabelänge polynomiell in n beschränkt ist.

Behauptung 32 *Es gibt genau dann eine Lösung der Instanz $(\tilde{G})$ von HAMILTONIAN CIRCUIT, wenn es eine Lösung der Instanz $(G, OD, 0, n^2, 2(n^2 + 1))$ von A-zGraph-Kreis gibt.*

Sei K eine Lösung von $(G, OD, 0, n^2, 2(n^2 + 1))$. Dann ist K ein Kreis und

$$l(K) \leq n^2. \tag{30}$$

Außerdem gilt

$$d(i', j') \leq 2(n^2 + 1) \tag{31}$$

für alle OD-Paare $\{i', j'\}$.
Für beliebiges $j' \in V'$ ist $[j, j'] \notin E_K$, weil sonst

$$l(K) \geq c_{jj'} = n^2 + 1 > n^2 \tag{32}$$

im Widerspruch zu (30).
Angenommen es existiert $i \in \tilde{V}, i \notin V_K$. Dann stände

$$\begin{aligned} d(i',j') =& d(i',i) + d(i,j) + d(j,j') \\ >& n^2 + 1 + 0 + n^2 + 1 \\ =& 2(n^2+1) \end{aligned} \tag{33}$$

im Widerspruch zu (31), da $i \notin T$ und deshalb $d(i,j) > 0$. Also ist K ein Kreis in $\tilde{G}$, der alle Knoten aus $\tilde{V}$ enthält, das heißt K löst die Instanz $(\tilde{G})$ von HAMILTONIAN CIRCUIT.
Sei nun K eine Lösung von $(\tilde{G})$. Man betrachte K in G:
K ist ein Kreis und $l(K) \leq \sum_{[i,j]\in\tilde{E}} c_{ij} = \sum_{[i,j]\in\tilde{E}} 1 < n^2$. Außerdem gilt:

$$\begin{aligned} d_K(i',j') &= d_K(i',i) + d_K(i,j) + d_K(j,j') \\ &= d(i',i) + d_K(i,j) + d(j,j') \\ &= n^2 + 1 + 0 + n^2 + 1 \\ &= 2(n^2+1), \end{aligned} \tag{34}$$

da $d_K(i,j) = 0 \;\forall\; i,j \in V_K = \tilde{V}$.
Also ist K auch Lösung für $(G, OD, 0, n^2, 2(n^2+1))$ von *A-zGraph-Kreis*. $\Rightarrow$ Der Kreis K ist genau dann Lösung von der Instanz (G) von HAMILTONIAN CIRCUIT, wenn er Lösung der Instanz $(G, OD, 0, l(\tilde{G}), 2M)$ von *A-zGraph-Kreis* ist. Das Entscheidungsproblem HAMILTONIAN CIRCUIT lässt sich also auf Instanzen I von *A-zGraph-Kreis* mit polynomiell in n beschränkten Eingabewerten reduzieren.
$\Rightarrow$ Das Entscheidungsproblem *A-zGraph-Kreis* ist stark NP-vollständig. ■

4.4 Bestimmung optimaler zusammenhängender Teilgraphen

Auch in diesem Fall kann man sich auf die Suche in einem zusammenhängenden Graph beschränken, da alle anderen Fälle identisch mit den schon genannten Unterproblemen sind (siehe Abschnitt 3.2).
Da die Suche nach einem Teilbaum in einem Baum der Suche nach einem zusammenhängenden Graph in einem Baum entspricht und ein Baum ein Spezialfall eines zusammenhängenden Graphen ist, folgt aus Satz 23 die starke NP-Vollständigkeit von *A-zGraph-zGraph*.

Satz 33 *Das Entscheidungsproblem A-zGraph-zGraph ist NP-vollständig.*

4.5 Bestimmung optimaler allgemeiner Teilgraphen

4.5.1 A-Pfad-Graph

Satz 34 *Das Entscheidungsproblem A-Pfad-Graph ist NP-vollständig.*

Der Beweis erfolgt durch Reduktion des im Folgenden definierten Problems SUBSET SUM auf *A-Pfad-Graph*.

Definition 35 *[GJ79, 223]* **SUBSET SUM**

- *Instanz* (A, B)*:*
  Endliche Menge $A = \{a_1, a_2, \ldots, a_n\}$,
  jedem Element ist eine Größe $s(a) \in \mathbb{Z}^+$ *zugeordnet,*
  Ganze Zahl $B \in \mathbb{Z}^+$
- *Frage:*
  Gibt es eine Teilmenge $A' \subseteq A$*, so dass* $\sum_{a_j \in A'} a_j = B$*?*

Satz 36 *[GJ79, 223]*
SUBSET SUM ist NP-vollständig.

Beweis. (von Satz 34)
Sei (A, B) mit $A = \{a_1, a_2, \ldots, a_n\}$ eine Instanz von SUBSET SUM. Der Pfad $P = (V, E)$ sei dann gegeben durch $V := \{i : i = 0, \ldots, n\}$ und $E := \{[i-1, i] : i = 1, \ldots, n\}$ mit Kantenlängen $c_{i-1i} := s(a_i)$. Das einzige OD-Paar sei $\{1, n\}$. Außerdem sei $\alpha := 0$, $l_0 := B$ und $d_0 := \sum_{i=1}^{n} a_i - B$.

Elemente a_i aus A	a_1	a_2	a_3	a_4	a_5
$s(a_i)$	3	1	1	4	2

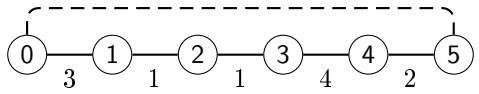

Abbildung 4: Beispiel für die Konstruktion von P aus A. Die durchgezogenen Linien stellen den Pfad dar, die gestrichelte Linie markiert das OD-Paar.

Behauptung 37 *Die Instanz* (A, B) *von SUBSET SUM hat genau dann eine Lösung, wenn die Instanz* $(P, OD, 0, B, \sum_{i=1}^{n} a_i - B)$ *von A-Pfad-Graph eine Lösung hat.*

Sei $A' \subseteq A$ eine Lösung der Instanz (A, B). Dann ist für $J := \{i : a_i \in A\}$

$$\sum_{j \in J} s(a_j) = B. \tag{35}$$

Sei der Graph g gegeben durch $g = (V_g, E_g)$ mit $V_g := \{j-1 : j \in J\} \cup \{j : j \in J\}$ und $E := \{[j-1, j] : j \in J\}$.
Dann ist

$$l(g) = \sum_{[j-1,j] \in E} c_{j-1j} = \sum_{j \in J} c_{j-1j} = \sum_{j \in J} s(a_j) = B. \tag{36}$$

und

$$\begin{aligned} d(1,n) =& \sum_{i=1}^{n} l_{i-1i} \\ =& \sum_{j=1,\ j \in J}^{n} \alpha c_{j-1j} + \sum_{i=1,\ i \notin J}^{n} c_{i-1i} \\ =& \sum_{i=1,\ i \notin J}^{n} c_{i-1i} \\ =& \sum_{i=1}^{n} c_{i-1i} - \sum_{i=1,\ i \in J}^{n} c_{i-1i} \\ =& \sum_{i=1}^{n} s(a_i) - \sum_{j \in J} s(a_j) \\ =& \sum_{i=1}^{n} s(a_i) - B \end{aligned} \tag{37}$$

mit (36). Also ist g eine Lösung von $(P, OD, 0, B, \sum_{i=1}^{n} s(a_i) - B)$.
Sei nun $g = (V_g, E_g)$ eine Lösung von $(P, OD, 0, B, \sum_{i=1}^{n} s(a_i) - B)$. Sei $A' := \{a_j : [j-1, j] \in E_g\}$.
Dann ist

$$\begin{aligned} \sum_{a_j \in A'} s(a_j) =& \sum_{[j-1,j] \in E_g} c_{j-1j} \\ =& l(g) \\ \leq& B \end{aligned} \tag{38}$$

Außerdem gilt, da $l_{ij} = 0$, falls $[i,j] \in E_g$:

$$\begin{aligned}
\sum_{a_i \in A} s(a_i) - \sum_{a_j \in A'} s(a_j) &= \sum_{[i-1,i] \in E} c_{i-1i} - \sum_{[j-1,j] \in E_g} c_{j-1j} \\
&= \sum_{[i-1,i] \in E \setminus E_g} c_{i-1i} \\
&= \sum_{[i-1,i] \in E} l_{i-1i} \\
&= d_g(1,n) \\
&\leq \sum_{a_i \in A} s(a_i) - B \\
&\Leftrightarrow \\
\sum_{a_j \in A'} s(a_j) &\geq B \qquad (39)
\end{aligned}$$

Aus (38) und (39) folgt:

$$\sum_{a_j \in A'} s(a_j) = B \qquad (40)$$

$\Rightarrow A'$ löst (A, B).

Das bedeutet, die Instanz (A, B) von SUBSET SUM hat genau dann eine Lösung, wenn die Instanz $(P, OD, 0, B, \sum_{i=1}^{n} a_i - B)$ von *A-Pfad-Graph* eine Lösung hat.

$\Rightarrow$ Das Entscheidungsproblem *A-Pfad-Graph* ist NP-vollständig. ■

4.5.2 A-Baum-Graph

Da ein Pfad ein Baum ist, ist das Entscheidungsproblem *A-Baum-Graph* eine Verallgemeinerung von *A-Pfad-Graph*. Aus der NP-Vollständigkeit von *A-Pfad-Graph* folgt also die NP-Vollständigkeit von *A-Baum-Graph*. Es gilt aber sogar:

Satz 38 *Das Entscheidungsproblem A-Baum-Graph ist stark NP-vollständig.*

Den Beweis hierzu kann man analog zu dem Beweis der starken NP-Vollständigkeit von *A-Baum-Baum* durchführen, denn der dort zur Reduktion von HITTING SET verwendete Sterngraph hat offensichtlich die Eigenschaft, dass Graphen (ohne isolierte Ecken, die für das Problem A irrelevant sind) Bäume sind.

4.5.3 A-Kreis-Graph

Satz 39 *Das Entscheidungsproblem A-Kreis-Graph ist NP-vollständig.*

Beweis. Es wird das Entscheidungsproblems *A-Pfad-Graph* auf *A-Kreis-Graph* reduziert.
Sei $(P, OD, \alpha, l_0, d_0)$ eine Instanz von *A-Pfad-Graph* mit Pfad $P = (V_P, E_P)$ gegeben durch $V_P = \{1, 2, \ldots, n\}$ und $E_P = \{[i, i+1] : i = 1, \ldots, n-1\}$.
Der zugehörige Kreis $K = (V_K, E_K)$ sei gegeben durch $V_K := V_P$, $E_K := E_P \cup \{[n, 1]\}$ mit

$$c_{n1} := \max\{l_0, d_0\} + 1. \tag{41}$$

α, l_0 und d_0 bleiben dabei unverändert.

Behauptung 40 *Es gibt genau dann eine Lösung der Instanz $(P, OD, \alpha, l_0, d_0)$ von A-Pfad-Graph, wenn es eine Lösung der Instanz $(K, OD, \alpha, l_0, d_0)$ von A-Kreis-Graph gibt.*

Sei $g \subset P$ eine Lösung von $(P, OD, \alpha, l_0, d_0)$. Dann ist g auch eine Lösung von $(K, OD, \alpha, l_0, d_0)$, da wegen $P \subseteq K$ nach Lemma 12 $d_g^K(i, j) \leq d_g^P(i, j)$.
Sei g eine Lösung von $(K, OD, \alpha, l_0, d_0)$. Dann ist

$$l(g) \leq l_0 < c_{n1} \tag{42}$$

und daraus folgt $[n, 1] \notin E_g \Rightarrow g \subset P$.
Sei für $\{u, v\} \in OD$ W_{uv}^g ein kürzester Weg zwischen u und v in K_g. Da g Lösung von $(K, OD, \alpha, l_0, d_0)$ ist $l(W_{uv}^g) \leq d_0$ und deshalb $[n, 1] \notin W_{u,v}^g$. Daraus folgt $W_{uv}^g \subset P$.
Also ist W_{uv}^g auch kürzester Weg zwischen u und v in P.
$\Rightarrow$ Für die Distanz zwischen u, v ($\{u, v\} \in OD$) in P_g gilt: $d_g(u, v) = l(W_{uv}^g) \leq d_0$.
Also ist g auch Lösung von $(P, OD, \alpha, l_0, d_0)$.
Das bedeutet, ein Graph g löst genau dann $(P, OD, \alpha, l_0, d_0)$, wenn g $(K, OD, \alpha, l_0, d_0)$ löst.
$\Rightarrow$ Das Entscheidungsproblem *A-Kreis-Graph* ist NP-vollständig. ■

4.5.4 A-zGraph-Graph

Da ein Baum ein zusammenhängender Graph ist, stellt *A-zGraph-Graph* eine Verallgemeinerung des NP-vollständigen Problems *A-Baum-Graph* dar. Aus der starken NP-Vollständigkeit von *A-Baum-Graph* folgt:

Satz 41 *Das Entscheidungsproblem A-zGraph-Graph ist stark NP-vollständig.*

4.6 Ergebnis

		gegeben			
		Pfad	Baum	Kreis	zGraph
gesucht	Pfad	$O(n^2)$	$O(n^5)$	$O(n^3)$	stark NP-vollständig
	Baum	$O(n^2)$	stark NP-vollständig	$O(n^3)$	stark NP-vollständig
	Kreis	nicht möglich	nicht möglich	$O(1)$	stark NP-vollständig
	zGraph	$O(n^2)$	stark NP-vollständig	$O(n^3)$	stark NP-vollständig
	Graph	NP-vollständig	stark NP-vollständig	NP-vollständig	stark NP-vollständig

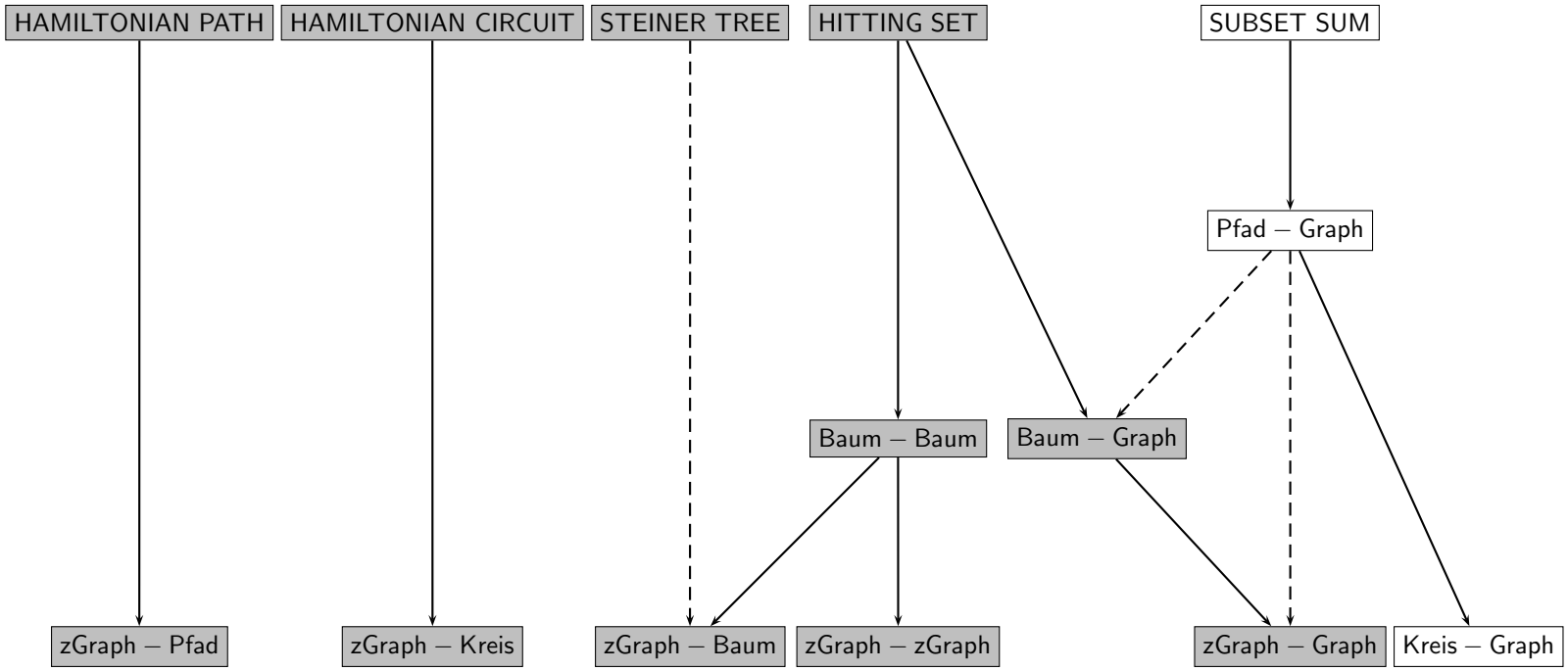

Abbildung 5: Übersicht über die NP-vollständigen Unterprobleme von Problem A. Die durchgezogenen Pfeile kennzeichnen in dieser Arbeit durchgeführte Reduktionen, gestrichelte Pfeile kennzeichnen einige ebenfalls mögliche Reduktionen zwischen den Problemen. Die grauen Felder kennzeichnen Probleme, die sogar stark NP-vollständig sind.

5 Problem B: Maximiere die Gesamtreisezeitersparnis

Sei $G = (V, E)$ ein Graph mit Kantenlängen $c_{ij} \in \mathbb{Z}^+$, OD eine Menge von OD-Paaren in G und l_0 eine natürliche Zahl. Es sei für jedes OD-Paar $\{u, v\} \in OD$ eine natürliche Zahl w_{uv} gegeben, die die Höhe der Nachfrage zwischen u und v angibt.
Betrachtet wird die Zielfunktion

$$\max_{g: l(g) \leq l_0} \sum_{\{u,v\} \in OD} w_{uv}(d(u, v) - d_g(u, v)). \tag{43}$$

Sieht man G als Straßennetzwerk an, in dem beispielsweise ein Busnetzwerk eingerichtet werden soll, so stellt für ein Teilnetzwerk g $d(u, v) - d_g(u, v)$ die Reisezeitersparnis zwischen u und v dar. Die Zielfunktion ist die mit $w_{uv} \in \mathbb{Z}^+$, der Anzahl der Reisenden zwischen u und v, gewichtete Summe dieser Reisezeitersparnisse, also die Gesamtreisezeitersparnis. Dabei darf die Länge des gesuchten Teilgraphen g l_0 nicht überschreiten.

Definition 42 Entscheidungsproblem zu Problem B

- *Instanz* $(G, OD, \alpha, l_0, d_0)$*:*
  Ungerichteter Graph $G = (V, E)$ *mit Kantenlängen* $c_{ij} \in \mathbb{Z}^+$ *für* $[i, j] \in E$*,*
  Menge von OD-Paaren OD *mit Nachfrage* $w_{uv} \in \mathbb{Z}^+$ *für* $\{u, v\} \in OD$*,*
  Beschleunigungsfaktor $\alpha \in [0, 1)$*,*
  Maximallänge des gesuchten Teilgraphen $l_0 \in \mathbb{Z}^+$*,*
  Minimale Reisezeitersparnis $m_0 \in \mathbb{Z}^+$

- *Frage:*
  Gibt es einen Teilgraph $g \subset G$*, so dass:*

$$l(g) \leq l_0, \tag{44}$$

$$\sum_{\{u,v\} \in OD} w_{uv}(d(u, v) - d_g(u, v)) \geq m_0 \ ? \tag{45}$$

Satz 43 *Problem B ist in NP. Dies gilt unabhängig vom gegebenen Graph G und der gewünschten Struktur des gesuchten Graphen g.*

Beweis. Sei $(G, OD, \alpha, l_0, d_0)$ eine Instanz von Entscheidungsproblem B. Für einen gegebenen Teilgraph $g \subset G$ ist es möglich, in Polynomialzeit zu überprüfen, ob g die gewünschte Struktur hat (zusammenhängend, Pfad, Kreis oder Baum), ob $l(g) \leq l_0$ (durch Addition der Kantenlängen der enthaltenen Kanten) und ob $\sum_{\{u,v\} \in OD} w_{uv}(d(u, v) - d_g(u, v)) \geq m_0$ (durch

Bestimmung der kürzesten Wege zwischen den OD-Paaren in G und in G_g). Also lässt sich in Polynomialzeit entscheiden, ob g eine Lösung des Entscheidungsproblems ist.
$\Rightarrow$ Problem B ist in NP. ■

Lemma 44 *In Graphen, in denen alle Wege zwischen den OD-Paaren eindeutig sind, also insbesondere in Bäumen, lässt sich die Zielfunktion (43) vereinfachen zu*

$$\max_{g:l(g)\leq l_0} \sum_{[i,j]\in g} (1-\alpha)c_{ij}m_{ij} \tag{46}$$

mit

$$m_{ij} = \sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv}, \tag{47}$$

wobei W_{uv} der eindeutig bestimmte Weg von u nach v ist.

Beweis. Sei G ein Graph, in dem für jedes OD-Paar $\{u,v\}$ der Weg W_{uv} zwischen den Knoten u und v eindeutig ist. Dann gilt:

$$d(u,v) = l(W_{uv}) = \sum_{[i,j]\in W_{uv}} c_{ij} \tag{48}$$

und

$$d_g(u,v) = \sum_{[i,j]\in W_{uv}\cap g} \alpha c_{ij} + \sum_{[i,j]\in W_{uv}\setminus g} c_{ij}. \tag{49}$$

Daraus folgt:

$$\begin{aligned} d(u,v) - d_g(u,v) &= \sum_{[i,j]\in W_{uv}} c_{ij} - \Big(\sum_{[i,j]\in W_{uv}\cap g} \alpha c_{ij} + \sum_{[i,j]\in W_{uv}\setminus g} c_{ij} \Big) \\ &= \sum_{[i,j]\in W_{uv}\cap g} (1-\alpha)c_{ij} \\ &= (1-\alpha) \sum_{[i,j]\in W_{uv}\cap g} c_{ij}. \end{aligned} \tag{50}$$

Sei nun für eine Kante $[i,j]$ die Gesamtnachfrage auf dieser Kante m_{ij} gegeben durch:

$$m_{ij} = \sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv}. \tag{51}$$

Dann lässt sich die Zielfunktion folgendermaßen umformen:

$$\begin{aligned}
\sum_{\{u,v\}\in OD} w_{uv}(d(u,v) - d_g(u,v)) &= \sum_{\{u,v\}\in OD} w_{uv}\Big((1-\alpha) \sum_{[i,j]\in W_{uv}\cap g} c_{ij}\Big) \\
&= (1-\alpha) \sum_{\{u,v\}\in OD} \sum_{[i,j]\in W_{uv}\cap g} w_{uv}c_{ij} \\
&= (1-\alpha) \sum_{[i,j]\in g} \sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv}c_{ij} \\
&= (1-\alpha) \sum_{[i,j]\in g} c_{ij} \sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv} \\
&= (1-\alpha) \sum_{[i,j]\in g} c_{ij}m_{ij}. \qquad (52)
\end{aligned}$$

■

Da der Faktor $(1-\alpha)$ konstant und positiv ist, lässt sich das Optimierungsproblem in (43) durch obige Umformung und Weglassen von $(1-\alpha)$ also vereinfachen zu:

Definition 45 *Problem B in Bäumen*
Sei $T = (V, E)$ ein Baum, OD die Menge der OD-Paare in T und l_0 eine positive Zahl. Gesucht ist ein Teilgraph $\hat{g} \subset T$ mit $l(\hat{g}) \leq l_0$, so dass

$$\sum_{[i,j]\in\hat{g}} c_{ij}m_{ij} = \max_{\hat{g}:l(\hat{g})\leq l_0} \sum_{[i,j]\in\hat{g}} c_{ij}m_{ij}. \qquad (53)$$

Betrachtet man das zugehörige Entscheidungsproblem, so ist zu beachten, dass wegen des entfernten Vorfaktors $(1-\alpha)$ die minimale Reisezeitersparnis m_0 durch $\acute{m}_0 = \frac{m_0}{1-\alpha}$ ersetzt wird.

Folgender Algorithmus bestimmt die Kantengesamtnachfrage m_{ij} für alle Kanten $[i,j]$ eines Baumes $T = (V, E)$:

Algorithmus 2 Bestimmung der m_{ij}

Eingabe: Rekursiv dargestellter Baum $T = (V, E)$ mit Knotenmenge $V = \{1, 2, \ldots, n\}$; OD Menge von OD-Paaren.

Ausgabe: $m_{iC(i)}$ $\forall\, [i, C(i)] \in E$.

1: **for** $u \in V$ **do**
2: **for** $v \in V, v < u$ **do**
3: $\tilde{w}_{uv} = \begin{cases} w_{uv} & \text{falls} \{u, v\} \in OD, \\ 0 & \text{sonst} \end{cases}$
4: **end for**
5: **end for**
6: **for** $u = n, n-1, \ldots, 2$ **do**
7: **for** jedes Knotenpaar $\{u, v\} \in V \times V$ mit $v < u$ **do**
8: **if** $v = C(u)$ **then**
9: $m_{uC(u)} = \tilde{w}_{uv}$
10: **else if** $\tilde{w}_{uv} > 0$ **then**
11: Addiere $\tilde{w}_{uv}$ zu $\tilde{w}_{uC(u)}$.
12: Addiere $\tilde{w}_{uv}$ zu $\tilde{w}_{C(u)v}$.
13: Setze $\tilde{w}_{uv} = 0$.
14: **end if**
15: **end for**
16: **end for**
17: **return** $m_{iC(i)} = \tilde{w}_{iC(i)}$ $\forall\ i \in V$

Lemma 46 *Die Berechnung der Kantengesamtnachfragen m_{ij} aus der Zielfunktion in (46) beziehungsweise (53) für Problem B in Bäumen ist in $O(n^2)$.*

Beweis. Zum Beweis betrachte man den Algorithmus zur Bestimmung der m_{ij} für einen rekursiv dargestellten Baum T. Das Vorgehen des Algorithmus kann folgendermaßen interpretiert werden: Die gegebene Menge von OD-Paaren wird schrittweise in eine Menge von OD-Paaren $\{u, v\}$ umgewandelt, so dass der (eindeutig bestimmte) Weg W_{uv} zwischen den Knoten dieses OD-Paares u und v aus nur einer Kante besteht. Während die Knoten K des rekursiv dargestellten Baumes absteigend durchlaufen werden, wird jedes in u startende OD-Paar $\{u, v\}$ mit $u < v$ aufgeteilt in ein OD-Paar $\{u, C(u)\}$ und ein OD-Paar $\{C(u), v\}$ mit gleicher Nachfrage $\tilde{w}_{uv}$, die zu der Nachfrage eventuell schon zwischen diesen Knoten bestehender OD-Paare addiert wird. Für jede Kante $[i, C(i)]$ wird die Kantennachfrage $m_{iC(i)}$ dabei offensichtlich nicht verändert und ist am Ende des Verfahrens identisch mit der Nachfrage zwischen den Knoten i und $C(i)$, also der Kantengesamtnachfrage $m_{iC(i)}$.
Aus jedem Baum $\tilde{T}$ mit dazugehörigen OD-Paaren erhält man nach Lemma 4 und Beobachtung 6 in $O(n^2)$ einen rekursiv dargestellten Baum und kann die OD-Paare entsprechend umbenennen. Es gibt $O(n^2)$ OD-Paare $\{u, v\}$ mit $v < u$. Schritt 1-5 sind also in $O(n^2)$. Die Anzahl der Additionen im u-ten Durchlauf beträgt in Schritt 11 und 12 für alle $v < u$ zusammen jeweils höchstens $u - 2$ und ist daher insgesamt in $O(n^2)$. Das Verfahren ist also von Komplexität $O(n^2)$. ■

Lemma 47 *Sei $G = (V, E)$ ein Graph mit Kantenlänge c_{ij} für $[i, j] \in E$. Seien $g = (V_g, E_g)$ und $g' = (V_{g'}, E_{g'})$ zwei Teilgraphen. Ist $g' \subset g$, so gilt:*

$$\sum_{\{u,v\} \in OD} w_{uv}(d(u, v) - d_g(u, v)) \geq \sum_{\{u,v\} \in OD} w_{uv}(d(u, v) - d_{g'}(u, v)). \tag{54}$$

Ist G ein Baum, so folgt:

$$\sum_{[i,j] \in E_g} c_{ij} m_{ij} \geq \sum_{[i,j] \in E_{g'}} c_{ij} m_{ij}. \tag{55}$$

Beweis. Aus Lemma 12 folgt:

$$d_g(u,v) \leq d_{g'}(u,v) \quad \forall \ \{u,v\} \in OD \tag{56}$$

$\Rightarrow$

$$d(u,v) - d_g(u,v) \geq d(u,v) - d_{g'}(u,v) \quad \forall \ \{u,v\} \in OD \tag{57}$$

$\Rightarrow$

$$\sum_{\{u,v\} \in OD} w_{uv}(d(u,v) - d_g(u,v)) \geq \sum_{\{u,v\} \in OD} w_{uv}(d(u,v) - d_{g'}(u,v)). \tag{58}$$

Ist G ein Baum, so folgt aus der Äquivalenz der Zielfunktionen (Lemma 44):

$$\sum_{[i,j] \in E_g} c_{ij} m_{ij} \geq \sum_{[i,j] \in E_{g'}} c_{ij} m_{ij}. \tag{59}$$

■

5.1 Bestimmung optimaler Teilpfade

5.1.1 B-Pfad-Pfad

Satz 48 *Das Problem B-Pfad-Pfad ist in* $O(n^2)$.

Beweis. Gegeben sei ein Pfad $P = (V, E)$, dessen Knoten ohne Beschränkung der Allgemeinheit entlang des Pfades in aufsteigender Reihenfolge angeordnet sind und eine Menge von OD-Paaren in P (siehe Beobachtung 5 und Bemerkung 6). Es reicht nach Lemma 47, die bezüglich Inklusion maximalen Teilpfade der Länge $\leq l_0$ auf Optimalität zu untersuchen. Durch Vergleich von $\sum_{[j,k] \in P_i} m_{jk} c_{jk}$ für jeden dieser Pfade P_i mit $i \in V$ findet man den bezüglich Problem *B-Pfad-Pfad* optimalen Pfad $\hat{P}$. Der Aufwand für die Bestimmung des optimalen Pfades $\hat{P}$ setzt sich also zusammen aus der Bestimmung der bezüglich Inklusion maximalen Pfade der Länge $\leq l_0$ (nach Lemma 7 in $O(n)$) und der Berechnung von $\sum_{[j-1,j] \in P_i} c_{j-1j} m_{j-jj}$. Die m_{j-1j} sind unabhängig vom gewählten Pfad und können nach Lemma 46 in $O(n^2)$ bestimmt werden. Die Berechnung der Zielfunktionswerte der Pfade P_i mit Startknoten i und Endknoten v_i kann nun für alle Pfade P_i zusammen in $O(n)$ erfolgen, indem die Summen $S_i = \sum_{[j-1,j] \in P_i} c_{j-1j} m_{j-1j}$ durch

$$\begin{aligned} S_1 &= \sum_{j=2}^{v_1} m_{j-1j} c_{j-1j}, \\ S_i &= S_{i-1} - m_{i-1i} c_{i-1i} + \sum_{j=v_{i-1}+1}^{v_i} m_{j-1j} c_{j-1j} \quad \forall \ i = 2, \ldots, n \end{aligned} \tag{60}$$

auseinander berechnet werden. Die Berechnung von $\hat{P}$ ist also im Allgemeinen in $O(n^2)$. Für einen in aufsteigender Reihenfolge nummerierten Pfad mit schon bekannten Kantengesamtnachfragen $m_{[jC(j)]}$ für alle $[j, C(j)] \in E$ ist die Bestimmung des optimalen Pfades sogar in $O(n)$. ■

5.1.2 B-Baum-Pfad

Satz 49 *Das Problem B-Baum-Pfad ist in* $O(n^3)$.

Beweis. Sei $T = (V, E)$ ein Baum und OD eine Menge von OD-Paaren in T. T lässt sich nach Lemma 4 in $O(n^2)$ rekursiv darstellen, die dazugehörigen OD-Paare ergeben sich ebenfalls in $O(n^2)$ (siehe Bemerkung 6). Ebenfalls in $O(n^2)$ kann man die $m_{iC(i)}$ $\forall i \in V$ bestimmen (Lemma 46). Nun betrachtet man alle Knotenpaare $\{i, j\}$ mit $i \neq j$ aus $V \times V$. Die Anzahl dieser Paare ist offensichtlich in $O(n^2)$. Für jedes feste Knotenpaar $\{i, j\}$ bestimmt man mit Hilfe von Algorithmus „Wege in Bäumen“ nach Lemma 9 in $O(n)$ den Weg W_{ij} zwischen i und j und addiert die c_{kl} und m_{kl} der Kanten $[k, l]$ aus W_{ij}. Durch Auswahl eines Pfades $\hat{P}$ mit maximalem Zielfunktionswert aus der Menge der Pfade, die $l(P) = \sum_{[k,l] \in E_P} c_{kl} \leq l_0$ nicht verletzen, erhält man also in $O(n^3)$ eine Optimallösung von *B-Baum-Pfad*. ■

5.1.3 B-Kreis-Pfad

Satz 50 *Das Problem B-Kreis-Pfad ist in* $O(n^4)$.

Beweis. Sei $K = (V, E)$ ein Kreis. Nach Lemma 5 seien die Knoten von K ohne Beschränkung der Allgemeinheit im Uhrzeigersinn aufsteigend angeordnet. OD sei eine Menge von OD-Paaren in K (vergleiche Bemerkung 6). Für jeden Knoten $i \in V$ sei $P_i = (V_i, E_i)$ der längste unter den im Uhrzeigersinn in i startenden Pfaden der Länge höchstens l_0. Es reicht nach Lemma 47, nur diese Pfade P_i zu betrachten, da jeder andere Pfad $\tilde{P}$ in einem dieser Pfade enthalten ist. Die Berechnung der Pfade P_i ist nach Lemma 7 in $O(n)$.
Die Distanzen zwischen den Knoten der OD-Paare in K und in K_{P_i} für jeden Pfad P_i erhält man für jedes der $O(n^2)$ OD-Paare durch Vergleich der Längen der zwei möglichen Wege W_1 und W_2. Da die Berechnung (siehe Lemma 9) und der Vergleich in $O(n)$ sind, ist dieser Schritt für alle OD-Paare und alle $O(n)$ Pfade in $O(n^4)$. Durch Auswahl von

$$\hat{P} := \text{argmax}_{P_i; l(P_i) \leq l_0} \sum_{\{u,v\} \in OD} w_{uv}(d(u, v) - d_g(u, v)) \tag{61}$$

$$\tag{62}$$

erhält man den optimalen Pfad $\hat{P}$. Die Bestimmung von $\hat{P}$ ist also in $O(n^4)$. ■

5.1.4 B-zGraph-Pfad

Satz 51 *Das Entscheidungsproblem B-zGraph-Pfad ist stark NP-vollständig.*

Der Beweis wird, ähnlich wie im Beweis zur NP-Vollständigkeit von *A-zGraph-Pfad* (Lemma 18), durch Reduktion des stark NP-vollständigen Entscheidungsproblems HAMILTONIAN PATH geführt.

Beweis. Sei $(\tilde{G})$ mit $\tilde{G} = (\tilde{V}, \tilde{E})$ und $\tilde{V} := \{1, 2, \ldots, n\}$ eine Instanz von HAMILTONIAN PATH.
Dann sei $G := (V, E)$ gegeben durch $V := \tilde{V} \cup V'$, wobei $V' := \{1', 2', \ldots, n'\}$ und $E := \tilde{E} \cup E'$ mit $E' := \Big\{[i, i'] : i \in \tilde{V}\Big\}$. Die Kantenlängen seien $c_{ij} := 1 \ \forall \ [i, j] \in \tilde{E}$ und $c_{ii'} := n^2 + 1$ für $i \in \tilde{V},\ i' \in V'$.
Die Menge der OD-Paare sei $OD := \{\{i', (i+1)'\} : i = 1, 2, \ldots, n-1\} \cup \{\{n', 1'\}\}$ mit Nachfrage $w_{uv} = 1 \ \forall \{u, v\} \in OD$. Sei $m_0 := \sum_{\{u,v\} \in OD}(d(u, v) - 2(n^2 + 1))$ und $\alpha := 0$.
Die Konstruktion von G aus $\tilde{G}$ kann in Polynomialzeit erfolgen, da die Berechnung von $d(u, v)$, das heißt das Bestimmen eines kürzesten Weges zwischen u und v, nach Lemma 9 in $O(n)$ ist. Da $c_{ij} = 1$ für alle $i, j \in \tilde{E}$ folgt $d(k, l) \leq n$ für alle $k, l \in \tilde{V}$. Deswegen gilt für m_0:

$$
\begin{aligned}
m_0 = & \sum_{\{u,v\} \in OD} (d(u, v) - 2(n^2 + 1)) \\
= & \Big(\sum_{i=1}^{n-1} d(i', (i+1)') - 2(n^2 + 1) \Big) + d(n', 1') - 2(n^2 + 1) \\
= & \Big(\sum_{i=1}^{n-1} d(i', i) + d(i, i+1) + d(i+1, (i+1)') - 2(n^2 + 1) \Big) \\
& + d(n', n) + d(n, 1) + d(1, 1') - 2(n^2 + 1) \\
\leq & \Big(\sum_{i=1}^{n-1} (n^2 + 1) + n + (n^2 + 1) - 2(n^2 + 1) \Big) + (n^2 + 1) + n + (n^2 + 1) - 2(n^2 + 1) \\
= & \sum_{i=1}^{n} n \\
= & n^2. \qquad (63)
\end{aligned}
$$

Also sind alle Zahlenwerte in $(T, OD, \alpha, n^2, m_0)$ polynomiell beschränkt in n.

Behauptung 52 *Es gibt genau dann eine Lösung der Instanz $(\tilde{G})$ von HAMILTONIAN PATH, wenn es eine Lösung der Instanz $(G, OD, 0, n^2, m_0)$ von B-zGraph-Pfad gibt.*

Sei $P = (V_P, E_P)$ eine Lösung von $(\tilde{G})$. Dann ist P ein Pfad und wegen $E_P \subset \tilde{E}$ gilt $l(P) \leq \sum_{[i,j]\in\tilde{E}} c_{ij} = \sum_{[i,j]\in\tilde{E}} 1 < n^2$. Außerdem ist

$$
\begin{aligned}
&\sum_{\{u,v\}\in OD} (d(u,v) - d_P(u,v)) \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \sum_{\{u,v\}\in OD} d_P(u,v) \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \Big[\Big(\sum_{j=1}^{n-1} d_P(j',(j+1)')\Big) + d_P(n',1')\Big] \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \Big[\Big(\sum_{j=1}^{n-1} d_P(j',j) + d_P(j,j+1) + d_P(j+1,(j+1)')\Big) \\
&+ d_P(n',n) + d_P(n,1) + d_P(1,1')\Big] \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \Big[\Big(\sum_{j=1}^{n-1}(n^2+1) + 0 + (n^2+1)\Big) + (n^2+1) + 0 + (n^2+1)\Big] \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \sum_{j=1}^{n} 2(n^2+1) \\
=&\sum_{\{u,v\}\in OD} \Big(d(u,v) - 2(n^2+n)\Big) \\
=&m_0. \qquad (64)
\end{aligned}
$$

$\Rightarrow P$ ist Lösung der Instanz $(G, OD, 0, n^2, m_0)$ von *B-zGraph-Pfad*.
Sei nun $P = (V_P, E_P)$ eine Lösung von $(G, OD, 0, n^2, m_0)$. Dann ist P ein Pfad,

$$l(P) \leq n^2 \qquad (65)$$

und

$$\sum_{\{u,v\}\in OD} (d(u,v) - d_P(u,v)) \geq m_0. \qquad (66)$$

Wegen (65) ist $[i,i'] \notin E_P \ \forall\ i \in \tilde{G}$, da sonst $l(P) \geq c_{ii'} = n^2 + 1 > n^2$. Außerdem ist $P \neq (\{i'\}, \emptyset) \forall i \in V$, da sonst $d(u,v) = d_p(u,v)$ für alle $\{u,v\} \in OD$, und das wäre ein Widerspruch zu (66). Also ist $P \subset \tilde{G}$.

Angenommen P ist kein Hamiltonscher Pfad in $\tilde{G}$, das heißt $\exists i \in \tilde{V} : \ i \notin V_P$. Dann ist wegen $c_{ij} > 0 \ \forall [i,j] \in E_G$:

$$\begin{aligned} d_P(i', (i+1)') =& d_P(i', i) + d_P(i, i+1) + d_P(i+1, (i+1)') \\ =& (n^2+1) + d_P(i, i+1) + (n^2+1) \\ >& (n^2+1) \end{aligned} \tag{67}$$

$\Rightarrow$

$$d(i', (i+1)') - d_P(i', (i+1)') < d(i', (i+1)') - 2(n^2+1). \tag{68}$$

Daraus folgt

$$\begin{aligned} \sum_{\{u,v\} \in OD} (d(u,v) - d_P(u,v)) =& \sum_{j=1}^{n-1} (d(j', (j+1)') - d_P(j', (j+1)')) + d(n', 1') - d_P(n', 1') \\ <& \sum_{j=1}^{n-1} (d(j', (j+1)') - 2(n^2+1)) + d(n', 1') - 2(n^2+1) \\ =& m_0. \end{aligned} \tag{69}$$

Also ist $\sum_{\{u,v\} \in OD} (d(u,v) - d_P(u,v)) < m_0$, was ein Widerspruch zu (66) ist. Deshalb muss i in V_P enthalten sein.

$\Rightarrow P$ ist Hamiltonscher Pfad.

Die Instanz $(\tilde{G})$ von HAMILTONIAN PATH hat also genau dann eine Lösung, wenn die Instanz $(G, OD, 0, n^2, m_0)$ von *B-zGraph-Pfad* eine Lösung hat.

Es wurde gezeigt, dass HAMILTONIAN PATH sich auf ein Unterproblem von *B-zGraph-Pfad* mit polynomiell in n beschränkten numerischen Eingaben reduzieren lässt.

$\Rightarrow$ *B-zGraph-Pfad* ist stark NP-vollständig. ■

5.2 Bestimmung optimaler Teilbäume

5.2.1 B-Baum-Baum

Satz 53 *Das Entscheidungsproblem B-Baum-Baum ist NP-vollständig.*

Der Beweis dieses Satzes erfolgt durch Reduktion des Rucksackproblems auf das Entscheidungsproblem *B-Baum-Baum*. Im Folgenden wird neben der Definition des Entscheidungsproblems auch die Definition des Optimierungsproblems gegeben, da diese in 5.5.2 benötigt wird.

Definition 54 *[GJ79, 247]* **KNAPSACK**

Entscheidungsproblem:

- *Instanz* (O, C, N)*:*
  Menge von Objekten $O := \{o_1, o_2, \ldots, o_n\}$ *mit zugeordnetem Nutzen* $a_i \in \mathbb{Z}^+$ *und Gewicht* $b_i \in \mathbb{Z}^+$ *für* $o_i \in O$*,*
  Kapazitätsbeschränkung $C \in \mathbb{Z}^+$*,*
  Minimalnutzen $N \in \mathbb{Z}^+$
- *Frage:*
  Gibt es eine Untermenge $R \subseteq O$*, so dass* $\sum_{o_i \in R} b_i \leq C$ *und* $\sum_{o_i \in R} a_i \geq N$*?*

Optimierungsproblem:

- *Instanz* (O, C)*:*
  Menge von Objekten $O := \{o_1, o_2, \ldots, o_m\}$ *mit zugeordnetem Nutzen* $a_i \in \mathbb{Z}^+$ *und Gewicht* $b_i \in \mathbb{Z}^+$ *für* $o_i \in O$*,*
  Kapazitätsbeschränkung $C \in Z^+$
- *Gesucht:*
  Untermenge $R \subseteq O$*, so dass* $\sum_{o_i \in R} b_i \leq C$ *und* $\sum_{o_i \in R} a_i$ *maximal.*

Satz 55 *[GJ79, 247] Das Entscheidungsproblem KNAPSACK ist NP-vollständig.*

Beweis. (von Satz 53)
Sei (O, C, N) eine Instanz von KNAPSACK, wobei $O := \{o_1, o_2, \ldots, o_n\}$ mit zugeordnetem Nutzen m_i und Gewicht b_i für $o_i \in O$. Durch $V := \{0, 1, \ldots, n\}$, $E := \{e_i = [0, i] : i = 1, \ldots n\}$ wird ein Baum, genauer gesagt ein Sterngraph, $S = (V, E)$ definiert. Die Kantenlängen seien gegeben durch $c_{0i} := b_i$. Sei $X := \prod_{i=1}^{n} b_i$. Die Menge der OD-Paare sei $OD := \{\{0, i\} : i = 1, \ldots, n\}$ mit $w_{0i} := \frac{a_i}{c_{0i}} X$. Sei $\alpha := 0$, $l_0 := C$, und $\acute{m}_0 := NX$.
Betrachtet wird also die Instanz $(S, OD, 0, C, (1 - \alpha)NX)$ von *B-Pfad-Graph*.

Ein Beispiel für die Konstruktion von S aus O:

Objekte o_i	o_1	o_2	o_3	o_4	o_5	o_6
Nutzen a_i	6	4	1	2	20	8
Gewicht b_i	2	3	1	1	4	3

$X = 2 \cdot 3 \cdot 1 \cdot 1 \cdot 4 \cdot 3 = 72$

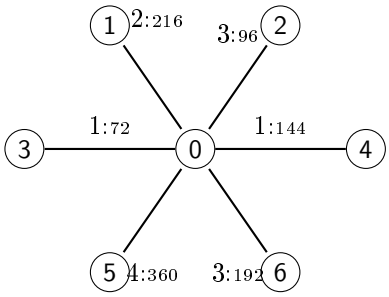

Abbildung 6: Beispiel zur Konstruktion eines Sterngraphen aus einer Instanz von KNAPSACK. Die Striche stellen gleichzeitig die Kanten von S und die OD-Paare dar. Die Zahlen an den Kanten $[i, j]$ stehen für $c_{ij} : w_{ij}$.

Behauptung 56 *Die Instanz (O, C, N) von KNAPSACK hat genau dann eine Lösung, wenn die Instanz $(S, OD, 0, C, (1-\alpha)NX)$ von B-Baum-Baum eine Lösung hat.*

Da in einem Sterngraph die Wege zwischen den Knoten eindeutig bestimmt sind, lässt sich für *B-Baum-Baum* die Charakterisierung (53) der Zielfunktion verwenden.
Angenommen $R \subset O$ löst (O, C, N). Dann gilt:

$$\sum_{o_i \in R} a_i \geq N \tag{70}$$

$$\sum_{o_i \in R} b_i \leq C. \tag{71}$$

Man betrachte nun die Instanz $(S, OD, 0, C, (1-\alpha)NX)$ von *B-Baum-Baum*. Sei $T = (V_T, E_T)$ mit $V_T := \{0\} \cup \{j \in V : o_j \in R\}$ und $E_g := \{[0, i] : o_i \in R\}$. T ist offensichtlich zusammenhängend. Es gilt:

$$\begin{aligned} \sum_{[i,j] \in E_T} c_{ij} &= \sum_{[0,i] \in E_T} c_{0i} \\ &= \sum_{[0,i] \in E_T} b_i \\ &= \sum_{o_i \in R} b_i \\ &\leq C \end{aligned} \tag{72}$$

und

$$\begin{aligned}\sum_{[i,j]\in E_T} m_{ij}c_{ij} &= \sum_{[0,i]\in E_T}^{n} w_{0i}c_{0i} \\ &= \sum_{[0,i]\in E_T}^{n} \frac{a_i}{c_{0i}} X c_{0i} \\ &= \sum_{[0,i]\in E_T} a_i X \\ &= \sum_{o_i\in R} a_i X \\ &\geq NX. \end{aligned} \tag{73}$$

Deswegen ist $(1-\alpha)\sum_{[i,j]\in E_T} m_{ij}c_{ij} \geq (1-\alpha)NX$. Die Instanz $(S, OD, 0, C, (1-\alpha)NX)$ wird also von T gelöst.
Sei nun $T = (V_T, E_T)$ eine Lösung von $(S, OD, 0, C, (1-\alpha)NX)$. Sei $R := \{o_i \in O : [0,i] \in E_T\}$. Dann gilt mit ähnlichen Rechnungen wie vorher:

$$\sum_{o_i\in R} b_i = \sum_{[0,i]\in E_T} c_{0i} \leq C \tag{74}$$

und

$$\begin{aligned}\sum_{o_i\in R} a_i &= \sum_{[0,i]\in E_T} \frac{w_{0i}c_{0i}}{X} \\ &= \Big(\sum_{[0,i]\in E_T} \big(\sum_{\{u,v\}\in OD:[0,i]\in W_{uv}} w_{uv} \big) c_{0i} \Big) / X \\ &= \Big(\sum_{[i,j]\in E_T} \big(\sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv} \big) c_{ij} \Big) / X \\ &= \Big(\sum_{[i,j]\in E_T} m_{ij}c_{ij} \Big) / X \\ &\geq \frac{NX}{X} \\ &= N. \end{aligned} \tag{75}$$

(O, C, N) hat also genau dann eine Lösung, wenn $(S, OD, 0, C, (1-\alpha)NX)$ eine Lösung hat.
$\Rightarrow$ Das Entscheidungsproblem *B-Baum-Baum* ist NP-vollständig. ■

Satz 57 *Sei (T, OD, α, l_0) eine Instanz des Optimierungsproblems B-Baum-Baum. m_{ij} bezeichne die Kantennachfrage der Kante $[i,j]$. Dann ist (T, OD, α, l_0) in $O(Qn^2 + n^3)$ mit*

$Q = \sum_{[i,j]\in E} m_{i,j}c_{ij}$ *lösbar.*

Zum Beweis betrachte man das PARTIALLY ORDERED KNAPSACK Problem [GJ79, 247] in seiner graphentheoretischen Formulierung. Kanten eines gerichteten Graphen werden im Folgenden mit (i, j) für Knoten i, j bezeichnet.

Definition 58 *[JN83]* **PARTIALLY ORDERED KNAPSACK**
Entscheidungsproblem:

- *Instanz* (G, C, N)*:*
  Gerichteter azyklischer Graph $G = (V, E)$ *mit Gewicht* $b_i \in Z_0^+$ *und Wert* $a_i \in Z_0^+$ *für alle* $i \in V$*,*
  $C, N \in \mathbb{Z}^+$
- *Frage:*
  Gibt es eine Untermenge $V' \subseteq V$*, so dass* $\sum_{i\in V'} b_i \leq C$*,* $\sum_{i\in V'} a_i \geq N$ *und aus* $i \in V'$ *und* $(j, i) \in E$ *folgt* $j \in V'$*?*

Optimierungsproblem:

- *Instanz* (G, C)*:*
  Gerichteter azyklischer Graph $G = (V, E)$ *mit Gewicht* $b_i \in Z_0^+$ *und Wert* $a_i \in Z_0^+$ *für alle* $i \in V$*,*
  $C \in \mathbb{Z}^+$
- *Gesucht:*
  Untermenge $V' \subseteq V$*, so dass* $\sum_{i\in V'} b_i \leq C$*, aus* $i \in V'$ *und* $(j, i) \in E$ *folgt* $j \in V'$*und* $\sum_{i\in V'} a_i$ *maximal.*

Das Entscheidungsproblem PARTIALLY ORDERED KNAPSACK ist für allgemeine Graphen stark NP-vollständig [JN83]. Ist der Graph G ein Baum, so wird das Problem als TREE KNAPSACK bezeichnet. Das zugehörige Optimierungsproblem ist weiterhin NP-schwer. Es ist aber im Gegensatz zum allgemeinen PARTIALLY ORDERED KNAPSACK durch dynamische Programmierung in pseudopolynomieller Zeit lösbar (siehe [JN83]).

Definition 59 *[JN83]*
Ein Out-Tree ist ein gerichteter Baum mit einem ausgezeichneten Startknoten v*, so dass alle Kanten* (j, k) *von* v *weggerichtet sind.*

Satz 60 *[JN83]*
TREE KNAPSACK lässt sich für Out-Trees in $O(nQ)$ mit $Q = \sum_{v \in V} a_v$ lösen.

Nun zum Beweis von Satz 57:

Beweis. Sei (T, OD, α, l_0) mit $T = (V, E)$ eine Instanz des Optimierungsproblems *B-Baum-Baum*. Für jedes $v \in V$ lässt sich in $O(n^2)$ eine rekursive Bezeichnung $A(T)$ von T finden, so dass $A(v) = 1$ (Lemma 4). Sei nun $V_v := A(V)$ und $E_v := \{(C(i), i) : i \in V_v \setminus v\}$. Dann ist $T_v = (V_v, E_v)$ nach Konstruktion ein Out-Tree. Man definiere nun Knotengewichte und Knotenwerte durch $b_v = a_v := 0$ für den Startknoten v und $b_j := c_{C(j)j}$ und $a_j := m_{C(j)j} c_{C(j)j}$ für $j \in V_v \setminus v$. Der Aufwand dieser Konstruktion ist für jeden Startknoten v in $O(n^2)$, außerdem sind a_j und b_j ganzzahlig, wie in Definition 58 gefordert.
Betrachtet man nun (T, OD, α, l_0) unter der zusätzlichen Bedingung, dass die Ecke v in der Lösung enthalten sein soll, so entspricht eine Lösung dieses Problems einer Lösung der Instanz (T_v, l_0) des Optimierungsproblems PARTIALLY ORDERED KNAPSACK mit gleichem Zielfunktionswert, da die Gewichte und Werte nur von den Kanten auf die Ecken umverteilt wurden. Für jeden der n Startknoten $v \in V$ wird T_v in $O(n^2)$ konstruiert und die Instanz (T_v, l_0) in $O(nQ)$ mit $Q = \sum_{v \in V} a_v = \sum_{[C(j),j] \in E} m_{C(j)j} c_{C(j)j}$ gelöst (Satz 60). Durch Auswahl der besten Lösung erhält man also eine Optimallösung von (T, OD, α, l_0) in $O(Qn^2 + n^3)$. ■

5.2.2 B-zGraph-Baum

Die NP-Vollständigkeit von *B-zGraph-Baum* folgt aus der Tatsache, dass schon *B-Baum-Baum* NP-vollständig ist (Satz 53) und ein Baum insbesondere ein zusammenhängender Graph ist. Durch Reduktion eines NP-vollständigen Spezialfalls des Steinerbaumproblems auf Instanzen von *B-zGraph-Baum*, deren numerische Eingabewerte polynomiell in der Eingabelänge beschränkt sind, lässt sich sogar die starke NP-Vollständigkeit von *B-zGraph-Baum* zeigen.

Satz 61 *Das Entscheidungsproblem B-zGraph-Baum ist stark NP-vollständig.*

Definition 62 *[GJ79, 208f.] Steinerbaum*
Sei $G = (V, E)$ ein zusammenhängender Graph mit Kantenlängen c_{ij} für $[i, j] \in E$ und $K \subseteq V$ eine Menge von Knoten. Ein Teilgraph T von G wird Steinerbaum genannt, wenn T ein Baum ist und alle Knoten aus K enthält.

Definition 63 *[GJ79, 208f.] STEINER TREE*

- *Instanz* $(\tilde{G}, \tilde{K}, l_0)$*:*
  Ungerichteter, zusammenhängender Graph $\tilde{G} = (\tilde{V}, \tilde{E})$ *mit Kantenlängen* $\tilde{c}_{ij} \in \mathbb{Z}^+$ *für alle* $[i,j] \in \tilde{E}$*,*
  Knotenmenge $\tilde{K} \subset \tilde{V}$*,*
  Positive ganze Zahl $\tilde{l}_0$*.*

- *Frage:*
  Gibt es einen Steinerbaum T *für* $\tilde{K}$ *in* $\tilde{G}$*, so dass* $l(T) \leq \tilde{l}_0$*?*

Satz 64 *[GJ79, 208f.] Das Entscheidungsproblem STEINER TREE ist NP-vollständig. Das gilt auch, wenn die Kantenlängen* $\tilde{c}_{ij}\ \forall\ [i,j] \in \tilde{E}$ *gleich sind.*

Beweis. (von Satz 61)
Sei $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ eine Instanz von STEINER TREE mit $\tilde{G} = (\tilde{V}, \tilde{E})$, $c \in \mathbb{Z}^+$ und $\tilde{c}_{ij} = c$ für alle $[i,j] \in \tilde{E}$. Daraus wird folgendermaßen eine Instanz $(G, OD, \alpha, l_0, m_0)$ von *B-zGraph-Baum* konstruiert:
Sei ohne Beschränkung der Allgemeinheit $\tilde{K} = \{1, 2, \ldots, k\} \subseteq \tilde{V}$ mit $k \in \mathbb{N}$. Nach Hinzufügen von k neuen Knoten $K' := \{1', 2', \ldots, k'\}$ sei $G = (V, E)$ gegeben durch $V := \tilde{V} \cup \{1', 2', \ldots, k'\}$ und $E := \tilde{E} \cup \left\{[i, i'] : i \in \tilde{K}\right\}$ mit $c_{ij} := 1\ \forall\ [i,j] \in E$. Es seien $OD := \left\{\{j, j'\} : j \in \tilde{K}\right\}$ mit $w_{jj'} := 1\ \forall\ \{j, j'\} \in OD$ die OD-Paare. Sei $\alpha := 0$, $l_0 := \frac{\tilde{l}_0}{c} + k$ und $m_0 := k$.
Ist $\tilde{l}_0 \geq l(G)$, so ist jeder spannende Baum von $\tilde{G}$ Lösung des Steinerbaumproblems. Weiter betrachtet werden muss also nur der Fall $\tilde{l}_0 < l(G) < cn^2$. Dann ist $l_0 = \frac{\tilde{l}_0}{c} + k < \frac{cn^2}{c} + n = n^2 + n$, das heißt, da auch $k < n$, die Zahleinträge der Instanzen $I = (T, OD, 0, l_0, m_0)$ sind polynomiell in n beschränkt.

Behauptung 65 *Es gibt genau dann eine Lösung der Instanz* $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ *von STEINER TREE, wenn es eine Lösung der Instanz* $(G, OD, 0, l_0, m_0)$ *von B-zGraph-Baum gibt.*

Sei $\tilde{T}$ eine Lösung der Instanz $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ von STEINER TREE. Dann gilt $l(\tilde{T}) \leq \tilde{l}_0$, $\tilde{T}$ ist ein Baum und $v \in \tilde{T}\ \forall v \in \tilde{K}$. Man betrachte nun $T = \left(V_{\tilde{T}} \cup K', E_{\tilde{T}} \cup \left\{[j, j'] : j \in \tilde{K}\right\}\right)$. Da jedes $j' \in K'$ Kantengrad 1 hat, können durch das Hinzufügen der Ecken und Kanten bei der Konstruktion von T keine Kreise entstehen. Da $\tilde{T}$ ein Baum ist, ist auch T ein Baum.
Außerdem gilt:

$$l(T) = \frac{l(\tilde{T})}{c} + \sum_{j \in \tilde{K}} c_{jj'} = \frac{l(\tilde{T})}{c} + k \leq \frac{\tilde{l}_0}{c} + k = l_0 \tag{76}$$

und

$$\begin{aligned}\sum_{\{u,v\}\in OD} w_{uv}(d(u,v) - d_T(u,v)) = & \sum_{j\in\tilde{K}} w_{jj'}(d(j,j') - d_T(j,j')) \\ = & \sum_{j\in\tilde{K}} 1\cdot(1-0) \\ = & \ k. \end{aligned} \tag{77}$$

Daraus folgt, dass T Lösung von $(G, OD, 0, l_0, m_0)$ ist.
Ist andererseits $T = (V, E)$ eine Lösung von $(G, OD, 0, l_0, m_0)$, so ist T ein Baum,

$$l(T) \leq l_0 \tag{78}$$

und

$$\sum_{\{u,v\}\in OD} w_{uv}(d(u,v) - d_T(u,v)) \geq k. \tag{79}$$

Man bemerke, dass $[j, j'] \in E_T$, da sonst

$$\begin{aligned}\sum_{\{u,v\}\in OD} w_{uv}(d(u,v) - d_T(u,v)) = & \sum_{j\in\tilde{K}} 1\cdot(1-d_T(u,v)) \\ < & \sum_{j\in\tilde{K}} 1 \\ = & \ k, \end{aligned} \tag{80}$$

was ein Widerspruch zu (79) ist. Also ist $v \in E \ \forall \ v \in \tilde{K}$.
Der Graph $\tilde{T}$, entstehend durch $V_{\tilde{T}} = V \setminus K'$ und $\tilde{E} = E \setminus \left\{[j, j'] : j \in \tilde{K}, j' \in K'\right\}$, ist weiterhin zusammenhängend und deshalb ein Baum. Außerdem gilt:

$$l(\tilde{T}) = l(T)c - \sum_{j\in\tilde{K}} c_{jj'} = l_0 c + k - k = \tilde{l}_0. \tag{81}$$

$\Rightarrow \tilde{T}$ ist Lösung von $(\tilde{G}, \tilde{K}, \tilde{l}_0)$.
Also hat die Instanz $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ von STEINER TREE genau dann eine Lösung, wenn die Instanz $(G, OD, 0, l_0, m_0)$ von *B-zGraph-Baum* eine Lösung hat.
$\Rightarrow$ Das Entscheidungsproblem *B-zGraph-Baum* ist stark NP-vollständig. ■

5.3 Bestimmung optimaler Kreise

Wie in Problem A lohnt auch hier nur die Betrachtung des Problems *B-zGraph-Kreis*.

Satz 66 *Das Entscheidungsproblem B-zGraph-Kreis ist stark NP-vollständig.*

Der Beweis erfolgt durch Reduktion des NP-vollständigen Entscheidungsproblem HAMILTONIAN CIRCUIT (siehe Definition 30) auf ein Teilproblem von *B-zGraph-Kreis*, analog zum Beweis von Satz 51.

Beweis. Sei $(\tilde{G})$ mit $\tilde{G} = (\tilde{V}, \tilde{E})$ und $\tilde{V} := \{1, 2, \ldots, n\}$ eine Instanz von HAMILTONIAN CIRCUIT. Dann definiere man $G = (V, E)$ durch $V := \tilde{V} \cup V'$, wobei $V' := \{1', 2', \ldots, n'\}$ und $E := \tilde{E} \cup E'$ mit $E' := \left\{[i, i'] : i \in \tilde{V}\right\}$.

Die Kantenlängen seien gegeben durch $c_{ij} := 1$ für $[i, j] \in \tilde{E}$ und $c_{ii'} := n^2+1$ für $[i, i'] \in E'$. Die Menge der OD-Paare sei $OD := \{\{i', (i+1)'\} : i = 1, 2, \ldots, n-1\} \cup \{\{n', 1'\}\}$ mit Nachfrage $w_{uv} := 1 \; \forall \{u, v\} \in OD$. Sei $m_0 := \sum_{\{u,v\} \in OD}(d(u, v) - 2(n^2 + 1))$, $l_0 := n^2$ und $\alpha := 0$.

Die Konstruktion von G kann in Polynomialzeit erfolgen, da die Berechnung von $d(u, v)$, das heißt das Bestimmen eines kürzesten Weges zwischen u und v, nach Lemma 9 in $O(n)$ ist.

Da wie im Beweis zu Satz 51 $m_0 < n^2$, sind alle Zahlen in der Instanz $(T, OD, 0, n^2, m_0)$ polynomiell beschränkt in n.

Behauptung 67 *Es gibt genau dann eine Lösung der Instanz $(\tilde{G})$ von HAMILTONIAN CIRCUIT, wenn es eine Lösung der Instanz $(G, OD, 0, n^2, m_0)$ von B-zGraph-Kreis gibt.*

Sei $K = (V_K, E_K)$ eine Lösung von $(\tilde{G})$. Dann ist K ein Kreis und da $E_K \subset \tilde{E}$, gilt

$$l(K) \leq \sum_{[i,j] \in \tilde{E}} c_{ij} = \sum_{[i,j] \in \tilde{E}} 1 < n^2. \tag{82}$$

Außerdem ist

$$
\begin{aligned}
&\sum_{\{u,v\}\in OD} (d(u,v) - d_K(u,v)) \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \sum_{\{u,v\}\in OD} d_K(u,v) \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \Big[\Big(\sum_{j=1}^{n-1} d_K(j',(j+1)')\Big) + d_K(n',1')\Big] \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \Big[\Big(\sum_{j=1}^{n-1} d_K(j',j) + d_K(j,j+1) + d_K(j+1,(j+1)')\Big) \\
&+ d_K(n',n) + d_K(n,1) + d_K(1,1')\Big] \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \Big[\Big(\sum_{j=1}^{n-1}(n^2+1) + 0 + (n^2+1)\Big) + (n^2+1) + 0 + (n^2+1)\Big] \\
=&\sum_{\{u,v\}\in OD} d(u,v) - \sum_{j=1}^{n} 2(n^2+1) \\
=&\sum_{\{u,v\}\in OD} \Big(d(u,v) - 2(n^2+n)\Big) \\
=&m_0. \qquad (83)
\end{aligned}
$$

$\Rightarrow K$ ist Lösung der Instanz $(G, OD, 0, n^2, m_0)$ von *B-zGraph-Kreis*.
Sei nun $K = (V_K, E_K)$ eine Lösung von $(G, OD, 0, n^2, m_0)$. Dann ist K ein Kreis,

$$l(K) \leq n^2 = l(\tilde{G}) \qquad (84)$$

und

$$\sum_{\{u,v\}\in OD} (d(u,v) - d_K(u,v)) \geq m_0. \qquad (85)$$

Wegen (84) ist $[i,i'] \notin E_K \ \forall\ i \in V_{\tilde{G}}$, da sonst

$$l(K) \geq c_{ii'} = n^2 + 1 > n^2. \qquad (86)$$

Wegen (85) gibt es kein $i \in V$, so dass $K = (\{i'\}, \emptyset)$, denn sonst wäre $d_K(u,v) = d(u,v)$ für alle $\{u,v\} \in OD$ und $\sum_{\{u,v\}\in OD} d(u,v) - d_k(u,v) = 0$. Also ist $K \subset G$.
Angenommen K ist kein Hamiltonscher Kreis in $\tilde{G}$, das heißt $\exists i \in \tilde{V} :\ i \notin V_K$. Dann ist wegen

$c_{ij} > 0 \; \forall [i,j] \in \tilde{E}$:

$$\begin{aligned} d_K(i', (i+1)') =& d_K(i', i) + d_K(i, i+1) + d_K(i+1, (i+1)') \\ =& (n^2+1) + d_K(i, i+1) + (n^2+1) \\ >& 2(n^2+1). \end{aligned} \tag{87}$$

$\Rightarrow$

$$d(i', (i+1)') - d_K(i', (i+1)') < d(i', (i+1)') - 2(n^2+1) \tag{88}$$

Daraus folgt:

$$\begin{aligned} & \sum_{\{l,u\} \in OD} (d(u,v) - d_K(u,v)) \\ =& \sum_{j=1}^{n-1} (d(j', (j+1)') - d_K(j, (j+1)')) + d(n', 1') - d_K(n', 1') \\ <& \sum_{j=1}^{n-1} (d(j', (j+1)') - 2(n^2+1)) + d(n', 1') - 2(n^2+1) \\ =& \sum_{\{u,v\} \in OD} (d(u,v) - 2(n^2+1)) \\ =& m_0. \end{aligned} \tag{89}$$

Also ist $\sum_{\{u,v\} \in OD} (d(u,v) - d_K(u,v)) < m_0$. Das ist ein Widerspruch zu (85). Es folgt $i \in V_K \; \forall \; i \in \tilde{V}$.

$\Rightarrow K$ ist ein Hamiltonscher Kreis in $\tilde{G}$.

Die Instanz $(\tilde{G})$ von HAMILTONIAN CIRCUIT hat also genau dann eine Lösung, wenn die Instanz $(G, OD, 0, n^2, m_0)$ von *B-zGraph-Kreis* eine Lösung hat.

$\Rightarrow$ *B-zGraph-Kreis* ist stark NP-vollständig. ■

5.4 Bestimmung optimaler zusammenhängender Teilgraphen

Bei der Bestimmung optimaler zusammenhängender Teilgraphen kann man sich auf das Problem *B-zGraph-zGraph* beschränken (siehe Kapitel 3.2).

Satz 68 *Das Entscheidungsproblem B-zGraph-zGraph ist stark NP-vollständig.*

Beweis. Sei T Lösung einer Instanz $(G, OD, \alpha, l_0, m_0)$ von *B-zGraph-Baum*. Dann löst T insbesondere auch die Instanz $(G, OD, \alpha, l_0, m_0)$ von *B-zGraph-zGraph*.

Ist andersherum g Lösung einer Instanz $(G, OD, 0, l_0, m_0)$ von *B-zGraph-zGraph* mit $\alpha = 0$, dann ist jeder spannende Baum von g ebenfalls Lösung von $(G, OD, \alpha, l_0, m_0)$. Für Instanzen mit $\alpha = 0$ ist also die Existenz einer Lösung von *B-zGraph-zGraph* äquivalent zu der Existenz einer Lösung von *B-zGraph-Baum*.
Da bei der Reduktion von STEINER TREE im Beweis der starken NP-Vollständigkeit von *B-zGraph-Baum* (Satz 61) nur Instanzen mit $\alpha = 0$ betrachtet werden, zeigt dieser Beweis gleichzeitig die starke NP-Vollständigkeit von *B-zGraph-zGraph*. ■

5.5 Bestimmung optimaler allgemeiner Teilgraphen

5.5.1 B-Pfad-Graph

Satz 69 *Das Problem B-Pfad-Graph ist NP-vollständig.*

Der Beweis erfolgt durch Reduktion des Entscheidungsproblems KNAPSACK (siehe Definition 54).

Beweis. (von Satz 69)
Sei (O, C, N) für $O := \{o_1, o_2, \ldots, o_n\}$ für $n \in \mathbb{N}$ mit zugeordnetem Nutzen a_i und Gewicht b_i für $o_i \in O$ eine Instanz von KNAPSACK. Durch $V := \{0, 1, \ldots, n\}$ und $E := \{[i-1, i] : i = 1, \ldots, n\}$ ist ein Pfad $P = (V, E)$ definiert. Die Kantenlängen seien gegeben durch $c_{i-1i} := b_i$. Sei $X = \prod_{i=1}^{n} b_i$. Die Menge der OD-Paare sei $OD := \{\{i-1, i\} : i = 1, \ldots, n\}$ mit $w_{i-1i} := \frac{a_i X}{c_{i-1i}}$.
Sei $\alpha := 0, l_0 := C$ und $\acute{m}_0 = NX$. Betrachtet wird nun also die Instanz $(P, OD, 0, C, (1-\alpha)NX)$ des Entscheidungsproblems *B-Pfad-Graph*.

Behauptung 70 *Die Instanz (O, C, N) von KNAPSACK hat genau dann eine Lösung, wenn die Instanz $(P, OD, 0, C, (1-\alpha)NX)$ von B-Pfad-Graph eine Lösung hat.*

Da in einem Pfad die Wege zwischen den Knoten eindeutig bestimmt sind, lässt sich für *B-Pfad-Graph* die Charakterisierung (53) der Zielfunktion verwenden. Angenommen die Teilmenge $R \subset O$ löst (O, C, N). Dann gilt:

$$\sum_{o_i \in R} a_i \geq N, \tag{90}$$

$$\sum_{o_i \in R} b_i \leq C. \tag{91}$$

Man betrachte nun die Instanz $(P, OD, 0, C, (1-\alpha)NX)$ von *B-Pfad-Graph*.
Sei der Teilgraph g definiert durch $g = (V_g, E_g)$ mit $V_g := \{j \in V : o_j \in R \text{ oder } o_{j+1} \in R\}$ und

$E_g := \{[i-1,i] : o_i \in R\}$.
Es gilt:

$$\begin{aligned}\sum_{[i,j]\in E_g} c_{ij} &= \sum_{[i-1,i]\in E_g} c_{i-1i} \\ &= \sum_{[i-1,i]\in E_g} b_i \\ &= \sum_{o_i\in R} b_i \\ &\leq C \end{aligned} \tag{92}$$

und

$$\begin{aligned}\sum_{[i,j]\in E_g} m_{ij}c_{ij} &= \sum_{[i,j]\in E_g} \Big(\sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv}\Big)c_{ij} \\ &= \sum_{[i-1,i]\in E_g} \Big(\sum_{\{u,v\}\in OD:[i-1,i]\in W_{uv}} w_{uv}\Big)c_{i-1i} \\ &= \sum_{[i-1,i]\in E_g} w_{i-1i}c_{i-1i} \\ &= \sum_{[i-1,i]\in E_g} \frac{a_iX}{c_{i-1i}}c_{i-1i} \\ &= \sum_{[i-1,i]\in E_g} a_iX \\ &= \Big(\sum_{o_i\in R} a_i\Big)X \\ &\geq NX, \end{aligned} \tag{93}$$

also $(1-\alpha)\sum_{[i,j]\in E_g} m_{ij}c_{ij} \leq (1-\alpha)NX$. Daraus folgt, dass g die Instanz $(P, OD, 0, C, NX)$ löst.
Sei nun $g = (V_g, E_g)$ eine Lösung von $(P, OD, 0, C, (1-\alpha)NX)$. Sei $R := \{o_i \in O : [i-1,i] \in E_g\}$. Dann gilt mit ähnlichen Rechnungen wie vorher:

$$\sum_{o_i\in R} b_i = \sum_{[i-1,i]\in E_g} c_{i-1i} \leq C \tag{94}$$

und

$$\begin{aligned}\sum_{o_i \in R} a_i &= \sum_{[i-1,i]\in E_g} \frac{w_{i-1i}c_{i-1i}}{X} \\ &= \Big(\sum_{[i-1,i]\in E_g} \big(\sum_{\{u,v\}\in OD:[i-1,i]\in W_{uv}} w_{uv} \big) c_{i-1i} \Big) / X \\ &= \Big(\sum_{[i,j]\in E_g} \big(\sum_{\{u,v\}\in OD:[i,j]\in W_{uv}} w_{uv} \big) c_{ij} \Big) / X \\ &\geq N \end{aligned} \tag{95}$$

(O, C, N) hat also genau dann eine Lösung, wenn $(P, OD, 0, C, (1-\alpha)NX)$ eine Lösung hat.
$\Rightarrow$ Das Entscheidungsproblem *B-Pfad-Graph* ist NP-vollständig. ■

In folgendem Abschnitt wird die pseudopolynomielle Lösbarkeit des Optimierungsproblems *B-Baum-Graph* bewiesen. Da *B-Pfad-Graph* ein Unterproblem von *B-Baum-Graph* ist, sind die dort erzielten Ergebnisse (Satz 73) auch auf das Optimierungsproblem *B-Pfad-Graph* anwendbar.

5.5.2 B-Baum-Graph

Wie in Satz 69 durch Reduktion des Rucksackproblems gezeigt, ist das Entscheidungsproblem *B-Pfad-Graph* NP-vollständig. Da ein Pfad insbesondere eine Baum ist, gilt:

Satz 71 *Das Entscheidungsproblem B-Baum-Graph ist NP-vollständig.*

Wie sich im Folgenden zeigen wird, lässt sich aber auch umgekehrt das Problem *B-Baum-Graph* (und damit auch das Unterproblem *B-Pfad-Graph*) in ein Rucksackproblem überführen und so lösen.
Sei (T, OD, α, l_0) eine Instanz des Optimierungsproblems *B-Baum-Graph* mit $T = (V, E)$. Sei ohne Beschränkung der Allgemeinheit T rekursiv dargestellt, wobei der erste Knoten abweichend von Definition 3 mit 0 bezeichnet sei, also $V = \{0, 1, 2, \ldots, n-1\}$. Die Kantenlängen des rekursiv dargestellten Baumes T seien für $i = 1, \ldots, n$ mit $c_{iC(i)}$ bezeichnet. Nach Lemma 46 lassen sich die Kantennachfragen $m_{iC(i)}$ für alle Kanten $[i, C(i)] \in E$ in $O(n^2)$ berechnen.
Der Instanz (T, OD, α, l_0) sei nun eine Instanz von KNAPSACK zugeordnet, die konstruiert wird durch $O := \{o_i : i = 1, \ldots, n\}$ mit $b_i := c_{iC(i)}$, $a_i := m_{iC(i)}c_{iC(i)}$ und $C := l_0$. Man beachte, dass alle diese Werte, wie in 54 gefordert, ganzzahlig sind.
Sei nun für eine Lösung $g = (V_g, E_g)$ von (T, OD, α, l_0) die Menge $R_g := \{o_i : [i, C(i)] \in E_g\} \subset O$ die entsprechende Lösung von (O, l_0). Umgekehrt sei R eine Lösung von (O, l_0). Dann ist

$g = (V_R, E_R) \subset T$ mit $E_R := \{[i, C(i)] : o_i \in R\}$ und V_R entsprechend die zugehörige Lösung von (T, OD, α, l_0).

Satz 72 *Sei (T, OD, α, l_0) eine Instanz von B-Baum-Baum und (O, l_0), wie oben konstruiert, die zugehörige Instanz von KNAPSACK. Seien g und R zwei einander zugeordnete Lösungen. Dann gilt für die Zielfunktionen:*

$$\sum_{\{u,v\} \in OD} w_{uv}(d(u,v) - d_g(u,v)) = (1-\alpha) \sum_{o_i \in R} a_i \tag{96}$$

beziehungsweise

$$\sum_{[i,j] \in E_g} c_{ij} m_{ij} = \sum_{o_i \in R} a_i. \tag{97}$$

Weiterhin ist R genau dann eine Optimallösung für (O, l_0), wenn g eine Optimallösung für (T, OD, α, l_0) ist.

Beweis. Seien (O, l_0) und (T, OD, α, l_0) ein Paar von Instanzen und R und g ein Paar von Lösungen wie oben.
Dann gilt für die Zielfunktionswerte:

$$\sum_{\{u,v\} \in OD} w_{uv}(d(u,v) - d_g(u,v)) = (1-\alpha) \sum_{[i,C(i)] \in g} c_{ij} m_{iC(i)} = (1-\alpha) \sum_{o_i \in R} a_i \tag{98}$$

nach Lemma 44 und Konstruktion der Paare von Instanzen von KNAPSACK und *B-Baum-Graph*. Ist R zulässig, so ist $l(g) = \sum_{[i,C(i)] \in E_R} c_{iC(i)} = \sum_{o_i \in R} b_i \leq l_0$, also ist g eine zulässige Lösung von (T, OD, α, l_0). Ist umgekehrt g zulässige Lösung von (T, OD, α, l_0), dann ist $\sum_{o_i \in R} b_i = \sum_{[i,C(i)] \in E_R} c_{iC(i)} \leq l_0$, daher ist R eine zulässige Lösung von (O, l_0). Daraus folgt die zweite Aussage des Satzes. ■

Das Optimierungsproblem *B-Baum-Baum* ist also nach Berechnung der Kantennachfragen m_{ij} ein Rucksackproblem mit Gewichten c_{ij} und Nutzen $m_{ij}c_{ij}$ für $[i, j] \in E$. Daher lassen sich alle Aussagen zur Lösbarkeit des Rucksackproblems auf *B-Baum-Graph* übertragen. Unter Anderem gilt also der folgende Satz:

Satz 73 *Sei (T, OD, α, l_0) eine Instanz des Optimierungsproblems B-Baum-Graph mit rekursiv dargestelltem Baum $T = (V, E)$, Kantenlängen $c_{iC(i)}$ und bekannter Kantennachfrage $m_{iC(i)}$ für alle $[i, C(i)] \in E$. Dann gilt:*

- *(T, OD, α, l_0) ist in $O(nl_0)$ exakt lösbar.*

- (T, OD, α, l_0) *ist in* $O(nC)$ *mit* $C = \sum_{[i,C(i)]\in E} c_{iC(i)}$ *exakt lösbar.*

Beweis. Die Aussagen folgen aus analogen Aussagen über das Rucksackproblem (siehe zum Beispiel [KV08, 439ff.]). ■

Da eine rekursive Darstellung von T, ebenso wie die Berechnung der $m_{iC(i)}$ in $O(n^2)$ erfolgen kann (siehe Lemma 4, Bemerkung 6 und Lemma 46), ist im Allgemeinen eine exakte Lösung einer Instanz (T, OD, α, l_0) des Optimierungsproblems *B-Baum-Baum* in $O(n^2 + nl_0)$ oder $O(n^2 + nC)$ mit $C = \sum_{[i,C(i)]\in E} c_{iC(i)}$ möglich.

5.5.3 Gegeben: Kreis

Satz 74 *Das Entscheidungsproblem B-Kreis-Graph ist NP-vollständig.*

Der Beweis des Satzes erfolgt durch Reduktion von *B-Kreis-Graph* auf *B-Pfad-Graph*.
Beweis. Sei die Instanz (P, OD, α, l_0, S) von *B-Pfad-Graph* mit $P = (V_P, E_P)$ gegeben, wobei $V = \{1, 2, ..., n\}$ und $E = \{[i, i+1] : i = 1, \ldots, n-1\}$. Durch Hinzufügen der Kante $[n, 1]$ mit Kantenlänge $c_{n1} = \max\{l_0, a_0\} + 1$ erhält man aus P den Kreis $K = (V_K, E_K)$. Für K gilt also $V_K = V_P$, $E_K = E_P \cup [n, 1]$.

Behauptung 75 *Es gibt genau dann eine Lösung der Instanz* $(K, OD, \alpha, l_0, m_0)$ *von B-Kreis-Graph, wenn es eine Lösung der Instanz* $(P, OD, \alpha, l_0, m_0)$ *von B-Pfad-Graph gibt.*

Sei $g \subset P$ eine Lösung von $(P, OD, \alpha, l_0, m_0)$. Dann ist g offensichtlich auch eine Lösung von $(K, OD, \alpha, l_0, m_0)$.
Sei umgekehrt g eine Lösung von $(K, OD, \alpha, l_0, m_0)$. Dann ist $l(g) \leq l_0 < c_{n1}$, also $[n, 1] \notin E_g$. Sei für ein OD-Paar $\{u, v\}$ $W_{uv}^{K_g}$ ein kürzester Weg zwischen u und v in K_g. Dann ist $[n, 1] \notin E_{W_{uv}^{K_g}}$, da $[n, 1] \notin E_g$ und

$$d_g^K(u, v) \leq d^K(u, v) \leq l(P) < c_{n1} \tag{99}$$

nach Lemma 12. Also ist $W_{uv}^{K_g} \subset P$ für alle $\{u, v\} \in OD$. Daraus folgt

$$d_g^K(u, v) = d_g^P(u, v) \tag{100}$$

und äquivalent gilt:

$$\sum_{\{u,v\}\in OD} d^K(u, v) - d_g^K(u, v) = \sum_{\{u,v\}\in OD} d^P(u, v) - d_g^P(u, v) \leq m_0. \tag{101}$$

Daher ist g auch Lösung von $(P, OD, \alpha, l_0, m_0)$.
Ein Graph g löst also genau dann die Instanz $(P, OD, \alpha, l_0, m_0)$ von Problem *B-Pfad-Graph*, wenn er die Instanz $(K, OD, \alpha, l_0, m_0)$ von *B-Kreis-Graph* mit dem entsprechend konstruiertem K löst.
$\Rightarrow$ Das Entscheidungsproblem *A-Kreis-Graph* ist NP-vollständig. ■

5.5.4 Gegeben: zusammenhängender Graph

Da das Problem *B-Pfad-Graph* schon NP-vollständig ist und ein Pfad insbesondere ein zusammenhängender Graph ist, folgt aus der NP-Vollständigkeit von *B-Pfad-Graph* die NP-Vollständigkeit von *B-zGraph-Graph*. Es lässt sich durch eine Modifikation des Beweises von Satz 61 aber sogar starke NP-Vollständigkeit beweisen.

Satz 76 *Das Entscheidungsproblem B-zGraph-Graph ist stark NP-vollständig.*

Beweis. Sei $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ eine Instanz von STEINER TREE mit $\tilde{G} = (\tilde{V}, \tilde{E})$ $c \in \mathbb{Z}^+$ und $\tilde{c}_{ij} = c \; \forall \; [i,j] \in \tilde{E}$. Daraus wird folgendermaßen eine Instanz $(G, OD, \alpha, l_0, m_0)$ von *B-zGraph-Baum* konstruiert:
Sei $\tilde{K} = \{1, 2, \ldots, k\} \subseteq \tilde{V}$ mit $k \in \mathbb{N}$. Nach Hinzufügen von k neuen Knoten $K' := \{1', 2', \ldots, k'\}$ sei $G = (V, E)$ gegeben durch $V := \tilde{V} \cup \{1', 2', \ldots, k'\}$ und $E := \tilde{E} \cup \bigcup_{i \in \tilde{K}} \{[i, i']\}$ mit $c_{ij} := 1$ für alle $[i,j] \in E$. Die OD-Paare seien gegeben durch $OD := \{\{i', j'\} : i', j' \in K'\}$ mit $w_{i'j'} := 1$.
Sei $\alpha := 0$, $l_0 := \frac{\tilde{l}_0}{c} + k$ und $m_0 := \sum_{i', j' \in K'} d(i', j')$.
Es ist $l_0 = \frac{\tilde{l}_0}{c} + k < \frac{n^2 c}{c} + n = n^2 + n$ und $m_0 \leq n^2$, da

$$d(i', j') + d(i, i') + d(i, j) + d(j, j') \leq 1 + n + 1 \tag{102}$$

und $|K'| = |K| \leq n$. Deshalb sind die numerischen Einträge in der konstruierten Instanz $I = (T, OD, 0, l_0, m_0)$ polynomiell in n beschränkt.

Behauptung 77 *Es gibt genau dann eine Lösung der Instanz $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ von STEINER TREE, wenn es eine Lösung der Instanz $(G, OD, 0, l_0, m_0)$ von B-zGraph-Graph gibt.*

Sei $\tilde{T}$ eine Lösung der Instanz $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ von STEINER TREE. Dann gilt $l(\tilde{T}) \leq \tilde{l}_0$ und $i \in \tilde{T}$ für alle $i \in \tilde{K}$. Man betrachte $g = (V_{\tilde{T}} \cup K', E_{\tilde{T}} \cup \{[j, j'] : j \in K'\})$. Es gilt:

$$l(g) = \frac{l(\tilde{T})}{c} + \sum_{j \in \tilde{K}} c_{jj'} = \frac{l(\tilde{T})}{c} + k \leq \frac{\tilde{l}_0}{c} + k = l_0. \tag{103}$$

Für alle i, j in K gilt nach Konstruktion von g: $d_g(i,j) = 0$, $d_g(i',i) = 0$ und $d_g(j',j) = 0$, also auch $d_g(i',j') = d_g(i',i) + d_g(i,j) + d_g(j,j') = 0$. Daraus folgt:

$$\begin{aligned}\sum_{\{i',j'\}\in OD} w_{i'j'}(d(i',j') - d_g(i',j')) &= \sum_{\{i',j'\}\in OD} w_{i'j'}d(i',j') \\ &= \sum_{\{i',j'\}\in OD} d(i',j') \\ &= m_0. \end{aligned} \tag{104}$$

Also ist g Lösung der Instanz $(G, OD, 0, l_0, m_0)$ von *B-zGraph-Graph*.
Ist g nun eine Lösung von $(G, OD, 0, l_0, m_0)$, so gilt:

$$l(g) \leq l_0 \tag{105}$$

und

$$\sum_{\{u,v\}\in OD} w_{uv}(d(u,v) - d_T(u,v)) \geq m_0. \tag{106}$$

Da $m_0 = \sum_{i',j'\in K'} d(i',j')$ und $OD = \{\{i',j'\} : i',j' \in K'\}$ mit $w_{i'j'} = 1$ folgt durch Einsetzen:

$$\Big(\sum_{\{i',j'\}\in OD} d(i',j') - d_g(i',j')\Big) \geq \sum_{\{i',j'\}\in OD} d(i',j'), \tag{107}$$

also ist $d_g(i',j') = 0 \;\forall\; i',j' \in K'$.
Daher muss es zwischen allen $i',j' \in K'$ einen Weg geben, der in g enthalten ist. Das heißt, dass g einen Baum $\hat{T}$ beinhaltet, der alle $i,j \in K$ enthält und für den gilt:

$$l(\hat{T}) \leq l(g) \leq l_0. \tag{108}$$

Sei $\tilde{T}$ der Baum $\hat{T}$ nach Entfernen der Kanten $[i,i'] \;\forall i \in K$, aufgefasst als Baum in $\tilde{G}$ (also mit Kantenlängen c). Dann enthält $\tilde{T}$ jeden Knoten aus K und es gilt, da $k \geq 1$:

$$l(\tilde{T}) = c\Big(l(\hat{T}) - \sum_{i\in K} c_{ii'}\Big) \leq c(l(g) - k) \leq c(l_0 - k) \leq cl_0 - k = \tilde{l_0} \tag{109}$$

$\Rightarrow \tilde{T}$ ist Lösung von $(\tilde{G}, \tilde{K}, \tilde{l}_0)$.
Die Instanz $(\tilde{G}, \tilde{K}, \tilde{l}_0)$ von STEINER TREE hat daher genau dann eine Lösung, wenn die Instanz $(G, OD, 0, l_0, m_0)$ von *B-zGraph-Graph* mit polynomiell in n beschränkten Einträgen eine Lösung hat.
Also ist das Entscheidungsproblem *B-zGraph-Graph* stark NP-vollständig. ■

5.6 Ergebnis

gesucht \ gegeben	Pfad	Baum	Kreis	zGraph
Pfad	$O(n^2)$	$O(n^3)$	$O(n^4)$	stark NP-vollständig
Baum	$O(n^2)$	NP-vollständig (pseudopolynomiell lösbar)	$O(n^4)$	stark NP-vollständig
Kreis	nicht möglich	nicht möglich	$O(1)$	stark NP-vollständig
zGraph	$O(n^2)$	NP-vollständig (pseudopolynomiell lösbar)	$O(n^4)$	stark NP-vollständig
Graph	NP-vollständig (pseudopolynomiell lösbar)	NP-vollständig (pseudopolynomiell lösbar)	NP-vollständig	stark NP-vollständig

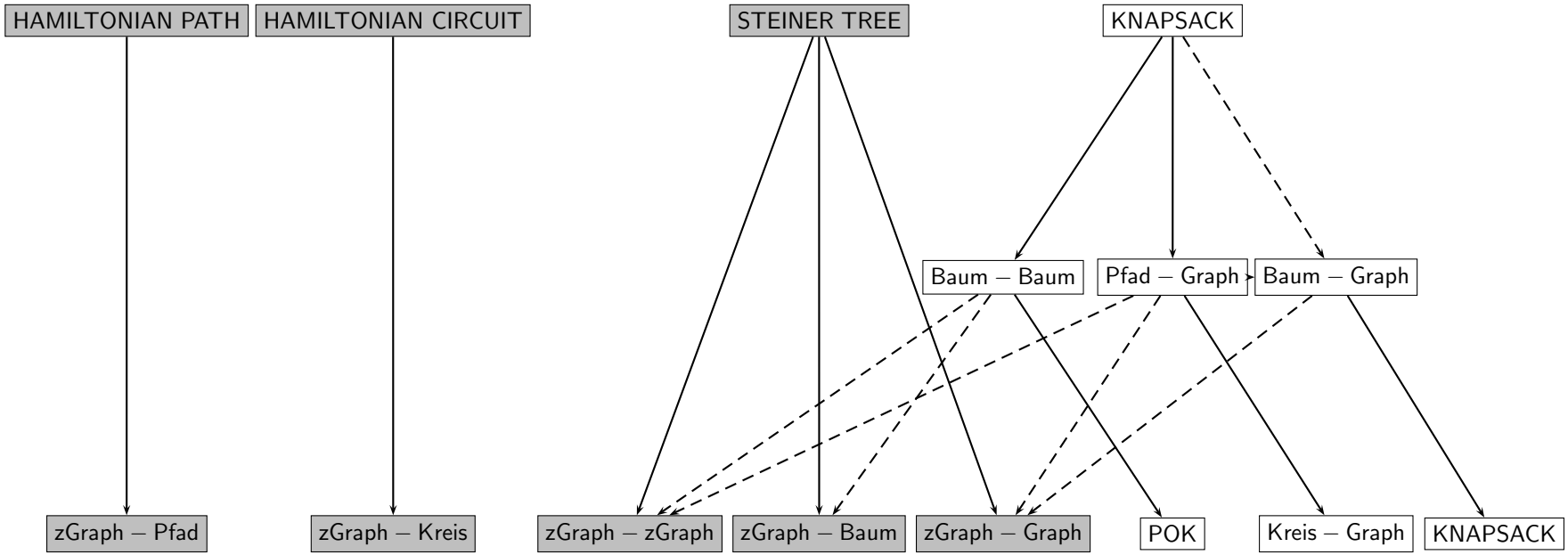

Abbildung 7: Übersicht über die NP-vollständigen Unterprobleme von Problem B. Die durchgezogenen Pfeile kennzeichnen in dieser Arbeit durchgeführte Reduktionen, gestrichelte Pfeile kennzeichnen einige ebenfalls mögliche Reduktionen zwischen den Problemen. Die grauen Felder kennzeichnen Probleme, die sogar stark NP-vollständig sind. Die Abkürzung „POK“ steht für das Problem PARTALLY ORDERED KNAPSACK.

6 Lösungsansätze für Problem *A-Baum-Baum*

6.1 Unterbäume

In einem Baum als unterliegendem Graph ist der Weg zwischen zwei Knoten i und j eindeutig bestimmt. Er wird im Folgenden auch durch die ihn bildende Knotensequenz oder verkürzt als (i, j), also durch Angabe von Start- und Zielknoten, dargestellt.
Es werden kürzeste Wege zwischen Graphen definiert:

Definition 78 Kürzeste Wege zwischen Graphen
Sei G ein Graph, $g = (V_g, E_g)$ und $g' = (V_{g'}, E_{g'})$ seien zwei Teilgraphen. Einen Weg $W_{g,g'}$, der unter den Wegen mit Startknoten in V_g und Endknoten in $V_{g'}$ minimale Länge hat, bezeichne man als einen kürzesten Weg zwischen g und g' in G. Gibt es ein $p \in V_g \cap V_{g'}$, so wird der nur aus dem Knoten p bestehende Teilgraph als kürzester Weg mit Länge 0 bezeichnet.

Bemerkung 79 *Ist T ein Baum mit Unterbäumen $T_1 = (V_{T_1}, E_{T_1})$ und $T_2 = (V_{T_2}, E_{T_2})$ mit $E_{T_1} \cap E_{T_2} = \emptyset$, dann gibt es genau einen Weg $W_{T_1,T_2} = (u_1, u_2, \ldots, u_m)$ mit $u_1 \in T_1$, $u_m \in T_2$ und $u_i \notin T_1 \cup T_2 \; \forall \; 1 \neq i \neq m$. Dieser Weg ist unter allen Wegen zwischen T_1 und T_2 inklusions- und längenminimal. Wie in obiger Definition besteht W_{T_1,T_2} im Fall $V_{T_1} \cap V_{T_2} \neq \emptyset$ nur aus einem Knoten. Man beachte, dass $|V_{T_1} \cap V_{T_2}| \leq 1$, da sonst $E_{T_1} \cap E_{T_2} \neq \emptyset$.*

Sei $(T, OD, \alpha, l_0, d_0)$ eine Instanz des Entscheidungsproblems *A-Baum-Baum*. Die Menge der kritischen OD-Paare sei $OD_{krit} := \{\{u, v\} \in OD : d(u, v) > d_0\}$, das heißt die Menge all der OD-Paare, deren Entfernung durch eine Lösung des Entscheidungsproblems verringert werden muss.
Haben diese kritischen OD-Paare nun die Eigenschaft, dass sie keine gemeinsame Kante haben, also

$$\bigcap_{i=1}^{|OD_{krit}|} E_{(u_i,v_i)} = \emptyset, \tag{110}$$

dann lässt sich mit Hilfe des folgenden Algorithmus ein Unterbaum T' von T konstruieren, der in jeder Lösung von (T, OD, α, l_0, d) mit $d \leq d_0$ enthalten sein muss. Dabei wird ausgenutzt, dass der entstehende Teilbaum zusammenhängend sein muss und somit mindestens die kürzesten Wege zwischen den kritischen OD-Paaren enthalten muss. Ist Eigenschaft 110 nicht erfüllt, lässt sich kein eindeutiger verbindender Unterbaum finden.

Algorithmus 3 Konstruktion Unterbaum

Eingabe: Baum $T = (V, E)$ mit Kantenlängen c_{ij} $\forall [i, j] \in E$; $OD = \{\{u, v\} : u \leq v\}$; l_0; d_0.

Ausgabe: Baum T' mit $l(T') \leq l_0$, der Unterbaum in jeder optimalen Lösung von (T, OD, α, l_0, d) mit $d \leq d_0$ ist.

→ Initialisierung

1: $T' = \emptyset$

2: $OD_{krit} = OD \setminus \{\{u, v\} : d(u, v) \leq d_0\}$

→ Erstellen des Grundpfades

3: **for** $j = 2, \ldots, |OD_{krit}|$ **do**

4: **if** $\bigcap_{i=1}^{j} E_{(u_i,v_i)} = \emptyset$ **then**

5: Bestimme kürzesten Weg $W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)}$ zwischen $\bigcap_{i=1}^{j-1}(u_i, v_i)$ und (u_j, v_j) Setze $T' = W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)}$.

6: $OD_{krit} = OD_{krit} \setminus \{\{u_i, v_i\} : i = 1, \ldots, j\}$

7: Gehe zu Schritt 11.

8: **end if**

9: **end for**

10: **return** $T' = \emptyset$

→ Hinzufügen von Kanten

11: **for** $\{u, v\} \in OD_{krit}$ **do**

12: Bestimme die kürzesten Wege zwischen T' und (u, v) und füge diese zu T' hinzu.

13: $OD_{krit} = OD_{krit} \setminus \{\{u, v\}\}$

14: **if** $l(T') > l_0$ **then**

15: **return** FAIL

16: **end if**

17: **end for**

18: **return** T'

Satz 80 *Algorithmus „Konstruktion Unterbaum“ konstruiert in Zeit $O(n^3)$ einen Unterbaum T', der in jedem Baum $\hat{T}$, der (T, OD, α, l_0, d) mit $d \leq d_0$ löst, enthalten ist. Dieser Unterbaum kann auch leer sein. Gibt er FAIL aus, so ist die Instanz $(T, OD, \alpha, l_0, d_0)$ nicht lösbar.*

Beweis. Sei $\hat{T}$ Lösung von (T, OD, α, l_0, d) mit $d \leq d_0$. Zu zeigen ist nun, dass der von Algorithmus „Konstruktion Unterbaum“ für eine Instanz $(T, OD, \alpha, l_0, d_0)$ von *A-Baum-Baum* konstruierte Graph T' Unterbaum von $\hat{T}$ ist.
Aus der Bedingung $d_{\hat{T}}(u, v) \leq d_0 \ \forall \ \{u, v\} \in OD$ folgt, dass $\hat{T}$ aus jedem Weg (u, v) zwischen den Knoten eines kritischen OD-Paares $\{u, v\} \in OD_{krit}$ mindestens eine Kante enthalten muss. Insbesondere muss $\hat{T}$ also aus jedem $\{u, v\} \in OD_{krit}$ einen Knoten enthalten. Da die Lösung $\hat{T}$ ein Unterbaum und damit zusammenhängend ist, müssen die Wege zwischen den enthaltenen Knoten ebenfalls in $\hat{T}$ liegen. Deswegen gilt für alle $\{\{u_i, v_i\} : \ i \leq j\} \subset OD_{krit}$ die Aussage $T' \supset \bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}$ und die Korrektheit von Schritt 5 folgt aus:

Behauptung 81 *Sei $\{\{u_i, v_i\} : i \leq j\}$ eine Teilmenge von OD_{krit} mit $\bigcap_{i=1}^{j-1}(u_i, v_i) \neq \emptyset$ und $\bigcap_{i=1}^{j}(u_i, v_i) = \emptyset$. Sei $W_{(u_i,v_i),(u_j,v_j)}$ der kürzeste Weg zwischen (u_i, v_i) und (u_j, v_j) sowie $W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)}$ der kürzeste Weg zwischen $\bigcap_{i=1}^{j-1}(u_i, v_i)$ und (u_j, v_j). Dann gilt:*

$$\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)} = W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)}. \tag{111}$$

Da für alle $i \leq j$ gilt $\bigcap_{i=1}^{j-1}(u_i, v_i) \subset (u_i, v_i)$, ist jeder Weg von (u_j, v_j) nach $\bigcap_{i=1}^{j-1}(u_i, v_i)$ insbesondere auch ein Weg von (u_j, v_j) nach (u_i, v_i). Also gilt für die kürzesten Wege zwischen $\bigcap_{i=1}^{j-1}(u_i, v_i)$ und (u_j, v_j) beziehungsweise (u_i, v_i) und (u_j, v_j):

$$W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)} \supseteq W_{(u_i,v_i),(u_j,v_j)} \ \forall \ i \leq j-1. \tag{112}$$

Daraus folgt

$$W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)} \supseteq \bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}. \tag{113}$$

Angenommen $\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}$ ist nicht zusammenhängend. Dann gibt es $k, l \leq j-1$, so dass $W_{(u_k,v_k),(u_j,v_j)} \cap W_{(u_l,v_l),(u_j,v_j)} = \emptyset$. Man schreibe nun $W_{(u_k,v_k),(u_j,v_j)} = (w_1^k, w_2^k, \ldots, w_{m^k}^k)$ und $W_{(u_l,v_l),(u_j,v_j)} = (w_1^l, w_2^l, \ldots, w_{m^l}^l)$ mit $w_1^k, w_1^l \in (u_j, v_j)$, $w_{m^k}^k \in (u_k, v_k)$, $w_{m^l}^l \in (u_l, v_l)$ und $w_i^k, w_j^l \notin ((u_k, u_l) \cup (u_l, v_l) \cup (u_j, v_j))$ (vergleiche Bemerkung 79). Da $\bigcap_{i=1}^{j-1}(u_i, v_i) \neq \emptyset$, ist $(w_{m_k}^k, w_{m_l}^l) \subset \bigcap_{i=1}^{j-1}(u_i, v_i)$.

Daher kann ein Kreis in $\hat{T}$ nach obigen Überlegungen durch folgende Abfolge von Pfaden definiert werden, die nur die Knoten, an denen sie zusammengesetzt werden, gemeinsam haben:

$$(w_1^k, w_{m^k}^k) \cup (w_{m^k}^k, w_{m^l}^l) \cup (w_{m^l}^l, w_1^l) \cup (w_1^l, w_1^k). \tag{114}$$

Die Existenz eines Kreises als Teilgraph von T steht im Widerspruch zu der Tatsache, dass T ein Baum ist. Also ist die Annahme falsch und $\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}$ ist zusammenhängend.
Da $\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}$ in dem Weg $W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)} = (w_1, w_2, \ldots, w_m)$ enthalten ist folgt, dass $\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}$ ebenfalls ein Weg ist und sich schreiben lässt als:

$$\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)} = (w_1^\cup, w_2^\cup, \ldots, w_{m^\cup}^\cup) \subset (w_1, w_2, \ldots, w_m). \tag{115}$$

Angenommen

$$\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)} \subsetneq W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)}, \tag{116}$$

also

$$(w_1, w_2 \ldots, w_m) \subsetneq (w_1^\cup, w_2^\cup, \ldots, w_{m^\cup}^\cup). \tag{117}$$

Es ist $w_1^\cup = w_1$, da sonst $(u_j, v_j) \cap (w_1^\cup, w_2^\cup, \ldots, w_{m^\cup}^\cup) = \emptyset$ gelten würde. Also muss gelten $w_m \notin (w_1^\cup, w_2^\cup, \ldots, w_{m^\cup}^\cup)$. Daraus folgt $(w_1^\cup, w_2^\cup, \ldots, w_{m^\cup}^\cup) = (w_1, w_2, \ldots, w_{m^\cup})$ mit $m^\cup < m$ und $(w_1, w_2, \ldots, w_{m^\cup}) \cap \bigcap_{i=1}^{j-1}(u_i, v_i) = \emptyset$.
Daher gibt es ein OD-Paar $\{u_s, v_s\}$ mit $s \leq j-1$ und $w_{m^\cup} \notin (u_s, v_s)$. Angenommen es gibt ein $t \leq m^\cup$ mit $w_t \in (u_s, v_s)$. Da $w_m \in \bigcap_{i=1}^{j-1}(u_i, v_i)$ ist insbesondere auch $w_m \in (u_s, v_s)$. Dann ergibt sich wegen $t < m^\cup < m$ aus $w_{m^\cup} \in (w_t, w_m) \subset (u_s, v_s)$ ein Widerspruch. Also kann es kein $w_t \in \bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}$ geben, das in (u_s, v_s) enthalten ist. Dies ist ein Widerspruch zu $\bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)} \supset W_{(u_t,v_t),(u_j,v_j)}$.
Es folgt, dass $w_{m^\cup} = w_m$. Also gilt

$$W_{\bigcap_{i=1}^{j-1}(u_i,v_i),(u_j,v_j)} = \bigcup_{i=1}^{j-1} W_{(u_i,v_i),(u_j,v_j)}. \tag{118}$$

Behauptung 82 *Sei $T(\tilde{OD})$ der inklusionsminimale Unterbaum, der für jedes $\{u, v\} \in \tilde{OD}$ mindestens einen Knoten enthält. Dann gilt $T(\hat{OD} \cup \{\hat{u}, \hat{v}\}) = T(\hat{OD}) \cup W_{T(\hat{OD}),(\hat{u},\hat{v})}$.*

Offensichtlich ist $T(\hat{OD}) \cup W_{T(\hat{OD}),(\hat{u},\hat{v})}$ zusammenhängend, weil $W_{T(\hat{OD}),(\hat{u},\hat{v})}$ einen Knoten aus $T(\hat{OD})$ enthält. Da $T(\hat{OD})$ für jedes $\{u, v\} \in \hat{OD}$ mindestens einen Knoten beinhaltet und $W_{T(\hat{OD}),(\hat{u},\hat{v})}$ einen Knoten aus $\{u, v\}$ enthält, gilt $T(\hat{OD} \cup \{\hat{u}, \hat{v}\}) \subseteq T(\hat{OD}) \cup W_{T(\hat{OD}),(\hat{u},\hat{v})}$.

Nach Definition ist $W_{T(\hat{OD}),(\hat{u},\hat{v})}$ auch der kürzeste Weg zwischen $T(\hat{OD})$ und $(\hat{u},\hat{v})$, also gilt $T(\hat{OD} \cup \{\hat{u},\hat{v}\}) = T(\hat{OD}) \cup W_{T(\hat{OD}),(\hat{u},\hat{v})}$.

Induktiv folgt hieraus, dass der durch Hinzufügen von Kanten in Schritt 12 konstruierte Unterbaum T' der Baum $T(OD_{krit})$ ist, also der inklusionsminimale Unterbaum, der für jedes $\{u,v\} \in OD_{krit}$ mindestens einen Knoten enthält. Nach den Überlegungen am Anfang des Beweises ist also $T' \subset \hat{T}$ für jede Lösung $\hat{T}$ von (T, OD, α, l_0, d) mit $d \leq d_0$.

Gibt Algorithmus „Konstruktion Unterbaum“ FAIL aus, so ist $l(T') > l_0$ für den konstruierten Unterbaum T'. Angenommen $\hat{T}$ sei Optimallösung von (T, OD, α, l_0, d) mit $d \leq d_0$. Dann gilt $T' \subset \hat{T}$ und somit $l_0 \geq l(\hat{T}) \geq l(T') > l_0$, was ein Widerspruch ist.

Nun zur Komplexität des Algorithmus: Ohne Beschränkung der Allgemeinheit sei T rekursiv dargestellt und seine kanonische rekursive Darstellung bekannt (siehe Lemma 4 und Bemerkung 6). Da die Menge $\bigcap_{i=1}^{j-1} E_{(u_i,v_i)}$ bereits beim vorherigen Durchlauf der Schleife berechnet wurde, muss zur Überprüfung der Bedingung in Schritt 4 nur für jedes Element aus E überprüft werden, ob es in beiden Mengen $\bigcap_{i=1}^{j-1} E_{(u_i,v_i)}$ und $E_{(u_j,v_j)}$ enthalten ist. Diese Überprüfung ist also in $O(n)$. Da sie höchstens $(|OD_{krit}| - 1)$-mal ausgeführt wird und $|OD_{krit}|$ in $O(n^2)$ ist, ist die Überprüfungszeit in $O(n^3)$.

Die Bestimmung des kürzesten Weges zwischen zwei Graphen in Schritt 5 und 12 erfolgt durch Bestimmung des Weges zwischen zwei beliebigen Knoten der Wege (nach Lemma 9 in $O(n)$) und Entfernen von bereits in den Kantenmengen der Wege enthaltenen Kanten und ist somit in $O(n)$. Schritt 5 wird einmal, Schritt 12 $O(n^2)$-mal ausgeführt. Also ist auch die Berechnung der Wege in $O(n^3)$. $\Rightarrow$ Die Komplexität von Algorithmus „Konstruktion Unterbaum“ ist $O(n^3)$.

■

Bemerkung 83 *Gibt Algorithmus „Konstruktion Unterbaum“ den leeren Baum aus, so stellen die Knoten $v \in \bigcap_{i=1}^{|OD_{krit}|} V_{(u_i,v_i)} \neq \emptyset$ die (nicht eindeutig bestimmten) kürzesten Wege zwischen den kritischen OD-Paaren dar. Daher muss, falls ein Baum existiert, der das Entscheidungsproblem A-Baum-Baum löst, mindestens ein solches v in diesem Baum enthalten sein.*

Lemma 84 *Sei T ein Baum und OD eine Menge von OD-Paaren in T. Seien $0 \leq s < t$ zwei nichtnegative, reelle Zahlen, so dass es kein OD-Paar $\{u,v\}$ mit $s \leq d(u,v) < t$ gibt.*
Dann ist für jedes $d_0 \in [s,t)$ der von Algorithmus „Konstruktion Unterbaum“ konstruierte Unterbaum identisch.

Beweis. Sei $d_0 \in [s,t)$. Dann ist für OD_{krit} aus Algorithmus „Konstruktion Unterbaum“ :

$$\begin{aligned} OD_{krit} &= OD \setminus \{\{u,v\} : d(u,v) \leq d_0\} \\ &= OD \setminus \{\{u,v\} : d(u,v) \leq s\}\,. \end{aligned} \tag{119}$$

Daraus folgt die Behauptung. ■

6.2 Lösung des Entscheidungsproblems für Spezialfälle von Problem *A-Baum-Baum*

Sei $(T, OD, \alpha, l_0, d_0)$ eine Instanz von *A-Baum-Baum*. Für einen gegebenen Unterbaum T'_E von T lässt sich das Problem *A-Baum-Baum* dahingehend verändern, dass nur noch Lösungen T' von $(T, OD, \alpha, l_0, d_0)$, die T'_E enthalten, als zulässig gelten:

Definition 85 *Problem A-Baum-Baum**
Entscheidungsproblem:

- *Instanz* $(T, OD, \alpha, l_0, d_0, T'_E)$*:*
  Baum $T = (V, E)$ *mit Kantenlängen* $c_{ij} \in \mathbb{Z}^+$ *für* $[i,j] \in E$,
  Menge von OD-Paaren OD,
  Beschleunigungsfaktor $\alpha \in [0,1)$,
  Maximallänge des gesuchten Teilbaumes $l_0 \in \mathbb{Z}^+$,
  Maximale Distanz in $G_{T'}$ *zwischen zwei OD-Paaren* $d_0 \in \mathbb{Z}_0^+$,
  Unterbaum T'_E *von* T

- *Frage:*
  Gibt es einen Teilgraph $T' \subset T$*, so dass* $T'_E \subset T'$ *und:*

$$l(T') \leq l_0 \tag{120}$$

$$d_{T'}(u,v) \leq d_0 \qquad \forall\ \{u,v\} \in OD\ ? \tag{121}$$

Optimierungsproblem:

- *Instanz* $(T, OD, \alpha, l_0, T'_E)$*:*
  Baum $T = (V, E)$ *mit Kantenlängen* $c_{ij} \in \mathbb{Z}^+$ *für* $[i,j] \in E$,
  Menge von OD-Paaren OD,
  Beschleunigungsfaktor $\alpha \in [0,1)$,
  Maximallänge des gesuchten Teilbaumes $l_0 \in \mathbb{Z}^+$,
  Unterbaum T'_E *von* T

- *Gesucht:*

  Unterbaum $T' \subset T$ *mit* $T'_E \subset T'$ *und* $l(T') \leq l_0$*, der* $\max_{\{u,v\} \in OD} d_{T'}(u,v)$ *minimiert.*

Bemerkung 86 *Sei* $(T, OD, \alpha, l_0, d_0)$ *eine Instanz von A-Baum-Baum. Ist* T'_E *ein von Algorithmus „Konstruktion Unterbaum“ für* d_0 *erzeugter Unterbaum, so ist* T'_E *nach Satz 80 in jeder Lösung von* (T, OD, α, l_0, d) *mit* $d \leq d_0$ *enthalten. Für* $d \leq d_0$ *sind die Instanzen* $(T, OD, \alpha, l_0, d, T'_E)$ *und* (T, OD, α, l_0, d) *der Entscheidungsprobleme A-Baum-Baum und A-Baum-Baum* also äquivalent.*

Sei $(T, OD, \alpha, l_0, d_0, T'_E)$ eine Instanz von *A-Baum-Baum**. Hat die Menge der OD-Paare für einen Unterbaum T', der T'_E enthält und in jeder Lösung einer Instanz von $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_0$ enthalten sein muss, die Eigenschaft

$$\begin{aligned} &(E_{u_1,v_1} \cap E_{u_2,v_2}) \setminus E_{T'} = \emptyset \\ &\forall\ \{u_1, v_1\}, \{u_2, v_2\} \in OD \text{ mit } d(u_i, v_i) > d_0, \end{aligned} \tag{122}$$

so lässt sich in Polynomialzeit entscheiden, ob es eine Lösung von $(T, OD, \alpha, l_0, d_0, T'_E)$ gibt. Diese Eigenschaft des Unterbaumes sei im Folgenden mit $Dist(T')$ bezeichnet. Die Konstruktion der Lösung beruht darauf, dass in diesem Fall, aufbauend auf dem Unterbaum T', für jedes OD-Paar einzeln eine Lösung gefunden wird und diese Einzellösungen zusammengesetzt werden. T' erhält man zum Beispiel durch Ausführung von Algorithmus „Konstruktion Unterbaum“ und anschließende Vereinigung des Ergebnisses mit T'_E.

Algorithmus 4 Hinzufügen von Kanten

Eingabe: Baum $T = (V, E)$ mit Kantenlängen $c_{ij}\ \forall [i,j] \in E$; nichtleerer Unterbaum $T'_E \subset T$; OD Menge von OD-Paaren; α; l_0; d_0.

Ausgabe: Unterbaum $\tilde{T}$ von T, der die Instanz $(T, OD, \alpha, l_0, d_0, T'_E)$ von *A-Baum-Baum** löst; Unterbaum T' von T mit $T'_E \subset T'$, der für $d \leq d_0$ in der Lösung des Problems enthalten sein muss ; FAIL falls es keinen solchen Baum $\tilde{T}$ gibt; „Eigenschaft $Dist(T'_E)$ nicht erfüllt“, falls dies der Fall ist und deshalb das Problem nicht entschieden werden kann.

1: **if** $l(T'_E) > l_0$ **then**
2: **return** FAIL
3: **end if**
4: $OD_{krit} = \left\{\{u,v\} \in OD : d_{T'_E}(u,v) > d_0\right\}$
5: **for** $\{u,v\} \in OD_{krit}$ **do**
6: **if** $(u,v) \subset T'_E$ **then**
7: **return** FAIL
8: **end if**
9: **end for**
10: $T' = T'_E$
11: **for** $\{u,v\} \in OD_{krit}$ **do**
12: **if** $(u,v) \cap T' = \emptyset$ **then**
13: Bestimme die kürzesten Wege zwischen T' und (u,v) und füge diese zu T' hinzu.
14: **end if**
15: **end for**
16: **if** $(E_{(u_1,v_1)} \cap E_{(u_2,v_2)}) \setminus E_{T'} \neq \emptyset$ für zwei OD-Paare $\{u_1,v_1\}, \{u_2,v_2\} \in OD_{krit}$ **then**
17: **return** „Eigenschaft $Dist(T'_E)$ nicht erfüllt.“
18: **end if**
19: Setze $\tilde{T} = T'$.
20: **for** $\{u,v\} \in OD_{krit}$ **do**
21: Bestimme alle Teilwege W^i_{uv} von (u,v), die $(u,v) \cap \tilde{T}$ enthalten und die inklusionsminimal mit der Eigenschaft $d_{W^i_{(u,v)}}(u,v) \leq d_0$ sind, und ihre Länge. Füge einen der kürzesten dieser Wege, genannt $W_{(u,v)}$, zu $\tilde{T}$ hinzu.
22: **if** Es gibt kein solches $W_{(u,v)}$ **then**
23: **return** FAIL
24: **end if**
25: **end for**

26: **if** $l(\tilde{T}) > l_0$ **then**

27: **return** FAIL

28: **end if**

29: **return** Lösung $\tilde{T}$ von $(T, OD, \alpha, l_0, d_0, T'_E)$; Unterbaum $T' \supset T'_E$

Lemma 87 *Sei $(T, OD, \alpha, l_0, d_0, T'_E)$ eine Instanz des Entscheidungsproblems A-Baum-Baum*. Gilt für einen Unterbaum $T' \subset T$ mit $T' \supset T'_E$, der in jeder Lösung von $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_0$ enthalten ist, Eigenschaft Dist(T'), also*

$$\begin{aligned} &(E_{u_1,v_1} \cap E_{u_2,v_2}) \setminus E_{T'} = \emptyset \\ &\forall\ \{u_1, v_1\}, \{u_2, v_2\} \in OD \text{ mit } d(u_i, v_i) > d_0, \end{aligned} \tag{123}$$

dann entscheidet Algorithmus „Hinzufügen von Kanten“ das Entscheidungsproblem A-Baum-Baum in $O(n^3)$ und gibt eine längenminimale Lösung $\tilde{T}$ der Instanz $(T, OD, \alpha, l_0, d_0, T'_E)$ aus, falls eine solche existiert. Weiterhin gibt der Algorithmus einen Unterbaum T' aus, der in jeder Lösung einer Instanz $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_0$ enthalten sein muss.*

Beweis. Sei $\hat{T}$ eine Lösung von $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_0$. Dann enthält $\hat{T}$ von jedem OD-Paar aus OD_{krit} mindestens eine Kante. Analog zum Beweis der Korrektheit von Algorithmus „Konstruktion Unterbaum“ folgt, dass die in Schritt 13 konstruierten kürzesten Wege in $\hat{T}$ liegen müssen und dass der so konstruierte Untergraph ein Baum ist. Also gilt $T' \subset \hat{T}$.
Sei $\tilde{T}$ die ausgegebene Lösung. Nach Konstruktion enthält $\tilde{T}$ den Unterbaum T'_E. Wegen Schritt 21 gilt $d_{\tilde{T}}(u, v) \leq d_0$ für alle $\{u, v\} \in OD_{krit}$. Für alle anderen OD-Paare $\{u, v\} \in OD \setminus OD_{krit}$ gilt das schon wegen $T'_E \subset \tilde{T}$ und der Definition von OD_{krit} in Schritt 4. Da T ein Baum ist, ist der Untergraph $\tilde{T}$ kreisfrei. Weil bei Konstruktion von $\tilde{T}$ alle Wege zusammenhängend angefügt werden, folgt induktiv, dass $\tilde{T}$ ein Baum ist. Es gilt $l(\tilde{T}) \leq l_0$, da sonst in Schritt 27 FAIL ausgegeben worden wäre. Also ist $\tilde{T}$ Lösung von $(T, OD, \alpha, l_0, d_0, T'_E)$.
Da $E_{(u_1,v_1)} \cap E_{(u_2,v_2)} \setminus E_{T'} = \emptyset$ für alle $\{u_1, v_1\}, \{u_2, v_2\} \in OD_{krit}$ gelten muss, damit Algorithmus „Hinzufügen von Kanten“ eine Lösung ausgibt, ist in Schritt 21 $W^i_{(u_1,v_1)} \cap W^j_{(u_2,v_2)} = \emptyset$ für alle $\{u_1, v_1\}, \{u_2, v_2\} \in OD_{krit}$ und alle betrachteten i und j. Die Auswahl aus den längenminimalen Wegen $W^l_{(u_k,v_k)}$ beeinflusst also nicht die Länge von $\bigcup_{\{u_k,v_k\} \in OD_{krit}} W_{(u_k,v_k)}$.
Angenommen es gibt eine Lösung T^* von $(T, OD, \alpha, l_0, d_0, T'_E)$ mit $l(T^*) < l(\tilde{T})$. Da, wie schon gezeigt, $T'_E \subset T^*$ und $T'_E \subset \tilde{T}$, muss gelten

$$l(T^* \setminus T'_E) < l(\tilde{T} \setminus T'_E) = l(\bigcup_{\{u_k,v_k\} \in OD_{krit}} W_{(u_k,v_k)} \setminus T'_E). \tag{124}$$

Das führt aber zum Widerspruch, da $\bigcup_{\{u_k,v_k\}\in OD_{krit}} W_{(u_k,v_k)}$ längenminimal gewählt wurde und es somit keinen kleineren zusammenhängenden Graph g geben kann, der $d_g(u,v) \leq d_0$ für alle $\{u,v\} \in OD_{krit}$ erfüllt. Also ist $\tilde{T}$ längenminimale Lösung von $(T, OD, \alpha, l_0, d_0, T'_E)$.
Gibt der Algorithmus „Konstruktion Unterbaum“ FAIL aus, so ist entweder $l(T'_E) > l_0$, es gilt $(u,v) \subset T'_E$ für ein $\{u,v\} \in OD_{krit}$, es gibt kein $W_{(u,v)}$ mit $d_{W_{(u,v)}}(u,v) \leq d_0$ oder $l(\tilde{T}) > l_0$. Im ersten Fall gilt für jeden Baum mit $\hat{T} \supset T'_E$ offensichtlich $l(\hat{T}) \geq l(T'_E) > l_0$, also kann es keine zulässige Lösung von $(T, OD, \alpha, l_0, d_0, T'_E)$ geben. Gilt $d_{T'_E}(u,v) > d_0$ für ein OD-Paar $\{u,v\}$, für das der Weg (u,v) vollständig in T' enthalten ist, so ist auch $d_{\hat{T}}(u,v) = d_{T'_E}(u,v) > d_0$ für jeden Unterbaum $\hat{T}$ von T. Also gibt es in diesem Fall ebenfalls keine Lösung des Problems. Gleiches gilt im dritten Fall, da wenn $d_{W^i_{(u,v)}}(u,v) > d_0$ für alle $W^i_{(u,v)}$, insbesondere (u,v) selbst, gilt, auch $d_{\hat{T}}(u,v) > d_0$ folgt. Ist im letzten Fall schließlich der konstruierte längenminimale Baum länger als l_0, so folgt ebenfalls sofort, dass es keine Lösung des Entscheidungsproblems geben kann.
Sei ohne Beschränkung der Allgemeinheit T rekursiv dargestellt (Lemma 4 und Bemerkung 6). Die Überprüfung der ersten Bedingung ist in $O(n)$. Die Berechnung der kritischen OD-Paare erfordert die Berechnung der Distanz von $O(n^2)$ OD-Paaren durch Berechnung des kürzesten Weges zwischen den beiden Knoten eines OD-Paares (siehe Lemma 9) und Addieren der Kantenlängen und ist damit in $O(n^3)$. Die Anzahl der kritischen OD-Paare ist allerdings beschränkt durch die Anzahl aller OD-Paare und damit in $O(n^2)$. Die Überprüfung in Schritt 5-9 ist also in $O(n^3)$. Ist kein kritisches OD-Paar ganz in T'_E enthalten, so ist die Anzahl der kritischen OD-Paare wegen Bedingung $Dist(T'_E)$ durch die Anzahl der Kanten von T beschränkt und damit in $O(n)$. Für ein OD-Paar ist wie im Beweis zu Satz 80 das Hinzufügen der Kanten in Schritt 13 in $O(n)$. Die Überprüfung der Bedingung in Schritt 16 geschieht, indem man für jede Kante in $T \setminus T'_E$ überprüft, ob sie in mehr als einem der $O(n^2)$ kritischen OD-Paare enthalten ist und ist deshalb in $O(n^3)$. Das Hinzufügen der Kanten in Schritt 21 kann folgendermaßen in $O(n)$ vorgenommen werden: Die Mindestlänge der noch hinzuzufügenden Kanten für ein kritisches OD-Paar beträgt $d^{uv} := (d_{\tilde{T}}(u,v) - d_0)/(1-\alpha)$. Bestimme (ähnlich wie im Beweis zu Lemma 15 beim Finden der maximalen Pfade der Länge höchstens l_0) in $O(n)$ die minimalen Teilpfade von (u,v) der Länge mindestens $d^{uv} + l((u,v) \cap \tilde{T})$, die $(u,v) \cap \tilde{T}$ enthalten. Wähle den kürzesten dieser Pfade und füge ihn zu $\tilde{T}$ hinzu. Also ist Schritt 20-28 in $O(n^3)$.
Algorithmus „Hinzufügen von Kanten“ ist somit in $O(n^3)$. ■

Daraus folgt:

Korollar 88 *Sei $(T, OD, \alpha, l_0, d_0)$ eine Instanz des Entscheidungsproblems A-Baum-Baum. Für einen nichtleeren Baum T'_E, sei bekannt, dass er in der Lösung des Entscheidungsproblems enthalten sein muss. Gilt für T'_E:*

$$\begin{aligned} &(E_{u_1,v_1} \cap E_{u_2,v_2}) \setminus E_{T'_E} = \emptyset \\ &\forall\ \{u_1, v_1\}, \{u_2, v_2\} \in OD \text{ mit } d(u_i, v_i) > d_0, \end{aligned} \tag{125}$$

dann entscheidet Algorithmus „Hinzufügen von Kanten" das Entscheidungsproblem in $O(n^3)$ und gibt eine längenminimale Lösung T^* der Instanz $(T, OD, \alpha, l_0, d_0)$ aus, falls sie existiert.

Bemerkung 89 *Einen solchen Baum T'_E erhält man zum Beispiel durch Ausführung von Algorithmus „Konstruktion Unterbaum" für $(T, OD, \alpha, l_0, d_0)$. In diesem Fall können bei der Ausführung von Algorithmus „Hinzufügen von Kanten" die Schritte 1-3 und 11-15 ausgelassen werden, da eine Überprüfung der Zulässigkeit der Länge und das Hinzufügen der kürzesten Wege schon in Algorithmus „Konstruktion Unterbaum" stattfanden.*

6.3 Optimalitätskriterien und Schranken

Es folgen zwei Kriterien zur Überprüfung der Optimalität von mit Hilfe von Algorithmus „Entscheidung A-Baum-Baum" gefundenen Lösungen des Entscheidungsproblems.

Lemma 90 *Sei T^* die von Algorithmus „Hinzufügen von Kanten" gegebene längenminimale Lösung einer Instanz $(T, OD, \alpha, l_0, d_0)$ des Entscheidungsproblems A-Baum-Baum für einen Baum $T = (V, E)$ und einen für d_0 mit Algorithmus „Konstruktion Unterbaum" erzeugten, nichtleeren Unterbaum T'. Setze $\delta := \min_{I_1, I_2} \left\{ \left| \sum_{e_i \in I_2} c_i - \sum_{e_i \in I_1} c_i \right| > 0 \right\}$ mit $I_1 \subset E$ und $I_2 \subset E \setminus I_1$, also als die minimale Längenveränderung die eintreten kann, wenn man Kanten aus einem Unterbaum von T entfernt und andere hinzufügt.*
Gilt $l(T^*) > l_0 - \delta$, dann ist T^* optimal.

Beweis. Angenommen T^* ist nicht optimal. Dann gibt es einen Baum $\hat{T}$ mit

$$l(\hat{T}) \leq l_0, \tag{126}$$

$$\max_{\{u,v\} \in OD} d_{\hat{T}}(u, v) < \max_{\{u,v\} \in OD} d_{T^*}(u, v). \tag{127}$$

Nach Satz 80 muss der von Algorithmus „Konstruktion Unterbaum" erzeugte Unterbaum T' auch Unterbaum von $\hat{T}$ sein. Sei $OD_{krit} = \{\{u, v\} \in OD : d_{T'}(u, v) > d_0\}$ wie in Algorithmus „Hinzufügen von Kanten".

Nach Lemma 87 ist T^* längenminimal mit der Eigenschaft $d_{T^*}(u,v) \leq d_0 \forall \{u,v\} \in OD_{krit}$. Also gilt:

$$l((u,v) \cap \hat{T}) \geq l((u,v) \cap T^*) \; \forall \{u,v\} \in OD_{krit} \tag{128}$$

und wegen $T' \subset \hat{T}$, $T' \subset T^*$

$$l(((u,v) \cap \hat{T}) \setminus T') \geq l(((u,v) \cap T^*) \setminus T') \; \forall \{u,v\} \in OD_{krit}. \tag{129}$$

Wegen (127) gibt es ein $\{u,v\} \in OD_{krit}$, so dass

$$l((u,v) \cap \hat{T}) > l((u,v) \cap T^*), \tag{130}$$

also auch

$$l(((u,v) \cap \hat{T}) \setminus T') > l(((u,v) \cap T^*) \setminus T'). \tag{131}$$

Nach Definition von δ gilt für dieses OD-Paar sogar:

$$l(((u,v) \cap \hat{T}) \setminus T') \geq l(((u,v) \cap T^*) \setminus T') + \delta. \tag{132}$$

Es folgt:

$$\begin{aligned} l(\hat{T}) &\geq l(T') + \sum_{\{u,v\} \in OD_{krit}} l(((u,v) \cap \hat{T}) \setminus T') \\ &\geq l(T') + \sum_{\{u,v\} \in OD_{krit}} l(((u,v) \cap T^*) \setminus T') + \delta \\ &= l(T^*) + \delta \\ &> l_0. \end{aligned} \tag{133}$$

Das ergibt einen Widerspruch zu (126). Also ist T^* optimal. ■

Lemma 91 *Sei $T = (V,E)$ ein Baum und $\tilde{T}$ eine Optimallösung der Instanz (T, OD, α, l_0) des Optimierungsproblems A-Baum-Baum. $D := \max_{\{u,v\} \in OD} d_{\tilde{T}}(u,v)$ bezeichne den optimalen Zielfunktionswert. Sei T^* die von Algorithmus „Entscheidung A-Baum-Baum“ gegebene längenminimale Lösung einer Instanz $(T, OD, \alpha, l_0, d_0)$ des Entscheidungsproblems A-Baum-Baum. Sei $\delta := \min_{I_1, I_2} \left\{ \left| \sum_{e_i \in I_2} c_i - \sum_{e_i \in I_1} c_i \right| > 0 \right\}$ mit $I_1 \subset E$, $I_2 \subset E \setminus I_1$, wie in Lemma 90. Sind Schranken s, t mit $0 \leq s < D \leq t$ für den optimalen Zielfunktionswert D bekannt, für die gilt*

$$t - s \leq (1 - \alpha)\delta \tag{134}$$

und gilt für die Lösung T^* des Entscheidungsproblems

$$\max_{\{u,v\}\in OD} d_{T^*}(u,v) \in (s,t], \tag{135}$$

dann ist T^* optimal.

Beweis. Angenommen T^* ist nicht optimal, also $D < \max_{\{u,v\}\in OD} d_{T^*}(u,v)$.
Sei $\{u,v\} \in OD$ mit $d_{T^*}(u,v) = \max_{\{u,v\}\in OD} d_{T^*}(u,v)$. $\tilde{W} := \tilde{T} \cap (u,v)$ und $W^* := T^* \cap (u,v)$ seien die in $\tilde{T}$ beziehungsweise T^* liegenden Teile des Weges von u nach v.
Dann ist, da der Zielfunktionswert von T^* innerhalb der Schranken s und t liegt und $\tilde{T}$ sogar noch eine bessere Lösung sein soll, $s < d_{\tilde{T}}(u,v) < d_{T^*}(u,v) \leq t$:

$$d_{T^*}(u,v) - d_{\tilde{T}}(u,v) < (t-s) \leq (1-\alpha)\delta \tag{136}$$

und

$$\begin{aligned} d_{T^*}(u,v) - d_{\tilde{T}}(u,v) =& \alpha \cdot l(W^*) + [l(u,v) - l(W^*)] - \left(\alpha \cdot l(\tilde{W}) + [l(u,v) - l(\tilde{W})]\right) \\ =& (1-\alpha)[l(\tilde{W}) - l(W^*)] \end{aligned} \tag{137}$$

(136) und (137) zusammengenommen ergeben $\left(l(\hat{W}) - l(W^*)\right) < \delta$.
Das ist ein Widerspruch zur Definition von δ.
$\Rightarrow T^*$ ist optimal. ■

Folgendes Lemma liefert erste Schranken für den optimalen Zielfunktionswert von *A-Baum-Baum*.

Lemma 92 *Sei T ein Baum, OD eine Menge von OD-Paaren in T, $\alpha \in [0,1)$ ein Beschleunigungsfaktor und l_0 die Maximallänge des gesuchten optimalen Unterbaumes $\hat{T}$ von T.*
Sei $d_1 := \max_{\{u,v\}\in OD} d(u,v)$. Dann gilt für jeden Teilbaum $\tilde{T}$ von T mit $l(\tilde{T}) \leq l_0$:

$$d_1 \geq \max_{\{u,v\}\in OD} d_{\tilde{T}}(u,v) \geq \begin{cases} \alpha d_1 & \text{falls } d_1 \leq l_0 \\ \alpha l_0 + (d_1 - l_0) & \text{falls } d_1 > l_0 \end{cases} \tag{138}$$

Beweis. Die erste Ungleichung folgt aus Lemma 12:

$$d(u,v) \geq d_{\tilde{T}}(u,v) \geq d_T(u,v) = \alpha l(u,v) \qquad \forall\ \{u,v\} \in OD \tag{139}$$

$\Rightarrow$

$$\max_{\{u,v\}\in OD} d_{\tilde{T}}(u,v) \geq \max_{\{u,v\}\in OD} d_T(u,v) = \alpha d_1 \tag{140}$$

Ist $l_0 < d_1$, so kann maximal ein Teilstück (i,j) der Länge $l(i,j) = l_0$ von (u,v) in $\tilde{T}$ enthalten sein, da $l(\hat{T}) \leq l_0$. Also ist

$$\max_{\{u,v\} \in OD} d_{\tilde{T}}(u,v) \geq \max_{\{u,v\} \in OD} d_{(i,j)}(u,v) \geq \alpha l_0 + (d_1 - l_0). \tag{141}$$

■

Sei $(T, OD, \alpha, l_0, T'_E)$ eine Instanz des Optimierungsproblems *A-Baum-Baum**. Der folgende Algorithmus „A-Baum-Baum 1“ berechnet für geeignete Bäume weitere Schranken, in der Optimallösung enthaltene Unterbäume und Lösungen der Instanz $(T, OD, \alpha, l_0, d_0, T'_E)$ des zugehörigen Entscheidungsproblems mit möglichst kleinem d_0.
Dazu löst es wiederholt Instanzen $(T, OD, \alpha, l_0, d, T')$ des Entscheidungsproblems *A-Baum-Baum* mit kleiner werdendem d. Die Wahl von d orientiert sich dabei an den Distanzen der OD-Paare in T.

Algorithmus 5 A-Baum-Baum 1

Eingabe: Baum $T = (V, E)$ mit Kantenlängen c_{ij} $\forall [i, j] \in E$; Unterbaum $T'_E \neq \emptyset$ von T; $OD = \{\{k, l\} : k \leq l\}$ Menge von OD-Paaren; α; l_0, obere Schranke d^s; untere Schranke d_s.

Ausgabe: Beste gefundene Lösung von $(T, OD, \alpha, l_0, T'_E)$ $\tilde{T}^*$ mit Zielfunktionswert d^s; untere Schranke d_s; Unterbaum T'^*, der enthalten ist in jedem Baum, der das Entscheidungsproblem $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_0$ löst.

→ Initialisierung

1: Bestimme OD_i, d_i so dass $d(u, v) = d_i \; \forall \; \{u, v\} \in OD_i$, $d_1 > d_2 > \ldots > d_k$

2: $a = \max\{i : d_i < d^s\}$, $b = \min\{i : d_i > d_s\}$

→ Erster Schritt

3: Führe Algorithmus „Hinzufügen von Kanten“ mit $d_0 = d^s$ und Unterbaum T'_E aus.

4: **if** Ausgabe = „Eigenschaft $Dist(T'_E)$ nicht erfüllt“ **then**

5: **return** FAIL.

6: **else**

7: Seien $\tilde{T}$ und T' die Ausgaben von Algorithmus „Hinzufügen von Kanten“, also die Lösung des Entscheidungsproblems und der Unterbaum. Setze $\tilde{T}^* = \tilde{T}$, $T'^* = T'$.

8: **end if**

→ i-ter Schritt

9: **for** i=a,. . . , b **do**

10: Führe Algorithmus „Hinzufügen von Kanten“ mit $d_0 = d_i$ und Unterbaum T'^* aus.

11: **if** Ausgabe = „Eigenschaft $Dist(T'^*)$ nicht erfüllt“ **then**

12: Gehe zu Schritt 19.

13: **else if** FAIL **then**

14: **return** : Setze untere Schranke $d_s = d_i$. Gehe zu Schritt 19.

15: **else**

16: Seien $\tilde{T}$ und T' die Ausgaben von Algorithmus „Hinzufügen von Kanten“, also die Lösung des Entscheidungsproblems und der Unterbaum. Setze $\tilde{T}^* = \tilde{T}$, $T'^* = T'$, $d^s = d_i$.

17: **end if**

18: **end for**

19: **return** Beste gefundene Lösung $\tilde{T}^*$ mit Zielfunktionswert d^s, untere Schranke d_s für den Zielfunktionswert, in der Optimallösung enthaltener Unterbaum T'^*.

Lemma 93 *Algorithmus „A-Baum-Baum 1“ ist korrekt.*

Beweis. Der Algorithmus führt für absteigende d_i Algorithmus „Hinzufügen von Kanten“ für den Unterbaum T'_E beziehungsweise T'^* aus. T'_E muss laut Problemstellung in jeder Lösung mit Zielfunktionswert $d \leq d^s$ enthalten sein. Induktiv zeigt man, dass das für jedes T'^* gilt:
Sei der im i-ten Schritt konstruierte Baum T'^*_i in jeder Lösung von $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_i$ enthalten. Das heißt die Instanzen $(T, OD, \alpha, l_0, d, T'_E)$ und $(T, OD, \alpha, l_0, d, T'^*_i)$ haben die gleichen Lösungen. Aus Lemma 87 folgt, dass das mit Hilfe von Algorithmus „Hinzufügen von Kanten“ konstruierte T'^*_{i+1} in jeder Lösung von $(T, OD, \alpha, l_0, d, T'^*_i)$ mit $d \leq d_{i+1}$ enthalten sein muss. Nach Induktionsvoraussetzung ist das für $d \leq d_{i+1} < d_i$ aber äquivalent dazu, dass T'^*_{i+1} in jeder Lösung von $(T, OD, \alpha, l_0, d, T'_E)$ mit $d \leq d_{i+1}$ enthalten ist.
Aus Lemma 87 folgt dann die Korrektheit von $\tilde{T}^*$ und d^s in jedem Schritt. Wegen der absteigenden Anordnung der d_j ist jede gefundene Lösung mit Zielfunktionswert d_i besser als die vorherige mit Zielfunktionswert d_{i-1}. In T'^*, $\tilde{T}^*$ und d^s werden also die besten bekannten Lösungen gespeichert und mit neu gefundenen Lösungen aktualisiert. Ist Eigenschaft $Dist(T'_E)$ erfüllt, aber Algorithmus „Hinzufügen von Kanten“ gibt FAIL aus, so kann nach Lemma 87 kein Unterbaum $\tilde{T}$ mit den gewünschten Eigenschaften gefunden werden, die untere Schranke in Schritt 14 ist korrekt. Die Werte in $\tilde{T}^*$, T'^*, d^s und d_s stellen die besten gefundenen Lösungen dar und werden in jedem Schritt aktualisiert. ■

Lemma 94 *Algorithmus „A-Baum-Baum 1“ ist in $O(n^5)$.*

Beweis. Sei ohne Beschränkung der Allgemeinheit T rekursiv dargestellt (Lemma 4 und Bemerkung 6). In Schritt 1 werden für $O(n^2)$ OD-Paare die Distanzen zwischen ihren Knoten bestimmt (durch Ausführung von Algorithmus „Wege in Bäumen“ in $O(n)$ und Addieren der Kantenlängen) und die OD-Paare dabei sortiert. Also ist Schritt 1 in $O(n^3)$. Die Ausführung von Algorithmus „Hinzufügen von Kanten“ ist nach Lemma 87 in $O(n^3)$. Da dieser Algorithmus höchstens $\big(|OD_{krit}| + 2\big)$-mal ausgeführt wird, ist das Verfahren also in $O(n^5)$. ■

Bemerkung 95 *Sind keine besseren Schranken d^s und d_s bekannt, so können die Schranken aus Lemma 92 verwendet werden. In diesem Fall ist in Schritt 1 $a = 2$.*

Korollar 96 *Wählt man als Eingabe von Algorithmus „A-Baum-Baum 1“ einen von Algorithmus „Konstruktion Unterbaum“ für ein d_0 konstruierten Unterbaum T'_E, so ist, falls der Algorithmus in Schritt 14 oder 19 mit Zielfunktionswert $d^s \leq d_0$ terminiert, der ausgegebene Baum T^* längenminimal unter den Bäumen, die die Instanz $(T, OD, \alpha, l_0, d^s)$ lösen. Der Baum T' ist dann Unterbaum jeder Lösung einer Instanz (T, OD, α, l_0, d) mit $d \leq d^s$.*

Beweis. Nach Satz 80 ist T'_E Unterbaum jeder Lösung einer Instanz (T, OD, α, l_0, d) mit $d \leq d_0$, also insbesondere auch Unterbaum einer Lösung von $(T, OD, \alpha, l_0, d^s)$. Da die Konstruktion von T' analog zum Hinzufügen von Kanten in Algorithmus „Konstruktion Unterbaum“ erfolgt, ergibt sich wie in Lemma 80, dass der Baum T' Unterbaum jeder Lösung einer Instanz (T, OD, α, l_0, d) mit $d \leq d^s$ ist. Der Rest des Korollars folgt aus Korollar 88 und Lemma 93. ■

In dem im Korollar betrachteten Fall kann die Lösung mit Hilfe von Lemma 90 und 91 auf Optimalität überprüft werden. Lassen diese Kriterien nicht auf Optimalität schließen, so kann keine weitere Optimalitätsaussage getroffen werden, da es möglich ist, dass der optimale Zielfunktionswert D zwischen d_s und d^s liegt. Dies stellt einen Nachteil gegenüber dem im nächsten Abschnitt dargestellten Verfahren „A-Baum-Baum 2“ dar.

6.4 Ein Verfahren zur exakten Lösung von Spezialfällen von *A-Baum-Baum*

Das Vorhandensein von Algorithmus „Entscheidung A-Baum-Baum“ zur Konstruktion von Lösungen für bestimmte Instanzen des Entscheidungsproblems *A-Baum-Baum* und des in Lemma 91 gegebenen zweiten Optimalitätskriteriums motiviert folgendes Verfahren, das für geeignete Instanzen des Optimierungsproblems *A-Baum-Baum* durch Intervallhalbierung exakte Lösungen berechnet.

Algorithmus 6 A-Baum-Baum 2

Eingabe: Baum $T = (V, E)$ mit Kantenlängen $c_{ij}\ \forall [i,j] \in E$; OD Menge von OD-Paaren; α; l_0; Schranken d^s und d_s; Schrittanzahl m; δ wie in Lemma 91 oder kleiner.

Ausgabe: Optimallösung $\hat{T}$ oder Lösung T^* des Entscheidungsproblems $(T, OD, \alpha, l_0, d_0)$ für das kleinste d_0, für das eine Lösung gefunden wurde; FAIL falls für kein d_0 eine Lösung gefunden wurde.

→ Initialisierung

Setze $T = \emptyset$, $d^0 = d^s$, $C = \frac{d^s - d_s}{2}$, $d^1 = d_s + C$, $t = d^s$, $s = d_s$

Berechne $d(u,v)$ für alle $\{u,v\} \in OD$, sortiere OD entsprechend.

→ Verfahren

for $i = 1, \ldots, m$ **do**

  $OD^i_{krit} = \left\{\{u,v\} \in OD : d(u,v) > d^i\right\}$

  Führe Algorithmus „Konstruktion Unterbaum“ für $(T, OD, \alpha, l_0, d_i)$ mit Ausgabe T'_i aus.

  if FAIL **then**

    $s = d^i,\ d^{i+1} = d^i + \frac{1}{2^i}C$

  else

    if $T' \neq \emptyset$ **then**

      Führe Algorithmus „Hinzufügen von Kanten“ mit $d_0 = d^{i+1}$ und Unterbaum T'_i aus.

      if Ausgabe = „Eigenschaft $Dist(T'_i)$ nicht erfüllt.“ **then**

        $\tilde{s} = d^i,\ d^{i+1} = d^i + \frac{1}{2^i}C$

      else if Ausgabe = FAIL **then**

        $s = d^i,\ d^{i+1} = d^i + \frac{1}{2^i}C$

      else

        $t = d^i,\ d^{i+1} = d^i - \frac{1}{2^i}C$, speichere Ausgabe als T^*

      end if

    else

      $\tilde{t} = d^i,\ d^{i+1} = d^i - \frac{1}{2^i}C$

    end if

  end if

  if $t - s \leq (1 - \alpha)\delta$ **then**

    return $\hat{T} = T^*$ ist Optimallösung

24: **end if**

25: **if** $t - \tilde{s} \leq (1-\alpha)\delta$ **then**

26: **return** T^* löst $(T, OD, \alpha, l_0, d^i)$

27: **end if**

28: **if** $\tilde{t} - s \leq (1-\alpha)\delta$ oder $\tilde{t} - \tilde{s} \leq (1-\alpha)\delta$ **then**

29: **return** FAIL

30: **end if**

31: **end for**

Satz 97 *Algorithmus „A-Baum-Baum 2“ ist korrekt.*

Beweis. Algorithmus „A-Baum-Baum 2“ berechnet Schranken s und t für die Optimallösung des Problems *A-Baum-Baum*. Gibt Algorithmus „Konstruktion Unterbaum“ FAIL aus, so gibt es nach Satz 80 keine Lösung der Instanz $(T, OD, \alpha, l_0, d^i)$, die untere Schranke in Schritt 7 ist also korrekt. Findet Algorithmus „Hinzufügen von Kanten“ eine Lösung der Instanz $(T, OD, \alpha, l_0, d^i)$, so ist d^i eine obere Schranke für den optimalen Zielfunktionswert. Gibt er FAIL aus, so ist die Instanz $(T, OD, \alpha, l_0, d^i)$ nicht lösbar und natürlich auch keine Instanz mit gleichem T, OD, α und l_0 und kleinerem d_0. Deswegen ist d^i in diesem Fall eine untere Schranke für den optimalen Zielfunktionswert. Die Optimalitätsaussage in Schritt 23 folgt dann aus Lemma 91. Die Aussage in Schritt 26 folgt aus aus der Definition von d^0, falls $d^i = d^0$ und sonst aus Korollar 88. ■

Bemerkung 98 *Als anfängliche Schranken können die Schranken aus Lemma 92 gewählt werden. Bei sinnvoller Wahl der Schranken ist $d^s \leq d_1$, im Folgenden wird von einer solchen sinnvollen Wahl ausgegangen.*

Lemma 99 *Seien δ, d^s und d_s die Eingabewerte von Algorithmus „A-Baum-Baum 2“ und sei*

$$\tilde{m} = \lceil (\log(d^s - d_s) - \log(1-\alpha) - \log(\delta)) / \log 2 \rceil - 1. \tag{142}$$

Dann bricht Algorithmus „A-Baum-Baum 2“ nach dem $\tilde{m}$-ten Schritt ab.

Beweis. Im i-ten Schritt beträgt die Länge des betrachteten Intervalls $\left(d^i, d^{i+1}\right]$ beziehungsweise $\left(d^{i+1}, d^i\right]$, in dem der Zielfunktionswert liegt

$$|d^{i+1} - d^i| = \frac{1}{2^i} C = \frac{1}{2^i} \frac{d^s - d_s}{2} = \frac{d^s - d_s}{2^{i+1}}, \tag{143}$$

im $\tilde{m}$-ten Schritt also

$$\begin{aligned}\frac{d^s - d_s}{2^{\tilde{m}+1}} &= \frac{d^s - d_s}{\exp(\log 2 \left\lceil \frac{\log(d^s - d_s) - \log(1-\alpha) - \log(\delta))}{\log 2} \right\rceil)} \\ &\leq \frac{d^s - d_s}{\exp(\log(d^s - d_s) - \log(1-\alpha) - \log(\delta)))} \\ &= \frac{d^s - d_s}{\exp(\log \frac{d^s - d_s}{(1-\alpha)\delta})} \\ &= \delta(1-\alpha). \end{aligned} \tag{144}$$

Darum stoppt der Algorithmus in 23, 26 oder 29. ■

Satz 100 *Sei $T = (V, E)$ ein Baum. Gilt mit $D := \min_{\hat{T}:\ l(\hat{T}) \leq l_0} \max_{\{u,v\} \in OD} d_{\hat{T}}(u, v)$ und $\tilde{OD} = \{\{u, v\} \in OD : d(u, v) > D - (1-\alpha)\delta\}$*

$$E_{u_1 v_1} \cap E_{u_2 v_2} = \emptyset \ \forall \ \{u_1, v_1\}, \{u_2, v_2\} \in \tilde{OD}, \tag{145}$$

so lässt sich die Instanz (T, OD, α, l_0) von Optimierungsproblem A-Baum-Baum optimal lösen in $O(n^3 \max(\log(d_1), -\log(1-\alpha), -\log \delta))$.

Beweis. Sei ohne Beschränkung der Allgemeinheit T rekursiv dargestellt (Lemma 4 und Bemerkung 6). Da die Berechnung der Wege zwischen den OD-Paaren für jedes der $O(n^2)$ OD-Paare in $O(n)$ ist (Lemma 9), ist Schritt 2 in $O(n^3)$. Schritt 4 ist in $O(n^2)$, da die Weglängen der $O(n^2)$ OD-Paare schon in Schritt 2 berechnet wurden und deshalb bekannt sind. Die Ausführung von Algorithmus „Konstruktion Unterbaum“ ist nach Satz 80 in $O(n^3)$. Algorithmus „Hinzufügen von Kanten“ ist in $O(n^3)$ nach Lemma 87. Da die Schritte 4-26 höchstens $\tilde{m}$-mal ausgeführt werden mit $\tilde{m} = \lceil \log(d^s - d_s) - \log(1-\alpha) - \log(\delta) \rceil$ (Lemma 99), ist der Algorithmus also in $O(\tilde{m} n^3) = O(n^3 \max(\log(d_1), -\log(1-\alpha), -\log \delta))$.
War das betrachtete Intervall $(s, t]$ mit echter unterer Schranke s, so ist nach Lemma 91 der von Algorithmus „Entscheidung A-Baum-Baum“ als letztes konstruierte Baum $\hat{T}$ optimal.
Die Bedingung

$$(E_{u_1 v_1} \cap E_{u_2 v_2}) = \emptyset \ \forall \ \{u_1, v_1\}, \{u_2, v_2\} \in \tilde{OD} \tag{146}$$

stellt sicher, dass von „A-Baum-Baum 2“ tatsächlich eine untere Schranke s gefunden wurde: Da die Grenzen für den optimalen Zielfunktionswert maximal den Abstand $\delta(1-\alpha)$ haben, ist $s > (D - (1-\alpha)\delta)$ und aus (146) folgt insbesondere

$$(E_{u_1 v_1} \cap E_{u_2 v_2}) \setminus E_{T'} = \emptyset \ \forall \ \{u_1, v_1\}, \{u_2, v_2\} \in \{\{u, v\} \in OD : d(u, v) > s\}. \tag{147}$$

Galt Bedingung (146), so löst der Algorithmus also in $O(n^3 \max(\log(d_1), -\log(1-\alpha), -\log\delta))$ das Optimierungsproblem *A-Baum-Baum.* ■

Bemerkung 101 *In Satz 100 kann man* $D = \min_{\hat{T}:\ l(\hat{T}) \leq l_0} \max_{\{u,v\} \in OD} d_{\hat{T}}(u,v)$ *durch eine untere Schranke für den optimalen Zielfunktionswert ersetzen.*

Bemerkung 102 *Auch Algorithmus „A-Baum-Baum 2" lässt sich so modifizieren, dass er auf Probleme mit schon gegebenem Unterbaum* T'*, der in der Lösung enthalten sein muss, anwendbar ist. Statt in Schritt 5 Algorithmus „Konstruktion Unterbaum" auszuführen und mit dem berechneten Unterbaum* $T'_i \neq \emptyset$ *fortzufahren, führe man nur Algorithmus „Hinzufügen von Kanten" mit dem gegebenen Unterbaum* T' *aus. In Schritt 23 stellt die ausgegebene Lösung dann allerdings nur noch eine Optimallösung des Optimierungsproblems, bei dem zusätzlich* T' *in der Lösung enthalten sein muss, dar. Die Abbruchbedingungen und die Komplexität des Verfahrens werden dadurch höchstens indirekt (über die Wahl der Schranken) beeinflusst.*

6.5 Heuristische Lösung von *A-Baum-Baum*

Das in Kapitel 6.4 beschriebene Verfahren zur Lösung des Optimierungsproblems *A-Baum-Baum* führt nur unter Voraussetzung bestimmter starker Bedingungen für die Wege zwischen den OD-Paaren zur optimalen Lösung einer Instanz (T, OD, α, l_0). Da schon das Entscheidungsproblem *A-Baum-Baum* stark NP-vollständig ist (Satz 23), wird sich unter der Annahme $P \neq NP$ im Allgemeinen kein polynomielles oder pseudopolynomielles Verfahren zur Konstruktion einer Optimallösung des Optimierungsproblems finden lassen. Deswegen ist dieses Kapitel der Entwicklung einer Heuristik gewidmet. Das den bisherigen Verfahren zugrunde liegende Prinzip, aufbauend auf einem Unterbaum, der in der Optimallösung enthalten sein muss, Kanten hinzuzufügen, um die Distanz zwischen den OD-Paaren zu vermindern, wird dabei beibehalten und heuristisch umgesetzt. Es werden folgende Bezeichnungen eingeführt:

Definition 103 Zusammenhangskanten/Distanzverringerungskanten
Sei $T = (V, E)$ *ein Baum,* $\hat{T} = (V_{\hat{T}}, E_{\hat{T}})$ *ein Unterbaum von* T*. Sei* $\{u, v\}$ *ein OD-Paar mit* $\hat{T} \cap (u, v) = \emptyset$*. Sei* $W_{\hat{T},(u,v)}$ *der Weg von* $\hat{T}$ *nach* (u, v)*. Mit Zusammenhangskanten seien im Folgenden die Kanten* $e \in E_{W_{\hat{T},(u,v)}}$ *bezeichnet, da sie, falls die Distanz zwischen* u *und* v *durch eine Erweiterung von* $\hat{T}$ *verkleinert werden soll, aufgenommen werden müssen, damit der entstehende Graph zusammenhängend und somit ein Baum bleibt.*
Distanzverringerungskanten werden dagegen die Kanten genannt, die zusätzlich aufgenommen werden, damit für den entstehenden Baum $\hat{T}_e$ *gilt* $d_{\hat{T}_e}(u, v) < d_{\hat{T}}(u, v)$*.*

6.5.1 Heuristische Bestimmung eines Unterbaumes

Im Folgenden werden zwei unterschiedliche Methoden zur Erstellung eines Unterbaumes vorgestellt.

Bestimmung eines Knotens als Unterbaum

Essenziell für das spätere Anfügen von Zusammenhangs- und Distanzverringerungskanten ist die Existenz eines Unterbaumes als „Ansatzpunkt", von dem ausgehend die Kanten zusammenhängend hinzugefügt werden. Mit folgendem Verfahren bestimmt man nur einen einzelnen Knoten als Unterbaum.

Algorithmus 7 Heuristik Konstruktion Unterbaum 1

Eingabe: Baum $T = (V, E)$ mit Kantenlängen $c_{ij}\ \forall\ [i, j] \in E$; Menge von OD-Paaren OD.

Ausgabe: Knoten $i \in V$.

1: Berechne $d(u, v)\ \forall\ \{u, v\} \in OD$.

2: Wähle (zufällig) OD-Paar $\{u, v\}$ mit $d(u, v)$ maximal.

3: Wähle ein $i \in (u, v)$ als Unterbaum.

Da der Knoten i im weiteren Verlauf der Lösungskonstruktion als Ansatzpunkt für die hinzuzufügenden Kanten dient, spielt seine Wahl eine wichtige Rolle bei der Bestimmung eines guten Baumes bezüglich der Zielfunktion von Problem A. Es folgt eine Liste von möglichen Auswahlregeln für den Knoten i, der als Unterbaum gewählt wird. Liefert eine Auswahlregel kein eindeutiges Ergebnis, so kann zufällig oder anhand einer weiteren Auswahlregel einer der ausgewählten Knoten als Unterbaum bestimmt werden.

1. *Wähle den Knoten, der bezüglich der Anzahl der Kanten in der Mitte des Pfades* (u, v) *liegt.*

   Sobald bei der Aufnahme der Distanzverringerungskanten für ein OD-Paar $\{u, v\}$ eine zu u oder v inzidente Kante aufgenommen wurde, gibt es keine Wahlmöglichkeit bei der Aufnahme weiterer Distanzverringerungskanten für den Pfad (u, v) mehr. Ein Start mit einem Knoten in der Mitte des Pfades (u, v) soll sicherstellen, dass die Endknoten, wenn überhaupt, erst nach möglichst vielen Schritten erreicht werden. Dieses Auswahlkriterium ist aber nur relevant, wenn mehrere Kanten vom Pfad (u, v) eingefügt werden. Die Zeitkomplexität dieses Verfahrens ist $O(n)$.

2. *Wähle den Knoten, der bezüglich der Länge in der Mitte des Pfades* (u, v) *liegt.*

   Wie bei Auswahlregel 1 soll auch mit dieser Regel dafür gesorgt werden, dass die Endkno-

ten des das OD-Paar verbindenden Pfades erst spät erreicht werden. Nimmt man als Maß für den Forschritt des Verfahrens nicht die Anzahl der aufgenommenen Kanten, sondern die Länge des erstellten Teilgraphen, kommt man auf diese Auswahlregel, die ebenfalls in $O(n)$ ist. Wie Regel 1 kommt ihr Vorteil aber nur zum Tragen, wenn viele Kanten des Weges zwischen dem betrachteten OD-Paar aufgenommen werden.

3. *Wähle den Knoten mit höchstem Kantengrad.*
   Ein hoher Kantengrad kann die Wahrscheinlichkeit erhöhen, dass sich im aufgenommenen Knoten i viele OD-Paare kreuzen und somit für diese OD-Paare später keine Zusammenhangskanten mehr angefügt werden müssen. Für einen rekursiv dargestellten Baum kann man den Kantengrad von i für $i = 1$ als Anzahl der Knoten mit $C(j) = i$, für $i > 1$ als (1+ Anzahl der Knoten) mit $C(j) = i$ in $O(n)$ bestimmen. Um einen Knoten i zu finden, in dem sich tatsächlich viele OD-Paare kreuzen, wende man Auswahlregel 6 an. Diese ist allerdings in $O(n^3)$.

4. *Wähle den Knoten, der adjazent zu der Kante in (u, v) mit kleinstem Kantengewicht ist.*
   Bei dieser Wahl steht im weiteren Verlauf der Konstruktion die Aufnahme einer Kante zur Auswahl, die die Distanz zwischen u und v verkleinert, die Länge des zu erstellenden Teilbaumes aber weniger stark vergrößert als andere solche Kanten. Das könnte in der Endphase des Verfahrens von Nutzen sein oder falls es viele OD-Paare mit hoher Distanz gibt und eine große Verringerung der maximalen Distanz nicht möglich ist. Die Auswahl erfolgt auch hier in $O(n)$.

5. *Wähle den Knoten für den die Summe der Länge der adjazenten Kanten, geteilt durch ihre Anzahl, minimal ist.*
   Die Wahl dieses Auswahlkriteriums ist mit ähnlichen Argumenten wie Regel 4 zu begründen. Auch sie ist in $O(n)$.

6. *Lege eine Schranke $\tilde{d}$ fest. Wähle $i \in V$ so, dass $\left| \left\{ \{\tilde{u}, \tilde{v}\} : d(u, v) > \tilde{d};\ i \in V_{(\tilde{u},\tilde{v})} \right\} \right|$ maximal ist.*
   Von dem gewähltem i ausgehend lassen sich Kanten zu vielen OD-Paaren hinzufügen, ohne dass Zusammenhangskanten eingefügt werden müssen. Wählt man $\tilde{d}$ zu klein, werden jedoch OD-Paare, deren Distanz kleiner als $\min_{l(T^*) \leq l_0} \max_{\{\tilde{u},\tilde{v}\} \in OD} d_{T^*}(u, v)$ ist, betrachtet und beeinflussen die Wahl von i, obwohl ihr Zielfunktionswert für die Bestimmung des Maximums nicht relevant ist. Für die $O(n)$ Knoten in (u, v) muss überprüft

werden, in wie vielen der $O(n^2)$ Wege zwischen OD-Paaren sie liegen, also ist dieses Vorgehen in $O(n^3)$.

Lemma 104 *Das Verfahren „Heuristik Konstruktion Unterbaum 1" ist in* $O(n^3)$.

Beweis. Das Bestimmen eines maximalen OD-Paares in Schritt 1 ist in $O(n^3)$, die Zeitkomplexität der Auswahlregeln ist direkt bei ihrer Beschreibung aufgeführt. ■

Zweite Möglichkeit zur Erstellung eines Unterbaumes

In diesem Verfahren wird ein Unterbaum erstellt, der im Allgemeinen aus mehr als einem Knoten besteht. Wird in Schritt 5 oder 7 auf eine heuristische Auswahlregel aus vorigem Abschnitt zurückgegriffen, so besteht er jedoch nur aus einem Knoten und auch bei Konstruktion eines Unterbaumes mit Hilfe des exakten Algorithmus ist ein nur aus einem Knoten bestehender Unterbaum möglich.

Algorithmus 8 Heuristik Konstruktion Unterbaum 2

Eingabe: Baum $T = (V, E)$ mit Kantenlängen $c_{ij}\ \forall\ [i,j] \in E$; Menge von OD-Paaren OD.

Ausgabe: Unterbaum T'.

1: Berechne $d(u,v)\ \forall\ \{u,v\} \in OD$.
2: Wähle eine Menge OD_{krit} von OD-Paaren mit zugehörigem $d_0 = \max_{\{u,v\} \in OD \setminus OD_{krit}} d(u,v)$ (siehe unten).
3: Führe Algorithmus „Konstruktion Unterbaum" für d_0 aus.
4: **if** FAIL **then**
5: Vergrößere d_0 und wiederhole das Vorgehen oder greife auf eine der oben aufgeführten Regeln zur Auswahl eines $i \in (u,v)$ mit $d(u,v)$ maximal zurück.
6: **else if** Ausgabe $T' = \emptyset$ **then**
7: Verkleinere d_0 und wiederhole das Vorgehen oder greife auf eine der oben aufgeführten Regeln zur Auswahl eines $i \in \bigcap_{\{u,v\} \in OD_{krit}} V_{(u,v)}$ zurück.
8: **else**
9: **return** T'
10: **end if**

Zwei mögliche Regeln zur Wahl von OD_{krit} und d_0 sind:

1. *Auswahl aller OD-Paare* $\{u,v\}$ *mit* $d(u,v) > \tilde{d}$ *für festgelegtes* $\tilde{d}$.
   Der Distanzvergleich für die $O(n^2)$ OD-Paare erfolgt in $O(n^2)$, da die Distanzen schon in Schritt 1 berechnet wurden.

2. *Auswahl der k OD-Paare mit größter Distanz für festgelegtes k und aller weiteren mit der gleichen Distanz wie das k-te OD-Paar.*

   Ein solches k wird festgelegt, damit die Schnittbildung bei der Konstruktion des Unterbaumes einen leeren Schnitt liefert und nicht schon dabei auf ein heuristisches Verfahren zurückgegriffen werden muss. Entscheidet man sich für diese Regelung, kann die minimale Distanz zwischen den Knoten eines OD-Paares aus OD_{krit} allerdings nicht mehr direkt kontrolliert werden. Die Auswahl erfolgt in $O(n^2)$.

Lemma 105 *Das Verfahren ist in $O(n^3)$, falls die Anzahl der zugelassenen erneuten Aufrufe des Verfahrens in Schritt 5 und 7 durch eine Zahl x unabhängig von n beschränkt wird.*

Beweis. Die Berechnung der Distanzen der OD-Paare erfolgt in $O(n^3)$. Die Durchführung von Algorithmus „Konstruktion Unterbaum“ ist in $O(n^3)$ (Satz 80), die Auswahlregeln sind in $O(n^2)$. Wird der Algorithmus nur höchstens x-mal durchlaufen, wobei x nicht von n abhängt, ergibt sich ohne Anwendung der Auswahlregeln aus dem vorherigen Abschnitt eine Komplexität von $O(n^3)$. Da diese Auswahlregeln auch in $O(n^3)$ sind, ist das gesamte Verfahren in $O(n^3)$. ■

Bemerkung 106 *Dieses zweite heuristische Verfahren zur Konstruktion eines Unterbaumes gibt einen Unterbaum aus, der in der optimalen Lösung enthalten sein muss, falls für die gewählte Menge OD_{krit} gilt:*

$$\min_{l(\hat{T})\leq l_0} \max_{\{u,v\}\in OD} d_{\hat{T}}(u,v) < \max_{\{u,v\}\in OD_{krit}} d(u,v) \tag{148}$$

und Algorithmus „Konstruktion Unterbaum“ bei der letzten Durchführung nicht $T = \emptyset$ oder FAIL ausgibt (vergleiche Lemma 80). Eine gute Wahl von OD_{krit} durch Festlegen von d_0 oder k erfordert allerdings im Allgemeinen Kenntnisse über das Problem, insbesondere die ungefähre Höhe des Zielfunktionswerts und die Wege zwischen den OD-Paaren.

6.5.2 Hinzufügen der Kanten an den erstellten Unterbaum

Beim Hinzufügen der Kanten gibt es zwei Möglichkeiten:

1. Der im vorherigen Schritt erstellte Baum $\hat{T}$ wird vollständig übernommen. Das heißt es wird für alle OD-Paare $\{u,v\}$ $d_{\hat{T}}(u,v)$ berechnet und davon abhängig das nächste kritische OD-Paar/ die nächsten kritischen OD-Paare bestimmt.

2. Der Baum $\hat{T}$ aus dem vorherigen Schritt wird nicht übernommen. Die Distanzverringerungskanten werden an einen neu erstellten oder aus dem vorherigen Schritt übernommenen Unterbaum $\hat{T}'$ aus Zusammenhangskanten angefügt.

Möglichkeit 1 - In jedem Schritt Übernahme des vollständigen Baumes $\hat{T}$

Algorithmus 9 Heuristik 1 Konstruktion Lösung von *A-Baum-Baum*

Eingabe: Baum $T = (V, E)$ mit Kantenlängen c_{ij} $\forall$ $[i, j] \in E$; Menge von OD-Paaren OD; Beschleunigungsfaktor α; Maximallänge l_0.

Ausgabe: Baum $\hat{T}$ mit $l(\hat{T}) \leq l_0$.

1: Bestimme den ersten $\hat{T}$ Unterbaum mit einem der Verfahren aus 6.5.1.
2: **while** $l(\hat{T}) \leq l_0$ **do**
3: Berechne $d_{\hat{T}}(u, v)$ für alle $\{u, v\} \in OD$.
4: Wähle (zufällig) ein maximales OD-Paar.
5: Füge die Zusammenhangskanten zwischen (u, v) und $\hat{T}$ zu $\hat{T}$ hinzu wie in Schritt 5-11 von Algorithmus Konstruktion Unterbaum.
6: Füge eine Kante zu $\hat{T}$ hinzu.
7: **end while**
8: Entferne die im letzten Durchlaufen von Schritt 3-6 hinzugefügten Kanten wieder.
9: **return** $\hat{T}$

Es folgen einige Auswahlregeln für das Hinzufügen der Kanten in Schritt 6. Man beachte dabei, dass diese Wahl immer nur zwischen (maximal) zwei Kanten aus dem Weg (u, v) geschieht. Ist trotzdem durch die Regel keine eindeutige Kante bestimmt, wähle man zufällig eine der beiden Kanten oder benutze eine andere Auswahlregel.

1. *Wähle die kürzere Kante.*
   Die Länge von $\hat{T}$ wird minimal vergrößert. Dafür nimmt die Distanz zwischen dem gewählten OD-Paar aber auch nur minimal ab. Diese Regel eignet sich, wenn es viele OD-Paare mit ähnlicher, großer Distanz, aber wenigen gemeinsamen Kanten gibt. Da nur zwischen zwei Kanten gewählt wird, ist diese Auswahlregel in $O(1)$.

2. *Wähle die längere Kante.*
   Die Distanz zwischen dem gewählten OD-Paar wird maximal verringert; es wird eine Kante aufgenommen, die im späteren Verlauf eventuell nicht mehr aufgenommen werden kann, da sie zu lang ist. Die Auswahlregel eignet sich für den Anfang des Verfahrens, wenn es wenige OD-Paare mit großer Distanz gibt oder diese viele Kanten gemeinsam haben. Wie Regel 1 ist auch diese Regel in $O(1)$.

3. *Wähle die Kante, die in den meisten Wegen zwischen OD-Paaren $\{u, v\}$ mit $d_{\hat{T}}(u, v) > \tilde{d}$ für eine Schranke $\tilde{d}$ liegt.*

   Diese Kante verringert gleichzeitig die Distanz zwischen mehreren OD-Paaren, so dass für diese weniger Kanten hinzugefügt werden müssen. Wählt man $\tilde{d}$ allerdings zu klein, so werden viele OD-Paare in die Auswahl einbezogen, deren Distanz in $T_{\hat{T}}$ schon kleiner als der optimale Zielfunktionswert ist, also gar nicht verkleinert zu werden braucht. Diese Auswahlregel ist in $O(n^3)$, da die in $O(n)$ berechneten Wege zwischen $O(n^2)$ OD-Paare überprüft werden müssen.

4. *Wähle ein $\tilde{d}$. Sei $\tilde{O} = \left\{\{u_i, v_i\} \in OD : d_{\hat{T}}(u_i, v_i) > \tilde{d}\right\}$. Betrachte die Wege $W_{\hat{T},(u_i,v_i)}$ zwischen $\hat{T}$ und den Wegen (u_i, v_i) für $\{u_i, v_i\} \in \tilde{OD}$. Wähle die Kante, die in den meisten $W_{\hat{T},(u_i,v_i)}$ enthalten ist.*

   Werden besagte OD-Paare aufgenommen, so müsste die gewählte Kante als Zusammenhangskante aufgenommen werden, ihre Aufnahme geht also in diesem Fall nicht zusätzlich in die Länge des Baumes $\hat{T}$ ein. Da für $O(n^2)$ OD-Paare überprüft werden muss, welche der beiden zur Auswahl stehenden Kanten auf dem in $O(n)$ berechneten Weg $W_{\hat{T},(u_i,v_i)}$ liegen, ist auch dieses Verfahren in $O(n^3)$. Bei der Wahl von $\tilde{d}$ tritt allerdings das selbe Problem wie in Regel 3 auf.

Lemma 107 *„Heuristik 1“ zur Konstruktion einer Lösung von A-Baum-Baum ist in $O(n^4)$.*

Beweis. Nach Lemma 104 und 105 erfolgt die Bestimmung des ersten Unterbaumes in $O(n^3)$. Die Berechnung der Distanzen zwischen allen OD-Paaren und die Auswahl von einem mit maximaler Distanz ist in $O(n^3)$. Die Zeitkomplexität der Auswahlregeln ist oben aufgeführt. Da in jedem Schritt mindestens eine Kante hinzukommt, sind nach spätestens n Schritten alle Kanten in $\hat{T}$ aufgenommen. Alle Auswahlregeln sind in $O(n^3)$, deswegen folgt, dass das Verfahren in $O(n^4)$ ist. ■

Es ist möglich, dass bei diesem Verfahren durch ungünstig gewählte Auswahlregeln Kanten aufgenommen werden, die im Nachhinein unnötig sind, da der Baum $\hat{T}$ auch nach ihrem Entfernen zusammenhängend bliebe und die maximale Distanz nicht abnehmen würde. Werden solche Kanten entfernt, nimmt die Länge von $\hat{T}$ ab, so dass durch neues Hinzufügen von Kanten eventuell der Zielfunktionswert verbessert werden kann, ohne dass die Maximallänge l_0 durch $\hat{T}$ überschritten wird. In diesem Fall kann möglicherweise eine Verbesserung des Zielfunktionswerts erreicht werden, indem statt Schritt 8 und 9 in der „Heuristik 1“ zur Konstruktion einer Lösung von *A-Baum-Baum* folgende Schritte ausgeführt werden:

Algorithmus 10 Verbesserung des Zielfunktionswerts

1: Berechne $d := \max_{\{u,v\} \in OD} d_{\hat{T}}(u, v)$.
2: **for all** Blätter b (auch die neu entstehenden) von $\hat{T}$ **do**
3: **if** $d_{\hat{T} \setminus e_b}(u, v) \leq d$ für die zu b inzidente Kante $e_b \in E_{\hat{T}}$ **then**
4: Entferne e_b aus $E_{\hat{T}}$.
5: **end if**
6: **if** $l(\hat{T}) > l_0$ **then**
7: Fahre in „Heuristik 1“ zur Konstruktion Lösung von *A-Baum-Baum* mit Schritt 8 fort.
8: **else**
9: Fahre in „Heuristik 1“ zur Konstruktion Lösung von *A-Baum-Baum* mit Schritt 3 fort.
10: **end if**
11: **end for**

Diese Berechnung der Distanzen in $T_{\hat{T}}$ und ihres Maximums erfolgt in $O(n^3)$, die Gesamtüberprüfungszeit, ob für alle OD-Paare $d_{\hat{T} \setminus e_b}(u, v) > d$, ist in $O(n^2)$. Da diese Überprüfung für jedes der $O(n)$ Blätter durchgeführt wird, ist dieses Vorgehen in $O(n^3)$. Damit die „Heuristik 1“ zur Konstruktion einer Lösung von *A-Baum-Baum* auch unter Einbezug von Algorithmus „Verbesserung des Zielfunktionswert“ terminiert, sollte die Anzahl der möglichen Aufrufe dieses Algorithmus in „Heuristik 1“ begrenzt werden.

In folgendem Abschnitt, bei der Beschreibung des zweiten Verfahrens, werden in jedem Schritt mehrere Kanten angefügt bis die Distanz des betrachteten OD-Paares kleiner als ein vorgeschriebener Wert $\tilde{d}$ ist. Auch bei der Konstruktion mit Übernahme des vollständigen Baumes $\hat{T}$ in jedem Schritt wäre ein solches Verfahren denkbar. Dieses Vorgehen scheint hier aber nur dann sinnvoller als die Auswahl nur einer Kante zu sein, wenn das zu erreichende Distanzmaximum $\tilde{d}$ bekannt ist. Außerdem ist es aufwendiger, da nicht nur zwischen zwei Kanten, sondern zwischen $O(n)$ Kantenkombinationen gewählt wird. Hinzu kommt, dass eine gute Wahl von $\tilde{d}$ schwer zu treffen ist.

Möglichkeit 2 - Sukzessive Konstruktion von Lösungen

Eine andere Möglichkeit zur Konstruktion einer Lösung des Optimierungsproblems *A-Baum-Baum* ist es, ähnlich den Verfahren „ A-Baum-Baum 1“ oder „A-Baum-Baum 2“ vorzugehen, also für eine Folge von Schranken $d_1, d_2, \ldots$ eine Lösung der Instanz $(T, OD, \alpha, l_0, d_i)$ des Entscheidungsproblems zu suchen. Ein Schritt des Verfahrens sieht dann folgendermaßen aus:

Algorithmus 11 Schritt in Heuristik 2 zur Konstruktion einer Lösung

Eingabe: Baum $T = (V, E)$ mit Kantenlängen c_{ij} $\forall$ $[i, j] \in E$; Menge von OD-Paaren OD; Beschleunigungsfaktor α; Maximallänge l_0; Schranke d_i.

Ausgabe: Schranke d_{i+1} für den nächsten Schritt oder Lösung $\hat{T}$ von *A-Baum-Baum*.

1: Bestimme ersten Unterbaum T' mit einem der Verfahren aus 6.5.1 oder übernehme ihn aus früherer Lösung.
2: **for all** $\{u, v\} \in OD$ mit $d_{T'}(u, v) > d_i$ **do**
3: Füge die Zusammenhangskanten zu $\hat{T}'$ hinzu wie in Schritt 11-13 von Algorithmus „Konstruktion Unterbaum“.
4: Wähle wie in Algorithmus „Hinzufügen von Kanten“ die längenminimale Kombination von Kanten in (u, v), so dass der entstehende Baum $\hat{T}$ zusammenhängend bleibt und $d_{\hat{T}}(u, v) \leq d_i$.
5: **end for**
6: Prüfe $l(\hat{T}) \leq l_0$.
7: Lege neue Schranke d_{i+1} fest oder gebe $\hat{T}$ aus.

Die Schwierigkeit besteht hier in einer guten Wahl der d_i. Möglich wäre es zum Beispiel, eine absteigende Folge $d_1 > d_2 > \ldots$ zu wählen (zum Beispiel die Anfangsdistanzen der OD-Paare) und den letzten gefundenen zulässigen Baum als $\hat{T}$ als Lösung zu erhalten wie in Algorithmus „A-Baum-Baum 1“. Ebenfalls möglich wäre ein Intervallhalbierungsverfahren wie in Algorithmus „A-Baum-Baum 2“. Da die Wege zwischen den OD-Paaren sich im Allgemeinen überschneiden, sind die hinzugefügten Distanzverringerungskanten zwar optimal für die Verringerung der Distanz des jeweiligen OD-Paares, es kann aber nicht mehr insgesamt auf Optimalität geschlossen werden. Dadurch lassen sich Aussagen über die untere Schranke nicht von den genannten exakten Verfahren übernehmen und Optimalitätsaussagen sind im Allgemeinen nicht möglich.
Das Hinzufügen einzelner Kanten wie im vorherigen Verfahren ist hier nicht sinnvoll, da selbst wenn es für alle kritischen OD-Paare gleichzeitig durchgeführt werden würde, bei gleichbleibenden Auswahlregeln für einzelne Kanten unabhängig von d_i in jedem Schritt die gleichen Kanten

gewählt werden würden.

Lemma 108 *Ein Schritt des oben beschriebenen Verfahrens ist in* $O(n^5)$.

Beweis. Die Konstruktion eines Unterbaumes mit einem der obigen Verfahren ist in $O(n^3)$ (Lemma 104 und 105). Die Berechnung der $d_{T'}(u,v)$ für alle $\{u,v\} \in OD$ ist in $O(n^3)$, die Berechnung der Distanzverringerungskanten ebenfalls (siehe Beweis zu Lemma 87). Da die Berechnung aller Zusammenhangskanten in $O(n^3)$ ist (wie auch im Beweis zu Satz 80), ist jede Durchführung eines der oben beschriebenen Schritte 3 und 4 in $O(n^3)$. Die Durchführung der For-Schleife ist also in $O(n^5)$. ■

Die Komplexität des gesamten Verfahrens hängt dann von der Auswahlregel für die d_i ab (vergleiche entsprechende Aussagen über Algorithmus „A-Baum-Baum 1“ und „A-Baum-Baum 2“).
Bei der Konstruktion des ersten Unterbaumes, also des Grundgerüsts, bietet sich die zweite Möglichkeit zur Konstruktion eines Unterbaumes mit Schranke d_i an, da dann die Berechnung der Zusammenhangskanten entfällt.

6.6 Exakte Lösung von *A-Baum-Baum* durch Branch-and-Bound

Die Anzahl der in einem Baum enthaltenen Unterbäume ist endlich, also ist es prinzipiell möglich, das Optimierungsproblem *A-Baum-Baum* durch vollständige Enumeration zu lösen. Da die Anzahl der enthaltenen Unterbäume jedoch exponentiell in n ist (vergleiche [YY06]), ist ein solches Verfahren sehr aufwendig und für große Eingabelängen praktisch nicht durchführbar.
In solchen Fällen werden oft Branch-and-Bound Verfahren angewendet. Branch-and-Bound Verfahren lösen ein Problem, indem sie es solange in Teilprobleme unterteilen, bis die entstehenden Probleme klein genug sind, um ihre optimalen Lösungen zu bestimmen. Dabei können durch Bestimmung von Schranken für die Teilprobleme oft ganze Gruppen von möglichen Lösungen ausgeschlossen werden und müssen nicht mehr betrachtet werden. (siehe [Min86, 248ff.])
Im Folgenden wird dargestellt, wie die Technik Branch-and-Bound auf das Optimierungsproblem *A-Baum-Baum* angewendet werden kann. Dabei wird zunächst ein Basisverfahren eingeführt, danach werden zur Verbesserung des Verfahrens die heuristischen und exakten Ergebnisse aus den vorangegangenen Kapiteln verwendet.

Der kleinste mögliche Unterbaum eines nichtleeren Baumes besteht aus einem Knoten. Durch die Hinzunahme weiterer Knoten wird eindeutig ein minimaler diese Knoten enthaltender Baum

gegeben. Deshalb sollen hier im Allgemeinen in jedem Schritt nach Wahl eines Knotens v zwei Teilprobleme erzeugt werden:

1. Teilproblem 1: Das ursprüngliche Problem wird unter der Bedingung betrachtet, dass der Knoten v in den Teilbaum $\tilde{T}$ aufgenommen wird.

2. Teilproblem 2: Das ursprüngliche Problem wird unter der Bedingung betrachtet, dass der Knoten v nicht aufgenommen wird.

Nicht zusammenhängende Teilgraphen sind keine Lösungen des Problems *A-Baum-Baum*. Alle hier betrachteten Teilgraphen sollen aber zulässige Lösungen sein. Wird also ein Knoten in den Teilbaum $\tilde{T}$ aufgenommen, so müssen auch alle Knoten und Kanten aufgenommen werden, die nötig sind, um den entstehenden Baum minimal zusammenhängend zu machen. (Das heißt ohne einen dieser Knoten oder Kanten wäre der Baum nicht mehr zusammenhängend.) Wird die Aufnahme eines Knotens v dagegen ausgeschlossen, so ist der vorher betrachtete Graph ohne v nicht mehr zusammenhängend, ein Teilbaum $\tilde{T}$ kann nur noch in einer der beiden Zusammenhangskomponenten liegen. Das Teilproblem kann also in einem reduzierten Graph betrachtet werden.
Wegen dieser Überlegungen wird im Folgenden ein Teilproblem $\tilde{P}$ durch Angabe des schon aufgenommen Teilbaumes $\tilde{T}$ und des durch Wegnahme von Knoten reduzierten Graphen $\tilde{G}$ charakterisiert.

Definition 109 Teilproblem
Zu gegebener Instanz (T, OD, α, l_0) des Optimierungsproblems A-Baum-Baum wird das Teilproblem $\tilde{P} = (\tilde{T}, \tilde{G})$ mit Bäumen $\tilde{T} \subset \tilde{G} \subset T$, definiert durch die Zielfunktion

$$\min_{\hat{T}} \max_{\{u,v\} \in OD} d^T_{\hat{T}}(u, v) \tag{149}$$

unter den Nebenbedingungen

$$\begin{aligned} &\hat{T} \subset \tilde{G} \; Unterbaum, \\ &\hat{T} \supset \tilde{T} \\ &und \; l(\hat{T}) \leq l_0. \end{aligned} \tag{150}$$

Seien $\tilde{P}_2 = (\tilde{T}_2, \tilde{G}_2)$ und $\tilde{P} = (\tilde{T}, \tilde{G})$ zwei Teilprobleme. $\tilde{P}_2$ ist Teilproblem von $\tilde{P}$, wenn

$$\tilde{T}_2 \supseteq \tilde{T} \;\; und \;\; \tilde{G}_2 \subseteq \tilde{G}. \tag{151}$$

Lemma 110 *Sei (T, OD, α, l_0) eine Instanz des Optimierungsproblems A-Baum-Baum. Ist T^* Optimallösung eines Teilproblems $\tilde{P}$, so ist*

$$\max_{\{u,v\} \in OD} d_{T^*}(u,v) \leq \max_{\{u,v\} \in OD} d_{\hat{T}}(u,v) \tag{152}$$

für jede zulässige Lösung $\hat{T}$ eines Teilproblems $\tilde{P}_2$ von $\tilde{P}$.
Ist für ein Teilproblem $\tilde{P} = (\tilde{T}, \tilde{G})$ $l(\tilde{T}) > l_0$, so gilt das auch für jede Lösung von $(\tilde{P})$ oder eines Unterproblems $\tilde{P}_2$.

Beweis. Ist $\hat{T}$ zulässig für $\tilde{P}_2$, dann ist

$$\begin{aligned} l(\hat{T}) &\leq l_0, \\ \hat{T} \subset \tilde{G}_2 \subset \tilde{G} \text{ und } \hat{T} &\supset \tilde{T}_2 \supset \tilde{T}, \end{aligned} \tag{153}$$

also ist $\hat{T}$ auch zulässig für $\tilde{P}$. Daraus folgt

$$\max_{\{u,v\} \in OD} d_{T^*}(u,v) \leq \max_{\{u,v\} \in OD} d_{\hat{T}}(u,v) \tag{154}$$

für jede zulässige Lösung $\hat{T}$ eines Teilproblems $\tilde{P}_2$ von $\tilde{P}$.
Die zweite Aussage folgt aus der Bedingung $\tilde{T} \subset \tilde{T}_2 \subset \hat{T}$ für jede Lösung $\hat{T}$ eines Teilproblems.
■

Ein Schritt des Branch-and-Bound-Verfahrens besteht nun aus der Auswahl eines Teilproblems $\tilde{P} = (\tilde{T}, \tilde{G})$ aus der Menge aller erstellten, aber noch nicht bearbeiteten Teilprobleme S, der Überprüfung, ob $\tilde{P}$ betrachtet werden muss, oder ausgeschlossen werden kann und der Lösung von $\tilde{P}$ oder der Aufteilung von $\tilde{P}$ in zwei neue Teilprobleme. Das Verfahren kann also wie folgt durchgeführt werden:

Algorithmus 12 Branch-and-Bound 1 für A-Baum-Baum

Eingabe: Instanz (T, OD, α, l_0) des Optimierungsproblems *A-Baum-Baum*; zulässige Lösung $\tilde{T}$ (zum Beispiel $\tilde{T} = \emptyset$).

Ausgabe: Optimallösung T^* von (T, OD, α, l_0) mit optimalem Zielfunktionswert d^*.

1: $T^* = \tilde{T}$, $d^* = \max_{\{u,v\} \in OD} d_{\tilde{T}}(u, v)$.
2: $S = \{(\emptyset, T)\}$.
3: **while** $S \neq \emptyset$ **do**
4: Wähle Teilproblem $(x, y) \in S$, entferne (x, y) aus S.
5: **if** $x = y$ **then**
6: Berechne $\tilde{d}_x = \max_{\{u,v\} \in OD} d_x(u, v)$.
7: **if** $\tilde{d}_x < d^*$ **then**
8: $T^* = x$, $d^* = \tilde{d}^*$.
9: **end if**
10: **else if** $x = \emptyset$ **then**
11: Lege einen Ausgangsknoten v fest.
12: $S = S \cup \{(v, y), (\emptyset, y_1^v), (\emptyset, y_2^v)\}$, wobei y_1^v und y_2^v die Zusammenhangskomponenten von $y \setminus v$ darstellen.
13: **else**
14: Bestimme hinzuzufügenden Knoten v. Bezeichne den durch Anfügen des kürzesten Weges zwischen x und v entstehenden Teilbaum als x_{neu}.
15: $S = S \cup \{(x, y_1^v)\}$ mit y_1^v der Zusammenhangskomponente von $y \setminus v$, die x enthält.
16: **if** $l(x_{neu}) \leq l_0$ **then**
17: $S = S \cup \{(x_{neu}, y)\}$.
18: **else**
19: Berechne $\tilde{d}_x = \max_{\{u,v\} \in OD} d_x(u, v)$.
20: **if** $\tilde{d}_x < d^*$ **then**
21: $T^* = \tilde{T}$ und $d^* = \tilde{d}_x$.
22: **end if**
23: **end if**
24: **end if**
25: Gehe zu Schritt 3.
26: **end while**
27: Gebe T^* und d^* aus.

Satz 111 *Der Algorithmus „Branch-and-Bound 1 für A-Baum-Baum" findet die optimale Lösung und den optimalen Zielfunktionswert einer Instanz (T, OD, α, l_0) des Optimierungsproblems A-Baum-Baum.*

Beweis. Das anfänglich in Schritt 2 betrachtete Teilproblem ist das durch (T, OD, α, l_0) gegebene Problem selbst. Im Allgemeinen wird das betrachtete Teilproblem (x, y) in jedem Schritt anhand eines ausgewählten Knotens, wie in Definition 109 beschrieben, in Teilprobleme geteilt. Die Korrektheit der Wahl von x_{neu}, y_1^v und gegebenenfalls y_2^v folgt dabei ebenfalls aus den Überlegungen auf Seite 91-92.
Es gilt für jedes neu aus $P = (x, y)$ erstellte Teilproblem $P_{neu} = (x_{neu}, y_{neu})$ nach Lemma 110 $x_{neu} \supseteq x$ und somit nach Lemma 12:

$$\max_{\{u,v\} \in OD} d_{x_{neu}}(u, v) \leq \max_{\{u,v\} \in OD} d_x(u, v). \tag{155}$$

Ist x_{neu} zulässig, kann die alte Lösung x also verworfen werden. Diese Zulässigkeit wird in Schritt 16 überprüft. Ist x_{neu} nicht zulässig, so ist nach Lemma 110 auch keines der Teilprobleme von (x_{neu}, y_{neu}) zulässig, in diesem Fall muss also keines dieser Teilprobleme mehr betrachtet werden. Gilt $x = y$, wird der Verfahrensschritt in Schritt 5 abgebrochen, da x die einzige Lösung und damit Optimallösung des Teilproblems (x, y) ist. Die Zielfunktionswerte aller Lösungen, die nicht verworfen werden, werden in Schritt 6-9 oder 19-22 überprüft, der beste wird mit dazugehöriger Lösung in Schritt 27 ausgegeben. ■

Lemma 112 *Jeder Schritt des Verfahrens „Branch-and-Bound 1 für A-Baum-Baum" ist in $O(n^3)$.*

Beweis. Ohne Beschränkung der Allgemeinheit sei T rekursiv dargestellt (Lemma 4 und Bemerkung 6). Die Berechnung von $\max_{\{u,v\} \in OD} d_x(u, v)$ für einen Unterbaum $x \subset T$, wie sie in Schritt 6 und 19 vorkommt, ist in $O(n^3)$. Die Bestimmung von x_{neu} in Schritt 14 (vergleiche Beweis von Satz 80) und von y_i^v in Schritt 12 und 15 ist in $O(n)$, da T rekursiv dargestellt ist. Auch die übrigen Schritte sind in $O(n)$. ■

Die Integration der exakten und heuristischen Erkenntnisse aus Kapitel 6.1-6.5 zur Lösung von *A-Baum-Baum* kann dazu verwendet werden, um in Spezialfällen schnell exakte Lösungen zu erhalten, Lösungen frühzeitig auszuschließen und die als nächstes zu betrachtenden Teilprobleme möglichst gut zu wählen. Wie dies geschehen kann, soll im Folgenden dargestellt werden. Außerdem werden hier einige zusätzliche Berechnungsmöglichkeiten für untere Schranken gegeben. Im Anschluss wird ein Branch-and-Bound-Algorithmus zur Lösung von *A-Baum-Baum* unter

Berücksichtigung all dieser Kriterien aufgestellt. Zur besseren Übersichtlichkeit wird schon bei den folgenden Erläuterungen angegeben, in welchen Schritten des nachfolgenden Algorithmus die Erkenntnisse angewendet werden.

Unter der Bedingung, dass ein in der Lösung enthaltener Unterbaum T' gegeben ist und für die kritischen OD-Paare $\{u_i, v_i\} \in OD_{krit}$ paarweise gilt, dass $\left(E_{(u_j,v_j)} \cap E_{(u_k,v_k)}\right) \setminus E_{T'} = \emptyset$, ist aus Kapitel 6.3 schon ein Verfahren zur exakten Lösung von *A-Baum-Baum* bekannt. Ist es möglich, für das betrachtete Problem $\tilde{P} = (\tilde{T}, \tilde{G})$ eine Untermenge $OD_{krit} \subset OD$ zu finden, so dass

$$\begin{aligned} & d_{\tilde{T}}(\tilde{u}, \tilde{v}) < d_{\tilde{T}}(u_j, v_j) && \forall\ \{u_j, v_j\} \in OD_{krit},\ \{\tilde{u}, \tilde{v}\} \in OD \setminus OD_{krit} \\ \textit{und}\quad & \left(E_{(u_j,v_j)} \cap E_{(u_k,v_k)}\right) \setminus E_{\tilde{T}} = \emptyset && \forall\ \{u_j, v_j\}, \{u_k, v_k\} \in OD_{krit}, \end{aligned} \tag{156}$$

so lässt sich dieses Verfahren unter der Modifikation, dass nur Kanten aus $\tilde{G}$ aufgenommen werden können, hier verwenden, um obere Schranken und eventuell eine Optimallösung des Teilproblems und damit eine scharfe untere Schranke für alle Lösungen $\hat{P} = (\hat{T}, \hat{G})$ mit $\hat{T} \supset \tilde{T}$ und $\hat{T} \subset \hat{G}$ zu finden. Dieses wird in Algorithmus „Branch-and-Bound 2 für A-Baum-Baum“ in Schritt 17 ausgeführt.
Wird auf diese Weise keine exakte Lösung für das Teilproblem $\tilde{P}$ gefunden, so muss $\tilde{P}$ durch Auswahl eines Knotens $v \in \tilde{G} \setminus \tilde{T}$ weiter unterteilt werden. Um schnell möglichst gute Lösungen und damit obere Schranken für den optimalen Zielfunktionswert zu finden, bietet es sich an, bei der Auswahl dieses Knotens in Schritt 29 analog zur „Heuristik 1“ zur Konstruktion einer Lösung von *A-Baum-Baum* in Kapitel 6.5 vorzugehen, also ein maximales OD-Paar zu wählen, Zusammenhangskanten hinzuzufügen und mit einer der in Kapitel 6.5 erwähnten Auswahlregeln eine hinzuzufügende Kante, und damit den Knoten v als Endpunkt dieser Kante zu wählen. Die beiden neu entstehenden Teilprobleme $\hat{P}_1$ und $\hat{P}_2$ sind dann gegeben durch $\hat{P}_1 = (\hat{T}_1, \hat{G}_1)$ mit dem neu entstandenen Baum als $\hat{T}_1$ und $\hat{G}_1 = \tilde{G}$ und $\hat{P}_2 = (\hat{T}_2, \hat{G}_2)$ mit $\hat{T}_2 = \tilde{T}$ und der Zusammenhangskomponente von $\tilde{G} \setminus v$, in der $\tilde{T} = \hat{T}_2$ liegt, als $\hat{G}_2$.
Ist $\tilde{T} = \emptyset$, kann in Schritt 26 mit Hilfe einer der Heuristiken aus Kapitel 6.5 ein Startknoten bestimmt werden.
Ein Teilproblem muss erst dann nicht mehr betrachtet werden, kann also mit all seinen Unterproblemen aus S entfernt werden, wenn seine Optimallösung durch Erreichen der unteren Schranke des Teilproblems in Schritt 31, Algorithmus „A-Baum-Baum 1“ oder „A-Baum-Baum 2“ in Schritt 17 oder Ausprobieren aller Lösungen (durch Unterteilung in Teilprobleme, die nur

noch eine Lösung zulassen) bekannt ist oder es ausgeschlossen wurde.
Unter anderem kann Algorithmus „Konstruktion Unterbaum“ verwendet werden, um Mengen von Lösungen auszuschließen. Wurde im Laufe des Verfahrens eine zulässige Lösungen T^* mit Zielfunktionswert $d^* = \max_{\{u,v\} \in OD} d_{T^*}(u,v)$ gefunden, so konstruiert dieser Algorithmus mit $d_0 = d^* - \epsilon$ für genügend kleines ϵ nach Satz 80 einen Unterbaum T', der in jeder Lösung, die besser als T^* ist, enthalten sein muss. Gilt $T' \not\subset \tilde{G}$ für ein Teilproblem $\tilde{P}$, so kann also keine bessere Lösung in $\tilde{G}$ gefunden werden. In diesem Fall können alle Teilprobleme von $\tilde{P}$ nach Lemma 110 verworfen werden. Ist dagegen $T' \subset \tilde{G}$, so kann $\tilde{T}$ durch $\tilde{T} \cup T'$ ersetzt werden.
Wenn Algorithmus „Konstruktion Unterbaum“ $T' = \emptyset$ ausgibt, können nach Bemerkung 83 die Teilprobleme ausgeschlossen werden, für die $\tilde{G} \cap \bigcap_{\{u,v\} \in OD:\ d(u,v) > d^* - \epsilon} V_{(u,v)}$ leer ist. Ist in $\tilde{G} \cap \bigcap_{\{u,v\} \in OD:\ d(u,v) > d^* - \epsilon} V_{(u,v)}$ genau ein Knoten enthalten, so muss dieser in $\tilde{P}$ aufgenommen werden. Diese Überlegungen werden in Schritt 7 für die in Schritt 13, 20, 33, und 40 berechneten Unterbäume T' umgesetzt.
Weitere Gruppen von Lösungen können durch das Berechnen von Schranken für den Zielfunktionswert ausgeschlossen werden. Sei $d_s^{\tilde{P}}$ eine untere Schranke für Lösungen des Teilproblems $\tilde{P}$. Sei d^* die beste bisher gefundene Lösung eines Teilproblems und somit eine obere Schranke. Ist $d_s^{\tilde{P}} \geq d^*$, können alle Lösungen von $\tilde{P}$ (und damit auch seiner Teilprobleme) verworfen werden, da für keines der Teilprobleme der Zielfunktionswert der Lösung besser als der beste bisher gefundene Zielfunktionswert sein kann. Diese Überprüfung geschieht in Schritt 6.
Anfänglich kann d^* auf $d_1 = \max_{\{u,v\} \in OD} d(u,v)$ gesetzt werden. Für das erste Problem $\tilde{P} = (\emptyset, T)$ setze man $d_s^{\tilde{P}}$ auf

$$d_s^{\tilde{P}} = \begin{cases} \alpha d_1 & \text{falls } d_1 \leq l_0 \\ \alpha l_0 + (d_1 - l_0) & \text{falls } d_1 > l_0 \end{cases} \tag{157}$$

(nach Lemma 92).
Sowohl Algorithmus „A-Baum-Baum 1“ als auch Algorithmus „A-Baum-Baum 2“ können zum Finden oberer Schranken herangezogen werden.
Die betrachteten unteren Schranken sind hier immer bezogen auf ein einzelnes Teilproblem, während die oberen Schranken sowohl teilproblemspezifisch (zur Auswahl des nächsten zu betrachtenden Problems, siehe Seite 98) als auch global gelten.
Zusätzliche untere Schranken erhält man aus folgendem Lemma:

Lemma 113 *Sei (T, OD, α, l_0) eine Instanz des Optimierungsproblems A-Baum-Baum. Sei $\tilde{P} = (\tilde{T}, \tilde{G})$ ein Teilproblem. Dann gilt für den Zielfunktionswert jeder zulässigen Lösung des*

Teilproblems:

$$\tilde{d} \geq \max_{\{u,v\} \in OD} d_{\tilde{G}}(u,v) \tag{158}$$

und

$$\tilde{d} \geq \max_{\{u,v\} \in OD} d_{\tilde{T}}(u,v) - (l_0 - l(\tilde{T}))(1-\alpha). \tag{159}$$

Beweis. Jede zulässige Lösung $\hat{T}$ von $\tilde{P}$ ist nach Definition 109 Unterbaum von $\tilde{G}$. Nach Lemma 12 folgt Aussage 158.
Nun zu Aussage 159. Angenommen es gibt einen Baum $\hat{T}$ mit $\tilde{T} \subset \hat{T}$ und

$$\max_{\{u,v\} \in OD} d_{\hat{T}}(u,v) < \max_{\{u,v\} \in OD} d_{\tilde{T}}(u,v) - (l_0 - l(\tilde{T}))(1-\alpha). \tag{160}$$

Dann müsste $\hat{T}$ die Distanz des bezüglich $\tilde{T}$ maximalen OD-Paares um mehr als $(l_0 - l(\tilde{T}))(1-\alpha)$ verringern, es müssten also Kanten der Länge größer als $(l_0 - l(\tilde{T}))$ aufgenommen werden. Dann ist aber

$$l(\hat{T}) > l(\tilde{T}) + (l_0 - l(\tilde{T})) = l_0, \tag{161}$$

also ist $\hat{T}$ nicht zulässig.

■

Da die oberen Schranken als Zielfunktionswerte schon bekannter Lösungen gefunden werden, bietet es sich an, bei der Auswahl des nächsten zu betrachtenden Teilproblems aus der Auswahlmenge S der bereits erstellten Teilprobleme in Schritt 4 das Teilproblem mit der niedrigsten oberen Schranke auszuwählen. In diesem Fall können durch die dadurch gegebene bessere obere Schranke wiederum hoffentlich Teilprobleme ausgeschlossen werden.

Unter Beachtung obiger Ausführungen kann der Branch-and-Bound Algorithmus folgendermaßen beschrieben werden:

Algorithmus 13 Branch-and-Bound 2 für A-Baum-Baum

Eingabe: Instanz (T, OD, α, l_0) des Optimierungsproblems *A-Baum-Baum*; zulässige Lösung $\tilde{T}$ (zum Beispiel $\tilde{T} = \emptyset$).

Ausgabe: Optimallösung T^* von (T, OD, α, l_0) mit optimalem Zielfunktionswert d^*.

1: $T^* = \tilde{T}$, $d^* = \max_{\{u,v\} \in OD} d_{\tilde{T}}(u, v)$.
2: $S = \{(\emptyset, T)\}$, $T' = \emptyset$.
3: **while** $S \neq \emptyset$ **do**
4: Wähle Teilproblem $(x, y) \in S$, entferne (x, y) aus S.
5: Bestimme untere Schranke $d_s^{(x,y)}$ für (x, y).
6: **if** $d_s^{(x,y)} < d^*$ **then**
7: Führe die beschriebenen Überprüfungen und Ergänzungen mit Hilfe von Algorithmus „Konstruktion Unterbaum“ für (x, y) und den in der optimalen Lösung enthaltenen Unterbaum T' durch.
8: **if** Alle Überprüfungen waren positiv und $l(x) \leq l_0$. **then**
9: **if** $x = y$ **then**
10: $\tilde{d} = \max_{\{u,v\} \in OD} d_x(u, v)$.
11: **if** $\tilde{d} < d^*$ **then**
12: $T^* = \tilde{T}$ und $d^* = \tilde{d}$.
13: Führe Algorithmus „Konstruktion Unterbaum“ mit $d_0 = d^* - \epsilon$ durch. Sei T'' die Ausgabe von Algorithmus „Konstruktion Unterbaum“. Setze $T' = T''$.
14: **end if**
15: Gehe zu Schritt 3.
16: **end if**
17: Berechne, falls möglich, mit Hilfe von Algorithmus „A-Baum-Baum 1“ oder „A-Baum-Baum 2“ die Optimallösung von (x, y) oder obere Schranken und dazugehörige Lösungen. Sei $\tilde{T}$ die beste gefundene Lösung mit Zielfunktionswert $\tilde{d}$.
18: **if** $\tilde{d} < d^*$ **then**
19: $T^* = \tilde{T}$ und $d^* = \tilde{d}$.
20: Führe Algorithmus „Konstruktion Unterbaum“ mit $d_0 = d^* - \epsilon$ durch. Sei T'' die Ausgabe von Algorithmus „Konstruktion Unterbaum“. Setze $T' = T''$.
21: **if** $\tilde{T}$ optimal für (x, y) **then**
22: Gehe zu Schritt 3.
23: **end if**
24: **end if**

25: **if** $x = \emptyset$ **then**

26: Lege einen Knoten v fest.

27: $S = S \cup \{(v, y), (\emptyset, y_1^v), (\emptyset, y_2^v)\}$, wobei y_1^v und y_2^v die Zusammenhangskomponenten von $y \setminus v$ darstellen.

28: **else**

29: Bestimme hinzuzufügenden Knoten v. Bezeichne den durch Hinzufügen des kürzesten Weges zwischen x und v entstehenden Teilbaum als x_{neu}.

30: Berechne $\tilde{d} = \max_{\{u,v\} \in OD} d_{x_{neu}}(u, v)$.

31: **if** $\tilde{d} = d_s^{(x,y)}$, $\tilde{d} < d^*$ und $l(x_{neu}) \leq l_0$ **then**

32: $T^* = x_{neu}$, $d^* = d_v^{(x,y)}$.

33: Führe Algorithmus „Konstruktion Unterbaum“ mit $d_0 = d^* - \epsilon$ durch. Sei T'' die Ausgabe von Algorithmus „Konstruktion Unterbaum“. Setze $T' = T''$.

34: Gehe zu Schritt 3.

35: **end if**

36: $S = S \cup \{(x, y_1^v)\}$ mit y_1^v der Zusammenhangskomponente von $y \setminus v$, die x enthält.

37: **if** $l(x_{neu}) \leq l_0$ **then**

38: **if** $\tilde{d} < d^*$ **then**

39: $T^* = x$, $d^* = \tilde{d}$.

40: Führe Algorithmus „Konstruktion Unterbaum“ mit $d_0 = d^* - \epsilon$ durch. Sei T'' die Ausgabe von Algorithmus „Konstruktion Unterbaum“. Setze $T' = T''$.

41: **end if**

42: $S = S \cup \{(x_{neu}, y)\}$.

43: **end if**

44: **end if**

45: **end if**

46: **end if**

47: **end while**

48: Gebe T^* und d^* aus.

Satz 114 *Algorithmus „Branch-and-Bound 2 für A-Baum-Baum“ ist korrekt.*

Beweis. Die Einbindung der heuristischen Auswahlregeln bei der Wahl des hinzuzufügenden Knotens oder des zu betrachtenden Teilproblems hat keinen Einfluss auf die Korrektheit des Verfahrens. Zu zeigen ist nun Zulässigkeit und Optimalität der ausgegebenen Lösung T^*.

Behauptung 115 *Sei (T, OD, α, l_0) eine Instanz von A-Baum-Baum. Für alle in S gespeicherten Teilprobleme (x, y) ist x zulässig für (T, OD, α, l_0).*

Anfänglich ist $S = \{(\emptyset, T')\}$ und $l(\emptyset) = 0 \leq l_0$. Neue Lösungen werden in Schritt 36 und 42 hinzugefügt. Die Zulässigkeit von Graph x aus dem Teilproblem (x, y_1^v) in Schritt 36 wurde schon festgestellt, da $(x, y) \in S$. $l(x_{neu}) \leq l_0$ für den Graph x_{neu} aus Schritt 42 wurde in Schritt 37 überprüft. x_{neu} ist ein Baum, da x ein Baum ist und die Konstruktion in Schritt 29 zusammenhängend erfolgt.

Behauptung 116 *Alle in T^* gespeicherten Lösungen sind zulässig.*

Anfänglich ist $T^* = \tilde{T}$ für die gegebene zulässige Lösung $\tilde{T}$. Der in Schritt 12 als T^* gespeicherte Graph ist zusammenhängend, also ein Baum, nach Konstruktion in Schritt 7. Seine Länge wird in Schritt 8 überprüft. Die Zulässigkeit des in Schritt 19 betrachteten Baumes ergibt sich aus der Zulässigkeit der Lösungen von Algorithmus „A-Baum-Baum 1“ und „A-Baum-Baum 2“ (Lemma 93 und 97). Der in Schritt 29 konstruierte Graph x_{neu} ist ein Baum, da die Konstruktion zusammenhängend erfolgt. Die Zulässigkeit der Länge der in Schritt 32 als T^* gespeicherten Lösung wird in Schritt 31 überprüft, für die in Schritt 39 gespeicherte geschieht dies in Schritt 37.

Behauptung 117 *Das in Algorithmus „Branch-and-Bound 2“ vorgenommene Ausschließen von Lösungen ist korrekt.*

Nach Lemma 110 können alle Teilprobleme von (x, y) ausgeschlossen werden, falls $l(x) > l_0$.
Man betrachte nun Schritt 6. Ist für ein Teilproblem (x, y) die untere Schranke für jede Lösung des Teilproblems größer oder gleich dem Zielfunktionswert einer bisher gefundenen Lösung, so kann dieses Teilproblem mit allen seinen Teilproblemen nach Lemma 110 verworfen werden. Deswegen muss das Teilproblem nur weiter betrachtet werden, wenn $d_s^{(x,y)} < d^*$.
Ist in Schritt 9 $x = y$, so ist x die einzige Lösung des Teilproblems (x, y). Nach eventuellem Ersetzen der besten bisher gefundenen Lösung muss (x, y) deshalb nicht weiter betrachtet werden.

In Schritt 13, 20, 33 und 40 wird, wenn eine bisher beste Lösung T^* mit Zielfunktionswert d^* gefunden wurde, ein Unterbaum T', der in jeder Lösung mit Zielfunktionswert $\leq d^* - \epsilon$, also in jeder besseren Lösung als T^*, enthalten sein muss, konstruiert. Die Überprüfungen in Schritt 7 werden anhand dieses Unterbaumes durchgeführt. Wie auf Seite 97 erklärt, sind diese Schritte korrekt.

Die durch die Anwendung von Algorithmus „A-Baum-Baum 1“ und „A-Baum-Baum 2“ in Schritt 17 entstehenden Bäume sind zulässig nach Lemma 93 und Bemerkung 102. Sind diese außerdem optimal für das Teilproblem, so müssen keine anderen Lösungen des Teilproblems mehr untersucht werden.

Wird durch eine zulässige Lösung des Teilproblems die untere Schranke des Teilproblems erreicht, so müssen alle weiteren Lösungen des Teilproblems nicht mehr betrachtet werden, da diese Lösung optimal für das Teilproblem ist. Nach der Überprüfung und eventuellen Ersetzung in Schritt 31 und 32 kann das Teilproblem also verworfen werden.

$\Rightarrow$ Die Regeln zum Ausschließen von Teilproblemen sind korrekt.

Von den zulässigen Teilproblemen werden also nur die ausgeschlossen, deren Zielfunktionswert nicht besser als der der am Ende ausgegebenen Lösung T^* ist. Durch Überprüfung aller weiteren Teilprobleme und Auswahl der besten Lösung findet Algorithmus „Branch-and-Bound 2“ die Optimallösung. ■

7 Zusammenfassung

In dieser Arbeit wurde eine Untersuchung der Komplexität der durch die Struktur von gegebenem und gesuchtem Graph definierten Teilprobleme zweier Optimierungsprobleme vorgenommen. Eines der Teilprobleme wurde ausgiebig betrachtet, exakte und heuristische Lösungsalgorithmen wurden angegeben.

Die Teilprobleme von Problem A, der Minimierung der maximalen Distanz zwischen den Knoten eines OD-Paares, lassen sich einteilen in polynomiell lösbare und NP- vollständige (also unter der Annahme P$\neq$ NP nicht polynomiell lösbare) Probleme. Tatsächlich erweisen sich nur genau die Probleme als in Polynomialzeit lösbar, bei denen die Anzahl der enthaltenen Teilgraphen des geforderten Typs schon polynomiell in der Eingabelänge begrenzt ist, so dass sich die Lösung auch durch naives Durchprobieren in Polynomialzeit finden ließe. Die aufgeführten Lösungsverfahren machen sich diese Tatsache zunutze, beschleunigen den Vorgang durch Definition maximaler Pfade und maximaler OD-Paare jedoch erheblich.

Durch Reduktion eines Spezialfalls des Entscheidungsproblems HITTING SET lässt sich schon für das Entscheidungsproblem *A-Baum-Baum* starke NP-Vollständigkeit zeigen, obwohl es aufgrund der durch die Baumstruktur gegebenen eindeutigen Wege zwischen Knotenpaaren auf den ersten Blick gut zugänglich erscheint. Daraus folgt sofort auch die starke NP-Vollständigkeit von *A-zGraph-Baum* und *A-zGraph-zGraph*. Die starke NP-Vollständigkeit von *A-zGraph-Pfad* und *A-zGraph-Kreis* ergibt sich durch Reduktion der bekannten Entscheidungsprobleme HAMILTONIAN PATH und HAMILTONIAN CIRCUIT. Wird die Bedingung des Zusammenhangs fallen gelassen, erweist sich sogar im Pfad die Suche nach einem optimalen Teilgraph als NP-vollständig. Der Beweis erfolgt allerdings durch Reduktion des pseudopolynomiell lösbaren Entscheidungsproblems SUBSET SUM, so dass keine starke NP-Vollständigkeit gezeigt werden kann. Andererseits erfolgt die Reduktion auf eine Instanz mit nur einem OD-Paar, so dass auf diesem Wege nicht auf pseudopolynomielle Lösbarkeit des Problems geschlossen werden kann. Die NP-Vollständigkeit lässt sich einfach auf das Problem *A-Kreis-Pfad* übertragen, während *A-Baum-Graph* und *A-zGraph-Graph* sogar stark NP-vollständig sind.

Außerdem wurde Problem B, die Maximierung der gesamten Reisezeitersparnis, betrachtet. Auch hier stellen sich wie in Problem A allein die Teilprobleme mit nur polynomiell beschränkter Anzahl von hinsichtlich der Struktur zulässigen möglichen Lösungen als in Polynomialzeit lösbar heraus. Bei der Betrachtung von Teilproblemen mit Bäumen als unterliegenden Graphen erweist sich die durch die Existenz eindeutiger Wege mögliche Verteilung der Nachfrage zwischen OD-Paaren auf die Kanten selbst als hilfreich. Für Problem *B-Pfad-Graph* und *B-Baum-Graph*

lässt sich auf diese Weise die Äquivalenz zu Rucksackproblemen zeigen. Sie sind somit zwar NP-vollständig, aber pseudopolynomiell lösbar. Ähnliches gilt für das Problem *B-Baum-Baum*, dessen Komplexität ebenfalls von KNAPSACK herrührt und das durch Rückführung auf ein spezielles, PARTIALLY ORDERED KNAPSACK genanntes Rucksackproblem in Pseudopolynomialzeit gelöst werden kann. Für das Problem *B-Kreis-Graph* kann NP-Vollständigkeit zwar aus der NP-Vollständigkeit von *B-Pfad-Graph* gefolgert werden, die Nachfrage kann aber nicht mehr auf die Kanten verteilt werden. Somit ist es zumindest auf diese Weise nicht möglich, hier Äquivalenz zu einem Rucksackproblem zu zeigen. Wie auch bei Problem A erweisen sich alle Probleme in allgemeinen zusammenhängenden Graphen als stark NP-vollständig. Der Beweis erfolgt für *B-zGraph-Pfad* und *B-zGraph-Kreis* wieder durch Reduktion von HAMILTONIAN PATH und HAMILTONIAN CIRCUIT. Für *B-zGraph-Baum*, *B-zGraph-zGraph* und sogar *B-zGraph-Graph* erweist sich die Reduktion des Entscheidungsproblems STEINER TREE als geeignet.

Im Gegensatz zu den ersten beiden Teilen dieser Arbeit, in denen Komplexitätsuntersuchungen im Vordergrund stehen, widmet sich das letzte Kapitel Lösungsansätzen für das Optimierungsproblem *A-Baum-Baum*. Da das zugehörige Entscheidungsproblem schon stark NP-vollständig ist, wird nicht der Versuch unternommen, das Problem im Allgemeinen in Polynomial- oder Pseudopolynomialzeit zu lösen. Die Eindeutigkeit der Wege zwischen Knotenpaaren aufgrund der unterliegenden Baumstruktur und die Forderung des Zusammenhangs für den gesuchten Graph bilden jedoch einen guten Ansatzpunkt für Lösungsverfahren. Bei geeigneter Struktur der OD-Paare lassen sich in $O(n^3)$ Unterbäume konstruieren, die in jeder guten Lösung des Optimierungsproblems enthalten sein müssen. Gilt für einen festen Wert d_0 sogar, dass die OD-Paare mit größerem Abstand $d > d_0$ außerhalb des schon konstruierten Unterbaumes keine gemeinsamen Kanten haben, so lässt sich das Entscheidungsproblem *A-Baum-Baum* mit Maximalabstand d_0 durch Anfügen von Kanten an den zuvor konstruierten Unterbaum in $O(n^3)$ lösen. Auf dieser Basis können Verfahren angegeben werden, die durch sukzessive Lösung von Entscheidungsproblemen im besten Fall die Optimallösung und anderenfalls zumindest Schranken mit zugehörigen Lösungen finden. Ist diese Bedingung nicht erfüllt, kann das Prinzip der Konstruktion eines Unterbaumes und des anschließenden Anfügens von Kanten auch heuristisch umgesetzt werden. Mit dem entwickelten Branch-and-Bound Algorithmus wurde ein Verfahren gefunden, das die zuvor gewonnenen Erkenntnisse und Verfahren zur exakten und heuristischen Lösung von Problem *A-Baum-Baum* und zur Berechnung von Schranken zur exakten Lösung jeder Instanz von *A-Baum-Baum* nutzt.

Im Rahmen dieser Arbeit wurden Heuristiken zur Lösung des Optimierungsproblems *A-Baum-Baum* angegeben. Neben den angestellten Überlegungen zur Anwendbarkeit und Eignung der heuristischen Regeln müssten diese zur Bestimmung der Güte der Verfahren implementiert und getestet werden. Gleiches gilt für den Branch-and-Bound-Algorithmus, bei dem eine Abwägung zwischen Nutzen und Zeitaufwand der eingeführten Auswahlregeln getroffen werden sollte. Leider war dieses im zeitlichen Rahmen dieser Arbeit nicht mehr möglich. Auch Aussagen zur Relevanz der in Kapitel 6 vorgestellten exakten Verfahren, die sich auf die tatsächliche Häufigkeit des Vorkommens der Spezialfälle stützen müssten, in denen die Verfahren anwendbar sind, konnten nicht getroffen werden. Offen bleibt ebenfalls die Frage nach der Möglichkeit einer pseudopolynomiellen Lösung für die Probleme *A-Pfad-Graph*, *A-Kreis-Graph* sowie *B-Kreis-Graph*. Ferner wäre eine Konstruktion heuristischer oder exakter Verfahren, wie in Kapitel 6 für Problem *A-Baum-Baum*, zur Lösung der weiteren stark NP-schweren Optimierungsprobleme denkbar.

Literatur

[AN92] Aneja, Y. P. ; Nair, K. P. K.: Location of a Tree Shaped Facility in a Network. In: *Information Systems and Operational Research* 30 (1992), S. 319–324

[CRC85] Current, J. R. ; ReVelle, C. S. ; Cohon, J. L.: The Maximum Covering/Shortest Path Problem: A Multiobjective Network Design and Routing Formulation. In: *European Journal of Operational Research* 21 (1985), Nr. 2, S. 189–199

[CS89] Current, J. R. ; Schilling, D. A.: The Covering Salesman Problem. In: *Transportation Science* 23 (1989), Nr. 3, S. 208–213

[CS94] Current, J. R. ; Schilling, D. A.: The Median Tour and Maximal Covering Tour Problems: Formulations and Heuristics. In: *European Journal of Operational Research* 73 (1994), Nr. 1, S. 114–126

[Die00] Diestel, R.: *Graphentheorie.* Zweite Auflage. Springer, 2000

[FM03] Fernández, P. ; Marín, A.: A Heuristic Procedure for Path Location with Multisource Demand. In: *Information Systems and Operational Research* 41 (2003), Nr. 2, S. 165–177

[GDOM07] Gutierrez, G. ; Donoso, M. ; Obreque, C. ; Marianov, V.: *Minimum Cost Path Location for Maximum Traffic Capture.* 2007. – unveröffentlicht, eingereicht bei Journal of Operational Research

[GJ79] Garey, M. R. ; Johnson, D. S.: *Computers and Intractability - A Guide to the Theory of NP-Completeness.* Bell, 1979

[GLS97] Gendreau, M. ; Laporte, G. ; Semet, F.: The Covering Tour Problem. In: *Operations Research* 45 (1997), Nr. 4, S. 568–576

[GR03] George, J. W. ; ReVelle, C. S.: Bi-Objective Median Subtree Location Problems. In: *Annals of Operations Research* 122 (2003), Nr. 1–4, S. 219–232

[HSL93] Hakimi, S. L. ; Schmeichel, E. F. ; Labbé, M.: On Locating Path- or Tree-Shaped Facilities on Networks. In: *Networks* 23 (1993), Nr. 6, S. 543–555

[JN83] Johnson, D. S. ; Niemi, K. A.: On Knapsacks, Partitions and a new Dynamic Programming Technique for Trees. In: *Mathematics of Operations Research* 8 (1983), Nr. 1, S. 1–14

[KLTW96] Kim, T. U. ; Lowe, Timothy J. ; Tamir, Arie ; Ward, James E.: On the Location of a Tree-Shaped Facility. In: *Networks* 28 (1996), Nr. 3, S. 167–175

[KLWF90] Kim, T. U. ; Lowe, T. J. ; Ward, J. E. ; Francis, R. L.: A Minimum-Length Covering Subtree of a Tree. In: *Naval Research Logistics* 37 (1990), Nr. 2, S. 309–326

[KMP+08] Körner, M. ; Mesa, J. A. ; Perea, F. ; Schöbel, A. ; Scholz, D.: The Maximum Trip Covering Station Location Problem with Alternative Mode of Transportation. In: *ARRIVAL Report* (2008), Nr. 148

[KV08] Korte, B. ; Vygen, J.: *Combinatorial Optimization - Theory and Algorithms.* Vierte Auflage. Springer, 2008

[MCH81] Mitchell Hedetniemi, S. ; Cockayne, E. J. ; Hedetniemi, S. T.: Linear Algorithms for Finding the Jordan Center and Path Center of a Tree. In: *Transportation Science* 15 (1981), Nr. 2, S. 98–114

[Min85] Minieka, E.: The Optimal Location of a Path or Tree in a Tree Network. In: *Networks* 15 (1985), Nr. 3, S. 309–321

[Min86] Minoux, M.: *Mathematical Programming - Theory and Algorithms.* Wiley, 1986

[MP83] Minieka, E. ; Patel, N. H.: On Finding the Core of a Tree with a Specified Length. In: *Journal of Algorithms* 4 (1983), Nr. 4, S. 345–352

[Ric90] Richey, M. B.: Optimal Location of a Path or Tree on a Network with Cycles. In: *Networks* 20 (1990), Nr. 4, S. 391–407

[Sla82] Slater, P. J.: Locating Central Paths in a Graph. In: *Transportation Science* 16 (1982), Nr. 1, S. 1–18

[YY06] Yan, W. ; Yeh, Y. N.: Enumeration of Subtrees of Trees. In: *Theoretical Computer Science* 369 (2006), Nr. 1–3, S. 256–268