Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Verzweigen und Beschränken

Branch and Bound

Sarah Bertulat, Jens Weber

Hochschule Zittau/Görlitz

27. November 2014

Gliederung

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Allgemeines

Funktionsprinzip

Rucksackproblem

Das Rundreiseproblem

Fragen?

Allgemeines

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Systematische Suche

- Lösungsbäume
  - Tiefensuche
  - Breitensuche

$\Rightarrow$ Problem des exponentiellen Aufwands

Lösung

Lösung ist das branch and bound

- Teilbäume bestimmt, die nicht als Lösung in frage kommen

  ↳ werden vorläufig entfernt

Allgemeines

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Vorteile von Branch and Bound

- man kann eine Aufwandsverbesserung bekommen
- worst case: der Aufwand stimmt mit dem der Systematischen Suche überein, da man keinen Zweig des Baumes findet den man entfernen kann

Funktionsprinzip

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Allgemein

- Für jeden Knoten wird eine Wertschranke berechnet
- Wertschranken gibt es in 2 Arten (Maximierungsproblem, Minimierungsproblem]

Branch and Bound

- von der Wurzel an wird je ein Knoten expandiert
- der Knoten mit dem besten Bound wird dann weiter expandiert
- wird der expandierte Knoten schlechter als ein noch nicht expandierter Knoten wird dieser aus der Warteschlange geholt

Funktionsprinzip

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Aufwand

- genaue Schrankenberechnung $\Rightarrow$ hoher Aufwand, genaue Lösung
- eine Schrankenbestimmung $\Rightarrow$ geringer Aufwand, nur Näherungslösung (reicht meist aus)

Rucksackproblem

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Rucksackproblem

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Mathematische Formulierung

$$\frac{w_i}{g_i} \geq \frac{w_{i+1}}{g_{i+1}}, \text{ für alle } 1 \leq i \leq n$$

Gesucht ist der Vektor $\vec{x} = (x_1, x_2, ..., x_i)$ mit $x_i \in \{0, 1\}$

,

$$\text{sodass } \sum_{i=1}^{n} x_i g_i \leq K \text{ sowie } \sum_{i=1}^{n} x_i w_i \longrightarrow max$$

Rucksackproblem

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Bound Berechnung

$$\sum_{i=1}^{n} x_i w_i + (K - \sum_{i=1}^{n} x_i g_i) * \frac{w_{k+1}}{g_{k+1}}$$

Beispiel

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

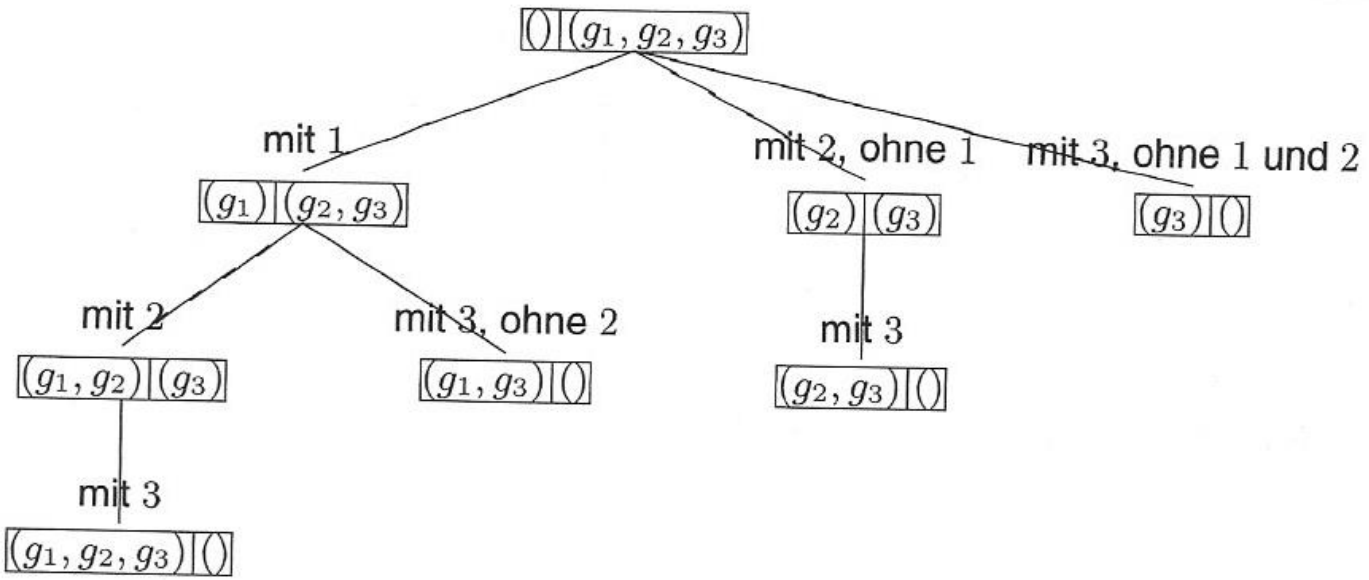
- $n = 4$ (Gegenstände
- $K = 8$ (Kapazität des Rucksacks)

i	1	2	3	4
w_i	10	5	6	3
g_i	5	3	4	2

Beispiel

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

5.3 BILD **Reduzierter Suchbaum für 0/1-Rucksackproblem**



Beispiel

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Lösung mit Bounds

- Berechnung des spezifischen Wertes (in absteigender Folge ordnen)

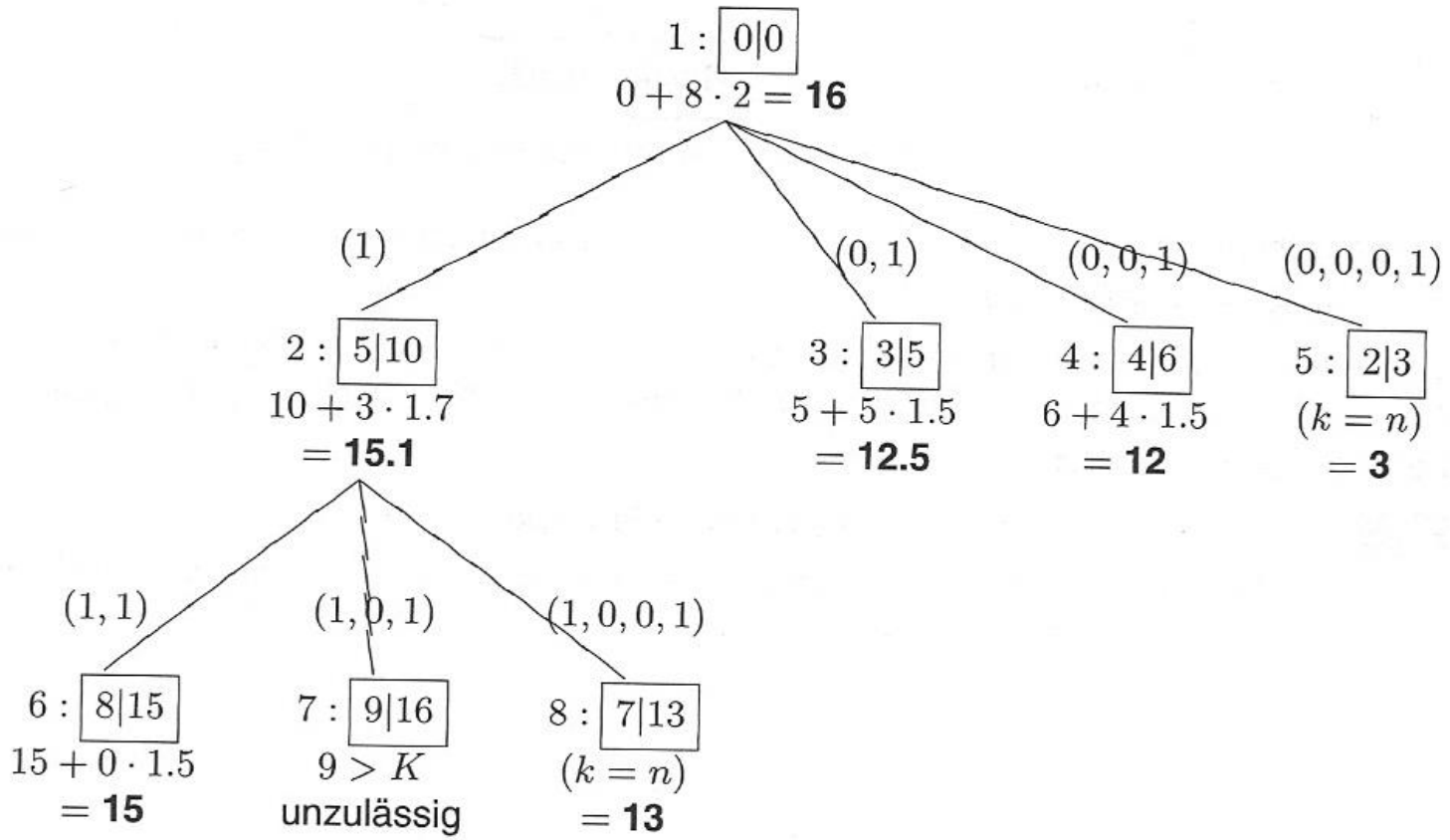
i	1	2	3	4
w_i	10	5	6	3
g_i	5	3	4	2
$\frac{w_i}{g_i}$	2	1,7	1,5	1,5

- Zuerst wird die Wurzel angegeben und dessen Bound berechnet
- danach wird der Knoten mit dem größten Bound expandiert

Beispiel

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

6.1 BILD **Lösungsbaum für Beispiel 6.1 mit Bounds für $K = 8$**



1

Anwendung

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Beispiel

Wo kann Verzweigen und Beschränken noch angewandt werden? Ein Beispiel ist das Rundreiseproblem.

Das Rundreiseproblem - Worum geht es?

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Begriff

- engl. traveling salesman problem $\rightarrow$ TSP
- auch: Problem des Vertreters

Das Rundreiseproblem - Worum geht es?

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Begriff

- engl. traveling salesman problem → TSP
- auch: Problem des Vertreters

Das Problem

- der Reisende will n Städte besuchen und am Ende wieder in der Startstadt ankommen
- dabei soll die Route mit der kürzesten Gesamtstrecke gesucht werden
  - also wo die Summe aller Einzelstrecken am kleinsten ist
- die Entfernung zwischen den Städten ist beim Erstellen der Route bekannt

Das Rundreiseproblem

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Anwendungsbeispiele

- Reiseunternehmen (Tourenplanung)
- Routenplaner
- (Post)-Boten
- Handelsvertreter
- Entwurf integrierter Schaltkreise

Das Rundreiseproblem

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Anwendungsbeispiele

- Reiseunternehmen (Tourenplanung)
- Routenplaner
- (Post)-Boten
- Handelsvertreter
- Entwurf integrierter Schaltkreise

Was ist gegeben?

- in einer Matrix werden die Entfernungen zwischen den Städten angegeben (in der Regel nicht symmetrisch)
- nicht zulässige Verbindungen (z.B. von Bautzen nach Bautzen) werden mit ∞ oder einem anderen Platzhalter markiert

Das Rundreiseproblem - Beispiel Vertreter

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Ein Vertreter ist im Kreis Görlitz unterwegs. Er soll seine Runde in Görlitz starten und noch Löbau, Zittau und Niesky besuchen. Welche Route ist für ihn die kürzeste?

	Görlitz	Löbau	Niesky	Zittau
Görlitz	∞	25	23	35
Löbau	25	∞	27	27
Niesky	22	27	∞	64
Zittau	36	27	64	∞

Das Rundreiseproblem - Intuitiver Lösungsansatz

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Das Rundreiseproblem - Intuitiver Lösungsansatz

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

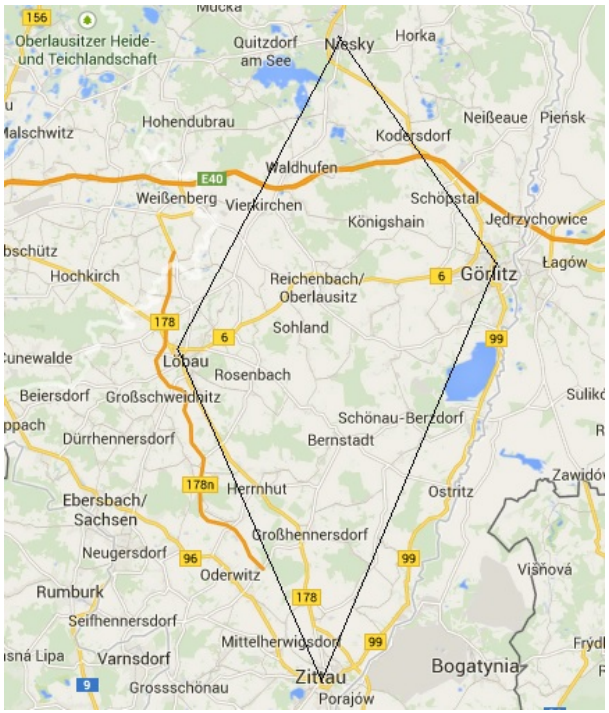
Abschätzen auf Karte

- sehr trivialer Ansatz
- nur für geringe Anzahl Reiseziele anwendbar
- sehr ungenau, wenn Straßensituation nicht bekannt ist

Beispiel Vertreter - Karte

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

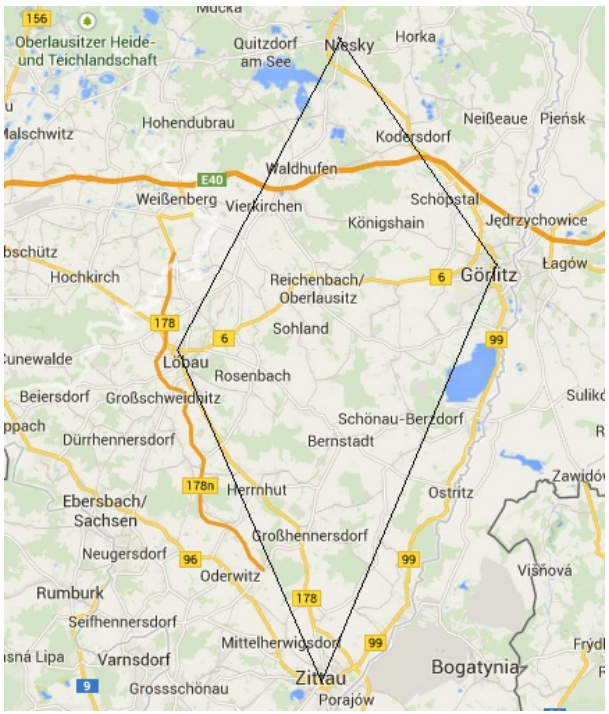
Für unseren Vertreter wäre es einfach, da die Orte auf der Karte günstig angeordnet sind.



Beispiel Vertreter - Karte

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Für unseren Vertreter wäre es einfach, da die Orte auf der Karte günstig angeordnet sind.

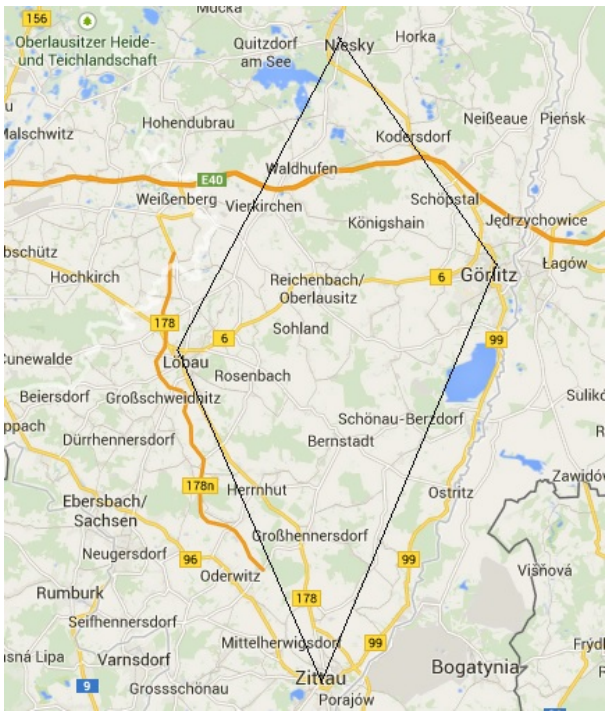


Aber was wäre, wenn es in der Luftlinie keine Wege gibt oder diese gesperrt sind? Wenn Umwege zu nehmen sind, kann die Karte falsch interpretiert werden.

Beispiel Vertreter - Karte

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Für unseren Vertreter wäre es einfach, da die Orte auf der Karte günstig angeordnet sind.



Aber was wäre, wenn es in der Luftlinie keine Wege gibt oder diese gesperrt sind? Wenn Umwege zu nehmen sind, kann die Karte falsch interpretiert werden.
$\Rightarrow$ mathematischer / algorithmischer Ansatz wäre besser

Algorithmischer Ansatz

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

intuitiver Algorithmus

- zuerst werden alle möglichen Reiserouten erzeugt
- dabei werden die Entfernungen der benutzten Einzelrouten addiert
- am Ende wird die Strecke mit der geringsten Entfernung gewählt

Algorithmischer Ansatz

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

intuitiver Algorithmus

- zuerst werden alle möglichen Reiserouten erzeugt
- dabei werden die Entfernungen der benutzten Einzelrouten addiert
- am Ende wird die Strecke mit der geringsten Entfernung gewählt

ABER!

- nur für kleine Anzahl Reiseziele realisierbar
- Grund:
  - es gibt $(n-1)!$ Kombinationsmöglichkeiten
  - $\Rightarrow$ exponentieller Zuwachs

Warum $(n-1)!$?

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

$(n-1)!$

- es gibt n Städte, in die der Vertreter reisen muss
- Beispiel 4 Städte
  - er startet in Stadt 1 und hat 3 Möglichkeiten, wo er hinreisen kann
  - in der nächsten Stadt sind es noch zwei
  - und in der nächsten Stadt hat er nur noch eine Möglichkeit
  - $\rightarrow$ d.h. er hat $3 \cdot 2 \cdot 1 = 3!$ Möglichkeiten
  - $\Rightarrow$ für $n = 4$ Städte gibt es also $(4-1)! = (n-1)!$ Möglichkeiten

Nachweis exponentieller Aufwand

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Stirling-Formel

$$n! = \sqrt{2 * \pi * n}(\frac{n}{e})^n(1 + \frac{1}{12n} + \frac{1}{288n^2} + \mathcal{O}(\frac{1}{n^3}))$$

- exponentieller Aufwand entsteht bereits für die Berechnung aller möglichen Reiserouten
- Aufwand für die Auswahl der besten Route (eigentliche Lösung) noch nicht beinhaltet
- $\Rightarrow$ auch dieser wäre nicht konstant, sondern von n abhängig (linear)

Nachweis exponentieller Aufwand

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Stirling-Formel

$$n! = \sqrt{2 * \pi * n}(\frac{n}{e})^n(1 + \frac{1}{12n} + \frac{1}{288n^2} + \mathcal{O}(\frac{1}{n^3}))$$

- exponentieller Aufwand entsteht bereits für die Berechnung aller möglichen Reiserouten
- Aufwand für die Auswahl der besten Route (eigentliche Lösung) noch nicht beinhaltet
- $\Rightarrow$ auch dieser wäre nicht konstant, sondern von n abhängig (linear)

$\Rightarrow$ auch dieser Ansatz (intuitiver Algorithmus) ist für größere Städtemengen nicht praktikabel

- n = 6 $\rightarrow$ 120 mögliche Routen
- n = 10 $\rightarrow$ 362.880 mögliche Reiserouten
- n = 20 $\rightarrow$ 19! = 121.645.100.408.832.000 (ca. 121,5 Billiarden) mögliche Reiserouten

Das Rundreiseproblem - Gerichteter Graph

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Modellierung als gerichteter Graph

- Darstellung als Modell
  - Knoten sind Städte
  - Kanten sind die möglichen Reiserouten
- Bewertung der Kanten muss nicht geometrisch korrekt sein
- Graph sollte vollständig sein (Verbindung von jeder Stadt zu jeder Stadt vorhanden)

- Zyklen (Sackgassen) sind in der Lösung erlaubt

Sonderfälle

- ungerichteter Graph (symmetrisches Reiseproblem)
- metrischer Graph (Dreiecksungleichung wird erfüllt)

Hinführen zu Branch and bound

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Um auch größere Städtemengen handhaben zu können, sollte Verzweigen und Beschränken angewandt werden!

Besserer Ansatz - Branch and bound

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Was wird benötigt?

- eine Funktion, die für bestimmte Knoten k einen Schrankenwert s_k berechnet
- der Schrankenwert soll angeben, dass durch Expansion von k kein kürzerer Weg entstehen kann, als durch s_k angezeigt ist

Besserer Ansatz - Branch and bound

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Was wird benötigt?

- eine Funktion, die für bestimmte Knoten k einen Schrankenwert s_k berechnet
- der Schrankenwert soll angeben, dass durch Expansion von k kein kürzerer Weg entstehen kann, als durch s_k angezeigt ist

$\Rightarrow$ bei der Wahl, welcher Knoten expandiert werden soll, ist hier immer der mit dem kleinsten Schrankenwert zu wählen (kürzester Weg ist gesucht)

Mögliche Vorgehensweise

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Weg zum nächsten Knoten bestimmen

- sehr einfach zu handhaben
- gibt aber keinerlei Rückschluss auf den weiteren Weg nach dem nächsten Knoten
- $\rightarrow$ deshalb nicht mal als grobes Näherungsverfahren geeignet

Richtiges Vorgehen

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Berücksichtigung aller zu k hin- UND wegführenden Kanten

- um Doppelzählungen zu vermeiden, ist die Summe zu halbieren
- es werden auch die (langen) Strecken mit berücksichtigt, die gar nicht genommen werden müssten
  - da die Stadt z.B. schon besucht worden ist
- $\rightarrow$ auch dieses Verfahren ist nur zur Bestimmung eines Näherungswertes geeignet

Richtiges Vorgehen

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Berücksichtigung aller zu k hin- UND wegführenden Kanten

- um Doppelzählungen zu vermeiden, ist die Summe zu halbieren
- es werden auch die (langen) Strecken mit berücksichtigt, die gar nicht genommen werden müssten
  - da die Stadt z.B. schon besucht worden ist
- $\rightarrow$ auch dieses Verfahren ist nur zur Bestimmung eines Näherungswertes geeignet

Ziel:

Eine Schranke finden, die relativ einfach und schnell zu berechnen ist UND aussagekräftig genug ist

- nicht die beim Algorithmus gewonnene Effizienz bei der Schrankenberechnung wieder verlieren

Richtiges Vorgehen

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Schaffen einer reduzierten Entfernungsmatrix

- in jeder Zeile wird ein Minimum gesucht und von jedem Wert abgezogen
- anschließend wird das gleiche in den Spalten der Matrix wiederholt (Nullen werden ignoriert)
- Ergebnis ist die reduzierte Entfernungsmatrix

Bildung reduzierte Entfernungsmatrix

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Schritt 1 - zeilenreduzierte Entfernungsmatrix

- in jeder Zeile das Minimum finden
- jeden Wert der Zeile um das Zeilen-Minimum reduzieren

Beispiel Vertreter:

	Görlitz	Löbau	Niesky	Zittau	
Görlitz	∞	25_2	23_0	35_{12}	23
Löbau	25_0	∞	27_2	27_2	25
Niesky	22_0	27_5	∞	64_{42}	22
Zittau	36_9	27_0	64_{37}	∞	27
					97

Bildung reduzierte Entfernungsmatrix

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Schritt 2 - reduzierte Entfernungsmatrix

- in jeder Spalte der zeilenreduzierten Matrix das Minimum finden
- jeden Wert der Spalte um das Spalten-Minimum reduzieren

Beispiel Vertreter:

	Görlitz	Löbau	Niesky	Zittau	
Görlitz	∞	2	0	10	23
Löbau	0	∞	2	0	25
Niesky	0	5	∞	40	22
Zittau	9	0	37	∞	27
	0	0	0	2	**99**

Schranke

Die gesuchte Schranke (bound-Wert) ist die Summe der Zeilen- und Spaltenminima.

Weiteres Vorgehen

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

- die Startmatrix muss immer expandiert werden
- im Folgenden wird immer nur die Matrix mit dem niedrigsten bound-Wert expandiert
- zur Auswahl stehen dabei jeweils alle Matrizen, die noch nicht expandiert wurden

Was passiert bei Expansion?

Bei Expansion von Knoten i wird die Kante (i, j) hinzugenommen.
Die Entfernungsmatrix E ist dabei folgendermaßen anzupassen:

- $E[z, j] = \infty$ für alle $z \neq i \rightarrow j$ wurde schon von i aus besucht
- $E[i, s] = \infty$ für alle $s \neq i \rightarrow$ von i darf nur zu j gegangen werden
- $E[j, i] = \infty \rightarrow$ Weg ist nicht mehr verfügbar

Für die restlichen Felder die Werte aus der Standardmatrix einsetzen und wieder die reduzierte Entfernungsmatrix bilden.

Beispiel Vertreter

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Weg Görlitz → Löbau:

	G	L	N	Z	
G	∞	25_0	∞	∞	25
L	∞	∞	27_0	27_0	27
N	22_0	∞	∞	64_{42}	22
Z	36_0	∞	64_{28}	∞	36
	0	0	0	0	**110**

Weg Görlitz → Niesky:

	G	L	N	Z	
G	∞	∞	23_0	∞	23
L	25_0	∞	∞	27_2	25
N	∞	27_0	∞	64_{37}	27
Z	36_9	27_0	∞	∞	27
	0	0	0	2	**104**

Weg Görlitz → Zittau:

	G	L	N	Z	
G	∞	∞	∞	35_0	35
L	25_0	∞	27_2	∞	25
N	22_0	27_5	∞	∞	22
Z	∞	27_0	64_{37}	∞	27
	0	0	2	0	**111**

Der Schranken-Wert 99 der Start-Matrix sagt aus, dass es keine Lösung gibt, die kleiner als 99 ist. Die Schranken-Werte aller durch die Expansion erzeugten Matrizen sind zwar größer, aber es ist keiner unter 99 dabei. Im nächsten Schritt wird die Matrix mit dem kleinsten Schranken-Wert expandiert ⇒ also der Weg von Görlitz nach Niesky

Beispiel Vertreter

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Weg Görlitz → Niesky → Löbau:

	G	L	N	Z	
G	∞	∞	23_0	∞	23
L	25_0	∞	∞	27_2	25
N	∞	27_0	∞	∞	27
Z	36_0	∞	∞	∞	36
	0	0	0	2	**113**

Weg Görlitz → Niesky → Zittau:

	G	L	N	Z	
G	∞	∞	23_0	∞	23
L	25_0	∞	∞	∞	25
N	∞	∞	∞	64_0	64
Z	36_9	27_0	∞	∞	27
	0	0	0	0	**139**

Die neu ausgerechneten Schranken sind alle größer als, als die Schranke von Görlitz → Löbau eine Stufe weiter oben. Deshalb wird als nächstes die Matrix Görlitz → Löbau expandiert.

Beispiel Vertreter

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Weg Görlitz → Löbau → Niesky:

	G	L	N	Z	
G	∞	25_0	∞	∞	25
L	∞	∞	27_0	∞	27
N	22_0	∞	∞	64_{42}	22
Z	36_0	∞	∞	∞	36
	0	0	0	42	**152**

Weg Görlitz → Löbau → Zittau:

	G	L	N	Z	
G	∞	25_0	∞	∞	25
L	∞	∞	∞	27_0	27
N	22_0	∞	∞	∞	22
Z	36_0	∞	64_{28}	∞	36
	0	0	28	0	**138**

Die neu ausgerechneten Schranken sind alle größer als, als die Schranke von Görlitz → Zittau eine Stufe weiter oben. Deshalb wird als nächstes die Matrix Görlitz → Zittau expandiert.

Beispiel Vertreter

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Weg Görlitz $\rightarrow$ Zittau $\rightarrow$ Löbau:

	G	L	N	Z	
G	∞	∞	∞	35_0	35
L	25_0	∞	27_2	∞	25
N	22_0	∞	∞	∞	22
Z	∞	27_0	∞	∞	27
	0	0	2	0	**111**

Weg Görlitz $\rightarrow$ Zittau $\rightarrow$ Niesky:

	G	L	N	Z	
G	∞	∞	∞	35_0	35
L	25_0	∞	27_2	∞	25
N	22_0	27_5	∞	∞	22
Z	∞	∞	64_0	∞	64
	0	5	2	0	**153**

Die noch nicht expandierte Matrix mit der kleinsten Schranke ist jetzt Görlitz $\rightarrow$ Zittau $\rightarrow$ Löbau mit 111. Diese wird jetzt nochmals expandiert.

Beispiel Vertreter

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Weg Görlitz $\rightarrow$
Zittau $\rightarrow$ Löbau $\rightarrow$
Niesky:

	G	L	N	Z	
G	∞	∞	∞	35_0	35
L	∞	∞	27_0	∞	27
N	22_0	∞	∞	∞	22
Z	∞	27_0	∞	∞	27
	0	0	0	0	**111**

111 ist die niedrigste Schranke und deshalb ist der Weg Görlitz $\rightarrow$ Zittau $\rightarrow$ Löbau $\rightarrow$ Niesky der kürzeste für die Route des Vertreters.

Implementierung des Reiseproblems - Ansatz

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

- um jeweils den nächsten zu expandierenden Knoten zu wählen, müssen die Knoten in einer Warteschlange hinterlegt werden
- hier eignet sich eine **Prioritäts-Warteschlange**, in der die Knoten mit dem kleinsten Schrankenwert an erster Stelle stehen
- neue Knoten werden mit *insert*() an der passenden Stelle eingefügt
- beim nächsten Expansionsschritt wird der vorderste Knoten mit *removeMin*() verbrauchend gelesen
- zur Implementation der Prioritäts-Warteschlange eignet sich z.B. der Heap
- die Effizienz von *insert*() und *removeMin*() liegt bei $\mathcal{O}(\log n)$

Implementierung - Beispiel in Java

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

```
import java.util.*;

public class PriorityQueue {
    Object[] inhalt;
    int[] schranken;
    int elements;

    PriorityQueue(){
        inhalt = new Object[0];
        schranken = new int[0];
        elements = 0;
    }

    void insert(Object o, int wert){
        Object[] inhalt2 = new
       Object[inhalt.length+1];
        for(int i=0;i<inhalt.length;i++){
            inhalt2[i] = inhalt[i];
        }
        inhalt = inhalt2;
        inhalt[inhalt.length-1]=o;
        int[] schranken2 = new
       int[schranken.length+1];
        for(int i=0;i<schranken.length;i++){
            schranken2[i] = schranken[i];
        }
        schranken = schranken2;
        schranken[schranken.length-1]=wert;
        elements = elements+1;
    }

    Object removeMin(){
        int min = Integer.MAX_VALUE;
        for(int i=0;i<schranken.length;i++){
            if(schranken[i]<min){
                min=i;
            }
        }
        Object ergebnis = inhalt[min];
        Object[] inhalt2 = new
       Object[inhalt.length-1];
        for(int i=0;i<min;i++){
            inhalt2[i] = inhalt[i];
        }
        for(int
       i=(min+1);i<(inhalt.length);i++){
            inhalt2[i-1] = inhalt[i];
        }
        inhalt = inhalt2;
        int[] schranken2 = new
       int[schranken.length-1];
        for(int i=0;i<min;i++){
            schranken2[i] = schranken[i];
        }
        for(int
       i=(min+1);i<(schranken.length);i++){
            schranken2[i-1] = schranken[i];
        }
        schranken = schranken2;
        elements = elements-1;
        return ergebnis;
    }
}
```

Fragen und Diskussion

Fakultät Elektrotechnik und Informatik
Hochschule Zittau/Görlitz

Vielen Dank für die Aufmerksamkeit!

Gibt es Fragen?