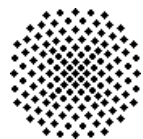
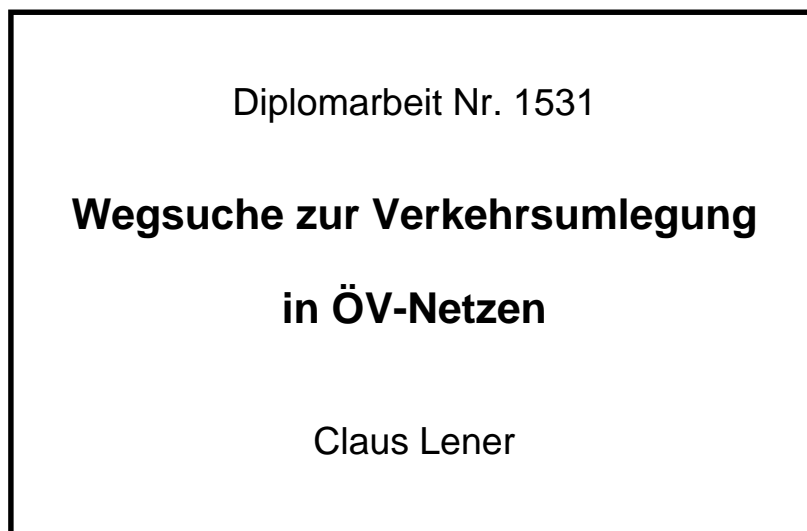


Prüfer: Prof. Dr. rer. nat. Volker Claus  
Betreuer: Dipl.-Inform. Friedhelm Buchholz  
Dipl.-Inform. Jürgen Hoffmann

begonnen am: 15.04.1997  
beendet am: 14.10.1997

CR-Klassifikation: G.2.2 Graph Theory



**Universität  
Stuttgart**

Fakultät Informatik  
Institut für Informatik

Breitwiesenstraße 20-22  
D-70565 Stuttgart

Fakultät Bauingenieurwesen  
Institut für Eisenbahn- und  
Verkehrswesen

Pfaffenwaldring 7  
D-70569 Stuttgart

## Inhaltsverzeichnis

<b>1. Aufgabenstellung</b> .....	<b>4</b>
<b>2. Datenstrukturen zur Wegsuche</b> .....	<b>6</b>
2.1 ÖV-Netzmodelle .....	6
2.1.1 Einleitung.....	6
2.1.2 Spezifikation von ÖV-Netzmodellen.....	8
2.1.3 Annahmen zur Größenordnung von H, L und F gegenüber B:.....	9
2.1.4 Umsteigebeziehungen in ÖV-Netzmodellen:.....	11
2.1.5 Wege in ÖV-Netzmodellen .....	12
2.2 Bewertung von Wegen in ÖV-Netzmodellen .....	14
2.3 Distanzgraphen .....	18
2.4 Konvertierung von ÖV-Netzmodellen in Distanzgraphen.....	20
2.5 Abbildung von Wegen in Distanzgraphen auf ÖV-Wege.....	24
2.6 Berücksichtigung von Parallelverkehr.....	26
2.6.1 Motivation .....	26
2.6.2 Verfahrensprinzipen.....	28
2.6.3 Bündeln von Parallelverkehr .....	29
<b>3. Spezifikation der gesuchten Wege</b> .....	<b>36</b>
<b>4. Algorithmen zur Bestimmung der gesuchten Wege</b> .....	<b>40</b>
4.1 Problemstellung und -lösung.....	40
4.1.1 Besonderheiten der Problemstellung .....	40
4.1.2 Ansätze zur Problemlösung .....	45
4.2 Lösungsansatz nach Minieka (N:N-Suchverfahren).....	47
4.2.1 Allgemeines .....	47
4.2.2 Grundprinzip .....	47
4.2.3 Anpassungen an die Problemstellung.....	47
4.2.4 Grundelemente und -operationen .....	48
4.2.5 Ablauf .....	49
4.2.6 Korrektheitsbeweis .....	50
4.2.7 Aufwandsabschätzung.....	51
4.2.8 Beispiel.....	52
4.3 Tiefensuche und Algorithmus nach Dijkstra (1:N-Suchverfahren) .....	55
4.3.1 Allgemeines .....	55
4.3.2 Algorithmus nach Dijkstra .....	55
4.3.3 Grundelemente .....	55
4.3.4 Ablauf .....	56
4.3.5 Korrektheitsbeweis .....	56
4.3.6 Aufwandsabschätzung.....	57
4.3.7 Beispiel.....	58
4.4 Prinzip der Linienverfolgung .....	59
4.4.1 Allgemeines .....	59
4.4.2 Grundprinzip .....	59
4.4.3 Grundelemente und -operationen .....	59
4.4.4 Ablauf .....	62
4.4.5 Korrektheitsbeweis .....	62
4.4.6 Aufwandsabschätzung.....	64
4.4.7 Beispiel.....	65
4.5 Ansätze zur Parallelisierung der Wegsuche .....	68
4.5.1 Parallelisierung des Verfahrens nach Minieka .....	68
4.5.2 Parallelisierung der Tiefensuche und des Algorithmus nach Dijkstra .....	70
4.5.3 Parallele Linienverfolgung.....	71

<b>5. Verkehrsumlegung .....</b>	<b>72</b>
<b>6. Implementierung und Tests .....</b>	<b>75</b>
6.1 Voraussetzungen .....	75
6.2 Datenstrukturen im Rechner .....	76
6.2.1 Aufbau und Organisation von ÖV-Netzmodellen.....	76
6.2.2 Datenstruktur von Distanzgraphen.....	78
6.2.3 Konvertierung von ÖV-Netzmodellen in Distanzgraphen .....	79
6.2.4 Datenstruktur von Wegeprotokollen.....	80
6.3 Implementierung der Wegsuche-Algorithmen .....	82
6.3.1 Implementierung des Verfahrens nach Minieka .....	82
6.3.2 Implementierung des Verfahrens nach Dijkstra .....	83
6.3.3 Implementierung des Verfahrens der Linienverfolgung.....	83
6.4 Laufzeitverhalten und Speicherplatzbedarf in der Praxis.....	84
<b>7. Zusammenfassung und Ausblick.....</b>	<b>87</b>
<b>8. Literaturquellen .....</b>	<b>90</b>

## Verzeichnis der Definitionen

<b>Def. 2.1:</b> ÖV-Netzmodell: $\text{ÖV} = (B, H, L, F, \tau)$ .....	8
<b>Def. 2.2:</b> Menge U der <b>Umsteigebeziehungen</b> in einem ÖV-Netzmodell .....	11
<b>Def. 2.3:</b> Menge S der <b>Linienabschnitte</b> in einem ÖV-Netzmodell .....	12
<b>Def. 2.4:</b> Menge T der <b>Fahrten</b> in einem ÖV-Netzmodell .....	12
<b>Def. 2.5:</b> Menge $W^{(\text{ÖV})}$ der <b>Wege</b> in ÖV-Netzmodellen .....	13
<b>Def. 2.6:</b> <b>Widerstands- oder Bewertungsfunktion</b> $\omega$ .....	14
<b>Def. 2.7:</b> Gerichteter, endlicher <b>Distanzgraph</b> : $G = (V, E, \gamma)$ .....	18
<b>Def. 2.8:</b> Menge W der <b>Wege</b> in einem Distanzgraph .....	18
<b>Def. 2.9:</b> <b>Weglängen</b> in einem Distanzgraph .....	19
<b>Def. 2.10:</b> <b>Linienbündel oder Parallelverkehr</b> .....	26
<b>Def. 2.11:</b> <b>Taktfolgezeit eines Linienbündels</b> .....	26
<b>Def. 3.1:</b> <b>Bestwege</b> .....	36
<b>Def. 3.2:</b> <b>Grenzwertfunktion</b> $\Gamma$ .....	37
<b>Def. 3.3:</b> <b>Umsteigehäufigkeit</b> $\mu$ eines ÖV-Weges.....	38
<b>Def. 3.4:</b> <b>Lösungsmenge</b> $L_{a,b}^{(\text{ÖV})}$ für $a,b \in B$ (gesuchte Wege) .....	39
<b>Def. 4.1:</b> Bei der Routensuche verwendete <b>oberen Schranken</b> $D_{i,j}$ .....	41
<b>Def. 4.2:</b> <b>Konkatenation von Wegen</b> .....	41
<b>Def. 4.3:</b> <b>Wegeprotokoll</b> $P_{i,j}$ für ein Knotenpaar $(i,j) \in V^2$ , $i \neq j$ .....	48
<b>Def. 4.4:</b> <b>Konkatenation zweier Wegeprotokolle</b> $P_{a,b}, P_{b,c}$ , $a,b,c \in V$ .....	48
<b>Def. 4.5:</b> Menge der <b>ÖV-Wegbeschreibungen</b> .....	59
<b>Def. 4.6:</b> <b>Umsteigehäufigkeit</b> $\mu$ einer ÖV-Wegbeschreibung.....	60
<b>Def. 4.7:</b> <b>Haltstellenfolge</b> einer Wegbeschreibung (besuchte Haltstellen) .....	60
<b>Def. 4.8:</b> Erweiterung der <b>Bewertungsfunktion</b> auf Wegbeschreibungen.....	60
<b>Def. 4.9:</b> <b>ÖV-Wegeprotokolle</b> .....	61
<b>Def. 4.10:</b> <b>Selektive Vereinigung</b> $\cup_k$ zweier Mengen von Wegbeschreibungen .....	61
<b>Def. 5.1:</b> <b>Gewichtungsfunktion</b> für Wege .....	73

## 1. Aufgabenstellung

Zur Prognose der Verkehrsnachfrage und zur Abschätzung der Auswirkungen von verkehrsplanerischen Maßnahmen werden seit vielen Jahren EDV-Programme eingesetzt, um schnellere und genauere Berechnungen durchführen zu können. Am Institut für Eisenbahn- und Verkehrswesen an der Universität Stuttgart (IEUV) wird im Rahmen von Gutachten und Bewertungen von Planungen im öffentlichen Verkehr (ÖV) ein Nachfragemodell entworfen, um das zukünftige Verhalten von Fahrgästen in ÖV-Netzen abschätzen zu können. Dabei werden folgende Maßnahmen ergriffen:

1. Erhebung des Mobilitätsbedarfs der Bevölkerung des Untersuchungsgebiets und Abschätzung des daraus resultierenden Verkehrsaufkommens (sogenannte Verkehrserzeugung),
2. Bestimmung der Verkehrsverflechtungen im Untersuchungsgebiet durch Aufstellen einer sogenannten Fahrtenmatrix, welche das Verkehrsaufkommen  $F_{ij}$  zwischen jeder Quelle  $i$  und jedem Ziel  $j$  beschreibt (Verkehrsverteilung),
3. Aufteilung der Verkehrsnachfrage auf verschiedene Verkehrsmittel („modal split“), wobei man sich in der Regel auf die Unterscheidung zwischen Individualverkehr und ÖV beschränkt,
4. Verteilung der Nachfrage im ÖV für alle Verkehrsbeziehungen auf geeignete Routen des ÖV-Netzes mit der Absicht, die Verkehrsbelastung auf allen Strecken des Netzes zu berechnen (Verkehrsumlegung).

Das obige Verfahren wird im Verkehrswesen als „4-Stufen-Algorithmus“ bezeichnet. Zu weiteren Informationen bezüglich der ersten drei Schritte des Verfahrens sei auf die Literatur verwiesen (z.B. [2]). Die Verkehrsumlegung als letzter Schritt soll in dieser Diplomarbeit näher betrachtet werden. Im Vordergrund steht dabei die Frage, welche Routen die Verkehrsteilnehmer wählen, um von einer Quelle an ein Ziel zu gelangen. Sind diese Routen bekannt, so kann man den Verkehrsfluß in allen Teilen des ÖV-Netzes ermitteln und feststellen, ob die zukünftige Nachfrage an bestimmten Streckenabschnitten das Verkehrsangebot übersteigt. Auf diese Weise können rechtzeitig Maßnahmen zum Ausbau oder zur Verlagerung des Angebots ergriffen werden. Gerade deshalb sind die Ergebnisse der Umlegung von zentraler Bedeutung für die Verkehrsplanung.

Schwerpunkt dieser Diplomarbeit ist die effiziente Suche nach geeigneten Wegen in ÖV-Netzen als Voraussetzung für die Verkehrsumlegung. Dabei ist in erster Linie die Frage zu klären, welche Wege aus Sicht der Fahrgäste als geeignet bzw. ungeeignet zu betrachten sind. Zur Durchführung der Routensuche in ÖV-Netzen gibt es die Möglichkeit, aus der Graphentheorie bekannte Wegsuche-Algorithmen heranzuziehen und an die Problemstellung anzupassen oder neue, spezielle Verfahren zu entwickeln, welche sich am Verkehrsnachfragemodell orientieren. Dabei ist vor allem die Effizienz der Algorithmen von Interesse. Im Hinblick auf die zunehmende Verbreitung von Parallelrechnern ist es sinnvoll, bei den grundlegenden Untersuchungen Aspekte der verteilten Verarbeitung einzubeziehen und Verbesserungen gegenüber der sequentiellen Verarbeitung aufzuzeigen. Abschließend sollen die analysierten Algorithmen implementiert und deren Effizienz sowie deren Tauglichkeit in der Praxis anhand von ÖV-Netzen, welche bei Verkehrsuntersuchungen des IEUV zum Einsatz kommen, demonstriert werden.

## 2. Datenstrukturen zur Wegsuche

### 2.1 ÖV-Netzmodelle

#### 2.1.1 Einleitung

Um Verkehrsprognosen mit Hilfe des Rechners durchführen zu können, müssen charakteristische Ausprägungen des betrachteten Verkehrsnetzes modellmäßig erfaßt werden. Das im Folgenden erläuterte Netzmodell für den öffentlichen Verkehr ist am IEUV gebräuchlich und stellt die Grundlage für die Wegsuche und die Verkehrs-umlegung dar.

Das ÖV-Netzmodell gliedert sich in zwei Schichten:

- 1.) Ebene des Untersuchungsgebiets bzw. der Verkehrsbezirke,
- 2.) Ebene des Verkehrsnetzes (hier: ÖV).

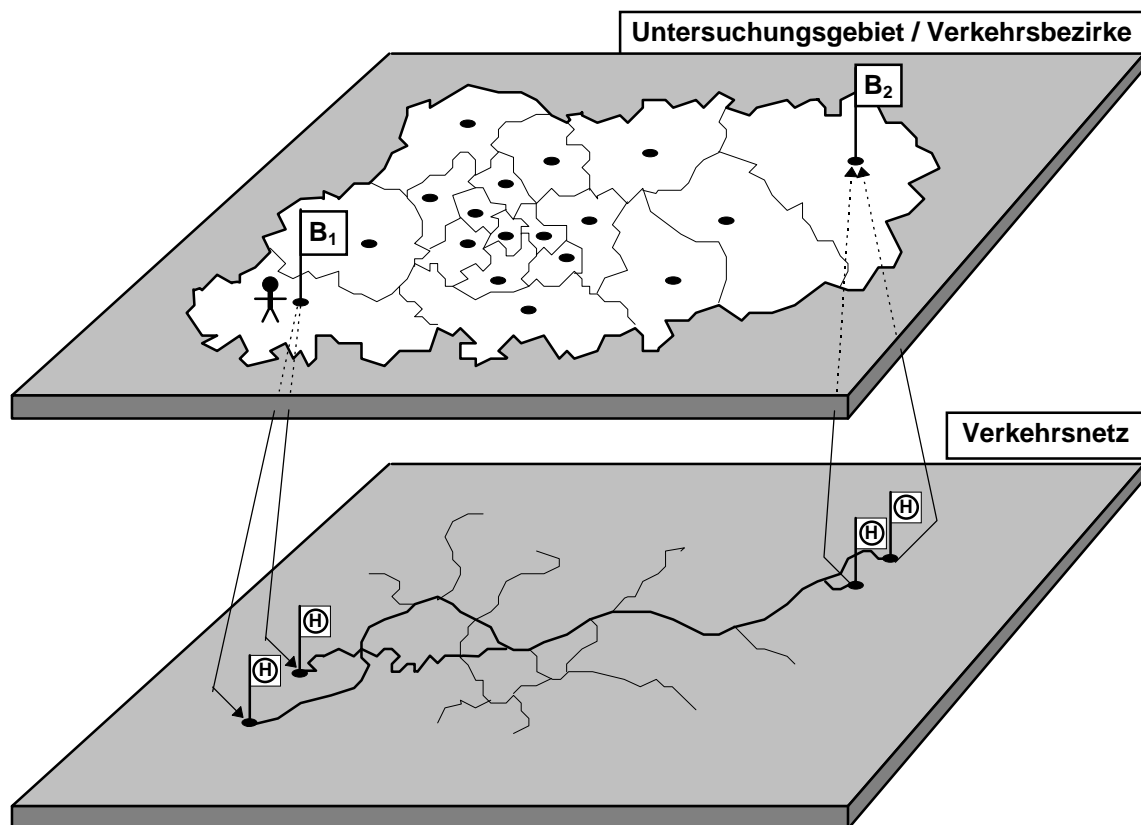


Abbildung 2.1: Beispiel für ein ÖV-Netzmodell: Die Verkehrsbezirke des Untersuchungsgebiets sind über Fußwege mit den jeweils nächstgelegenen Haltestellen verknüpft. Im Verkehrsnetz sind die Haltestellen über verschiedene Linien miteinander verbunden. Auf diese Weise ergeben sich mehrere (sinnvolle) Wege für Personen, welche beispielsweise von B<sub>1</sub> nach B<sub>2</sub> reisen wollen.

zu 1.)

Um das Datenmaterial überschaubar zu halten, werden die Fahrten im Verkehrsnetz nicht einzeln (Haus-zu-Haus-Verkehr), sondern kollektiv behandelt. Zu diesem Zweck wird das Untersuchungsgebiet in Verkehrszellen (Verkehrsbezirke) eingeteilt. Für die Region Stuttgart beispielsweise dient der Regionalverkehrsplan (siehe [5]) als Vorlage. Die Gliederung basiert in der Regel auf statistischen Bezirken, wobei der Umfang der Verkehrszellen im Rahmen von Untersuchungen an planerische Vorgaben und örtliche Gegebenheiten angepaßt werden muß: In Teilbereichen, in denen Maßnahmen durchgeführt werden sollen (Planungsraum), wird eine besonders feine Gliederung angestrebt. Im Gegensatz dazu genügt in den Randgebieten des Untersuchungsgebiets eine großflächige Einteilung.

Zur Vereinfachung der Berechnungen ist im ÖV-Netzmodell die Bevölkerung eines Verkehrsbezirks im Schwerpunkt der Siedlungsfläche zentriert, so daß für alle Personen, welche eine Reise zu einem anderen Bezirk unternehmen wollen, dieselben Voraussetzungen gelten. Der Zellbinnenverkehr spielt bei Verkehrsplanungen eine untergeordnete Rolle. Der Verkehr über das Untersuchungsgebiet hinaus kann vernachlässigt werden, wenn das Gebiet hinreichend groß dimensioniert ist.

zu 2.)

Das Verkehrsnetz stellt die Grundlage für Ortsveränderungen von Personen dar. ÖV-Netze bestehen aus Haltestellen sowie Linien, welche die Haltestellen bedienen. Das Netz muß stark zusammenhängend sein, d.h. von einer beliebigen Haltestelle ausgehend sollte jede andere Haltestelle erreichbar sein.

Um herauszufinden, auf welchen Routen Fahrten im ÖV-Netz durchgeführt werden, müssen zuerst Verbindungen zwischen Verkehrsbezirken und Haltestellen festgelegt werden. Jedem Bezirk werden die dem Siedlungsschwerpunkt nächstgelegenen Haltestellen zugeordnet (eingebunden). Auf diese Weise erhält die Bevölkerung über einen oder mehrere Anbindungsfußwege Zugang zum ÖV-Netz. Die Länge der Fußwege wird anhand der Entfernung zwischen Schwerpunkt und Haltestelle grob abgeschätzt. Verlassen wird das Netz über dieselben Haltestellen und Fußwege wie beim Zugang. Linien stellen die Verbindung zwischen den Zu- und Abgangshaltestellen her.

### 2.1.2 Spezifikation von ÖV-Netzmodellen

Die Struktur von ÖV-Netzmodellen läßt sich formal wie folgt beschreiben:

**Def. 2.1: ÖV-Netzmodell:**  $\text{ÖV} = (B, H, L, F, \tau)$

**B :** endliche Menge von Verkehrsbezirken (Verkehrszellen).

**H :** endliche Menge von Haltestellen,  $H \cap B = \{ \}$ , wobei jede Haltestelle von mindestens einer Linie bedient wird:

$$\forall h \in H: \exists l \in L: h \in l .$$

**$L \subseteq H^2 \times H^*$ :** endliche Menge von Linien: Eine Linie ist eine Haltestellenfolge, die mindestens einen Start- und einen Endpunkt enthält. Dabei ist nicht ausgeschlossen, daß eine Haltestelle mehrfach in einer Folge vorkommt. Richtung und Gegenrichtung einer realen Linie werden im Modell auf zwei verschiedene Folgen abgebildet. Da jede Linie einen festen Verlauf besitzen muß, werden unterschiedliche Fahrwege einer Linie (Streckenabschnitte, welche nur zeitweise bedient werden) im Modell durch verschiedene Linien repräsentiert.

**$F \subseteq B \times H$  :** Menge von (Anbindungs-)Fußwegen zwischen den Verkehrsbezirken und den Haltestellen. Zu- und Abgang werden über dieselben Fußwege abgewickelt. Dabei muß jeder Bezirk mindestens einen Zugang zum Liniennetz haben:

$$\forall b \in B: \exists h \in H: (b, h) \in F .$$

**$\tau: L \rightarrow \mathbb{R}^+$ :** Taktfolgezeit einer Linie in der werktäglichen Hauptverkehrszeit: Die Taktfolgezeit ist die Zeitdauer zwischen zwei aufeinanderfolgenden Fahrten einer Linie. Sie wird bei der Wegsuche berücksichtigt, da sie ein wichtiges Qualitätsmerkmal einer Linie ist (je kleiner desto besser).

### **2.1.3 Annahmen zur Größenordnung von H, L und F gegenüber B:**

Bei der Einteilung des Untersuchungsgebiets in Verkehrsbezirke wird darauf geachtet, daß die Anzahl der Haltestellen je Verkehrszelle einen bestimmten Wert nicht überschreitet, weil bei zu groß dimensionierten Bezirken die Genauigkeit einer Verkehrsprognose leidet. Die Differenzierung der Verkehrsströme im ÖV-Netz ist optimal, wenn jeder Bezirk dem Einzugsgebiet einer Haltestelle entspricht und somit  $|B| = |H|$  ist. In der Praxis läßt sich dies kaum verwirklichen, weil die zur Berechnung von Verkehrsströmen notwendigen verkehrszellenbezogenen Strukturdaten auf dieser kleinräumigen Ebene nicht verfügbar sind. Dennoch ist anzunehmen, daß es eine obere Schranke für die Anzahl der Haltestellen je Verkehrszelle gibt, d.h.  $|H|$  ein konstantes Vielfaches von  $|B|$  ist, so daß gilt:

$$|H| \in O(|B|).$$

Bei der Konzeption von Linien wird in der Regel darauf geachtet, daß der Fahrweg, also die Folge der bedienten Haltestellen nicht zu lang ist, weil bei großen Umläufen die Anfälligkeit für Verspätungen entsprechend groß ist. Daher ist die Annahme zulässig, daß es für die Anzahl der bedienten Haltestellen je Linie eine obere Grenze gibt, d.h.

$$\forall l \in L: |l| \in O(1).$$

Darüber hinaus kann man davon ausgehen, daß es bei Mehrfachbedienung einer Haltestelle obere Schranken für die Anzahl der abfahrenden Linien gibt, weil einerseits das Verkehrsaufkommen je Haltestelle und andererseits das Bündeln von Linien insbesondere in Verkehrszentren begrenzt ist, daher die Annahme:

$$|L| \in O(|H|) = O(|B|).$$

Jeder Verkehrsbezirk ist gemäß Def. 2.1 über mindestens einen Anbindungsfußweg mit einer Haltestelle verbunden. Gibt es mehrere Fußwege, so kann die ortsansässige Bevölkerung je nach Reiseziel zwischen den Alternativen auswählen. Da die Anzahl der Haltestellen je Bezirk beschränkt ist (s.o.), ist auch die Anzahl dieser Alternativen begrenzt, so daß  $|F|$  ein konstantes Vielfaches von  $|B|$  ist, d.h.

$$|F| \in O(|B|).$$

Zusammenfassend kann man feststellen, daß bei ÖV-Netzmodellen  $|H|$ ,  $|L|$ ,  $|F|$  und  $|B|$  als proportional anzusehen sind.

**Beispiel 2.1:**

Sei  $\text{ÖV}_{\text{Holzdorf}} = (B, H, L, F, \tau)$  ein **ÖV-Netzmodell** mit folgenden Eigenschaften:

$B = \{ \text{West, Mitte, Ost} \}$ ,

$H = \{ \text{Rathaus, Bhf, Schule, Kirche, Friedhof, Park} \}$ ,

$L = \{ ( \text{Park, Rathaus, Bhf, Kirche} ),$   
 $( \text{Kirche, Bhf, Rathaus, Park} ),$   
 $( \text{Rathaus, Schule, Bhf, Friedhof} ),$   
 $( \text{Friedhof, Bhf, Schule, Rathaus} ) \}$ ,

$F = \{ ( \text{West, Kirche} ), ( \text{West, Friedhof} ),$   
 $( \text{Mitte, Schule} ), ( \text{Ost, Park} ) \}$ ,

$\tau(L) = \{ 10, 10, 20, 20 \}$ .

Verkehrsbezirke,

Haltestellen

Linie 1a (1-Richtung)

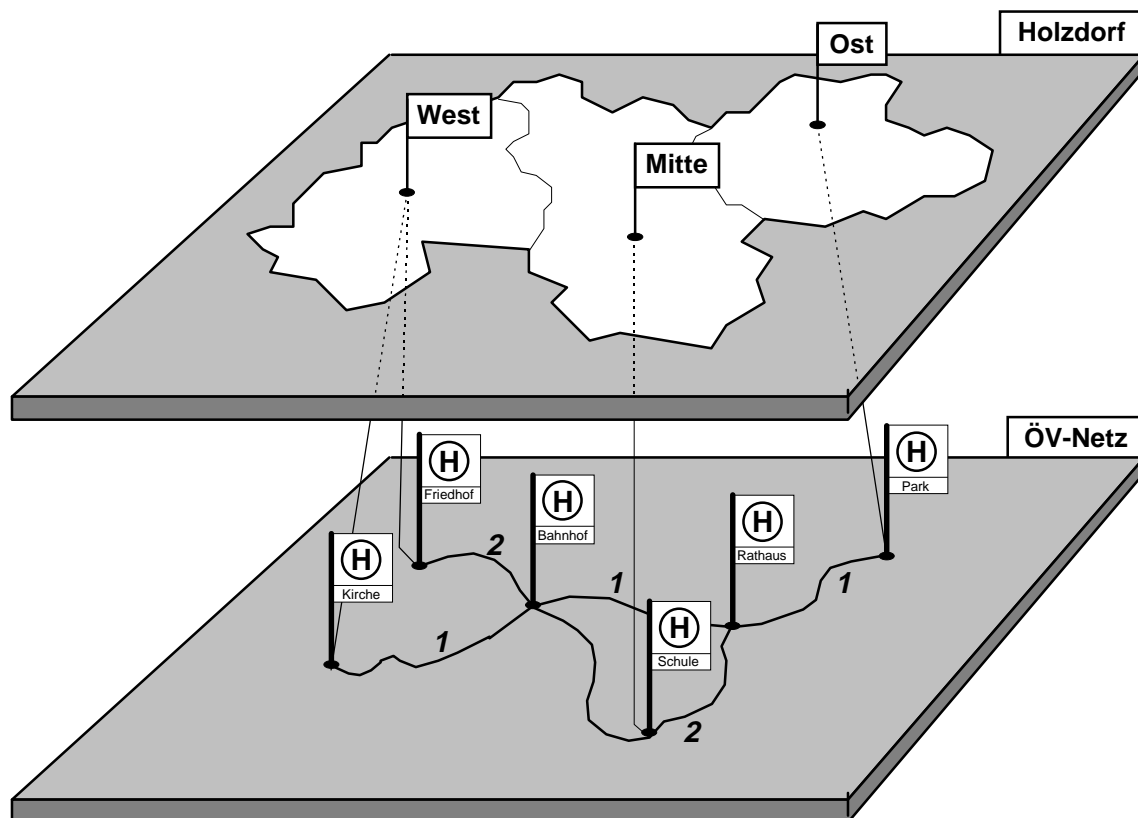
Linie 1b (1-Gegenrichtung)

Linie 2a (2-Richtung)

Linie 2b (2-Gegenrichtung)

Anbindungsfußwege

Taktfolgezeiten in Minuten



**Abbildung 2.2:** ÖV-Netzmodell „Holzdorf“ mit drei Verkehrsbezirken, sechs Haltestellen, vier Linien (Richtung und Gegenrichtung unterschieden) sowie vier Anbindungsfußwegen.

### **2.1.4 Umsteigebeziehungen in ÖV-Netzmodellen:**

Normalerweise gibt es für die Mehrheit der Verkehrsbeziehungen in einem ÖV-Netz keine Direktverbindung mittels einer einzigen Linie. Bei der Wegsuche müssen daher Umsteigebeziehungen berücksichtigt werden. Sie ergeben sich implizit an allen Haltestellen, an denen mindestens zwei Linien aufeinandertreffen (sogenannte Verknüpfungspunkte):

**Def. 2.2:** Menge  $U$  der **Umsteigebeziehungen** in einem ÖV-Netzmodell:

Sei  $\ddot{O}V = (B, H, L, F, \tau)$  ein ÖV-Netzmodell. Dann ist

$$U = \{ (l_1, l_2, h) \in L^2 \times H \mid h \in l_1 \text{ und } h \in l_2 \text{ und } l_1 \neq l_2 \}.$$

$U$  enthält alle Umsteigebeziehungen zwischen paarweise verschiedenen Linien, wobei die Angabe des Umsteigepunkts notwendig ist, da es in vielen Fällen mehrere Möglichkeiten gibt, zwischen zwei bestimmten Linien umzusteigen.

### **Beispiel 2.2:**

Für  $\ddot{O}V_{\text{Holzdorf}}$  aus Beispiel 2.1 ergeben sich folgende 23 **Umsteigebeziehungen**:

$$U = \{ (1a, 1b, Rathaus), (1a, 2a, Rathaus), (1b, 1a, Rathaus), (1b, 2a, Rathaus), \\ (2b, 1a, Rathaus), (2b, 1b, Rathaus), (2b, 2a, Rathaus), \\ (1a, 1b, Bhf), (1a, 2a, Bhf), (1a, 2b, Bhf), (1b, 1a, Bhf), (1b, 2a, Bhf), (1b, 2b, Bhf), \\ (2a, 1a, Bhf), (2a, 1b, Bhf), (2a, 2b, Bhf), (2b, 1a, Bhf), (2b, 1b, Bhf), (2b, 2a, Bhf), \\ (2a, 2b, Schule), (2b, 2a, Schule), (1b, 1a, Kirche), (2a, 2b, Friedhof), (1a, 1b, Park) \}.$$

Dabei sind 1a, 1b, 2a und 2b die Kürzel für die vier Linien von  $\ddot{O}V_{\text{Holzdorf}}$ .

### 2.1.5 Wege in ÖV-Netzmodellen

Um Wege formal beschreiben zu können, müssen zunächst alle Linienverläufe in deren kleinsten Elemente, den Linienabschnitten, zerlegt werden:

**Def. 2.3:** Menge S der **Linienabschnitte** in einem ÖV-Netzmodell:

Sei  $\ddot{O}V = (B, H, L, F, \tau)$  ein ÖV-Netzmodell. Dann ist

$$S = \{ (h_i, h_{i+1}, l) \in H^2 \times L \mid l = (h_1, \dots, h_i, h_{i+1}, \dots, h_n) \in L, 1 \leq i < n \}$$

Linienabschnitte sind Verbindungen zwischen zwei in einem Linienverlauf aufeinanderfolgenden Haltestellen. Da zwischen zwei Haltestellen mehrere Linien parallel mit unterschiedlicher Fahrtdauer verkehren können, ist für jeden Abschnitt  $s \in S$  die Angabe der Linie zur Differenzierung der Fahrzeit unverzichtbar.

#### Beispiel 2.3:

Für  $\ddot{O}V_{\text{Holzdorf}}$  aus Beispiel 2.1 ergeben sich folgende **Linienabschnitte**:

$$S = \{ (\text{Park}, \text{Rathaus}, 1a), (\text{Rathaus}, \text{Bhf}, 1a), (\text{Bhf}, \text{Kirche}, 1a), \\ (\text{Kirche}, \text{Bhf}, 1b), (\text{Bhf}, \text{Rathaus}, 1b), (\text{Rathaus}, \text{Park}, 1b), \\ (\text{Rathaus}, \text{Schule}, 2a), (\text{Schule}, \text{Bhf}, 2a), (\text{Bhf}, \text{Friedhof}, 2a), \\ (\text{Friedhof}, \text{Bhf}, 2b), (\text{Bhf}, \text{Schule}, 2b), (\text{Schule}, \text{Rathaus}, 2b) \}.$$

Aufeinanderfolgende Linienabschnitte lassen sich zu längeren Verbindungen zusammenbauen (transitiver Abschluß von S bezüglich jeder Linie):

**Def. 2.4:** Menge T der **Fahrten** in einem ÖV-Netzmodell:

Sei  $\ddot{O}V = (B, H, L, F, \tau)$  ein ÖV-Netzmodell. Dann ist

$$T = \{ (h_i, h_j, l) \in H^2 \times L \mid l = (h_1, \dots, h_i, h_{i+1}, \dots, h_j, \dots, h_n) \in L, 1 \leq i < j \leq n \}$$

Fahrten sind durch Benutzung einer bestimmten Linie mögliche (Direkt-) Verbindungen zwischen zwei Haltestellen. Eine Verbindung  $t = (h_i, h_j, l) \in T$  beschreibt eine Fahrt von Haltestelle  $h_i$  zu Haltestelle  $h_j$  mit Linie  $l$ .

**Beispiel 2.4:**

Für  $\text{ÖV}_{\text{Holzdorf}}$  aus Beispiel 2.1 ergeben sich folgende **Fahrten**:

$$T = S \cup \{ (\text{Park, Bahnhof, 1a}), (\text{Park, Kirche, 1a}), (\text{Rathaus, Kirche, 1a}), \\ (\text{Kirche, Rathaus, 1b}), (\text{Kirche, Park, 1b}), (\text{Bahnhof, Park, 1b}), \\ (\text{Rathaus, Bahnhof, 2a}), (\text{Rathaus, Friedhof, 2a}), (\text{Schule, Friedhof, 2a}), \\ (\text{Friedhof, Schule, 2b}), (\text{Friedhof, Rathaus, 2b}), (\text{Bahnhof, Rathaus, 2b}) \}.$$

Dabei ist S die Menge der Linienabschnitte aus Beispiel 2.3.

**Def. 2.5:** Menge  $W^{(\text{ÖV})}$  der **Wege** in ÖV-Netzmodellen:

$$W^{(\text{ÖV})} \subseteq F \times T \times (U \times T)^* \times F$$

Jeder Weg beginnt und endet mit einem Fußweg, dazwischen muß mindestens eine Fahrt mit einer Linie stattfinden. Durch Umsteigen lassen sich weitere Fahrten anschließen. Dabei muß auf den Zusammenhang zwischen den Komponenten eines Weges geachtet werden:

Menge der **Wege zwischen zwei Verkehrsbezirken**  $a, b \in B$ :

$$W_{a,b}^{(\text{ÖV})} = \{ \underbrace{(a, h_0)}_{\in F}, \underbrace{(h_0, h_1, l_1)}_{\in T}, \underbrace{(l_1, l_2, h_1)}_{\in U}, \underbrace{(h_1, h_2, l_2)}_{\in T}, \dots, \underbrace{(l_{n-1}, l_n, h_{n-1})}_{\in U}, \underbrace{(h_{n-1}, h_n, l_n)}_{\in T}, \underbrace{(b, h_n)}_{\in F} \mid \\ n \geq 1 \text{ und } \forall 0 \leq i \leq n : h_i \in H \text{ und } \forall 1 \leq i \leq n : l_i \in L \}.$$

**Bemerkung:**

Def. 2.5 schließt nicht aus, daß auf einem Weg Haltestellen mehrfach vorkommen und damit Schleifen auftreten. Da Wege beliebig viele Schleifen enthalten können, ist  $W_{a,b}^{(\text{ÖV})}$  im Allgemeinen nicht endlich.

**Beispiel 2.5:**

Ein möglicher **Weg**  $w$  von Holzdorf-Ost nach Holzdorf-Mitte (vgl. Beispiel 2.1) ist:

$w = ($	(Ost, Park),	.....	Anmarschweg zur Haltestelle Park,
	(Park, Rathaus, 1a),	.....	Fahrt mit Linie 1 (Richtung) zum Rathaus,
	(1a, 2a, Rathaus),	.....	Umsteigen in Linie 2 (Richtung),
	(Rathaus, Schule, 2a),	.....	Fahrt mit Linie 2 (Richtung) zur Haltestelle Schule,
	(Mitte, Schule) ).	.....	Abmarschweg zum Bezirk Mitte.

Die Menge  $W_{\text{Ost,Mitte}}^{(\text{ÖV})}$  ist nicht endlich, da es die Möglichkeit gibt, über die Haltestellen Rathaus, Bahnhof und Schule beliebig viele Schleifen zu fahren.

## 2.2 Bewertung von Wegen in ÖV-Netzmodellen

Eine der zentralen Aufgaben von Wegsuche-Algorithmen ist die Selektion von geeigneten Routen aus der in aller Regel unendlichen Menge von Wegen in einem ÖV-Netzmodell. Hierzu ist es notwendig, die Güte von Verkehrsverbindungen zu quantifizieren, so daß ein Vergleich von Routen untereinander möglich ist. Zu diesem Zweck wird jedem Weg ein Widerstand zugeordnet, der dem materiellen und immateriellen Aufwand einer Person für die Reise auf diesem Weg entspricht. Dabei werden nur solche Kriterien berücksichtigt, welche quantifizierbar sind und maßgeblichen Einfluß auf die Routenwahl haben. Als signifikante Einflußgrößen gelten die Reisezeit, die Fahrtkosten und der Reisekomfort. Die Bewertungsansätze stammen mehr oder minder aus dem Schriftwerk „Standardisierte Bewertung von Verkehrswegeinvestitionen des ÖPNV“ ([1]), einem Leitfaden für Nutzen-Kosten-Untersuchungen, mit dessen Hilfe unterschiedliche Nahverkehrsprojekte nach einheitlichen Maßstäben beurteilt werden können.

Der Widerstand eines Weges setzt sich aus den einzelnen Widerständen von Fußwegen, Linienabschnitten und Umsteigebeziehungen zusammen. Zur Ermittlung des Widerstands von Verbindungen aller Art ist eine Funktion  $\omega$  definiert, deren Werte einer gewichteten Zeit entsprechen:

**Def. 2.6:** Widerstands- oder Bewertungsfunktion  $\omega$ :

$$\omega: F \cup S \cup T \cup U \cup W^{(\text{ÖV})} \rightarrow \mathbb{R}^+$$

a) für Ein- und Ausbindungsfußwege  $f = (b, h) \in F$ :

$$\omega(f) = g_i \cdot \frac{\text{Entfernung}(f)}{v_{\text{Fuß}}}$$

$\omega(f)$  ist der gewichtete Zeitbedarf für das Zurücklegen des Fußwegs  $f$ , der sich aus der Weglänge und der Fußgängergeschwindigkeit  $v_{\text{Fuß}}$  ergibt. Die Gewichtungsfaktoren  $g_i$  werden unterschieden nach Zugang ( $i = 1$ ) und Abgang ( $i = 2$ ) bezüglich der Haltestelle. Weil Fahrgäste lange Fußwege als großes Hindernis betrachten, sind  $g_1$  und  $g_2$  deutlich höher zu bewerten als die Gewichte für Fahrten mit Verkehrsmitteln (s.u.).

b) für **Linienabschnitte**  $s = (h, \bar{h}, l) \in S$ :

$$\omega(s) = g_3 \cdot \text{Fahrzeit}(s)$$

$\omega(s)$  ist der gewichtete Zeitbedarf für eine Fahrt von Haltestelle  $h$  nach Haltestelle  $\bar{h}$  mit Linie  $l$ .

c) für **Fahrten**  $t = (h_i, h_j, l) \in T$ :

Widerstände von Fahrten ergeben sich aus den Widerständen der einzelnen Linienabschnitte. Sei  $l = (h_1, \dots, h_i, h_{i+1}, \dots, h_j, \dots, h_n) \in L$ ,  $1 \leq i < j \leq n$ . Dann gilt:

$$\omega(t) = \sum_{x=i}^{j-1} \gamma((h_x, h_{x+1}, l))$$

d) für **Umsteigebeziehungen**  $u = (l_1, l_2, h) \in U$ :

$$\omega(u) = g_4 \cdot \text{UWZ} + \text{KV}$$

$$\text{UWZ} = \text{FPA} \cdot \min\{\tau(l_2), \text{UWZ}_{\max}\}$$

Der Widerstand  $\omega(u)$  setzt sich aus der gewichteten Umstiegswartezeit **UWZ** und einem Zeitzuschlag **KV** zusammen. Weil im ÖV-Netzmodell keine detaillierten Fahrpläne verwendet werden, muß zur Berechnung der Wartezeit die Taktfolgezeit  $\tau(l_2)$  der Linie, in die umgestiegen wird, herangezogen werden. Die Wartezeit liegt theoretisch zwischen 0 und  $\tau(l_2)$  und hängt davon ab, wie gut die Fahrpläne der Linien  $l_1$  und  $l_2$  aufeinander abgestimmt sind. Vereinfachend wird für alle Umsteigebeziehungen ein netzglobaler Fahrplanabstimmungsfaktor **FPA** eingesetzt. Der FPA ist der Anteil der Wartezeit an der Taktfolgezeit von  $l_2$ . Der FPA wird meistens auf 0,5 gesetzt, d.h. die Umstiegswartezeit entspricht der halben Taktfolgezeit.

Das ÖV-Netzmodell geht von der Annahme aus, daß die Fahrplanabstimmung zumindest für selten verkehrende Linien gewährleistet ist, so daß es eine obere Schranke **UWZ<sub>max</sub>** für die Umstiegswartezeit gibt.

Außer der Wartezeit gibt es einen Zeitzuschlag für den Komfortverlust **KV** durch Umsteigen, wodurch die grundsätzliche Abneigung von Fahrgästen gegen das Umsteigen modellmäßig erfaßt werden soll.

e) für **Wege** in ÖV-Netzmodellen:

Sei  $w = (f_a, t_0, u_1, t_1, \dots, u_n, t_n, f_b) \in W_{a,b}^{(OV)}$ ,  $a, b \in B$ ,  $n \geq 0$ ,  $\forall 0 \leq i \leq n: t_i = (h_i, h_{i+1}, l_{i+1}) \in T$ .

Dann gilt:

$$\omega(w) = \omega(f_a) + g_5 \cdot \text{EWZ} + \sum_{i=0}^n \omega(t_i) + \sum_{i=1}^n \omega(u_i) + \omega(f_b) \\ + g_6 \cdot \text{Kosten} + g_7 \cdot \max\{\tau(l_i) \mid 1 \leq i \leq n\} + g_8 \cdot \text{SBA}$$

$$\text{EWZ} = 0,5 \cdot \min\{\tau(l_1), \text{EWZ}_{\max}\}$$

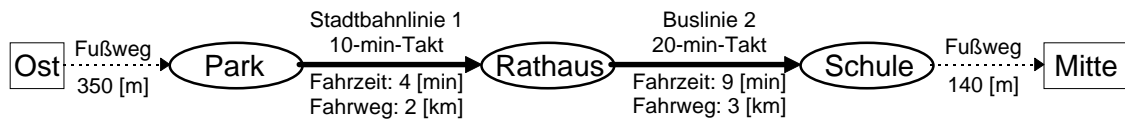
Der Widerstand eines Weges besteht aus

- den Einzelwiderständen von Fußwegen, Fahrten und Umsteigebeziehungen.
- der gewichteten Einstiegswartezeit **EWZ**, die für jeden Weg genau einmal anfällt. Sobald ein Fahrgast zu Fuß die Einstiegshaltestelle erreicht, muß dieser zwischen 0 und  $\tau(l_1)$  Minuten auf Linie  $l_1$  warten. Angenommen, der Anmarsch von Fahrgästen ist zufällig (gleich verteilt), so ergibt sich im Schnitt eine Wartezeit von  $0,5 \cdot \tau(l_1)$ . Verkehrswissenschaftliche Untersuchungen haben ergeben, daß es eine obere Schranke **EWZ<sub>max</sub>** für die Wartezeit gibt, bis zu welcher die Fahrgäste zufällig an der Einstiegshaltestelle erscheinen. Bei selten verkehrenden Linien ( $\tau(l_1) > \text{EWZ}_{\max}$ ) werden die Abfahrtszeiten des Verkehrsmittels bei der Reiseplanung von Fahrgästen berücksichtigt, so daß die durchschnittliche Einstiegswartezeit im Endeffekt nie über  $0,5 \cdot \text{EWZ}_{\max}$  liegt.
- den in einen Zeitwert umgerechneten Fahrtkosten.
- der gewichteten **maximalen Taktfolgezeit** aller benutzten Linien. Die Linie, die am wenigsten häufig verkehrt, stellt einen Engpaß für den gesamten Weg dar, weil auf deren Fahrplan die gesamte Reiseplanung abgestimmt sein muß. Häufig verkehrende Zubringerlinien verbessern die Situation nicht. Die maximale Taktfolgezeit bestimmt letztendlich die Gesamtzahl der Fahrmöglichkeiten auf dem betrachteten Weg und ist mit entscheidend für die Qualität der Verkehrsverbindung.
- dem gewichteten Schnellbahnanteil **SBA**, also dem Anteil des mit Stadt-, S- und U-Bahnen zurückgelegten Weges an der gesamten Weglänge. Auf diese Weise wird der besondere Komfort und die Zuverlässigkeit von Schnellbahnen hervorgehoben.

Der Widerstand eines Weges wird im Folgenden vereinfachend Weglänge genannt.

**Beispiel 2.6:**

Gegeben sei der Weg  $w$  aus Beispiel 2.5 (schematisch dargestellt):



Weitere Angaben:

- Fahrpreis: 3 [DM],
- Fußgängergeschwindigkeit:  $v_{\text{Fuß}} = 70$  [m/min],
- Fahrplanabstimmungsfaktor:  $\text{FPA} = 0,5$ ,
- maximale Einstiegswartezeit:  $\text{EWZ}_{\text{max}} = 13$  [min],
- maximale Umstiegswartezeit:  $\text{UWZ}_{\text{max}} = 30$  [min],
- Komfortverlust (durch Umstieg):  $\text{KV} = 3$  [min],
- Gewichtungsfaktoren:  $g_1 = g_2 = g_4 = g_5 = 2$  (Wartezeit zählt doppelt),  
 $g_3 = 1$  (Fahrzeit zählt einfach),  
 $g_6 = 3$  [min/DM] (Zeitwert für Fahrtkosten),  
 $g_7 = 0,1$  (Anteil von max. Taktfolgezeit)  
 $g_8 = -5$  [min] (bei 100% Schnellbahnanteil).

**Widerstand** von  $w$ :

- Fußwege:  $\omega(f_{\text{Ost}}) = 2 \cdot \frac{350}{70} = 10$  [min],  
 $\omega(f_{\text{Mitte}}) = 2 \cdot \frac{140}{70} = 4$  [min],
- Einstiegswartezeit:  $\text{EWZ} = 0,5 \cdot \min\{10, 13\} = 5$  [min],
- Fahrtwiderstand:  $\omega(t) = 1 \cdot (4 + 9) = 13$  [min],
- Umstiegswiderstand:  $\omega(u) = 2 \cdot 0,5 \cdot \min\{20, 30\} + 3 = 23$  [min],
- maximale Taktfolgezeit:  $\max\{10, 20\} = 20$  [min],
- Schnellbahnanteil:  $\text{SBA} = \frac{2}{2+3} = 0,4$ ,
- Gesamtwiderstand:  $\omega(w) = 10 + 2 \cdot 5 + 13 + 23 + 4 + 3 \cdot 3 + 0,1 \cdot 20 + (-5) \cdot 0,4$   
 $= \underline{69}$  [min].

Man beachte, daß die Zeitangabe keine reale, sondern eine gewichtete Zeit ist. Die tatsächliche Reisezeit ist  $5+5+4+10+9+2 = 35$  [min], was für die zurückgelegte Entfernung von etwa 5,5 [km] sehr schlecht ist, nämlich im Schnitt 9,4 [km/h] von Haus zu Haus (mit dem Fahrrad wäre man wesentlich schneller am Ziel). Dabei sind die Fußwege keinesfalls überdurchschnittlich lange (5+2 [min]). Vielmehr fällt der Umsteigevorgang stark ins Gewicht (10 [min]). Man erkennt, daß der öffentliche Verkehr schon bei geringer Umsteigehäufigkeit stark an Attraktivität verlieren kann.

### 2.3 Distanzgraphen

Zahlreiche, in der Praxis bewährte Algorithmen zur Wegsuche arbeiten auf der Grundlage von Graphen oder einer speziellen Form davon, den Distanzgraphen:

**Def. 2.7:** Gerichteter, endlicher **Distanzgraph**:  $G = (V, E, \gamma)$

$V = \{ 1, 2, \dots, N \}$  : endliche Menge von Knoten ( $N \in \mathbb{IN}$ ),

$E \subseteq V^2$  : Menge von gerichteten Kanten,

$\gamma: E \rightarrow \mathbb{IR}_0^+$  : Bewertungsfunktion für Kanten (Länge von Kanten).

Zwei Knoten  $i, j \in V$  heißen benachbart, falls  $(i, j) \in E$ .

#### Annahme zur Größenordnung von E gegenüber V:

Wegen  $E \subseteq V^2$  gilt für allgemeine Graphen  $|E| \in O(|V|^2)$ . Im Extremfall gilt  $E = V^2$ , d.h. alle Knoten sind paarweise miteinander verbunden. Die im Rahmen der Problemstellung betrachteten Distanzgraphen sind jedoch bei weitem nicht so dicht: Wegen der zweidimensionalen Ausprägung von ÖV-Netzen sind entsprechende Graphen weitgehend planar, d.h. jeder Knoten ist nur mit den Knoten seiner engsten Umgebung verbunden. Dabei können sich einzelne Kanten durchaus schneiden. Trotzdem kann man davon ausgehen, daß es für die Anzahl der Kanten, welche mit einem Knoten verknüpft sind, eine obere Schranke gibt, die auch von den zentral gelegenen Knoten großer Netze nicht überschritten wird. Im Folgenden wird angenommen, daß gilt:

$$|E| \in O(|V|) = O(N).$$

**Def. 2.8:** Menge  $W$  der **Wege** in einem Distanzgraph  $G = (V, E, \gamma)$ :

$$W = \{ (v_1, v_2, \dots, v_n) \subseteq V^2 \times V^* \mid \forall 1 \leq x \leq n-1: (v_x, v_{x+1}) \in E \}$$

Ein Weg muß mindestens einen Start- und einen Endpunkt enthalten.

Menge der **Wege zwischen zwei Knoten**  $i, j \in V$ :

$$W_{i,j} = \{ (v_1, v_2, \dots, v_n) \in W \mid i = v_1 \text{ und } j = v_n \}$$

#### Bemerkung:

Def. 2.8 schließt nicht aus, daß auf einem Weg Knoten mehrfach vorkommen und damit Schleifen auftreten. Da Wege beliebig viele Schleifen enthalten können, ist  $W_{i,j}$  in allgemeinen Distanzgraphen nicht endlich.

**Def. 2.9:** **Weglängen** in einem Distanzgraph  $G = (V, E, \gamma)$ :

Die Länge eines Weges  $w = (v_1, v_2, \dots, v_n) \in W$  ergibt sich aus den Längen der einzelnen Kanten:

$$\gamma(w) = \sum_{x=1}^{n-1} \gamma((v_x, v_{x+1}))$$

**Beispiel 2.7:**

Sei  $G = (V, E, \gamma)$  ein **Distanzgraph** mit folgenden Eigenschaften:

$$V = \{ 1, 2, 3, 4, 5 \},$$

$$E = \{ (1,2), (1,4), (2,4), (2,5), (3,1), (4,1), (4,2), (4,3), (5,4) \},$$

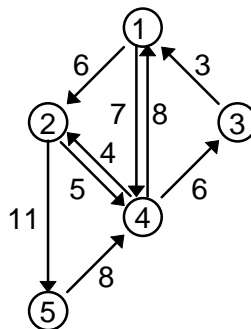
$$\gamma(E) = \{ 6, 7, 5, 11, 3, 8, 4, 6, 8 \}.$$

**Wege** von Knoten 1 nach Knoten 5 sind unter anderem:

$$W_{1,5} = \{ (1,2,5), (1,4,2,5), (1,2,4,2,5), (1,4,1,2,5), (1,4,3,1,2,5), \dots \}$$

Die **Länge** dieser Wege ist:

$$\gamma(W_{1,5}) = \{ 17, 22, 26, 32, 33, \dots \}$$



**Abbildung 2.3:** exemplarischer Distanzgraph mit fünf Knoten und neun Kanten inklusive Längenangaben

**Bemerkung:**

Man erkennt, daß die Berechnung von Weglängen in Distanzgraphen wesentlich einfacher ist als die Bewertung von Routen in ÖV-Netzmodellen. Darum kann es vorteilhaft sein, Wegsuche-Algorithmen auf Graphen anstatt auf Netzmodellen zu definieren. Der Vorteil kann allerdings nur dann ausgenutzt werden, wenn alle im Netzmodell auftretenden Widerstände sich den Kanten fest zuordnen lassen, was leider nicht der Fall ist (siehe nächsten Abschnitt).

## 2.4 Konvertierung von ÖV-Netzmodellen in Distanzgraphen

Um bewährte Wegsuche-Algorithmen der Graphentheorie anwenden zu können, muß jedes ÖV-Netzmodell in einen äquivalenten Distanzgraphen umgewandelt werden.

**Gegeben:** ÖV-Netzmodell  $\ddot{O}V = (B, H, L, F, \tau)$  mit einer Bewertungsfunktion  $\omega$  gemäß Def. 2.6;  $S$  sei die Menge der Linienabschnitte gemäß Def. 2.3 und  $U$  die Menge der Umsteigebeziehungen gemäß Def. 2.2.

**Gesucht:** der zu  $\ddot{O}V$  äquivalente Distanzgraph  $G = (V, E, \gamma)$ .

Er besitzt folgende Eigenschaften:

### Knoten:

$$V = \{ b^{an}, b^{ab} \mid b \in B \}$$

$$\cup \{ h_i^{an} \mid l = (h_1, h_2, \dots, h_n) \in L \text{ und } h \in l \setminus \{h_1\} \}$$

$$\cup \{ h_i^{ab} \mid l = (h_1, h_2, \dots, h_n) \in L \text{ und } h \in l \setminus \{h_n\} \},$$

Ein- und Ausgangspunkt für jeden Verkehrsbezirk;  
Haltestellen splitten: für jede Linie Ein- und Ausgang einer Haltestelle (Erläuterung s.u.);

### Kanten:

$$E = \{ (b^{ab}, h_i^{ab}) \mid (b, h) \in F, h \in l \text{ und } l \in L \}$$

$$\cup \{ (h_i^{an}, b^{an}) \mid (b, h) \in F, h \in l \text{ und } l \in L \}$$

$$\cup \{ (h_i^{ab}, \bar{h}_i^{an}) \mid (h, \bar{h}, l) \in S \}$$

$$\cup \{ (h_{l_1}^{an}, h_{l_2}^{ab}) \mid (l_1, l_2, h) \in U \}$$

$$\cup \{ (h_i^{an}, h_i^{ab}) \mid h \in l \text{ und } l \in L \}.$$

Einbindungsfußwege;  
Ausbindungsfußwege;  
Linienabschnitte;  
Umsteigebeziehungen;  
Haltestellenaufenthalt im Fahrzeug;

**Kantenlängen:** (vgl. Widerstände in Def. 2.6)

- Einbindungsfußwege:  $\gamma((b^{ab}, h_i^{ab})) = g_1 \cdot \frac{\text{Entfernung}((b, h))}{v_{Fu\beta}} + g_5 \cdot \text{EWZ}$ ,  
 $\text{EWZ} = 0,5 \cdot \min\{ \tau(l), \text{EWZ}_{\max} \}$
- Ausbindungsfußwege:  $\gamma((h_i^{an}, b^{an})) = g_2 \cdot \frac{\text{Entfernung}((b, h))}{v_{Fu\beta}}$ ,
- Linienabschnitte:  $\gamma((h_i^{ab}, \bar{h}_i^{an})) = \omega((h, \bar{h}, l)) = g_3 \cdot \text{Fahrzeit}((h, \bar{h}, l))$ ,
- Umsteigebeziehungen:  $\gamma((h_{l_1}^{an}, h_{l_2}^{ab})) = \omega((l_1, l_2, h)) = g_4 \cdot \text{UWZ} + \text{KV}$ ,  
 $\text{UWZ} = \text{FPA} \cdot \min\{ \tau(l_2), \text{UWZ}_{\max} \}$ ,
- Haltestellenaufenthalt:  $\gamma((h_i^{an}, h_i^{ab})) = 0$  (siehe Erläuterungen).

**Erläuterungen zur Konvertierung:****a) Knoten für Bezirke und Kanten für Anbindungsfußwege:**

Da jeder ÖV-Weg an einem Bezirk beginnen und enden muß, wird jeder Verkehrsbezirk in zwei voneinander getrennte Knoten aufgespalten, nämlich einem Eingangspunkt, der nur einmündende Kanten besitzt (Ausbindungsfußwege), sowie einem Ausgangspunkt, der nur ausgehende Kanten besitzt (Einbindungsfußwege). Die Aufspaltung verhindert, daß bei der Wegsuche über einen Bezirkspunkt „hinweggefahren“ wird.

**b) Knoten für Haltestellen und Kanten für Linienabschnitte:**

Da  $E$  eine Menge ist, kann es zwischen zwei Knoten  $v_1, v_2 \in V$  höchstens eine Kante  $(v_1, v_2) \in E$  geben. Dagegen kann es in ÖV-Netzmodellen zwischen zwei Haltestellen mehrere Linienabschnitte mit unterschiedlichem Widerstand geben. Darüberhinaus gibt es innerhalb Haltestellen Umsteigebeziehungen mit individuellen Widerständen. Weil Knoten definitionsgemäß keine Widerstände (Längen) besitzen können, muß bei der Konvertierung jede Haltestelle in mehrere Knoten zerlegt werden (Haltestellenbereich): Für jede Linie, welche die Haltestelle bedient, sind zwei Knoten vorgesehen, jeweils ein Eingangsknoten, in den der vorige Linienabschnitt einmündet, und ein Ausgangsknoten, von dem der folgende Linienabschnitt ausgeht. Die Kante dazwischen repräsentiert den Haltestellenaufenthalt der Linie. Davon ausgenommen sind Start- bzw. Endhaltestellen einer Linie, weil in diesem Fall an der Haltestelle der Eingang bzw. Ausgang fehlt.

**c) Kanten für Haltestellenaufenthalte**

Haltestellenaufenthalte gehen nicht in die Bewertung von Wegen ein, da sie derzeit modellmäßig nicht erfaßt sind, bieten aber eine Erweiterungsmöglichkeit für ÖV-Netzmodelle bei zukünftigen Untersuchungen.

**d) Kanten für Umsteigebeziehungen**

Die Kanten innerhalb eines Haltestellenbereichs (Aufenthalte nicht inbegriffen) repräsentieren Umsteigebeziehungen. Sie verlaufen von einem Eingangspunkt einer Linie zu einem Ausgangspunkt einer anderen Linie. Die Trennung in Ein- und Ausgang verhindert, daß bei der Wegsuche Routen berechnet werden, bei denen innerhalb des Haltestellenbereichs mehrmals hintereinander umgestiegen wird.

### e) Widerstände als Kantenlängen

Der Widerstand von Fußwegen, Linienabschnitten und Umsteigevorgängen des ÖV-Netzmodells ist eine statische Größe und kann deshalb den Kanten des äquivalenten Distanzgraphen als Länge zugewiesen werden. Die Einstiegs-wartezeit (EWZ) ist zumindest im Graph statisch, weil für jede Linie, welche an der Einstiegshaltestelle abfährt, eine besondere Fußwegkante existiert. Die EWZ kann daher den Kanten, welche Einbindungsfußwege repräsentieren, fest zugeordnet werden.

### f) Widerstände, die im Graph nicht berücksichtigt werden können

Die maximale Taktfolgezeit, der Schnellbahnanteil und die Fahrtkosten eines Weges lassen sich nicht auf die Kanten umlegen, weil sie nicht statisch sind. Die genannten Größen können erst berechnet werden, wenn der Weg vollständig bekannt ist. Gibt es beispielsweise Tarifzonen zur Ermittlung des Fahrpreises, dann müßte das Überfahren jeder Zonengrenze einen festen Geldbetrag kosten, was aber in der Praxis nicht der Fall ist, denn beim Wiedereintritt in eine bereits besuchte Zone fallen keine zusätzlichen Kosten an.

### Beispiel 2.8:

Gegeben sei das ÖV-Netzmodell  $\text{ÖV}_{\text{Holzdorf}}$  aus Beispiel 2.1. Linie 1 fährt alle 10 Minuten, Linie 2 alle 20 Minuten. Die Widerstände der zugehörigen Linienabschnitte und Fußwege sind aus Abbildung 2.4 zu entnehmen. Die Randbedingungen (Gewichtungsfaktoren u.ä.) seien dieselben wie in Beispiel 2.6.

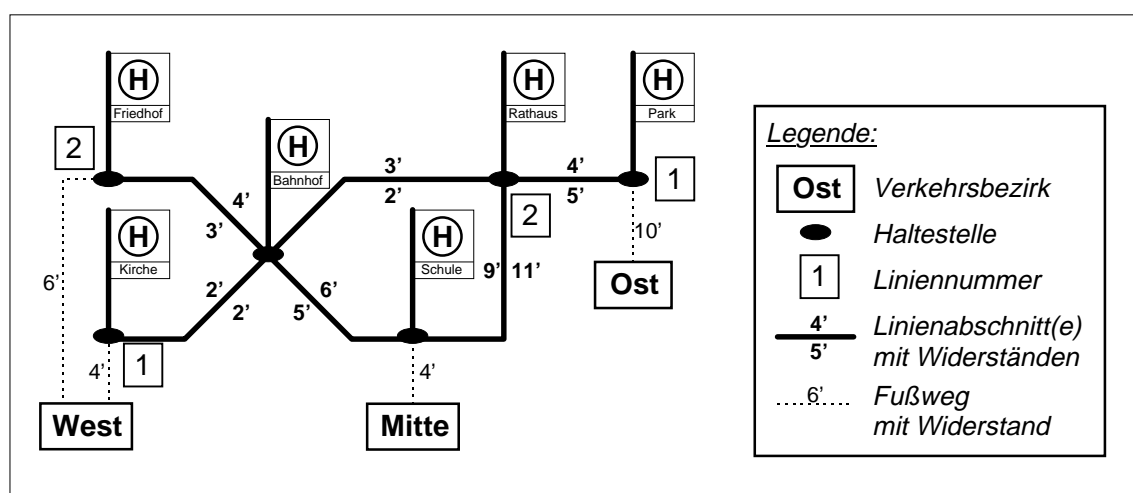


Abbildung 2.4: ÖV-Netzmodell „Holzdorf“ mit Angabe der Widerstände.



**Bemerkungen:**

Man erkennt, daß der Umfang des Netzmodells bei der Konvertierung deutlich zunimmt, vor allem aufgrund der Zerlegung von Haltestellen. Andererseits bietet die feine Zergliederung eine feste Zuordnung von Einstiegs- und Umstiegswiderständen zu Kanten und damit einen effizienteren Zugriff auf Widerstände während der Routensuche.

Die Tatsache, daß von jedem Abfahrtsknoten einer Haltestelle genau eine Kante ausgeht, läßt den Verdacht aufkommen, daß man die besagten Knoten einsparen kann, indem man jeweils die einmündenden Kanten auf das Ziel der ausgehenden Kante umleitet. Dabei würde jedoch das Problem auftreten, daß die von den Bezirken ausgehenden Fußwege nicht mehr zu den eingebundenen Haltestellen führen würden, was wiederum Fehlinterpretationen bei der Auswertung berechneter Wege zur Folge hätte (s.u.).

**2.5 Abbildung von Wegen in Distanzgraphen auf ÖV-Wege**

Wird die Routensuche in einem Distanzgraph  $G$  durchgeführt, so ergeben sich Wege in Form von Knotenfolgen, welche zunächst bezüglich des zugehörigen ÖV-Netzmodells  $\ddot{O}V$  interpretiert werden müssen. Da bei der Konvertierung aus jedem Bestandteil von  $\ddot{O}V$  mindestens ein Knoten bzw. eine Kante in  $G$  entsteht, läßt sich umgekehrt jeder Knoten und jede Kante aus  $G$  eindeutig einem Bestandteil von  $\ddot{O}V$  zuordnen. Auf dieser Grundlage läßt sich eine eindeutige Abbildung von Wegen in  $G$  auf Wege in  $\ddot{O}V$  konstruieren, sofern Start- und Zielknoten zu einem Bezirk gehören (in  $\ddot{O}V$  müssen Wege in Bezirken beginnen und enden, in  $G$  nicht).

**Gegeben:**  $\ddot{O}V$ -Netzmodell  $\ddot{O}V = (B, H, L, F, \tau)$  mit der Menge  $W^{(\ddot{O}V)}$  der Wege gemäß Def. 2.5 sowie der zugehörige Distanzgraph  $G = (V, E, \gamma)$  mit der Menge  $W$  der Wege gemäß Def. 2.8. Weiterhin sei ein  $w \in W$  gegeben,  $w = (v_1, v_2, \dots, v_m)$ ,  $m \geq 2$ , wobei  $v_1 = b_s^{ab}$  und  $v_m = b_z^{an}$  mit  $b_s, b_z \in B$ .

**Gesucht:** Der zu  $w$  äquivalente Weg  $w' \in W^{(\ddot{O}V)}$ .  $w'$  ist von der Form  
 $w' = (f_s, t_0, u_1, t_1, \dots, u_n, t_n, f_z)$ , wobei  $f_s, f_z \in F$ ,  $\forall 0 \leq i \leq n: t_i \in T$ ,  $\forall 1 \leq i \leq n: u_i \in U$ .

**Vorbereitung:**

Da Bezirksknoten entweder keine einmündenden oder ausgehenden Kanten besitzen, müssen alle Zwischenknoten in  $w$  zu Haltestellen gehören, d.h.

$$\forall 2 \leq i \leq m-1: \exists h \in H, \exists l \in L: v_i = h_l^x, x \in \{an, ab\}$$

Weil alle Kanten vom Startbezirk  $b_s^{ab}$  aus zu Abfahrtsknoten führen, gilt  $v_2 = h_1^{S,ab}$  für ein geeignetes  $h^S \in H$ ,  $l \in L$  und damit  $f_s = (b_s, h^S)$ . Entsprechend beginnen alle Kanten, welche in den Zielbezirk  $b_z^{an}$  einmünden, bei Ankunfts-knoten, also gilt  $v_{m-1} = h_1^{Z,an}$  für ein geeignetes  $h^Z \in H$ ,  $l \in L$  und damit  $f_z = (b_z, h^Z)$ .

$G$  ist so konstruiert, daß Kanten zwischen Haltestellenknoten entweder von einem Abfahrtsknoten zu einem Ankunfts-knoten oder umgekehrt verlaufen. Daraus folgt:

$$\begin{aligned} \forall 2 \leq i \leq m-1, \text{ i gerade: } & \exists h \in H, \exists l \in L: v_i = h_l^{ab}, \\ \forall 2 \leq i \leq m-1, \text{ i ungerade: } & \exists h \in H, \exists l \in L: v_i = h_l^{an}. \end{aligned}$$

Da  $v_{m-1}$  ein Ankunfts-knoten ist, muß  $m$  geradzahlig sein. Außerdem gilt  $m \geq 4$ , weil es keine Kanten zwischen Bezirken gibt. Die Anzahl der besuchten Haltestellen ist  $m/2$ .

### Ablauf:

<pre> n:= 0; h*:= h<sup>Z</sup>; l*∈L ergibt sich aus v<sub>2</sub> = h<sup>*</sup><sub>l*</sub><sup>ab</sup>; FOR i:= 3 TO m-1 DO   Falls i ungerade,     l'∈H ergibt sich aus v<sub>i</sub> = h<sup>'</sup><sub>l'</sub><sup>an</sup> für ein l'∈L;   ansonsten     l'∈L ergibt sich aus v<sub>i</sub> = h<sup>'</sup><sub>l'</sub><sup>ab</sup> für ein h'∈H;     Falls l' ≠ l*,       t<sub>n</sub>:= (h*, h', l*);       n:= n + 1;       u<sub>n</sub>:= (l*, l', h*);       l*:= l'; END; t<sub>n</sub>:= (h*, h', l*); </pre>	<pre> Annahme für Umsteigehäufigkeit; Einstiegshaltestelle; erste benutzte Linie;  Zwischenknoten aufzählen; (v<sub>i-1</sub>, v<sub>i</sub>) ist Linienabschnitt; aktuelle Haltestelle h' ermitteln;  aktuelle Linie l' ermitteln; (v<sub>i-1</sub>, v<sub>i</sub>) ist Umsteigebeziehung; Fahrt registrieren; Umsteigehäufigkeit erhöhen; Umsteigebeziehung registrieren; zuletzt benutzte Linie festhalten;  letzte Fahrt registrieren (h' letzte Haltestelle, da m-1 ungerade); </pre>
---	--

### Beispiel 2.9:

Gegeben sei ein Weg  $w$  im Distanzgraph  $G$  aus Beispiel 2.8:

$$w = (\text{West}^{ab}, \text{Kirche}_{1b}^{ab}, \text{Bhf}_{1b}^{an}, \text{Bhf}_{2b}^{ab}, \text{Schule}_{2b}^{an}, \text{Mitte}^{an}).$$

Der zu  $w$  äquivalente Weg  $w'$  im ÖV-Netzmodell ist von der Form  $w' = (f_s, t_0, u_1, t_1, f_z)$ ,

<pre> mit f<sub>s</sub> = (West, Kirche),      t<sub>0</sub> = (Kirche, Bhf, 1b),      u<sub>1</sub> = (1b, 2b, Bhf),      t<sub>1</sub> = (Bhf, Schule, 2b),      f<sub>z</sub> = (Mitte, Schule). </pre>	<pre> Anmarschweg zur Haltestelle Kirche, Fahrt mit Linie 1 (Gegenrichtung) zum Bahnhof, Umsteigen in Linie 2 (Gegenrichtung), Fahrt mit Linie 2 (Gegenrichtung) zur Schule, Abmarschweg zum Bezirk Mitte. </pre>
--	---

## 2.6 Berücksichtigung von Parallelverkehr

### 2.6.1 Motivation

In einem ÖV-Netzmodell kommt es vor allem in den Kernzonen häufig vor, daß zwischen zwei Haltestellen mehrere Linienabschnitte existieren, was als Parallelverkehr im weiteren Sinne bezeichnet wird, d.h. auf einem Streckenabschnitt verkehren mehrere Linien parallel. Falls die Linienabschnitte unterschiedliche Widerstände besitzen, so ist anzunehmen, daß Fahrgäste bei der Routenwahl Linienabschnitte mit kleinen Widerständen gegenüber den anderen bevorzugen. Dies gilt nicht für die Konstellation, daß parallele Linienabschnitte denselben Widerstand besitzen, weil sich dann die Linien wechselseitig vertreten können. Die parallel geführten Linien werden aus Sicht der Fahrgäste als eine einzige Linie betrachtet, was als Parallelverkehr im engeren Sinne bezeichnet wird.

**Def. 2.10: Linienbündel oder Parallelverkehr** (im engeren Sinne) zwischen zwei Haltestellen  $h_1, h_2 \in H$ :

Sei  $K = \{ l \in L \mid (h_1, h_2, l) \in S \text{ und } \omega((h_1, h_2, l)) = r \}$  für ein festes  $r \in \mathbb{R}^+$ .

$K$  ist Parallelverkehr, falls  $|K| \geq 2$ .

Parallelverkehr ist eine mindestens zweielementige Menge von Linien, welche zwischen  $h_1$  und  $h_2$  bei gleichem Widerstand  $r \in \mathbb{R}^+$  verkehren:

Bei Parallelverkehr sind die Nummern der beteiligten Linien aus Sicht des Fahrgastes unerheblich. Weil die Linien gleichwertige Alternativen darstellen, lassen sie sich zu einer Kombination bündeln, welche vom Fahrgast als eine einzige Linie betrachtet wird. Der Widerstand des Bündels ist derselbe wie jener der einzelnen Linien. Dagegen ist die Taktfolgezeit der Kombination besser als die Taktfolgezeit jeder beteiligten Linie:

**Def. 2.11: Taktfolgezeit eines Linienbündels:**

Sei  $K = \{l_1, l_2, \dots, l_n\}$  ein Linienbündel ( $\forall 1 \leq i \leq n: l_i \in L$ ). Dann gilt:

$$\tau(K) = \left( \sum_{i=1}^n \tau(l_i)^{-1} \right)^{-1} \quad [\text{min}].$$

Der Reziprokwert der Taktfolgezeit ist die Bedienungshäufigkeit [Fahrten/min], welche über die parallel verkehrenden Linien aufsummiert werden kann.

**Beispiel 2.10:**

Gegeben sei ein Ausschnitt eines ÖV-Netzmodells  $\text{ÖV}_{\text{Parallel}}$  mit Widerständen gemäß Abbildung 2.6. Alle Linien verkehren im 12-Minuten-Takt. Die Randbedingungen (Gewichtungsfaktoren u.ä.) seien dieselben wie in Beispiel 2.6.

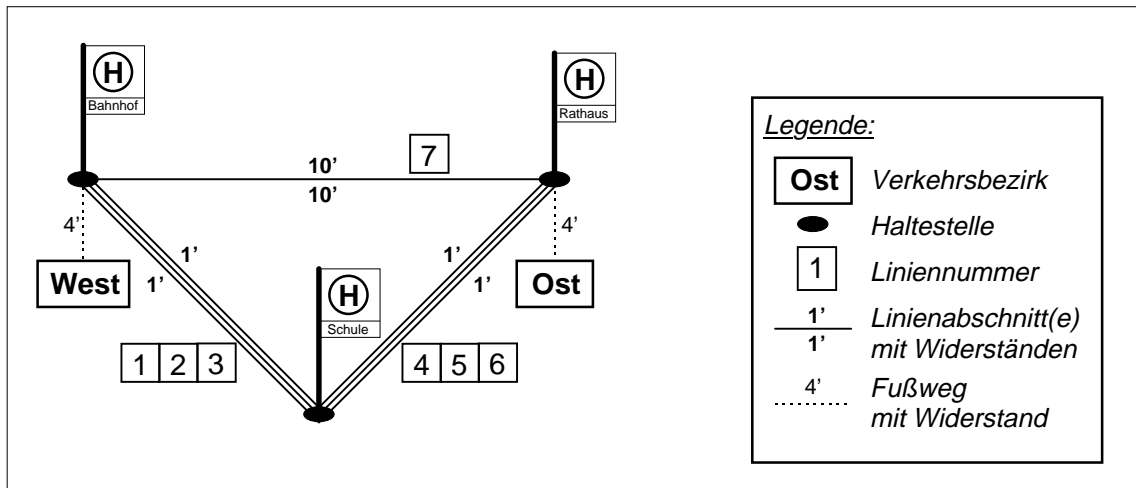


Abbildung 2.6: Ausschnitt aus einem ÖV-Netzmodell mit Parallelverkehr

Mögliche Wege von West nach Ost sind unter anderem:

mit Linie	Einstiegs-widerstand	Fahrzeit gesamt	Umstiegs-widerstand	Summe
7	2·6'	10'	0'	16'
z.B. 1, dann 4	2·6'	3'	2·6' + 3'	24'

Die Direktverbindung vom Bahnhof zum Rathaus schneidet trotz der großen Fahrzeit deutlich besser ab als jede Alternative über die Schule. Dementsprechend würde Linie 7 wesentlich mehr Fahrgäste auf sich ziehen als die Linien 1 bis 6, was jedoch unrealistisch wäre. Bei Berücksichtigung von Parallelverkehr ergibt sich ein anderes Bild:  $K_1 = \{1, 2, 3\}$  und  $K_2 = \{4, 5, 6\}$  sind Linienbündel gemäß Def. 2.10. Die Taktfolgezeit der Bündel ist:

$$\tau(K_1) = \tau(K_2) = (12^{-1} + 12^{-1} + 12^{-1})^{-1} = 4 \text{ [min]}$$

Damit ergeben sich zwei sinnvolle Wege von West nach Ost:

mit Linie / Bündel	Einstiegs-widerstand	Fahrzeit gesamt	Umstiegs-widerstand	Summe
$K_1$ , dann $K_2$	2·2'	3'	2·2' + 3'	14'
7	2·6'	10'	0'	16'

**Bemerkung:**

Die Taktfolgezeit wird durch Überlagerung von Linien deutlich verbessert. Dadurch verringern sich Einstiegs- und Umstiegswartezeiten, was die Attraktivität der Verkehrsverbindung erhöht. Der Gewinn kann so groß sein, daß Verbindungen, die wegen zu großer Widerstände aus Sicht der Fahrgäste nicht in Frage kommen, erst durch Bündeln von Linien relevant für die Verkehrsumlegung werden. Weil Parallelverkehr offensichtlich entscheidenden Einfluß auf die Menge der gesuchten Wege haben kann, ist dessen Berücksichtigung bei der Wegsuche notwendig.

**2.6.2 Verfahrensprinzipien**

Es gibt drei Methoden, Parallelverkehr zu berücksichtigen:

- 1.) bei der Konvertierung von ÖV-Netzmodellen in Distanzgraphen,
- 2.) während der Wegsuche oder
- 3.) nach Abschluß der Wegsuche.

zu 1.)

Zur Berücksichtigung von Parallelverkehr wird das Konvertierungsverfahren aus Abschnitt 2.4 um Regeln ergänzt, mit denen ein Distanzgraph diesbezüglich angepaßt werden kann. Linienbündel werden im Graph durch eigene Kanten repräsentiert, welche über besondere Zu- und Abgänge mit dem übrigen Netz verbunden sind. Vorteil ist, daß der Anpassungsaufwand ausschließlich vor der Wegsuche anfällt und die Wegsuche-Algorithmen nicht verändert werden müssen. Nachteil ist allerdings der größere Speicherplatzbedarf für die angepaßten Graphen.

zu 2.)

Bei der Wegsuche wird geprüft, ob es beim Ein- oder Umsteigen in eine Linie gleichwertige Alternativen gibt. In solchen Fällen wird ein besonderer Ein- bzw. Umstiegswiderstand berechnet und die Menge der parallel verlaufenden Linien registriert. Da die Suche nach Linienbündeln zeitaufwendig ist, sollte dies vor Beginn der Wegsuche durchgeführt werden. Die Berücksichtigung von Parallelverkehr in der Datenstruktur, d.h. die Anpassung von Distanzgraphen ist mehr oder weniger unvermeidlich.

zu 3.)

Nach Abschluß der Wegsuche erkennt man Parallelverkehr daran, daß sich mehrere Wege zwischen zwei Bezirken nur durch Liniennummern auf einigen Streckenabschnitten unterscheiden. Auf diese Weise müssen alle Wege eines Bezirkspaares auf Parallelverkehr geprüft und nachträglich gebündelt werden. Leider ist die Korrektheit dieses Verfahrens nicht gewährleistet, weil ein Teil der Routen, welche zum Bündeln herangezogen werden müssen, aufgrund zu hoher Widerstände eventuell nicht berechnet oder verworfen wurden.

Zusammenfassend kann man feststellen, daß Methode (1) am geeignetsten ist. Das unten erläuterte Verfahren ist als Ergänzung zum Konvertierungsverfahren aus Abschnitt 2.4 zu betrachten.

### **2.6.3 Bündeln von Parallelverkehr**

**Gegeben:** beliebiges ÖV-Netzmodell  $\ddot{O}V = (B, H, L, F, \tau)$  mit einer Bewertungsfunktion  $\omega$  gemäß Def. 2.6.  $U$  sei die Menge der Umsteigebeziehungen gemäß Def. 2.2 und  $S$  die Menge der Linienabschnitte gemäß Def. 2.3. Sei  $G = (V, E, \gamma)$  der zu  $\ddot{O}V$  äquivalente Distanzgraph laut Abschnitt 2.4.

**Gesucht:** Anpassungen in  $G$  zur Berücksichtigung von Parallelverkehr nach Def. 2.10.  $G$  wird in zwei Schritten angepaßt.

#### **1. Schritt:**

##### **Grundprinzip:**

Zunächst werden alle Streckenabschnitte des ÖV-Netzmodells auf Parallelverkehr untersucht: Für alle Linienabschnitte mit gemeinsamem Widerstand werden Kanten hinzugefügt, welche Linienbündel repräsentieren. Alle Zugänge zu den beteiligten Linienabschnitten werden auf die Kante des Bündels umgeleitet (Sperrung der zugehörigen Knoten), mit Ausnahme von Haltestellenaufenthalte, da Fahrgäste, die mit einer der beteiligten Linien ankommen, nicht vom Parallelverkehr profitieren können, sondern dieselbe Linie weiterbenutzen müssen. Lediglich Einsteiger und Umsteiger können Linienbündel benutzen. Weil deren Zugangswiderstände (Wartezeiten) verschieden sein können, müssen alle Bündel dupliziert werden, d.h. Einsteiger und Umsteiger werden über verschiedene Kanten geleitet (in Beispiel 2.11, Teil 2 ist dieser Sachverhalt näher erläutert).

**Ablauf:**

Sei  $\psi: V \rightarrow \{\text{frei, gesperrt}\}$  die Funktion, die alle Startknoten von Linienabschnitten kennzeichnet, welche Teil eines Linienbündels sind und somit gesperrt werden müssen. Da zu Beginn keine Bündel existieren, gilt:

$$\forall h \in H: \forall l \in L \text{ mit } h \in l: \psi(h_l^{ab}) := \text{frei};$$

$\forall h, \bar{h} \in H$  mit  $h \neq \bar{h}$ : Suche Linienabschnitte mit gleichem Widerstand:

Für ein  $r \in \mathbb{R}^+$  sei  $K = \{ l \in L \mid \exists s = (h, \bar{h}, l) \in S \text{ mit } \omega(s) = r \}$ .

Falls  $|K| \geq 2$  (Definition von Parallelverkehr), verfare wie folgt:

$\forall l \in K: \psi(h_l^{ab}) := \text{gesperrt},$	⋮ Zugänge zu einzelnen Linien ent-
$E := E \setminus \{ (v, h_l^{ab}) \mid v \in V \setminus \{h_l^{an}\} \text{ und } l \in K \},$	⋮ fernen außer Aufenthaltskante;
falls $\exists b \in B$ mit $(b, h) \in F$ ,	⋮ falls <b>Einsteiger</b> bei $h$ vorhanden,
$V := V \cup \{ h_{K_e}^{ab}, \bar{h}_{K_e}^{an} \},$	⋮ neue Kante (incl. Knoten) als Linien-
$E := E \cup \{ (h_{K_e}^{ab}, \bar{h}_{K_e}^{an}) \}$	⋮ bündel $K$ nur für Einsteiger;
$\cup \{ (b^{ab}, h_{K_e}^{ab}) \mid (b, h) \in F \},$	⋮ Einsteiger auf $K$ bei $h$ ;
$\gamma((h_{K_e}^{ab}, \bar{h}_{K_e}^{an})) := r,$	⋮ gemeinsamer Widerstand $r \in \mathbb{R}^+$ ;
$\forall (b, h) \in F: \gamma((b^{ab}, h_{K_e}^{ab})) := g_1 \cdot \frac{\text{Entfernung}((b, h))}{V_{Fu\beta}} + g_5 \cdot \text{EWZ}(K),$	⋮
$\psi(h_{K_e}^{ab}) := \text{frei},$	⋮ Zugang zu Linienbündel freigeben;
falls $\exists h_l^{an} \in V$ mit $l \notin K$ ,	⋮ falls <b>Umsteiger</b> bei $h$ vorhanden,
$V := V \cup \{ h_{K_u}^{ab}, \bar{h}_{K_u}^{an} \},$	⋮ neue Kante (incl. Knoten) als Linien-
$E := E \cup \{ (h_{K_u}^{ab}, \bar{h}_{K_u}^{an}) \}$	⋮ bündel $K$ nur für Umsteiger;
$\cup \{ (h_l^{an}, h_{K_u}^{ab}) \mid h \in l, l \in L \setminus K \},$	⋮ Umsteiger auf $K$ bei $h$ ;
$\gamma((h_{K_u}^{ab}, \bar{h}_{K_u}^{an})) := r,$	⋮ Linienbündel $K$ ;
$\forall h_l^{an} \in V$ mit $l \notin K: \gamma((h_l^{an}, h_{K_u}^{ab})) := g_4 \cdot \text{UWZ}(K) + KV,$	⋮
$\psi(h_{K_u}^{ab}) := \text{frei}.$	⋮ Zugang zu Linienbündel freigeben;

**Bemerkungen:**

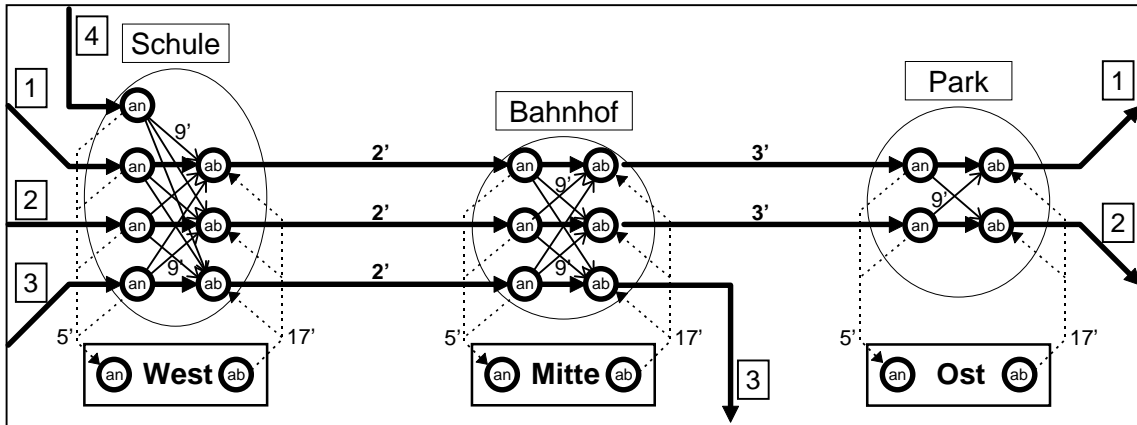
Weiterführende Kanten von  $K$  bei  $\bar{h}$  werden erst im 2. Schritt betrachtet, wenn alle Linienbündel im Graph vorhanden sind. Für die Wartezeiten ergibt sich laut Def. 2.6:

- Einstiegswartezeit:  $\text{EWZ}(K) = 0,5 \cdot \min\{ \tau(K), \text{EWZ}_{\max} \},$
- Umstiegswartezeit:  $\text{UWZ}(K) = \text{FPA} \cdot \min\{ \tau(K), \text{UWZ}_{\max} \},$  wobei  $K \subseteq L$ .

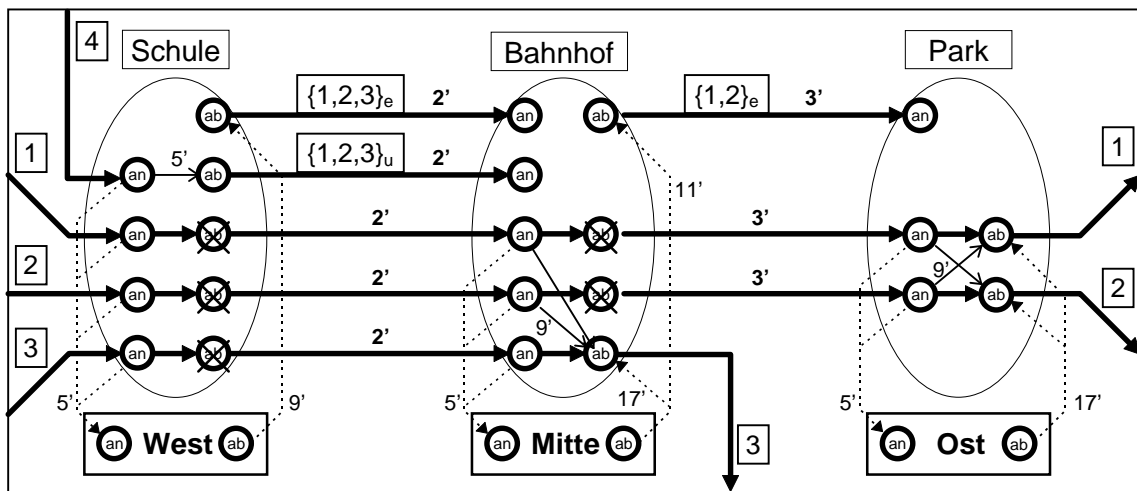
Der entscheidende Fortschritt ist die Taktfolgezeit  $\tau(K)$  des Linienbündels (laut Def. 2.11), welche die Wartezeit deutlich vermindern kann. Fahrgäste, die bei  $h$  ein- oder umsteigen und bei  $\bar{h}$  aussteigen, profitieren auf alle Fälle vom Parallelverkehr. Dagegen müssen Personen, welche mit einer bestimmten, am Bündel beteiligten Linie bei  $h$  durchfahren, dieselbe Linie weiterverwenden. Aus diesem Grund können die gesperrten Kanten nicht abgebaut werden.

**Beispiel 2.11:**

In den folgenden Abbildungen ist ein Ausschnitt eines ÖV-Netzmodells mit Parallelverkehr dargestellt. Widerstände sind nur an Stellen aufgeführt, an denen sie von Belang sind. Alle beteiligten Linien verkehren im 12-Minuten-Takt. Der Fahrplanabstimmungsfaktor FPA sei hier 0,25 (betrifft nur Umsteigende).



**Abbildung 2.7:** Zustand ohne Berücksichtigung von Parallelverkehr: Zwischen den Haltestellen Schule und Bahnhof verkehren die Linien 1, 2 und 3 parallel, zwischen Bhf und Park die Linien 1 und 2 (Legende: vgl. Abbildung 2.5). Alle Fußwege besitzen den Widerstand 5'. Der Einstiegswiderstand ist überall  $2 \cdot 0,5 \cdot \min\{12', 13'\} = 12'$ , der Umstiegswiderstand  $2 \cdot 0,25 \cdot \min\{12', 30'\} + 3' = 9'$ .



**Abbildung 2.8:** Bündeln von Parallelverkehr, 1. Schritt: Wegen  $\tau(\{1,2,3\}) = 4'$  (Taktfolgezeit des Bündels) ist der Einstiegswiderstand auf  $\{1,2,3\}$   $2 \cdot 0,5 \cdot \min\{4', 13'\} = 4'$  und der Umstiegswiderstand  $2 \cdot 0,25 \cdot \min\{4', 30'\} + 3' = 5'$ . Durchfahrende dürfen die Linie nicht wechseln (Sperren von Knoten). Am Bahnhof gibt es nur Einsteiger auf  $\{1,2\}$ . Wegen  $\tau(\{1,2\}) = 6'$  ist der Widerstand  $2 \cdot 0,5 \cdot \min\{6', 13'\} = 6'$ . Fehlende Anschlüsse werden im 2. Schritt ergänzt.

**2. Schritt:****Grundprinzip:**

Für einige Streckenabschnitte des ÖV-Netzmodells seien nun spezielle Kanten repräsentativ für Parallelverkehr vorhanden. Die Zugänge werden im 1. Schritt entworfen. Im 2. Schritt werden die Fortsetzungsmöglichkeiten für alle Bündel untersucht. Hierzu werden an jeder Haltestelle alle ankommenden Linienbündel mit weiterführenden Linien bzw. Bündeln verknüpft.

1) **Bündel K mit Linie l** verknüpfen:

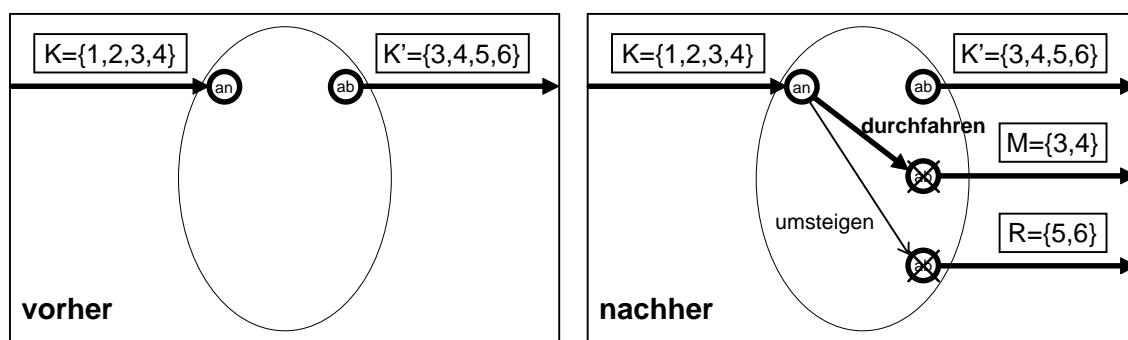
Die Verbindung ist unzulässig, wenn l Teil eines Bündels K' ist (gesperrter Zugang), weil dann das Bündel K' als Verbindungspartner für K herangezogen wird (siehe Fall 2). Ist l dagegen unabhängig, dann treten zwei Fälle auf:

- $l \in K$  entspricht dem (evtl. teilweisen) Ende des Parallelverkehrs und damit einer Einschränkung des Verkehrsangebots, was sich als zusätzlicher Widerstand beim Übergang von K auf l bemerkbar macht (Differenz der Wartezeiten von l und K, siehe Abbildung 2.10, Haltestelle Bahnhof).
- $l \notin K$  entspricht einem gewöhnlichen Umsteigevorgang von K nach l.

2) **Bündel K mit Bündel K'** verknüpfen: K' wird zerlegt in

- $M = K' \cap K$ , also alle durchgehenden Linien von K nach K', was einer Einschränkung des Verkehrsangebots analog zu Fall 1a entspricht ( $M \subseteq K$ ).
- $R = K' - K$ , also alle Linien, in welche man von K aus umsteigen muß. Dies ergibt einen gewöhnlichen Umsteigevorgang von K nach R (vgl. Fall 1b).

Durch Zerlegung von K' können neue Bündel entstehen, welche über mehrere Streckenabschnitte probagiert werden müssen (siehe Abbildung 2.9). Zu einem Linienbündel K müssen im schlimmsten Fall  $2^{|K|}$  Teilbündel konstruiert werden.



**Abbildung 2.9:** K' wird zerlegt in Hilfsbündel für Durchfahrende bzw. Umsteigende von K. Das Einfügen neuer Bündel verursacht weitere Untersuchungen an der Zielhaltestelle von M bzw. R.

**Ablauf:**

Sei  $\varphi: H \rightarrow \{\text{unerledigt, erledigt}\}$  die Funktion, die Haltestellen kennzeichnet, welche noch zu bearbeiten sind, was zu Beginn auf alle zutrifft:

$$\forall h \in H: \varphi(h) := \text{unerledigt};$$

$\forall h \in H$ mit $\varphi(h) =$ unerledigt: $\varphi(h) :=$ erledigt , $\forall K \subseteq L$ mit $ K  \geq 2$ und $\exists h_{Kx}^{an} \in V, x \in \{e, u\}$ : $\forall b \in B$ mit $(b, h) \in F: \forall x \in \{e, u\}: \exists h_{Kx}^{an} \in V$ : $E := E \cup \{ (h_{Kx}^{an}, b^{an}) \},$ $\gamma((h_{Kx}^{an}, b^{an})) := g_2 \cdot \frac{\text{Entfernung}((b, h))}{V_{Fu\beta}};$ $\forall l \in L: \exists h_l^{ab} \in V$ mit $\psi(h_l^{ab}) =$ frei: falls $\exists h_{Ke}^{an} \in V,$ $E := E \cup \{ (h_{Ke}^{an}, h_l^{ab}) \},$ $\gamma((h_{Ke}^{an}, h_l^{ab})) := \begin{cases} g_5 \cdot (EWZ(l) - EWZ(K)), & \text{falls } l \in K \\ g_4 \cdot UWZ(l) + KV, & \text{falls } l \notin K \end{cases}$ falls $\exists h_{Ku}^{an} \in V,$ $E := E \cup \{ (h_{Ku}^{an}, h_l^{ab}) \},$ $\gamma((h_{Ku}^{an}, h_l^{ab})) := \begin{cases} g_4 \cdot (UWZ(l) - UWZ(K)), & \text{falls } l \in K \\ g_4 \cdot UWZ(l) + KV, & \text{falls } l \notin K \end{cases}$ $\forall \bar{K} \subseteq L$ mit $ \bar{K}  \geq 2$ und $\exists h_{Kx}^{ab} \in V$ mit $\psi(h_{Kx}^{ab}) =$ frei, $x \in \{e, u\}$ : sei $s = (h, \bar{h}, l) \in S$ mit $l \in \bar{K}$ beliebig, sei $M = K \cap \bar{K}, R = \bar{K} - M$ falls $M \neq \{ \},$ $\forall x \in \{e, u\}: \exists h_{Kx}^{an} \in V:$ falls $\nexists h_{Mx}^{ab} \in V,$ $V := V \cup \{ h_{Mx}^{ab}, \bar{h}_{Mx}^{an} \},$ $E := E \cup \{ (h_{Mx}^{ab}, \bar{h}_{Mx}^{an}) \},$ $\gamma((h_{Mx}^{ab}, \bar{h}_{Mx}^{an})) := \omega(s),$ $\psi(h_{Mx}^{ab}) :=$ gesperrt, $\varphi(\bar{h}) :=$ unerledigt, $E := E \cup \{ (h_{Kx}^{an}, h_{Mx}^{ab}) \},$ $\gamma((h_{Kx}^{an}, h_{Mx}^{ab})) := \begin{cases} g_5 \cdot (EWZ(M) - EWZ(K)), & \text{falls } x = e \\ g_4 \cdot (UWZ(M) - UWZ(K)), & \text{falls } x = u \end{cases}$ falls $R \neq \{ \},$ falls $\nexists h_{Ru}^{ab} \in V,$ $V := V \cup \{ h_{Ru}^{ab}, \bar{h}_{Ru}^{an} \},$ $E := E \cup \{ (h_{Ru}^{ab}, \bar{h}_{Ru}^{an}) \},$ $\gamma((h_{Ru}^{ab}, \bar{h}_{Ru}^{an})) := \omega(s),$ $\psi(h_{Ru}^{ab}) :=$ gesperrt, $\varphi(\bar{h}) :=$ unerledigt, $\forall x \in \{e, u\}: \exists h_{Kx}^{an} \in V:$ $E := E \cup \{ (h_{Kx}^{an}, h_{Ru}^{ab}) \},$ $\gamma((h_{Kx}^{an}, h_{Ru}^{ab})) := g_4 \cdot UWZ(R) + KV.$	unerledigte Haltestellen bearbeiten; vorläufig, evtl. später widerrufen; bei h <b>ankommende Linienbündel</b> ; für alle <b>Aussteiger</b> von K bei h: Abgangsfußwege von K einfügen, Abgangswiderstand gemäß Def.2.6a; von h <b>ausgehende Linienabschnitte</b> , welche zu keinem Bündel gehören; falls <b>Bündel für Einsteiger</b> einmündet, Bündel K mit Linie l verbinden; Durchfahrende bei h; Umsteiger bei h; falls <b>Bündel für Umsteiger</b> einmündet, Bündel K mit Linie l verbinden; Durchfahrende bei h; Umsteiger bei h; von h <b>ausgehende Linienbündel</b> außer Hilfsbündel; Nachbarhaltestelle $\bar{h}$ ermitteln; gemeinsame und restliche Linien; Hilfsbündel M für <b>Durchfahrende</b> bei h; ankommende Linienbündel; M fehlt noch; Start- und Endknoten für neue Kante repräsentativ für M; Widerstand von Linienabschnitt s; Zugang zu Hilfsbündel sperren; Überarbeitung von $\bar{h}$ notwendig; Durchfahrende von K auf M; Hilfsbündel R für <b>Umsteiger</b> bei h; R fehlt noch; Start- und Endknoten für neue Kante repräsentativ für R; Widerstand von Linienabschnitt s; Zugang zu Hilfsbündel sperren; Überarbeitung von $\bar{h}$ nötig ankommende Linienbündel; Umsteiger von K auf R:
--	--

**Beispiel 2.11:** (Fortsetzung)

1) An der **Schule** gibt es keine ankommenden Linienbündel, also keine zusätzlichen Verknüpfungen.

2) Am **Bahnhof** trennt sich Linie I = 3 vom Bündel K = {1,2,3}.

a) Widerstand von K auf I ( $\tau(3) = 12'$ ,  $\tau(\{1,2,3\}) = 4'$ ):

- für Einsteiger (an der Schule):  $2 \cdot (0,5 \cdot \min\{12', 13'\} - 0,5 \cdot \min\{4', 13'\}) = 8'$
- für Umsteiger (von Linie 4):  $2 \cdot (0,25 \cdot \min\{12', 30'\} - 0,25 \cdot \min\{4', 30'\}) = 4'$

Damit ist der niedrige Zugangswiderstand an der Schule für Personen, die mit Linie 3 über den Bahnhof hinausfahren, aufgehoben, entsprechend der Tatsache, daß vom Parallelverkehr K nur Personen profitieren, welche am Bahnhof aussteigen. Da Einstiegs- und Umstiegswiderstände auf K verschieden sind, ergeben sich auch bei den Abgängen von K auf einzelne Linien oder Teilbündel Unterschiede zwischen Einsteigern und Umsteigern. Aus diesem Grund sind für K zwei Kanten erforderlich.

b) Widerstand von K auf  $K' = \{1,2\}$  ( $\tau(K') = 6'$ ):

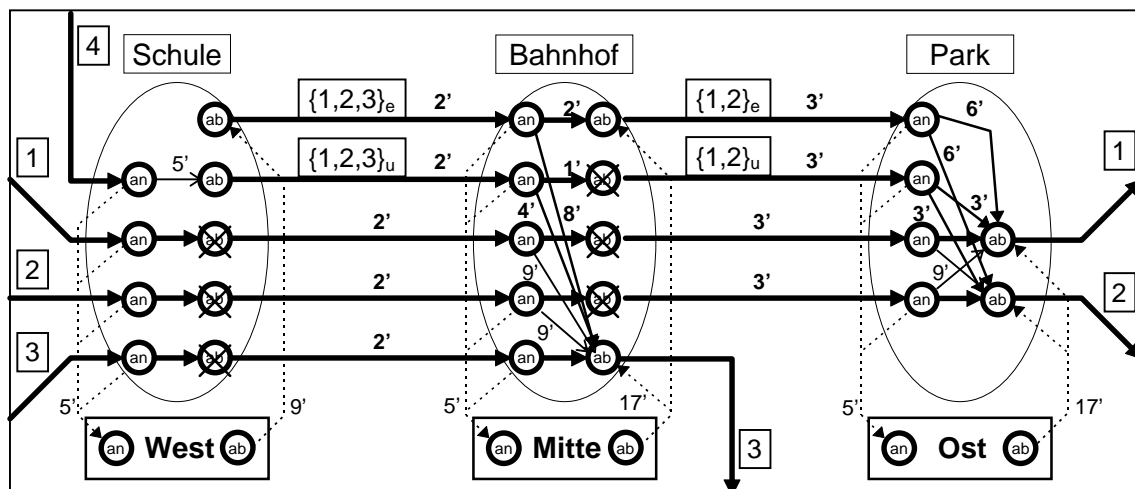
- für Einsteiger (an der Schule):  $2 \cdot (0,5 \cdot \min\{6', 13'\} - 0,5 \cdot \min\{4', 13'\}) = 2'$
- für Umsteiger (von Linie 4):  $2 \cdot (0,25 \cdot \min\{6', 30'\} - 0,25 \cdot \min\{4', 30'\}) = 1'$

Für die Umsteiger muß eine Hilfskante  $\{1,2\}_u$  vom Bahnhof zum Park erzeugt werden, was zusätzliche Verknüpfungen am Park impliziert.

3) Am **Park** endet  $K'$  (beide Kanten). Der Widerstand von  $K'$  auf Linie 1 ist:

- für Einsteiger (an der Schule):  $2 \cdot (0,5 \cdot \min\{12', 13'\} - 0,5 \cdot \min\{6', 13'\}) = 6'$
- für Umsteiger (von Linie 4):  $2 \cdot (0,25 \cdot \min\{12', 30'\} - 0,25 \cdot \min\{6', 30'\}) = 3'$

Die Widerstände von  $K'$  auf Linie 2 sind identisch, weil  $\tau(1) = \tau(2) = 12'$ .

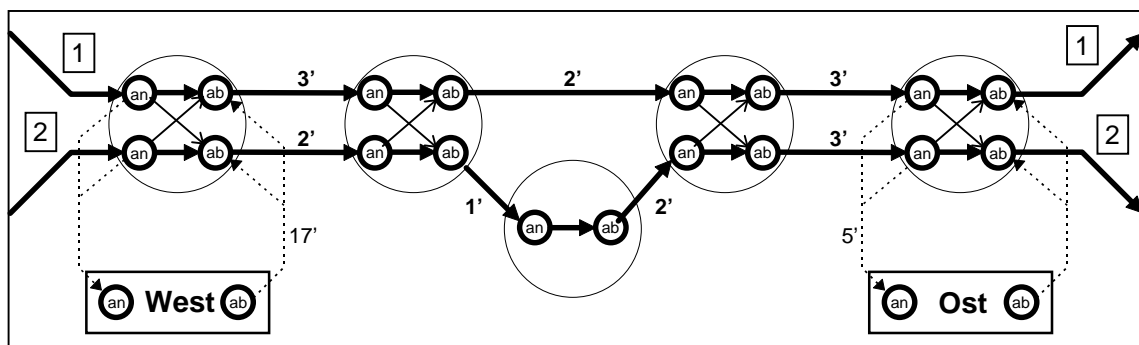


**Abbildung 2.10:** Bündeln von Parallelverkehr, 2. Schritt (Legende: vgl. Abb. 2.5)

**Bemerkungen:**

Das Bündeln von Linienabschnitten hat folgende Konsequenzen für die Wegsuche:

- Alle Wege, die sich ausschließlich durch eine Liniennummer auf einem Teilabschnitt unterscheiden, fallen zu einem einzigen Weg zusammen. Die Anzahl der zu berechnenden Wege verringert sich damit erheblich.
- Das aus Sicht von Fahrgästen nicht sinnvolle mehrfache Umsteigen zwischen parallel verlaufenden Linien ist durch die Konstruktion der Kanten im Distanzgraph von vornherein ausgeschlossen, wodurch sich die Anzahl der zu berechnenden Wege weiter verringert.
- Einmaliges Umsteigen zwischen zwei parallel verlaufenden Linien ist nur dann möglich, wenn eine der beiden Linien über den gemeinsamen Streckenabschnitt hinaus genutzt wird. Umstiegspunkt ist dann die letzte gemeinsame Haltestelle der beiden Linien (vgl. Beispiel 2.11). Andere Möglichkeiten gibt es konstruktionsbedingt nicht und werden im Interesse der Übersichtlichkeit auch nicht betrachtet.
- Der unter Umständen starke Zuwachs an Kanten für Linienbündel und deren Verknüpfungen zu anderen Kanten wird durch das Entfernen redundanter Umsteigebeziehungen abgemildert.
- Def. 2.10 ist eine strenge Auslegung des Begriffs Parallelverkehr, aus der Perspektive von Fahrgästen wahrscheinlich in einigen Fällen zu streng. Abbildung 2.11 verdeutlicht, welche Konstellationen durch eine großzügigere Interpretation des Begriffs auftreten können. Die Umsetzung dieser Interpretation ist vermutlich sehr aufwendig: Die Suche nach parallelen Wegen entspricht einer Routensuche im kleinen. Sicher ist nur, daß weitergehende Untersuchungen zum Thema Parallelverkehr den Rahmen dieser Diplomarbeit sprengen würde.



**Abbildung 2.11:** Parallelverkehr im weitesten Sinne: Der Fahrweg zwischen West und Ost spielt keine Rolle, entscheidend ist allein der Widerstand (Legende: vgl. Abbildung 2.5).

### 3. Spezifikation der gesuchten Wege

Zur Berechnung der zukünftigen Verkehrsnachfrage ist es notwendig, die Routen zu spezifizieren, welche die Fahrgäste wählen, um an das Reiseziel zu gelangen. Die Routenwahl läßt sich nachvollziehen, indem man aus der meist unendlichen Menge von ÖV-Wege solche Routen aussondert, welche aus Sicht der Fahrgäste als geeignet bzw. ungeeignet zu betrachten sind. Im Bezug auf das am IEUV verwendete Verkehrsnachfragemodell sind folgende drei Eigenschaften von Wegen in Betracht zu ziehen:

- 1) Widerstand,
- 2) Umsteigehäufigkeit,
- 3) Vermeidung von Schleifen.

#### zu 1)

Das Verkehrsnachfragemodell gründet auf der Annahme, daß Fahrgäste Wege mit möglichst geringem Widerstand wählen, so daß der materielle und immaterielle Aufwand für die Reise möglichst gering ist.

**Def. 3.1:** a) In einem **ÖV-Netzmodell**  $\mathcal{ÖV}$  ist der **Bestweg**  $w_0 \in W_{a,b}^{(\mathcal{ÖV})}$  zwischen zwei Verkehrsbezirken  $a, b \in B$  der Weg mit dem kleinsten Widerstand:

$$\forall w \in W_{a,b}^{(\mathcal{ÖV})}: \omega(w) \geq \omega(w_0).$$

Im Folgenden sei  $d_{a,b} = \omega(w_0)$  die Länge des Bestwegs zwischen den Bezirken  $a$  und  $b$  (Entfernung).

b) In einem **Distanzgraph**  $G$  ist der **Bestweg**  $w_0 \in W_{i,j}$  zwischen zwei Knoten  $i, j \in V$  der Weg mit der geringsten Länge:

$$\forall w \in W_{i,j}: \gamma(w) \geq \gamma(w_0).$$

Im Folgenden sei  $d_{i,j} = \gamma(w_0)$  die Länge des Bestwegs zwischen den Knoten  $i$  und  $j$  (Entfernung).

Berücksichtigt man bei der Verkehrsumlegung nur Bestwege, dann ist die rechnerische Verkehrsnachfrage auf Linienabschnitten, welche nicht Teil von Bestwegen sind, gleich null, obwohl diese in vielen Fällen nahezu gleichwertige Alternativen zu den kürzesten Wegen darstellen. Um ein realistischeres Bild von der Verkehrsnachfrage zu bekommen ist es sinnvoll, die  $k$ -besten Wege ( $k \in \mathbb{N}$ ) in die Umlegung mit einzubringen. Die Qualität der Alternativrouten ist allerdings je nach Verkehrsbeziehung sehr unterschiedlich. Bei geringen Entfernungen gibt es meist

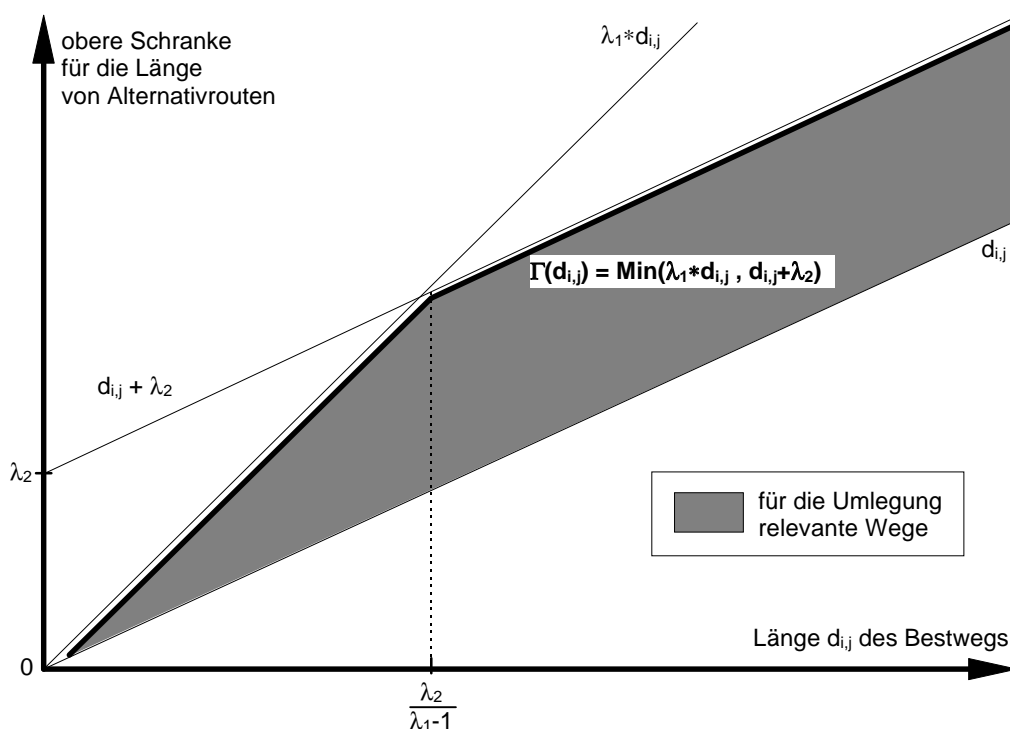
wenig Auswahl an brauchbaren Routen, bei großen Distanzen sehr viele gute Fahrmöglichkeiten. Darum kann der Wert von  $k$  nicht vor der Wegsuche fixiert werden. Maßgebendes Kriterium ist vielmehr der Widerstand von Alternativrouten, welcher innerhalb festgelegter Grenzen liegen muß. Untere Schranke ist die Länge  $d_{i,j}$  des kürzesten Wegs, die obere Schranke ist davon abhängig. Normalerweise sind für die Umlegung alle Wege relevant, deren Widerstand höchstens um einen konstanten Zeitzuschlag von beispielsweise 15 Minuten größer ist als die Bestweglänge  $d_{i,j}$ . Ist die Entfernung zwischen Start und Ziel gering (z.B. 5 Minuten), dann ist dieser Zeitzuschlag unrealistisch groß. Bei geringen Entfernung wird daher anstelle des absoluten Zeitzuschlags ein relativer Zuschlag (z.B. 20%) verwendet.

**Def. 3.2:** Die **Grenzwertfunktion**  $\Gamma$  berechnet allgemein aus der Länge  $d$  des Bestwegs die obere Schranke:

$$\Gamma(d) = \text{Min} (\lambda_1 * d, d + \lambda_2),$$

Standardwerte für die Parameter  $\lambda_1, \lambda_2$  sind:

$$\lambda_1 = 1,2 \text{ und } \lambda_2 = 15 \text{ [min].}$$



**Abbildung 3.1:** Abhängigkeit der oberen Schranke  $\Gamma$  für die Länge der Alternativrouten von der Bestweglänge  $d_{i,j}$  ( $i, j \in V$  oder  $i, j \in B$ , je nach Modell): Ist der Bestweg kurz, dann ist die Grenze ein konstantes Vielfaches von  $d_{i,j}$ . Bei großen Entfernungen gilt der additive Zeitzuschlag  $\lambda_2$ .

Bei der Verkehrsumlegung werden also sämtliche Wege berücksichtigt, deren Widerstand um bis zu 20%, jedoch nicht mehr als 15 [min] größer ist als die Bestweglänge.

### **zu 2)**

Die Anzahl der Umsteigevorgänge ist neben dem Widerstand ein weiteres Ausschlußkriterium von Alternativrouten. Untersuchungen von Verkehrsunternehmen haben ergeben, daß die Bereitschaft von Fahrgästen zum Umsteigen begrenzt ist. Routen mit mehr als vier Umsteigevorgängen werden im Allgemeinen nicht akzeptiert. Beispiel 2.6 verdeutlicht zudem, wie stark die Attraktivität von Verkehrsverbindungen durch Umsteigevorgänge leidet.

**Def. 3.3:** Umsteigehäufigkeit  $\mu$  eines ÖV-Weges:

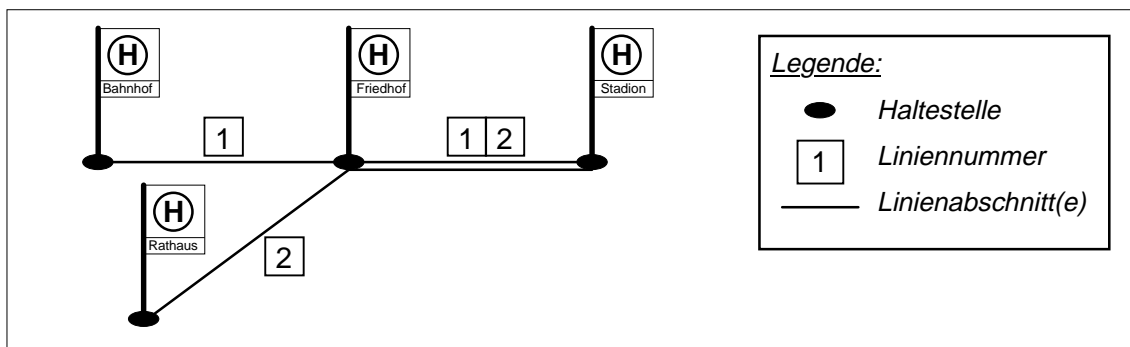
$$\mu: W^{(\text{ÖV})} \rightarrow \text{IN}$$

Sei  $w = (f_A, t_0, u_1, t_1, \dots, u_n, t_n, f_B) \in W^{(\text{ÖV})}$ . Dann ist  $\mu(w) = n$ .

Die maximale Umsteigehäufigkeit von Wegen, die bei der Verkehrsumlegung berücksichtigt werden, sei  $u_{\text{max}}$ . Diese Beschränkung ist insbesondere für Wegsuche- Algorithmen vorteilhaft, weil dadurch die Anzahl der berechneten Wege drastisch reduziert wird. Im Folgenden wird  $u_{\text{max}} = 5$  angenommen. Falls es im Extremfall zwischen zwei Bezirken keinen ÖV-Weg mit höchstens fünf Umsteigevorgängen gibt, muß die Wegsuche mit vergrößertem  $u_{\text{max}}$  wiederholt werden.

### **zu 3)**

Wege mit Schleifen sind aus Sicht von Fahrgästen nicht sinnvoll, weil die Reisezeit für Schleifenfahrten nutzlos verstreicht. Bei der Verkehrsumlegung werden deshalb nur zyklensfreie Wege berücksichtigt, d.h. keine Haltestelle wird mehrmals durchfahren.



**Abbildung 3.2:** Ausschnitt aus einem ÖV-Netzmodell: Vom Bahnhof zum Rathaus muß am Friedhof umgestiegen werden. Die Fahrt über das Stadion wäre nicht sinnvoll (Zeitverschwendung).

Die Anzahl Personen, welche in Schleifen fahren, kann vernachlässigt werden. Leider gibt es Linien, welche Haltestellen mehrfach bedienen. Beispielsweise könnte Linie 1 (vgl. Abbildung 3.2) vom Bahnhof über das Stadion zum Rathaus weiterfahren. In diesem Fall wäre es plausibel, die Haltestelle Friedhof doppelt zu befahren, um einen Umsteigevorgang einzusparen. Um Wege mit Schleifen nicht berücksichtigen zu müssen, werden im ÖV-Netzmodell folgende Anpassungen durchgeführt:

$$F \subseteq B \times H \cup H \times H$$

$$\forall l = (h_1, h_2, \dots, h_n) \in L:$$

$$\forall 1 \leq x \leq n:$$

$$M_x := \{ y \mid x \leq y \leq n \text{ und } h_y = h_x \};$$

Falls  $|M_x| \geq 2$ ,

Sei  $M_x = \{ i_1, i_2, \dots, i_k \}$  mit  $k = |M_x|$   
und  $\forall 1 \leq z \leq k: x \leq i_z \leq n$ ;

$$h_x^{(1)} := h_x;$$

$$H := H \cup \{ h_x^{(z)} \mid 2 \leq z \leq k \};$$

$\forall 2 \leq z \leq k$ : ersetze in  $l$   $h_{i_z}$  durch  $h_x^{(z)}$ ;

$$F := F \cup \{ (h_x^{(u)}, h_x^{(v)}) \mid 1 \leq u < v \leq k \};$$

Erweiterung von Def. 2.1: Fußwege werden für Verbindungen zwischen Haltestellen benötigt.

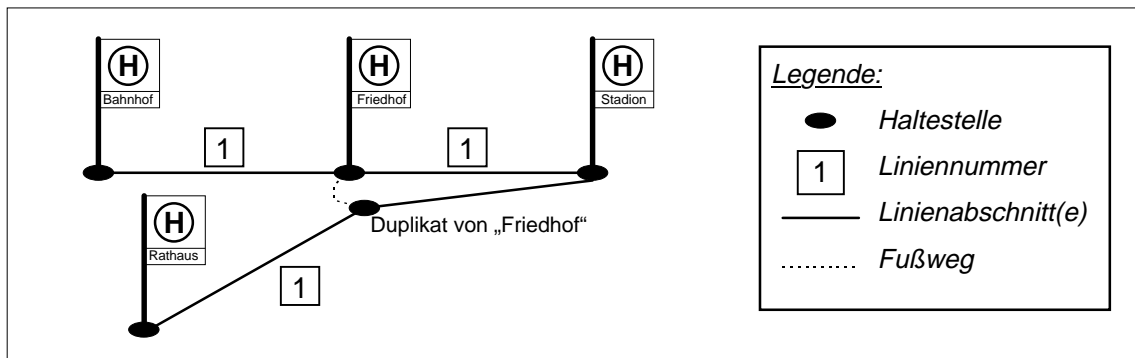
Teste alle Linien auf Mehrfachbedienug.

Ermittle alle Vorkommen von  $h_x$  in  $l$ .  
Falls  $h_x$  in  $l$  mehrfach enthalten ist,

wird jedes Auftreten von  $h_x$  in  $l$  außer das erste (Original)

durch ein Duplikat  $h_x^{(z)}$  ersetzt.

Alle Varianten von  $h_x$  werden über Fußwege miteinander verbunden.



**Abbildung 3.3:** Linie 1 bedient die Haltestelle Friedhof doppelt. Der Haltepunkt wird dupliziert, so daß die Fahrt vom Bahnhof über das Stadion zum Rathaus schleifenfrei ist. Duplikat und Original sind über einen Fußweg miteinander verbunden.

**Zusammenfassung:**

Gesucht sind alle zyklenfreie Wege, deren Länge die vom Bestweg abhängige obere Schranke nicht überschreitet und deren Umsteigehäufigkeit höchstens  $u_{max}$  beträgt.

**Def. 3.4:** Lösungsmenge  $L_{a,b}^{(OV)}$  für  $a, b \in B$  (Menge der für die Verkehrsumlegung relevanten Wege zwischen zwei Bezirken):

$$L_{a,b} = \{ w \in W_{a,b} \mid w \text{ zyklenfrei und } \omega(w) \leq \Gamma(d_{a,b}) \text{ und } \mu(w) \leq u_{max} \}.$$

## 4. Algorithmen zur Bestimmung der gesuchten Wege

### 4.1 Problemstellung und -lösung

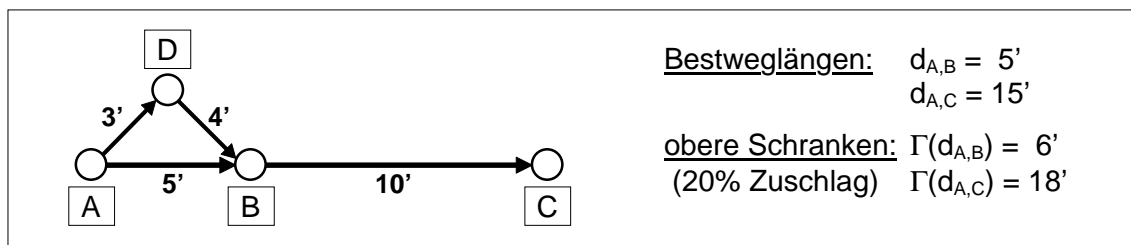
#### 4.1.1 Besonderheiten der Problemstellung

Bei der Konzeption von Algorithmen zur Bestimmung der gesuchten Wege müssen folgende Besonderheiten berücksichtigt werden:

- 1) Die Berechnung der oberen Schranken mit der Grenzwertfunktion  $\Gamma$  (Def. 3.2) verletzt die sogenannte Greedy-Bedingung, welche die Grundlage aller in diesem Kapitel betrachteten Algorithmen ist.
- 2) Die Berechnung von Fahrtkosten, maximaler Taktfolgezeit und Schnellbahnanteil während der Wegsuche verletzt die Greedy-Bedingung.
- 3) Die Problemstellung entspricht nicht dem  $k$ -kürzeste-Wege-Problem, weil die Anzahl  $k$  der gesuchten Wege je nach Verkehrsbeziehung unterschiedlich sein kann.

#### zu 1)

Die Greedy-Bedingung drückt aus, daß beim Vergleich von Alternativrouten während der Wegsuche bei jedem Zwischenschritt bereits endgültig entschieden werden kann, welche der Alternativen bezüglich der Minimierung des Widerstands optimal ist. Bei Greedy-Algorithmen sind alle „lokalen“ Entscheidungen bezüglich der Routenwahl auch nach Ablauf der Wegsuche noch gültig. Bei der Selektion von Wegen, deren Widerstand innerhalb der oberen Schranke  $\Gamma(d_{i,j})$  liegt, kann der Fall auftreten, daß sich eine „lokale“ Entscheidung während der Wegsuche zu einem späteren Zeitpunkt als falsch herausstellt, wie in Abbildung 4.1 verdeutlicht.



**Abbildung 4.1:** Beispielgraph mit vier Knoten bzw. Kanten; Der Weg  $w=(A,D,B)$  gehört wegen  $\gamma(w) = 7' > \Gamma(d_{A,B})$  nicht zur Lösungsmenge  $L_{A,B}$ . Wird  $w$  verworfen, so kann  $w' = (A,D,B,C)$  nicht berechnet werden, obwohl  $\gamma(w') = 17' < \Gamma(d_{A,C})$  und damit  $w' \in L_{A,C}$ .

Das „Prinzip der Vorentscheidung“, ob ein Weg länger als erlaubt ist, funktioniert offensichtlich nicht bei Verwendung relativer Zeitzuschläge. Daher kommen nur additive Zeitzuschläge in Frage.

**Def. 4.1:** Die bei der Routensuche verwendeten **oberen Schranken**  $D_{i,j}$  für die Weglänge zwischen zwei Objekten  $i, j$  (Knoten, Bezirke, Haltestellen) werden wie folgt berechnet:

$$D_{i,j} = d_{i,j} + \lambda_2$$

wobei  $d_{i,j}$  die Bestweglänge zwischen  $i$  und  $j$  ist.  $\lambda_2$  ist der konstante Zeitzuschlag aus Def. 3.2.

Es bleibt zu zeigen, daß das Prinzip der Vorentscheidung bei additiven Zeitzuschlägen funktioniert. Zur Vereinfachung werden nur Wege in Distanzgraphen betrachtet. Der Beweis für ÖV-Wege läßt sich analog zu führen (mit Haltestellen und Bezirken anstelle von Knoten). Zunächst muß die Verkettung von Wegen definiert werden:

**Def. 4.2: Konkatenation von Wegen**  $w_1 \in W_{i,k}$ ,  $w_2 \in W_{k,j}$  mit  $w_1 = (u_1, u_2, \dots, u_m)$ ,  $w_2 = (v_1, v_2, \dots, v_n)$  und  $u_m = v_1 = k$  (sogenannter Verbindungsknoten):

$$w_1 \bullet w_2 := (u_1, u_2, \dots, u_m, v_2, \dots, v_n) \in W_{i,j}.$$

**Satz 4.1: Prinzip der Vorentscheidung:**

Fällt bei der Routensuche ein Weg  $w \in W_{i,j}$  an mit  $\gamma(w) > D_{i,j}$  ( $i, j \in V$ ), so gilt:

$$\begin{aligned} &\forall w' \in W_{j,k} (k \in V): \gamma(w \bullet w') > D_{i,k} \\ \text{und } &\forall w' \in W_{k,i} (k \in V): \gamma(w' \bullet w) > D_{k,i} \end{aligned}$$

d.h. wenn  $w$  zu lang ist, dann ist jede Erweiterung von  $w$  ebenfalls zu lang. Damit wird  $w$  in späteren Berechnungen nicht mehr benötigt und kann verworfen werden.

Satz 4.1 bedeutet anschaulich, daß der durch Fahren von Umwegen entstandene Zeitverlust gegenüber dem Bestweg im weiteren Verlauf nicht mehr abgebaut werden kann, weil ansonsten Abkürzungen im Distanzgraph existieren müßten, welche gegenüber den Bestwegen Zeitgewinne ermöglichen, was im Widerspruch zu Def. 3.1 steht.

**Beweis von Satz 4.1:**

Gegeben sei zwei beliebige Wege  $w \in W_{i,j}$ ,  $w' \in W_{j,k}$  und  $\gamma(w) > D_{i,j}$  ( $i, j, k \in V$ ).

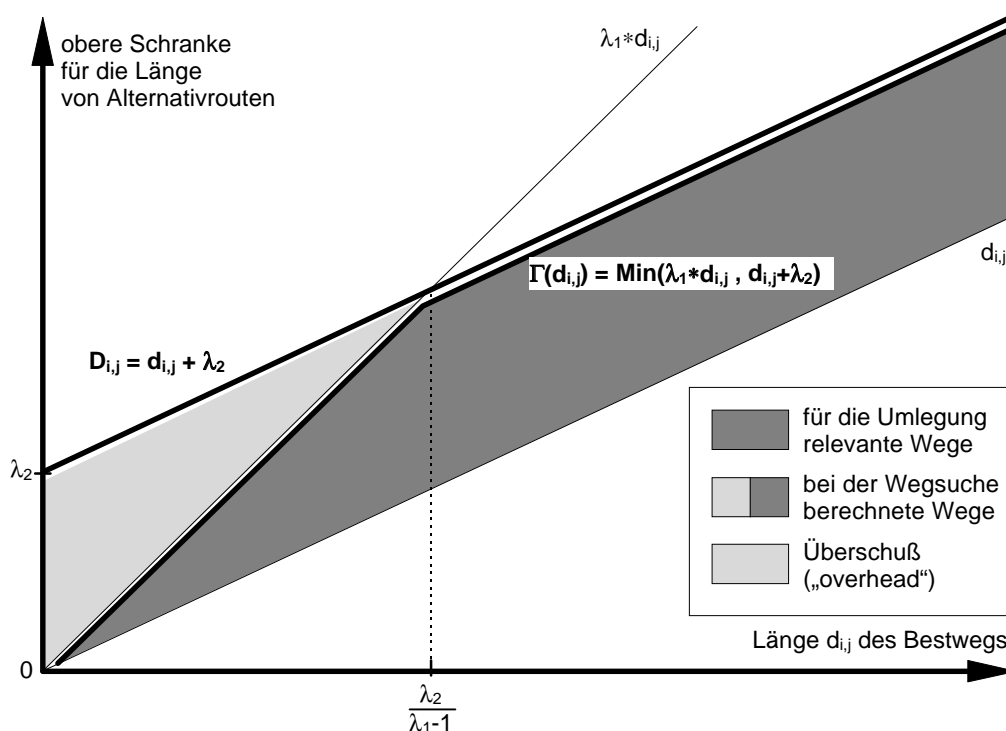
$$\begin{aligned} \gamma(w \bullet w') &= \gamma(w) + \gamma(w') \\ &\stackrel{\text{nach Voraussetzung}}{>} D_{i,j} + \gamma(w') = d_{i,j} + \lambda_2 + \gamma(w') \\ &\geq d_{i,j} + \lambda_2 + d_{j,k} && \vdots \text{ (#) gemäß Def. 3.1 gilt:} \\ &\stackrel{\text{(#)}}{\geq} d_{i,k} + \lambda_2 = D_{i,k}, && \vdots \quad d_{i,k} \leq d_{i,j} + d_{j,k} \end{aligned}$$

also  $\gamma(w \bullet w') > D_{i,k}$ .

Der Beweis für  $w' \bullet w$  kann analog zu  $w \bullet w'$  geführt werden. (q.e.d.)

**Folgerung aus Satz 4.1:**

Wegsuche-Algorithmen müssen unter Umständen wesentlich mehr Wege berechnen als notwendig, da vor allem bei geringen Entfernungen zwischen zwei Knoten  $i, j$  die obere Schranke  $D_{i,j}$  gegenüber Def. 3.2 zu hoch angesetzt ist. Im Anschluß an die Wegsuche müssen die oberen Schranken nach unten korrigiert und die überflüssigen Wege beseitigt werden.



**Abbildung 4.2:** Überschuß an Wegen nach Ablauf der Routensuche aufgrund der überhöhten oberen Schranken  $D_{i,j}$  für die Länge von Wegen zwischen  $i$  und  $j$  ( $i, j \in V$  oder  $i, j \in B$ , je nach Modell)

## zu 2)

- a) Die **Fahrtkosten** richten sich in vielen ÖV-Netzen nach einem Zonentarif, d.h. das Netz ist in Zonen gegliedert und der Fahrpreis hängt von der Anzahl der Zonen ab, welche durchfahren werden. Die Preisabstufung ist in der Regel nicht linear, außerdem gibt es obere Schranken für den Preis. Beim Überfahren einer Zonengrenze hängt der Kostenzuwachs von der Menge der bereits besuchten Zonen ab, so daß für grenzüberschreitende Wege unterschiedliche Mehrkosten anfallen. Dabei ist nicht garantiert, daß der Bestweg den geringsten Zuwachs aufweist, was die Greedy-Bedingung verletzt (vgl. Abbildung 4.3).

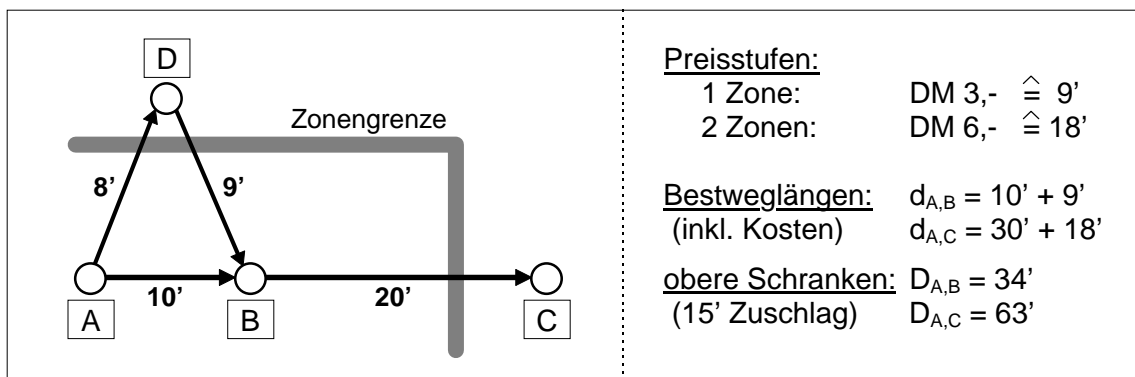


Abbildung 4.3: Beispielgraph mit vier Knoten bzw. Kanten und zwei Tarifzonen; Fahrtkosten werden für alle Wege berücksichtigt.  $w=(A,D,B)$  gehört wegen  $\gamma(w) = 17' + 18' = 35' > D_{A,B}$  nicht zur Lösungsmenge  $L_{A,B}$ . Wird  $w$  verworfen, so kann  $w' = (A,D,B,C)$  nicht berechnet werden, obwohl  $\gamma(w') = 37' + 18' = 55' < D_{A,C}$  und damit  $w' \in L_{A,C}$ . Es gilt sogar  $\gamma(w') < \Gamma(d_{A,C}) = 48' + 20\% = 57,6'$ , d.h.  $w'$  muß bei der Verkehrsumlegung berücksichtigt werden.

- b) Die **maximale Taktfolgezeit** betrifft die am wenigsten häufig verkehrende Linie, bezogen auf alle benutzten Linien im Verlauf eines Wegs. Die Ermittlung des zugehörigen Widerstands während der Wegsuche würde Probleme verursachen: Beim Zustieg auf eine selten verkehrende Linie hängt der Zuwachs des Widerstands von der Menge der bereits benutzten Linien ab. Der Zuwachs ist für den Bestweg nicht notwendigerweise minimal, was die Greedy-Bedingung verletzt.
- c) Für den **Schnellbahnanteil** läßt sich analog zu den Fahrtkosten und der maximalen Taktfolgezeit ein Beispiel finden, so daß die Greedy-Bedingung verletzt ist.

Auf die Berechnung von Fahrtkosten, maximaler Taktfolgezeit und Schnellbahnanteilen während der Wegsuche muß verzichtet werden. Die Minimierung des Widerstands erfolgt daher im wesentlichen auf Basis von Reisezeitkriterien. Die anderen Kriterien können erst nach Abschluß der Wegsuche berücksichtigt werden.

**zu 3)**

Die Problemstellung entspricht nicht dem k-kürzeste-Wege-Problem, sondern dem Problem „Wegsuche mit Zeitschranke“. Hauptproblem ist die Abschätzung der Größenordnung von k, also der Anzahl der gesuchten Wege, die je nach Verkehrsbeziehung unterschiedlich sein kann. Da die Anzahl der Wege in einem Graph gegenüber  $N = |V|$  exponentiell wächst, kann auch k exponentiell zunehmen, wie in Abbildung 4.4 verdeutlicht.

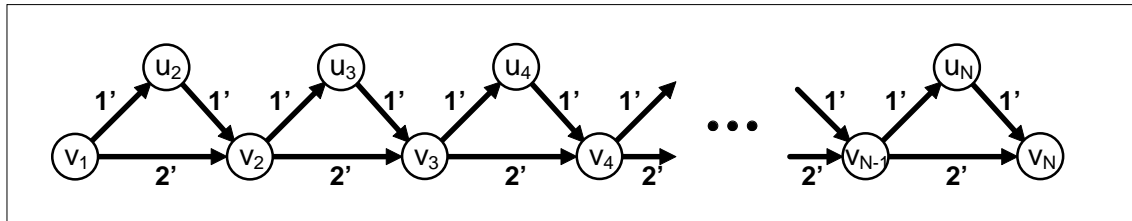


Abbildung 4.4: Beispielgraph mit  $2 \cdot N - 1$  Knoten und  $3 \cdot N - 3$  Kanten, wobei  $N \geq 2$  ist. Alle Wege von  $v_1$  nach  $v_N$  besitzen dieselbe Länge, also sind alle gleichermaßen relevant. Die Anzahl k der gesuchten Wege beträgt damit  $k = 2^{N-1}$ .

In der Praxis wächst k wesentlich langsamer gegenüber N, was auf die radiale Struktur von ÖV-Netzen zurückzuführen ist: In den meisten Netzen gibt es einen zentralen Verknüpfungspunkt, über den der Großteil des Gesamtverkehrs fließt. Weil das Liniennetz stark auf das Zentrum ausgerichtet ist, sind Tangentialverbindungen verhältnismäßig unattraktiv, so daß es zwischen zwei beliebigen Verkehrszellen im Endeffekt nur wenige sinnvolle Wege gibt, welche meist über das Zentrum führen.

Bei der Verkehrsumlegung sind ohnehin nur begrenzte Mengen von Routen interessant, da die Anzahl der zu verteilenden Fahrgäste je Verkehrsstrom in der Praxis gering ist. Weil ohne Begrenzung von k ein Vergleich verschiedener Wegsuche-Algorithmen unsinnig wäre (generell exponentieller Aufwand), wird für die Aufwandsabschätzungen aller Verfahren folgende Bedingung vorausgesetzt:

$$\exists k \in \mathbb{N}: \forall i, j \in V: |L_{i,j}| \leq k,$$

d.h. der Umfang der Lösungsmengen ist k-beschränkt. Die exponentielle Komponente wird dabei in k „versteckt“, falls vorhanden.

### **4.1.2 Ansätze zur Problemlösung**

Die Graphentheorie bietet eine Vielzahl von Verfahren zur Bestimmung der k-kürzesten Wege in allgemeinen Distanzgraphen, aber keines für die in Kapitel 3 spezifizierten Wege in ÖV-Netzmodellen. Zur Lösung dieses Problems gibt es prinzipiell zwei Ansätze:

1. Das ÖV-Netzmodell wird auf einen Distanzgraph abgebildet, so daß Standardalgorithmen der Graphentheorie zum Einsatz kommen können. Um die gesuchten Wege zu ermitteln, müssen die Algorithmen der Problemstellung entsprechend angepaßt werden. Vorteil dieser Vorgehensweise ist der große Vorrat an vielfach erprobten und bewährten Algorithmen zur Wegsuche, Nachteil ist der Aufwand für die Konvertierung des ÖV-Netzmodells in einen Distanzgraphen (vor allem Speicherplatzbedarf) sowie die Abbildung der berechneten Knotenfolgen auf ÖV-Wege. Die Besonderheiten der Problemstellung erfordern zum Teil drastische Anpassungen der Algorithmen, wodurch in vielen Fällen deren Effizienz leidet.
2. Die Entwicklung „maßgeschneiderter“, d.h. auf dem ÖV-Netzmodell basierender Algorithmen verspricht aufgrund des engen Bezugs zur Problemstellung deutliche Vorteile im Hinblick auf Effizienz. Allerdings existieren bislang weder Vorbilder, noch praktische Erfahrungen mit Algorithmen dieser Art. Deren Untersuchung stellt aus diesem Grund einen wichtigen Teil der Diplomarbeit dar (siehe Abschnitt 4.4).

Algorithmen zur Wegsuche lassen sich in drei Gruppen einteilen:

- a) 1:1-Suchverfahren („single pair“): Gesucht werden die kürzesten Wege zwischen zwei Punkten eines Netzes.
- b) 1:N-Suchverfahren („single source“): Von einem Startpunkt ausgehend werden Wege nach allen Richtungen hin verfolgt, wobei jeweils die kürzesten Wege zwischen dem Startpunkt und jedem anderen Punkt des Netzes erfaßt werden.
- c) N:N-Suchverfahren („all pairs“): Von allen Punkten des Netzes ausgehend werden schematisch Verbindungen nach allen Richtungen hin verfolgt und für jedes Paar von Punkten die kürzesten Wege selektiert.

**Bemerkungen:**

- zu a) 1:1-Suchverfahren sind problematisch im Hinblick auf Effizienz: Bei einer vollständigen Wegsuche („blindes“ Verfahren) werden Verbindungen nach allen Richtungen hin verfolgt, so daß fast alle berechneten Wege nicht zur Lösung gehören. Durch die Verwendung von Heuristiken, welche die Suche in Richtung auf das Ziel lenken, kann der Aufwand zwar eingeschränkt werden, aber die Lösung ist unter Umständen nicht mehr korrekt. 1:1-Suchverfahren sind nur dann sinnvoll, wenn nur Routen zwischen zwei bestimmten Bezirken gesucht sind. Für die Verkehrsumlegung müssen jedoch Wege zwischen allen Bezirken berechnet werden, so daß 1:N- bzw. N:N-Verfahren vorzuziehen sind.
- zu b) Bei 1:N-Verfahren tritt das oben genannte Problem nicht auf, weil die Wege ohnehin nach allen Richtungen hin verfolgt werden müssen. Außerdem kann ein Teilabschnitt eines Weges für verschiedene Relationen weiterverwendet werden, während bei 1:1-Verfahren für jede Relation der Algorithmus erneut gestartet und damit der Teilabschnitt neu berechnet werden muß. Ein 1:N-Algorithmus kann daher wesentlich effizienter sein als das N-malige Ausführen eines 1:1-Verfahrens.
- zu c) Bei N:N-Verfahren kann der Vorteil, daß Informationen von Teilberechnungen für viele Relationen weiterverwendet werden können, optimal ausgenutzt werden. Nachteilig ist jedoch der hohe Speicherplatzbedarf, weil die gesuchten Wege für alle Relationen gleichzeitig im Speicher gehalten werden müssen. In der Praxis kann dieses Problem die Laufzeitvorteile völlig aufheben.

Da bei einer Verkehrsumlegung in einem ÖV-Netz alle Verkehrsbeziehungen betrachtet werden müssen, ist bei einem Vergleich verschiedener Algorithmen der Gesamtaufwand einer N:N-Wegsuche maßgebend. 1:N-Verfahren müssen hierbei N-mal, 1:1-Verfahren  $N^2$ -mal durchgeführt werden. Aufgrund der oben genannten Nachteile werden letztere Verfahren nicht weiter betrachtet.

In den folgenden Abschnitten sind drei Algorithmen (N:N- bzw. 1:N-Verfahren) beschrieben, welche im Hinblick auf Effizienz als aussichtsreich gelten. Im Vordergrund der Untersuchungen steht dabei nicht der Vergleich, sondern die Implementierbarkeit der Verfahren.

## **4.2 Lösungsansatz nach Minieka (N:N-Suchverfahren)**

### **4.2.1 Allgemeines**

N:N-Verfahren kommen in der Graphentheorie relativ selten vor. Ein effizientes Verfahren ist der Algorithmus nach Minieka ([12]), der im Jahre 1974 veröffentlicht wurde. Mit dem Algorithmus kann man die k-kürzesten Wege in einem Distanzgraphen ermitteln. Das Verfahren ist aus dem Algorithmus nach Floyd ([9]) hervorgegangen, einem Standardalgorithmus zur Berechnung von Bestwegen in Distanzgraphen.

### **4.2.2 Grundprinzip**

Für jedes Knotenpaar  $(i,j) \in V^2$  werden die k-kürzesten Wege ermittelt (k ist fest), indem ein Hilfsknoten m herangezogen wird und alle Wege zwischen i und m mit jenen zwischen m und j kombiniert werden, wodurch sich neue Wege für das Paar (i,j) ergeben. Dabei werden nur die k besten Wege selektiert, alle anderen werden verworfen. Die Kombination der Wege von und nach m wird für jeden Knoten  $m \in V$  genau einmal durchgeführt, so daß kein Weg auf verschiedene Art und Weise zusammengebaut werden kann.

### **4.2.3 Anpassungen an die Problemstellung**

- Da die ermittelten Wege in der Regel nicht zyklonfrei sind, müssen besondere Vorkehrungen zur Vermeidung von Zyklen getroffen werden: Für jeden berechneten Weg muß die Menge der besuchten Knoten festgehalten werden. Beim Aneinanderhängen zweier Wege muß die Schnittmenge leer sein (siehe Def. 4.4).
- Minieka berücksichtigt auch reflexive Beziehungen an Knoten. Da nur schleifenfreie Wege gesucht sind, entfallen alle zugehörigen Berechnungen.
- Minieka definiert für alle Knotenpaare eine beliebige, aber feste Anzahl k von Wegen (k-Tupel für Weglängen). Bei der Routensuche in Verkehrsnetzen gibt es lediglich obere Schranken  $D_{i,j}$  für die Weglänge, jedoch nicht für die Anzahl von Alternativrouten. Zur Flexibilisierung des Algorithmus werden deshalb anstelle von k-Tupeln Mengen eingesetzt.
- Das Protokollieren von Streckenverläufen während der Routensuche ist ursprünglich nicht vorgesehen, für die Verkehrsumlegung aber unverzichtbar. Deshalb muß für jeden berechneten Weg die Folge der besuchten Knoten festgehalten werden.

- Die oberen Schranken  $D_{i,j}$ , die vom Bestweg abhängen, müssen zu Beginn des Verfahrens bekannt sein, weil sonst im Extremfall alle möglichen Wege erfaßt werden müßten, ehe der Bestweg gefunden würde. Folglich müssen in einer Vorbereitungsphase die Längen der Bestwege ermittelt werden. Hierfür kann der Algorithmus nach Dijkstra herangezogen werden, welcher besonders für planare Graphen geeignet ist (siehe Abschnitt 4.3).
- Der Algorithmus nach Minieka setzt voraus, daß für alle Knotenpaare die ermittelten Wege paarweise verschiedene Längen besitzen, was in allgemeinen Graphen nicht der Fall ist. Aufgrund der Verwendung von Weglängen in Kombination mit Mengen werden im Algorithmus alle Wege gleicher Länge verworfen (bis auf einen). Ersetzt man die Weglängen durch Knotenfolgen (Wegprotokolle), so tritt dieses Problem nicht mehr auf.

#### **4.2.4 Grundelemente und -operationen**

In einer  $N \times N$ -Knotenmatrix wird für jedes Feld, d.h. für jedes Knotenpaar ein Protokoll für die gesuchten Wege zwischen diesen Knoten geführt:

**Def. 4.3:** Ein **Wegeprotokoll**  $P_{i,j}$  für ein Knotenpaar  $(i,j) \in V^2$ ,  $i \neq j$  ist eine Menge von Wegen, die zur Lösungsmenge  $L_{i,j}$  gehören:

$$P_{i,j} = \{ w = (v_1, v_2, \dots, v_n) \in W_{i,j} \mid \forall 1 \leq x < y \leq n: v_x \neq v_y \\ \text{und } \gamma(w) \leq D_{i,j} \text{ und } \mu(w) \leq u_{\max} \}$$

Die wesentliche Operation des Algorithmus ist das Kombinieren von Teilstrecken zur Bildung größerer Wege.

**Def. 4.4:** Die **Konkatenation zweier Wegeprotokolle**  $P_{a,b}$ ,  $P_{b,c}$ ,  $a, b, c \in V$  ist die Menge der Wegekombinationen aus  $P_{a,b}$  und  $P_{b,c}$  mit Einschränkungen:

$$P_{a,b} \bullet P_{b,c} := \{ w_1 \bullet w_2 \mid w_1 \in P_{a,b}, w_2 \in P_{b,c} \text{ und } \gamma(w_1) + \gamma(w_2) \leq D_{i,j} \\ \text{und } \mu(w_1) + \mu(w_2) \leq u_{\max} \text{ und } w_1 \cap w_2 = \{b\} \}$$

$w_1 \bullet w_2$  ist die Konkatenation von Wegen gemäß Def. 4.2. Die Schleifenbedingung  $w_1 \cap w_2 = \{b\}$  garantiert, daß bis auf den Verbindungsknoten  $b$  kein Knoten in beiden Folgen vorkommt, so daß nur Wege entstehen, welche zur Lösungsmenge gehören.

**Hinweis:** Falls  $P_{a,b} = \{ \}$  oder  $P_{b,c} = \{ \}$ , dann ist auch  $P_{a,b} \bullet P_{b,c} = \{ \}$ .

Zur Darstellung des Rechenfortschritts im Algorithmus erhalten alle Wegeprotokolle zusätzlich einen Hochindex:

$$P_{i,j}^{(0)}, P_{i,j}^{(1)}, \dots, P_{i,j}^{(N)}$$

#### **4.2.5 Ablauf**

**Gegeben:** Distanzgraph  $G = (V, E, \gamma)$  nach Def. 2.7.

**Gesucht:** die besten zyklensfreien Wege von  $i$  nach  $j$  ( $i, j \in V, i \neq j$ ) mit Zeitschranke  $D_{i,j}$  gemäß Def. 4.1 und maximaler Umsteigehäufigkeit  $u_{\max}$ . Die gesuchten Wege werden in  $P_{i,j}$  protokolliert.

**Vorbereitung:**

- $\forall i, j \in V, i \neq j$ : Ermittle die Länge  $d_{i,j}$  des Bestwegs von  $i$  nach  $j$ .  
Dies kann beispielsweise mit dem bewährten Verfahren nach Dijkstra durchgeführt werden (vgl. Abschnitt 4.3). Da der Algorithmus ein 1:N-Suchverfahren ist, muß er für jeden der  $N$  Knoten wiederholt werden.
- $\forall i, j \in V, i \neq j$ :  $D_{i,j} := d_{i,j} + \lambda_2$  (provisorische obere Schranke)

**Initialisierung:**

Wegeprotokolle enthalten zunächst nur Kantenverbindungen, falls deren Längen innerhalb der zuvor berechneten Schranken liegen:

$$\forall i, j \in V, i \neq j: P_{i,j}^{(0)} := \{ (i, j) \}, \text{ falls } (i, j) \in E \text{ und } \gamma((i, j)) \leq D_{i,j},$$

$$P_{i,j}^{(0)} := \{ \} \text{ sonst.}$$

**Hauptteil:**

```

FOR m:= 1 TO N DO
  FOR i:= 1 TO N DO
    FOR j:= 1 TO N DO
      IF i ≠ m AND j ≠ m AND i ≠ j THEN
         $P_{i,j}^{(m)} := P_{i,j}^{(m-1)} \cup (P_{i,m}^{(m-1)} \bullet P_{m,j}^{(m-1)});$ 
        END;
      END;
    END;
  END;
END;
```

: : : Konstruktion neuer Wege  
: : : durch Zusammenbau „•“ und  
: : : Selektion der Wege  
: : : mittels Zeitschranke  $D_{i,j}$ ;

**Korrektur der oberen Schranken:**

$$\forall i, j \in V, i \neq j: P_{i,j} := \{ w \in P_{i,j}^{(N)} \mid \gamma(w) \leq \Gamma(d_{i,j}) \}; \quad \text{: : : mit } \Gamma \text{ gemäß Def. 3.2}$$

### 4.2.6 Korrektheitsbeweis

Für jedes Knotenpaar  $(i,j) \in V^2$  mit  $i \neq j$  ist folgende Lösungsmenge definiert:

$$L_{i,j}^* = \{ w = (v_1, v_2, \dots, v_n) \in W_{i,j} \mid \forall 1 \leq x < y \leq n: v_x \neq v_y \text{ und } \gamma(w) \leq D_{i,j} \text{ und } \mu(w) \leq u_{\max} \},$$

d.h. gesucht sind alle zyklensfreien Wege, deren Länge innerhalb der Schranke  $D_{i,j}$  liegt und deren Umsteigehäufigkeit  $u_{\max}$  nicht überschreitet. Zu zeigen ist, daß für die Wegeprotokolle  $P_{i,j}$  nach Ablauf des Algorithmus gilt:

$$P_{i,j}^{(N)} = L_{i,j}^*.$$

Man kann durch Induktion über  $m$  (Berechnungsfortschritt der äußeren Schleife) beweisen, daß  $P_{i,j}^{(m)}$  alle Wege aus  $L_{i,j}^*$  enthält, welche ausschließlich über die ersten  $m$  Knoten führen, d.h.

$$\forall i,j \in V, i \neq j: P_{i,j}^{(m)} = \{ w = (v_1, v_2, \dots, v_n) \in L_{i,j}^* \mid \forall 2 \leq x \leq n-1: v_x \in \{1, 2, \dots, m\} \}$$

#### Induktionsanfang (m=0):

Die Initialisierung des Algorithmus ergibt:

$$\forall i,j \in V, i \neq j: P_{i,j}^{(0)} = \{ (i,j) \}, \text{ falls } (i,j) \in E \text{ und } \gamma((i,j)) \leq D_{i,j},$$

$$P_{i,j}^{(0)} = \{ \} \text{ sonst.}$$

Die Wege in  $P_{i,j}^{(0)}$  sind Kanten (falls vorhanden) und damit zyklensfrei,  $\mu((i,j)) \leq 1 \leq u_{\max}$  und  $\gamma((i,j)) \leq D_{i,j}$ . Da in Kanten keine Zwischenknoten auftreten, ist die Behauptung erfüllt.

Induktionsvoraussetzung (IV): Für ein  $m \in V$  gelte:

$\forall i,j \in V, i \neq j: P_{i,j}^{(m-1)}$  enthält alle Wege aus  $L_{i,j}^*$ , welche über die ersten  $m-1$  Knoten führen.

Induktionsbehauptung (IB): Für  $m$  aus (IV) gilt:

$\forall i,j \in V, i \neq j: P_{i,j}^{(m)}$  enthält alle Wege aus  $L_{i,j}^*$ , welche über die ersten  $m$  Knoten führen.

#### Induktionsschritt:

Für alle Knotenpaare  $(i,j) \in V^2$ ,  $i \neq j$  wird berechnet:

$$P_{i,j}^{(m)} := P_{i,j}^{(m-1)} \cup (P_{i,m}^{(m-1)} \bullet P_{m,j}^{(m-1)})$$

Sei  $M = P_{i,m}^{(m-1)} \bullet P_{m,j}^{(m-1)} \stackrel{\text{Def. 4.4}}{=} \{ w_1 \bullet w_2 \mid w_1 \in P_{i,m}^{(m-1)}, w_2 \in P_{m,j}^{(m-1)}, \gamma(w_1) + \gamma(w_2) \leq D_{i,j}$   
und  $\mu(w_1) + \mu(w_2) \leq u_{\max}$  und  $w_1 \cap w_2 = \{m\} \}$

Gemäß (IV) führen alle Wege aus  $P_{i,m}^{(m-1)}$  und  $P_{m,j}^{(m-1)}$  über die ersten  $m-1$  Knoten. Bei der Konkatenation  $P_{i,m}^{(m-1)} \bullet P_{m,j}^{(m-1)}$  wird  $m$  als Verbindungsknoten zweier Wege hinzugefügt.

Daher gilt für alle Wege  $w = (v_1, v_2, \dots, v_n) \in M$ :

- $w$  führt über die ersten  $m$  Knoten, d.h.  $\forall 2 \leq x \leq n-1: v_x \in \{1, 2, \dots, m\}$ .
  - $w$  ist zyklonfrei,  $\mu(w) \leq u_{\max}$  und  $\gamma(w) \leq D_{i,j}$  (durch Def. 4.4 garantiert), also ist  $w \in L_{i,j}^*$ .
- $M$  enthält also gesuchte Wege über  $m$  und die ersten  $m-1$  Knoten.  $P_{i,j}^{(m-1)}$  enthält gemäß (IV) alle gesuchten Wege über die ersten  $m-1$  Knoten (ohne  $m$ ).

Wegen  $P_{i,j}^{(m)} = P_{i,j}^{(m-1)} \cup M$  enthält  $P_{i,j}^{(m)}$  alle Wege aus  $L_{i,j}^*$  über die ersten  $m$  Knoten.

(q.e.d.)

#### **4.2.7 Aufwandsabschätzung**

Wie in Abschnitt 4.1.1 angekündigt, wird für die Aufwandsabschätzung folgende Bedingung vorausgesetzt:

$$\exists k \in \mathbb{N}: \forall i, j \in V, i \neq j: |L_{i,j}| \leq k,$$

d.h. der Umfang der Lösungsmengen ist  $k$ -beschränkt. Ohne Begrenzung wäre ein Vergleich von Wegsuche-Algorithmen sinnlos (alle Verfahren liegen in  $O(2^N)$ ). Aus  $|P_{i,j}| \leq |L_{i,j}|$  folgt schließlich  $|P_{i,j}| \leq k$ , d.h. der Umfang der Protokolle ist  $k$ -beschränkt.

#### **Laufzeitverhalten:**

Vorbereitung: Die Bestwegsuche nach Dijkstra benötigt für planare Graphen  $O(N \cdot \log(N))$  (vgl. [11]). Als 1:N-Verfahren muß es für jeden der  $N$  Knoten wiederholt werden.

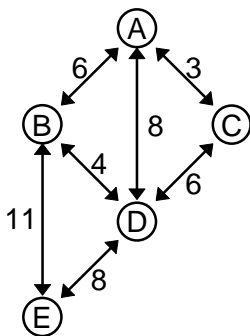
• N:N-Bestwegsuche:	$\in O(N^2 \cdot \log(N))$
• Berechnung der oberen Schranken $D_{i,j}$ :	$N^2$ Schritte
Initialisierung:	$N^2$ Schritte
Konkatenation zweier Wegeprotokolle $P_{i,m}, P_{m,j}$ :	
• Schleifenprüfung ( $w_1 \cap w_2$ ):	$\in O(N \cdot \log(N))$ (Weg als Baum)
• Verkettung $w_1 \bullet w_2$ :	$N$ Schritte (Weg kopieren)
• Anzahl Wegekombinationen aus $P_{i,m}$ und $P_{m,j}$ :	$k^2$ (wegen $ P_{i,j}  \leq k$ )
• Konkatenation insgesamt:	$k^2 \cdot (N \cdot \log(N) + N)$
Anzahl Schleifendurchläufe gesamt:	
	$N \cdot (N-1) \cdot (N-2)$ (wegen $m \neq i \neq j$ )
Korrektur der oberen Schranken:	$N^2$ Schritte
Beseitigen überflüssiger Wege:	$N^2 \cdot k$ Schritte
<hr/>	
<b>Gesamtaufwand:</b>	$N^2 \cdot \log(N) + 2 \cdot N^2 + N \cdot (N-1) \cdot (N-2) \cdot k^2 \cdot (N \cdot \log(N) + N) + N^2 + N^2 \cdot k$
	$\in O(N^4 \cdot \log(N) \cdot k^2)$

**Vergleich mit Original-Algorithmus:**

Der Algorithmus nach Miniéka benötigt in der Originalfassung nur  $O(N^3)$  Laufzeit. Ursache für den Unterschied ist im wesentlichen die fehlende Prüfung der Wege auf Zyklen, die jeweils bis zu  $N \cdot \log_2(N)$  Schritte erfordert.

**Speicherplatzbedarf:**

Der Speicherplatzbedarf des Algorithmus ist vor allem wegen der Protokolle  $P_{i,j}$  sehr groß: Weil für jede der  $N^2$  Relationen gleichzeitig  $k$  Wege mit jeweils bis zu  $N$  Knoten ermittelt werden müssen, liegt der Aufwand in  $O(N^3 \cdot k)$ .

**4.2.8 Beispiel**

nach von	A	B	C	D	E
A	-	6..11	3..8	8..13	16..21
B	6..11	-	9..14	4..9	11..16
C	3..8	9..14	-	6..11	14..19
D	8..13	4..9	6..11	-	8..13
E	16..21	11..16	14..19	8..13	-

Abbildung 4.5: Beispielgraph (links) und zulässige Wertebereiche für die Weglänge je Knotenpaar  $(i,j) \in V^2$ ,  $i \neq j$ : die untere Schranke ist die Länge  $d_{i,j}$  des Bestwegs, die obere Schranke  $D_{i,j}$  sei hier um  $\lambda_2 = 5$  größer als  $d_{i,j}$ .

**Initialisierung** der Protokolle  $P_{i,j}$  mit Kantenverbindungen:

nach von	A	B	C	D	E
A	-	{ (A,B) }	{ (A,C) }	{ (A,D) }	{ }
B	{ (B,A) }	-	{ }	{ (B,D) }	{ (B,E) }
C	{ (C,A) }	{ }	-	{ (C,D) }	{ }
D	{ (D,A) }	{ (D,B) }	{ (D,C) }	-	{ (D,E) }
E	{ }	{ (E,B) }	{ }	{ (E,D) }	-

**1. Schritt:** Verbindungen über Knoten A hinzufügen (neue Wege sind **fett** gedruckt; sie werden verworfen, falls deren Länge außerhalb der zulässigen Grenzen liegt):

nach von	A	B	C	D	E
A	-	{ (A,B) }	{ (A,C) }	{ (A,D) }	{ }
B	{ (B,A) }	-	<b>{ (B,A,C) }</b>	{ (B,D), <del>(B,A,D)</del> }	{ (B,E) }
C	{ (C,A) }	<b>{ (C,A,B) }</b>	-	{ (C,D), <b>(C,A,D)</b> }	{ }
D	{ (D,A) }	{ (D,B), <del>(D,A,B)</del> }	<b>{ (D,C), (D,A,C) }</b>	-	{ (D,E) }
E	{ }	{ (E,B) }	{ }	{ (E,D) }	-

**2. Schritt:** Verbindungen über Knoten B hinzufügen (neue Wege, die bei der Schleifenprüfung durchfallen, sind *kursiv* gedruckt und werden verworfen:

von \ nach	A	B	C	D	E
A	-	{ (A,B) }	{ (A,C), <i>(A,B,A,C)</i> }	{ (A,D), <i>(A,B,D)</i> }	{ (A,B,E) }
B	{ (B,A) }	-	{ (B,A,C) }	{ (B,D) }	{ (B,E) }
C	{ (C,A), <i>(C,A,B,A)</i> }	{ (C,A,B) }	-	{ (C,D), <i>(C,A,D)</i> , <i>(C,A,B,D)</i> }	{ <i>(C,A,B,E)</i> }
D	{ (D,A), <i>(D,B,A)</i> }	{ (D,B) }	{ (D,C), <i>(D,A,C)</i> , <i>(D,B,A,C)</i> }	-	{ (D,E), <i>(D,B,E)</i> }
E	{ (E,B,A) }	{ (E,B) }	{ <i>(E,B,A,C)</i> }	{ (E,D), <i>(E,B,D)</i> }	-

**3. Schritt:** Verbindungen über Knoten C hinzufügen:

von \ nach	A	B	C	D	E
A	-	{ (A,B), <i>(A,C,A,B)</i> }	{ (A,C) }	{ (A,D), <i>(A,B,D)</i> , <i>(A,C,D)</i> , <i>(A,C,A,D)</i> }	{ (A,B,E) }
B	{ (B,A), <i>(B,A,C,A)</i> }	-	{ (B,A,C) }	{ (B,D), <i>(B,A,C,D)</i> , <i>(B,A,C,A,D)</i> }	{ (B,E) }
C	{ (C,A) }	{ (C,A,B) }	-	{ (C,D), <i>(C,A,D)</i> }	{ }
D	{ (D,A), <i>(D,B,A)</i> , <i>(D,C,A)</i> , <i>(D,A,C,A)</i> }	{ (D,B), <i>(D,C,A,B)</i> , <i>(D,A,C,A,B)</i> }	{ (D,C), <i>(D,A,C)</i> }	-	{ (D,E) }
E	{ (E,B,A) }	{ (E,B) }	{ }	{ (E,D) }	-

**4. Schritt:** Verbindungen über Knoten D hinzufügen:

von \ nach	A	B	C	D	E
A	-	{ (A,B), <i>(A,D,B)</i> , <i>(A,B,D,B)</i> , <i>(A,C,B,B)</i> }	{ (A,C), <i>(A,D,C)</i> , + 5 weitere }	{ (A,D), <i>(A,B,D)</i> , <i>(A,C,D)</i> }	{ (A,B,E), <i>(A,D,E)</i> , <i>(A,B,D,E)</i> , <i>(A,C,D,E)</i> }
B	{ (B,A), <i>(B,D,A)</i> , <i>(B,D,B,A)</i> , <i>(B,D,C,A)</i> }	-	{ (B,A,C), <i>(B,D,C)</i> , <i>(B,D,A,C)</i> }	{ (B,D) }	{ (B,E), <i>(B,D,E)</i> }
C	{ (C,A), <i>(C,D,A)</i> , + 5 weitere }	{ (C,A,B), <i>(C,D,B)</i> , <i>(C,A,D,B)</i> }	-	{ (C,D), <i>(C,A,D)</i> }	{ <i>(C,D,E)</i> , <i>(C,A,D,E)</i> }
D	{ (D,A), <i>(D,B,A)</i> , <i>(D,C,A)</i> }	{ (D,B) }	{ (D,C), <i>(D,A,C)</i> }	-	{ (D,E) }
E	{ (E,B,A), <i>(E,D,A)</i> , <i>(E,D,B,A)</i> , <i>(E,D,C,A)</i> }	{ (E,B), <i>(E,D,B)</i> }	{ <i>(E,D,C)</i> , <i>(E,D,A,C)</i> }	{ (E,D) }	-

## 5. Schritt: Verbindungen über Knoten E hinzufügen:

nach von	A	B	C	D	E
A	-	{ (A,B), <del>(A,D,E,B)</del> , + 7 weitere }	{ (A,C), <del>(A,B,E,D,C)</del> , + 7 weitere }	{ (A,D), (A,B,D), (A,C,D), + 4 weitere }	{ (A,B,E), (A,D,E), (A,B,D,E), (A,C,D,E) }
B	{ (B,A), <del>(B,E,D,A)</del> , + 7 weitere }	-	{ (B,A,C), (B,D,C), <del>(B,E,D,C)</del> , + 3 weitere }	{ (B,D), <del>(B,E,D)</del> , <del>(B,D,E,D)</del> }	{ (B,E), (B,D,E) }
C	{ (C,A), <del>(C,E,D,B,A)</del> , + 7 weitere }	{ (C,A,B), (C,D,B), <del>(C,D,E,B)</del> , + 3 weitere }	-	{ (C,D), (C,A,D), <del>(C,D,E,D)</del> , <del>(C,A,D,E,D)</del> }	{ (C,D,E), (C,A,D,E) }
D	{ (D,A), (D,B,A), (D,C,A), + 4 weitere }	{ (D,B), <del>(D,E,B)</del> , <del>(D,E,D,B)</del> }	{ (D,C), (D,A,C), <del>(D,E,D,C)</del> , <del>(D,E,D,A,C)</del> }	-	{ (D,E) }
E	{ (E,B,A), (E,D,A), (E,D,B,A), (E,D,C,A) }	{ (E,B), (E,D,B) }	{ (E,D,C), (E,D,A,C) }	{ (E,D) }	-

Die oberen Schranken müssen nach unten korrigiert werden (sei  $\lambda_1 = 1,2$ ), was neue Wertebereiche für die Weglängen ergibt:

nach von	A	B	C	D	E
A	-	6..7	3..4	8..10	16..19
B	6..7	-	9..11	4..6	11..13
C	3..4	9..11	-	6..7	14..17
D	8..10	4..6	6..7	-	8..10
E	16..19	11..13	14..17	8..10	-

Damit erhält man endgültig die Mengen der gesuchten Wege für alle Knotenpaare. Man erkennt, daß lediglich vier Wege nachträglich gelöscht werden müssen:

nach von	A	B	C	D	E
A	-	{ (A,B) }	{ (A,C) }	{ (A,D), (A,B,D), (A,C,D) }	{ (A,B,E), (A,D,E), (A,B,D,E), (A,C,D,E) }
B	{ (B,A) }	-	{ (B,A,C), (B,D,C) }	{ (B,D) }	{ (B,E), (B,D,E) }
C	{ (C,A) }	{ (C,A,B), (C,D,B) }	-	{ (C,D), <del>(C,A,D)</del> }	{ (C,D,E), <del>(C,A,D,E)</del> }
D	{ (D,A), (D,B,A), (D,C,A) }	{ (D,B) }	{ (D,C), <del>(D,A,C)</del> }	-	{ (D,E) }
E	{ (E,B,A), (E,D,A), (E,D,B,A), (E,D,C,A) }	{ (E,B), (E,D,B) }	{ (E,D,C), <del>(E,D,A,C)</del> }	{ (E,D) }	-

### **4.3 Tiefensuche und Algorithmus nach Dijkstra (1:N-Suchverfahren)**

#### **4.3.1 Allgemeines**

Im Jahre 1959 hat E. W. Dijkstra einen Algorithmus veröffentlicht ([8]), welcher der Bestimmung der kürzesten Wege in einem Graph dient. Der Algorithmus ist einer der bekanntesten der Graphentheorie und zählt nicht zuletzt wegen seiner Effizienz zu den Standardalgorithmen für die Wegsuche in Graphen. Die Ermittlung der Bestwege wird jedoch der Problemstellung nicht gerecht. Durch Erweiterung des Verfahrens auf die Berechnung aller Wege mit Zeitschranke  $D_{i,j}$  verschlechtert sich die Effizienz dramatisch. Daher kann dieses Verfahren nur dazu dienen, die Entfernung zwischen zwei Knoten zu berechnen.

Zur Ermittlung der gesuchten Wege eignet sich eher Tiefensuche, einem Standardverfahren, mit dessen Hilfe man in einem Graph systematisch alle möglichen Wege absuchen kann. Damit das Verfahren terminiert, müssen Schleifen vermieden werden, was jedoch kaum Aufwand verursacht. Werden darüberhinaus obere Schranken für Widerstände berücksichtigt (begrenzte Tiefensuche), so läßt sich dieses Verfahren zur Lösung des Problems „Wegsuche mit Zeitschranke“ optimal einsetzen.

#### **4.3.2 Algorithmus nach Dijkstra**

Von einem Startknoten  $a \in V$  ausgehend werden Kanten nach allen Richtungen hin verfolgt (Suchpfade), wobei der aktuell kürzeste Suchpfad jeweils die Suchrichtung diktiert, so daß die Knoten in der Reihenfolge ihrer Entfernung zu  $a$  aufgesucht werden. Jeder Knoten wird somit genau einmal über den kürzesten Weg besucht. Alternativrouten zu demselben Knoten werden verworfen. Zu weiteren Informationen bezüglich des Verfahrens sei auf die Literatur verwiesen (z.B. [7]).

#### **4.3.3 Grundelemente**

Für jeden Knoten  $v \in V$  steht ein Protokoll  $P_{a,v}$  zur Verfügung, in welchem alle gesuchten Wege zwischen dem Startknoten  $a \in V$  und  $v$  gespeichert werden. Die Protokolle weisen die in Def. 4.3 spezifizierte Form auf.

**4.3.4 Ablauf**

**Gegeben:** Distanzgraph  $G = (V, E, \gamma)$  nach Def. 2.7 mit einem Startknoten  $a \in V$ .

$\psi: V \rightarrow \{\text{frei, gesperrt}\}$  sei Markierung von Knoten.

**Gesucht:** die besten zyklensfreien Wege von  $a$  nach  $v$  ( $v \in V \setminus \{a\}$ ) mit Zeitschranke  $D_{a,v}$  gemäß Def. 4.1 und maximaler Umsteigehäufigkeit  $u_{\max}$ . Die gesuchten Wege werden in  $P_{a,v}$  protokolliert.

**Vorbereitung:**

Führe Algorithmus nach Dijkstra mit Startknoten  $a$  aus  $\Rightarrow \forall v \in V \setminus \{a\}: d_{a,v}$ ;

$\forall v \in V \setminus \{a\}$ :

$D_{a,v} := d_{a,v} + \lambda_2$ ;

provisorische obere Schranke;

$\psi(v) := \text{frei}$ ;

Freigabe aller Knoten;

**Hauptteil:**

Tiefensuche ( $a, 0, 0, 1$ );

$\forall v \in V \setminus \{a\}: P_{a,v} := \{ w \in P_{a,v} \mid \gamma(w) \leq \Gamma(d_{a,v}) \}$ ;

Korrektur der oberen Schranke mit  $\Gamma$  gemäß Def. 3.2;

**Prozedur** Tiefensuche ( $i \in V$ ;  $r \in \mathbb{R}^+$ ;  $u, t \in \mathbb{N}$ )

$i$  = aktueller Knoten

$r$  = Entfernung von  $i$  zu  $a$ ;

$u$  = aktuelle Umsteigehäufigkeit;

$t$  = Anzahl Knoten im Suchpfad

falls  $\psi(i) = \text{frei}$  und  $r < D_{a,i}$  und  $u \leq u_{\max}$ :

Schleifen- und Schrankenprüfung;

$\psi(i) := \text{gesperrt}$ ;

Wiederholung ausschließen;

$v_i := i$ ;

aktuellen Knoten registrieren;

$P_{a,i} := P_{a,i} \cup \{ (v_1, v_2, \dots, v_t) \}$ ;

Suchpfad ist ein gesuchter Weg

$\forall e \in E$  mit  $e = (i, j)$ : Tiefensuche ( $j, r + \gamma(e), u + \mu(e), t + 1$ ); (rekursiver Aufruf)

$\psi(i) := \text{frei}$ ;

Knoten freigeben („backtracking“);

**4.3.5 Korrektheitsbeweis**

Da von einem Knoten  $v \in V$  ausgehend alle Kanten verfolgt werden, genügt zu zeigen:

1) In die Protokolle  $P_{a,v}$  werden nur Wege aufgenommen, die zur Lösungsmenge  $L_{a,v}^*$  gehören, wobei gilt:

$$L_{a,v}^* = \{ w = (v_1, v_2, \dots, v_n) \in W_{a,v} \mid \forall 1 \leq x < y \leq n: v_x \neq v_y \text{ und } \gamma(w) \leq D_{a,v} \text{ und } \mu(w) \leq u_{\max} \}.$$

2) Verworfen Suchpfade werden in späteren Berechnungen nicht mehr benötigt.

zu 1)

Sei  $i \in V$  der aktuelle Knoten und  $w = (v_1, v_2, \dots, v_t)$  aktueller Suchpfad,  $\forall 1 \leq x \leq t: v_x \in V$ ,  $v_1 = a$ ,  $v_t = i$ ,  $t \in \mathbb{N}$ .  $w$  wird genau dann ins Protokoll  $P_{a,i}$  aufgenommen, wenn die Eingangsbedingung von Prozedur Tiefensuche erfüllt ist, wenn also gilt:

$$\psi(i) = \text{frei} \text{ und } r < D_{a,i} \text{ und } u \leq u_{\max},$$

was gleichbedeutend ist mit

$$(\forall 1 \leq x < t: v_x \neq i) \text{ und } \gamma(w) \leq D_{a,i} \text{ und } \mu(w) \leq u_{\max}.$$

Da jede Teilfolge  $(v_1, v_2, \dots, v_y)$  von  $w$  ( $1 \leq y \leq t$ ) die Eingangsbedingung erfüllen muß, gilt

$$\forall 1 \leq x < y \leq t: v_x \neq v_y \text{ und } \gamma(w) \leq D_{a,i} \text{ und } \mu(w) \leq u_{\max}$$

und damit  $w \in L_{a,i}^*$ . Also wird  $w$  ins Protokoll  $P_{a,i}$  aufgenommen, wenn  $w \in L_{a,i}^*$ .

zu 2)

Die Behauptung folgt direkt aus Satz 4.1 (Prinzip der Vorentscheidung).

(q.e.d.)

#### 4.3.6 Aufwandsabschätzung

Wie in Abschnitt 4.1.1 angekündigt, wird für die Aufwandsabschätzung folgende Bedingung vorausgesetzt:

$$\exists k \in \mathbb{N}: \forall v \in V \setminus \{a\}: |L_{a,v}| \leq k.$$

Aus  $|P_{a,v}| \leq |L_{a,v}|$  folgt  $|P_{a,v}| \leq k$ , d.h. der Umfang von Protokollen und Lösungsmengen ist  $k$ -beschränkt.

#### Laufzeitverhalten:

Vorbereitung:

- Bestwegsuche nach Dijkstra für planare Graphen:  $\in O(N \cdot \log(N))$  (siehe [11])
- Initialisierung von  $D_{a,v}$  und  $\psi(v)$ :  $2 \cdot N$  Schritte

Prozedur Tiefensuche (Besuch eines Knotens  $i \in V$ ):

- Eingangsbedingung (Schleifenprüfung u.a.): 3 Schritte
- aktuellen Suchpfad ins Protokoll  $P_{a,i}$  kopieren:  $t$  Schritte  $\in O(N)$
- Knoten  $i$  sperren bzw. freigeben: je 1 Schritt
- rekursive Aufrufe von Tiefensuche:  $\in O(1)$  (bei planarem  $G$ )

Anzahl Knotenbesuche (es gibt  $\leq k$  Wege nach  $i \in V$ ):  $|V| \cdot k = N \cdot k$

---

Gesamtaufwand für Verfahren:  $O(N \cdot \log(N)) + 2 \cdot N + N \cdot k \cdot (5 + O(N) + O(1)) \in \underline{O(N^2 \cdot k)}$

**Gesamtaufwand** für  $N:N$ -Wegsuche (Verfahren  $N$ -mal ausführen):  $\in \underline{O(N^3 \cdot k)}$

**Speicherplatzbedarf:**

Maßgebend ist der Umfang der Wegeprotokolle  $P_{a,v}$  beim Terminieren des Algorithmus: Für alle Zielknoten  $v \in V$  müssen gleichzeitig  $k$  Wege bestehend aus bis zu  $N$  Knoten ermittelt werden, so daß der Speicheraufwand in  $O(N^2 \cdot k)$  liegt.

**4.3.7 Beispiel**

Gegeben sei ein Distanzgraph  $G = (V, E, \gamma)$  mit  $V = \{A,B,C,D,E\}$  und Kanten(-längen) gemäß Abbildung 4.6. A sei der Startknoten.

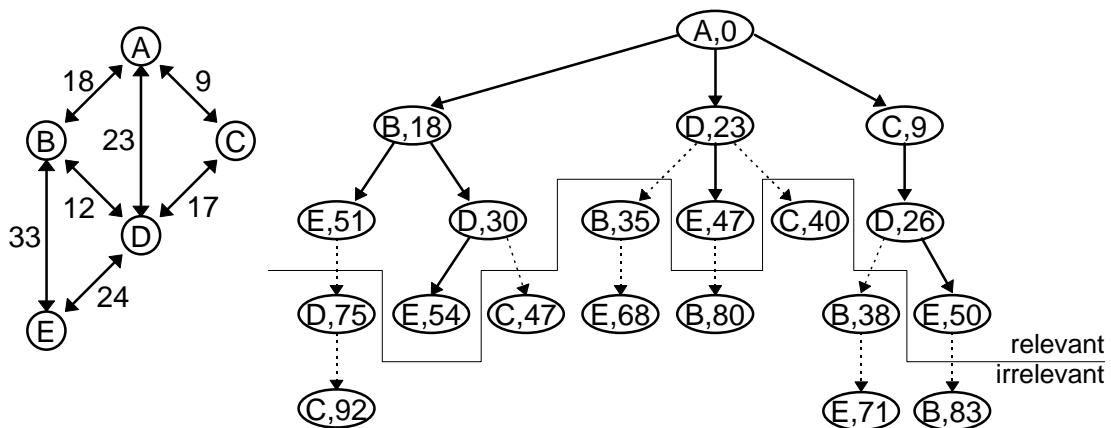


Abbildung 4.6: Beispielgraph  $G$  (links) und vollständiger Suchbaum mit Startknoten A als Ausgangspunkt, wobei nur schleifenfreie Wege in  $G$  dargestellt sind. Für jeden erreichten Knoten ist im Suchbaum die Entfernung von A angegeben. Die Querlinie trennt den relevanten Teil des Baums vom Bereich, in dem Entfernungsgrenzen überschritten sind. Einige Teilbäume werden nicht untersucht (gestrichelte Pfeile).

Zielknoten	B	C	D	E
$d_{A,v}$	18	9	23	47
$D_{A,v} = d_{A,v} + 15$	33	24	38	62
Protokolle $P_{A,v}$ nach Abschluß der Wegsuche	{ (A,B) } 18	{ (A,C) } 9	{ (A,D), 23 (A,C,D), 26 <del>(A,B,D)</del> } 30	{ (A,D,E), 47 (A,C,D,E), 50 (A,B,E), 51 (A,B,D,E) } 54
$\Gamma(d_{A,v}) = 1,2 \cdot d_{A,v}$	21,6	10,8	27,6	56,4

Nach Abschluß der Wegsuche fällt aufgrund der Absenkung der oberen Schranken die Route (A,B,D) aus dem Protokoll  $P_{A,D}$ .

## **4.4 Prinzip der Linienverfolgung**

### **4.4.1 Allgemeines**

Die Linienverfolgung ist ein vom Verfasser dieser Diplomarbeit entwickeltes Verfahren zur Suche der für die Verkehrsumlegung relevanten Wege in einem ÖV-Netz. Der Algorithmus ist im Sinne der Problemstellung „maßgeschneidert“ und arbeitet auf der Grundlage von ÖV-Netzmodellen nach Def. 2.1. Die Suche beginnt in einem Startbezirk und führt zu allen anderen Bezirken (1:N-Suchverfahren).

### **4.4.2 Grundprinzip**

Von allen in den Startbezirk eingebundenen Haltestellen ausgehend, werden vorhandene Linien bis zu ihrer Endhaltestelle verfolgt und an jeder besuchten Haltestelle der zurückgelegte Weg protokolliert (Zwischenspeichern von Suchpfaden). Auf diese Weise werden alle Haltestellen registriert, welche ohne Umsteigen vom Startbezirk aus erreichbar sind. Im nächsten Schritt werden alle Linien vollständig „durchfahren“, wobei an jeder besuchten Haltestelle die Suchpfade aufgegriffen und in Richtung der Linie verlängert werden. Somit erhält man alle Haltestellen, die mit einmaligem Umsteigen erreichbar sind. Der letztgenannte Schritt wird  $u_{\max}$ -mal durchgeführt. Während beim Dijkstra-Algorithmus Weglängen die Suchrichtung diktieren, orientiert sich dieses Verfahren an den Linien, so daß die Haltestellen in der Reihenfolge der Umsteigehäufigkeit aufgesucht werden.

### **4.4.3 Grundelemente und -operationen**

Für jede Haltestelle muß protokolliert werden, auf welchem Weg sie vom Startbezirk aus zu erreichen ist. Hierzu werden Wegbeschreibungen verwendet, welche leicht erweiterbar sind:

**Def. 4.5:** Menge der **ÖV-Wegbeschreibungen:**

$$WB = H \times (L \times H)^n, \quad 0 \leq n \leq u_{\max} + 1$$

$u_{\max}$  ist die maximal zulässige Umsteigehäufigkeit eines ÖV-Weges gemäß

Def. 3.3. Sei  $w = (h_0, l_1, h_1, \dots, l_n, h_n) \in WB$ .

$h_0$  :                   Einstieghaltestelle

$h_n$  :                   Ausstiegshaltestelle (bei  $n \geq 1$ )

$h_i$  mit  $1 \leq i \leq n-1$ : Umstiegshaltestellen (nur bei  $n \geq 2$ ), wobei gilt:  
 $(l_i, l_{i+1}, h_i) \in U$     (Umsteigen an Haltestelle  $h_i$ )

$l_i$  mit  $1 \leq i \leq n$ :   benutzte Linien (bei  $n \geq 1$ ), wobei gilt:  
 $(h_{i-1}, h_i, l_i) \in T$    (Fahrt mit Linie  $l_i$ )

Wegbeschreibungen mit  $n=0$  treten nur zu Beginn der Wegsuche auf. Sie werden vom Algorithmus sukzessive erweitert (Entwicklung nach der Umsteigehäufigkeit):

**Def. 4.6:** **Umsteigehäufigkeit**  $\mu$  einer ÖV-Wegbeschreibung:

$$\mu: WB \rightarrow \{-1, 0, 1, \dots, u_{\max}\}$$

Sei  $w = (h_0, l_1, h_1, \dots, l_n, h_n) \in WB$ . Dann ist  $\mu(w) = n - 1$ .

Um die Entstehung von Schleifen in einem Weg zu verhindern, muß die Folge der auf diesem Weg liegenden Haltestellen bekannt sein. Zur Ermittlung der Haltestellenfolge anhand einer Wegbeschreibung sei eine Funktion  $\varphi$  definiert:

**Def. 4.7:** **Haltestellenfolge** einer Wegbeschreibung (besuchte Haltestellen):

$$\varphi: WB \rightarrow H^+$$

Sei  $w = (k_0, l_1, k_1, \dots, l_m, k_m) \in WB$ .

**Fall 1:** Für  $m = 0$  ist  $\varphi(w) = (k_0)$ .

**Fall 2:**  $m \geq 1$ :

Sei  $\forall 1 \leq i \leq m$ :  $l_i = (h_{1}^{(i)}, h_{2}^{(i)}, \dots, h_{n(i)}^{(i)})$ ,  $k_{i-1} = h_{A(i)}^{(i)}$  (Einstiegshaltestelle)

und  $k_i = h_{E(i)}^{(i)}$  für  $1 \leq A(i) < E(i) \leq n(i)$  (Ausstiegshaltestelle)

Dann ist

$$\begin{aligned} \varphi(w) = & (h_{A(1)}^{(1)}, h_{A(1)+1}^{(1)}, h_{A(1)+2}^{(1)}, \dots, h_{E(1)}^{(1)}, \\ & h_{A(2)+1}^{(2)}, h_{A(2)+2}^{(2)}, \dots, h_{E(2)}^{(2)}, \\ & \dots, \\ & h_{A(n)+1}^{(n)}, h_{A(n)+2}^{(n)}, \dots, h_{E(n)}^{(n)} ). \end{aligned}$$

Von großer Bedeutung für die Wegsuche ist die quantitative Bewertung von Wegbeschreibungen, welche sich analog zur Berechnung des Widerstands von ÖV-Wegen durchführen läßt:

**Def. 4.8:** Erweiterung der **Bewertungsfunktion** auf Wegbeschreibungen:

$$\omega: WB \rightarrow IR_0^+$$

Sei  $w = (h_0, l_1, h_1, \dots, l_n, h_n) \in WB$  und  $b_0 \in B$  der Startbezirk. Dann gilt:

(1) Für  $n = 0$  ist  $\omega(w) = \omega((b_0, h_0))$ . (nur Zugangszeit)

(2) Für  $n \geq 1$  gilt:

$$\begin{aligned} \omega(w) = & \omega((b_0, h_0)) + g_5 \cdot EWZ + \sum_{i=1}^n \omega((h_{i-1}, h_i, l_i)) + \sum_{i=1}^{n-1} \omega((l_i, l_{i+1}, h_i)) \\ & + g_6 \cdot \text{Kosten} + g_7 \cdot \max\{\tau(l_i) \mid 1 \leq i \leq n\} + g_8 \cdot \text{SBA} \end{aligned}$$

$$EWZ = 0,5 \cdot \min\{\tau(l_1), EWZ_{\max}\}$$

Der Widerstand einer Wegbeschreibung (vgl. Def. 2.6e) besteht aus

- den Einzelwiderständen von Zugangsfußweg, Fahrten und Umsteigebeziehungen, falls vorhanden,
- der gewichteten Einstiegswartezeit (EWZ),
- den in einen Zeitwert umgerechneten Fahrtkosten,
- dem gewichteten Schnellbahnanteil (SBA),
- der gewichteten maximalen Taktfolgezeit aller benutzten Linien.

**Def. 4.9:** Ein **ÖV-Wegeprotokoll**  $Q_h$  für eine Haltestelle  $h \in H$  ist eine Menge von ÖV-Wegbeschreibungen zwischen dem Startbezirk  $b_0 \in B$  und  $h$ :

$$Q_h \subseteq \{ w = (h_0, l_1, h_1, \dots, l_n, h_n) \in WB \mid (b_0, h_0) \in F \text{ und } h_n = h \}$$

Ein ÖV-Wegeprotokoll  $Q_b$  für einen Verkehrsbezirk  $b \in B \setminus \{b_0\}$  ist eine Menge von ÖV-Wegbeschreibungen zwischen dem Startbezirk  $b_0 \in B$  und einer in  $b$  eingebundenen Haltestelle:

$$Q_b \subseteq \{ w = (h_0, l_1, h_1, \dots, l_n, h_n) \in WB \mid (b_0, h_0) \in F \text{ und } (b, h_n) \in F \}$$

Damit die Anzahl der Wege in  $Q_h$  nicht überhand nimmt, müssen sinnvolle Wege selektiert werden. Leider sind zu Beginn der Wegsuche keine absoluten Zeitschranken  $D_{i,j}$  verfügbar. Die Bestwegsuche mit Hilfe des Dijkstra-Algorithmus funktioniert nicht auf Grundlage von ÖV-Netzmodellen. Entscheidendes Kriterium bei der Selektion von Wegen ist deshalb die Länge des kürzesten Weges im Protokoll  $Q_h$ : Alternativrouten werden verworfen, wenn deren Widerstand gegenüber dem aktuell kürzesten Weg um einen Differenzbetrag zu groß ist.

**Def. 4.10:** **Selektive Vereinigung**  $\cup_k$  zweier Mengen von Wegbeschreibungen:

Seien  $M_1, M_2 \subseteq WB$  und  $d = \min\{\omega(w) \mid w \in M_1 \cup M_2\}$ . Dann ist

$$M_1 \cup_k M_2 = \{ w \in M_1 \cup M_2 \mid \omega(w) \leq d + \lambda_2 \}$$

$\lambda_2$  ist der konstante Zeitzuschlag aus Def. 3.2.

Zur Darstellung des Rechenfortschritts im Algorithmus erhalten alle Wegeprotokolle für Haltestellen  $h \in H$  zusätzlich einen Hochindex:

$$Q_h^{(0)}, Q_h^{(1)}, \dots, Q_h^{(U_{\max})}$$



Induktionsanfang (u=0):

Die Initialisierung des Algorithmus ergibt:

$$\forall h \in H: \quad Q_h^{(0)} := \{ (h) \}, \text{ falls } h \text{ in den Startbezirk } b_0 \text{ eingebunden ist, d.h. } (b_0, h) \in F, \\ Q_h^{(0)} := \{ \}, \quad \text{sonst}$$

Die Wegbeschreibungen in  $Q_h^{(0)}$  sind direkte Verbindungen vom Startbezirk zur Haltestelle  $h$  (falls vorhanden) und damit zyklensfrei, deren Länge innerhalb der oberen Schranke  $d + \lambda_2$ , weil es keine kürzere Verbindung gibt. Damit ist die Behauptung erfüllt.

Induktionsvoraussetzung (IV): Für ein  $0 \leq u \leq u_{\max} - 1$  gelte  $\mathcal{B}(u)$  (s.o.).

Induktionsbehauptung (IB):  $\mathcal{B}(u+1)$  für  $u$  aus (IV), d.h.:

$$\forall h \in H: \quad \text{Sei } d = \min\{ \omega(w) \mid Q_h^{(u+1)} \}, \text{ dann ist} \\ Q_h^{(u+1)} = \{ w \in WB \mid \varphi(w) = (h_1, h_2, \dots, h_n) \text{ mit } (b_0, h_1) \in F, h_n = h, \\ \forall 1 \leq x < y \leq n: h_x \neq h_y \text{ und } \omega(w) \leq d + \lambda_2 \text{ und } \mu(w) \leq u+1 \}$$

Induktionsschritt:

Man betrachte eine beliebige Haltestelle  $h \in H$ . Im Schleifendurchlauf  $u$  werden alle Linien „durchfahren“. Linien, welche  $h$  nicht anfahren, sind hier nicht von Belang, da sie  $Q_h^{(u+1)}$  nicht verändern. Betrachte also:

$\forall l = (h_1, h_2, \dots, h_n) \in L$  mit  $h \in l$ : Sei  $1 \leq x \leq n$  der Index mit  $h_x = h$ .

Falls  $x=1$ , dann ist  $l$  unbedeutend, da  $Q_h^{(u+1)}$  nicht verändert wird.

Falls  $x > 1$ , dann gilt im Durchlauf  $i=x-1$  der inneren Schleife:

$$R = \bigcup_{r=1}^i \{ w \in Q_{h_r}^{(u)} \mid \mu(w) = u \text{ und } w \text{ zyklensfrei} \} \quad (\text{selektive Vereinigung})$$

Sei  $M = \{ w \bullet (l, h_{i+1}) \mid w \in R \}$ . Man beachte, daß gilt:  $h_{i+1} = h_x = h$ , also  $M = \{ w \bullet (l, h) \mid w \in R \}$ .

Bei der Konkatenation „ $\bullet$ “ wird jede Wegbeschreibung um eine Fahrt ergänzt, d.h. es fällt ein zusätzlicher Umsteigevorgang an. Da jeder Weg  $w \in R$  Umsteigehäufigkeit  $u$  besitzt, gilt:

$$\forall w \in M: \mu(w) = u+1.$$

$\forall w \in R$ : Falls  $w$  die Schleifenprüfung  $h_{i+1} = h \notin \varphi(w)$  besteht, gilt:

$$Q_{h_{i+1}}^{(u+1)} = Q_h^{(u+1)} = Q_h^{(u)} \cup_k M.$$

Gemäß (IV) gilt:  $\forall w \in Q_h^{(u)}: \mu(w) \leq u$  und  $w$  zyklensfrei.

Daher gilt für alle Wege  $w \in Q_h^{(u+1)}$ :

(1)  $\mu(w) \leq u+1$ .

(2)  $w$  ist zyklensfrei ( $h \notin \varphi(w)$ ).

(3) Sei  $d = \min\{ \omega(w) \mid Q_{h_{i+1}}^{(u)} \}$ . Dann ist  $\omega(w) \leq d + \lambda_2$  (durch Def. 4.10 garantiert).

Damit gilt die Behauptung.

(q.e.d.)

#### 4.4.6 Aufwandsabschätzung

Für die Aufwandsabschätzung wird analog zu Abschnitt 4.2.7 (Algorithmus nach Minieka) folgende Bedingung vorausgesetzt:

$$\exists k \in \mathbb{N}: (|R| \leq k \text{ und } \forall h \in H: |Q_h| \leq k).$$

#### Laufzeitverhalten:

Initialisierung:	$ H $ Schritte	$\in O(N)$
Suchpfade in R aufnehmen:	$k$ Schritte	(je Haltestelle)
Schleifenprüfung $h_{i+1} \in \varphi(w)$ :	$\log_2( H )$ Schritte	(Weg als Binärbaum)
Suchpfad kopieren und erweitern:	$ H $ Schritte	
Neuen Weg in Protokoll einfügen:	$\log_2(k)$ Schritte	(binäres Einfügen)
Anzahl Schleifendurchläufe:	$u_{\max} \cdot  L  \cdot \text{konst} \cdot k$	$\in O(N \cdot k)$
Abschluß:	$ B  \cdot \text{konst}$	$\in O(N)$

Gesamtaufwand für Verfahren:

$$|H| + u_{\max} \cdot |L| \cdot \text{konst} \cdot (k + k \cdot (\log_2(|H|) + |H| + \log_2(k))) + |B| \cdot \text{konst} \quad \in \underline{O(N^2 \cdot k)}$$

**Gesamtaufwand** für N:N-Wegsuche (Verfahren N-mal ausführen):  $\in \underline{O(N^3 \cdot k)}$

#### Bemerkung:

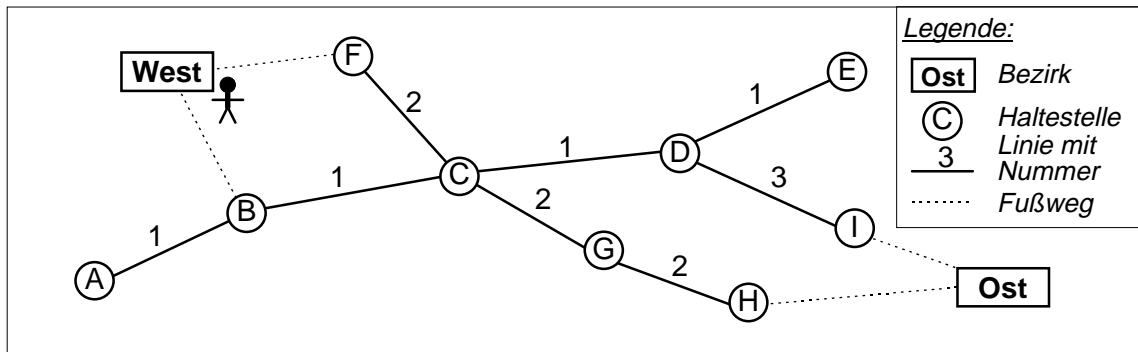
Der Gesamtaufwand ist von derselben Größenordnung wie der Aufwand für die Tiefensuche. Die Linienverfolgung hat aber den Vorteil, daß die Konvertierung des ÖV-Netzes in einen Distanzgraph entfällt und daß die Anzahl der Haltestellen und Linien wesentlich kleiner ist als die Anzahl der Knoten und Kanten im äquivalenten Distanzgraph.

#### Speicherplatzbedarf:

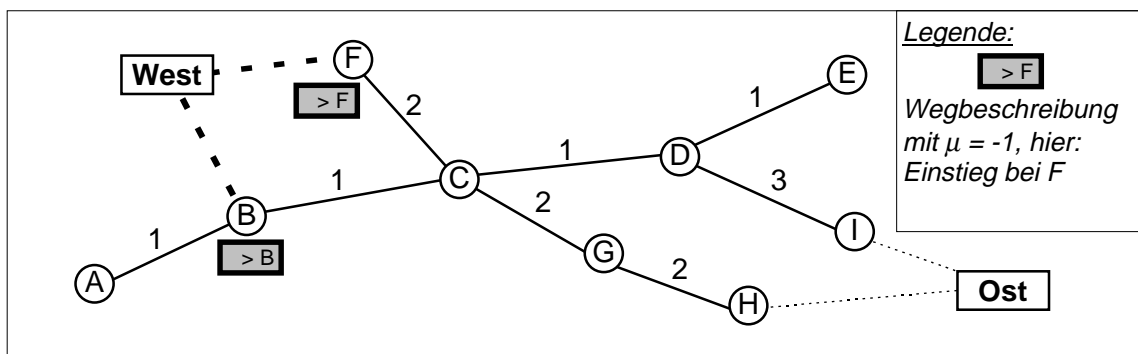
Maßgebend ist der Umfang der Wegeprotokolle  $Q_h$  beim Terminieren des Algorithmus. Für alle Haltestellen  $h \in H$  müssen gleichzeitig  $k$  Wege bestehend aus bis zu  $|H|$  Haltestellen ermittelt werden, so daß der Speicheraufwand in  $O(|H|^2 \cdot k) = \underline{O(N^2 \cdot k)}$  liegt.

**4.4.7 Beispiel**

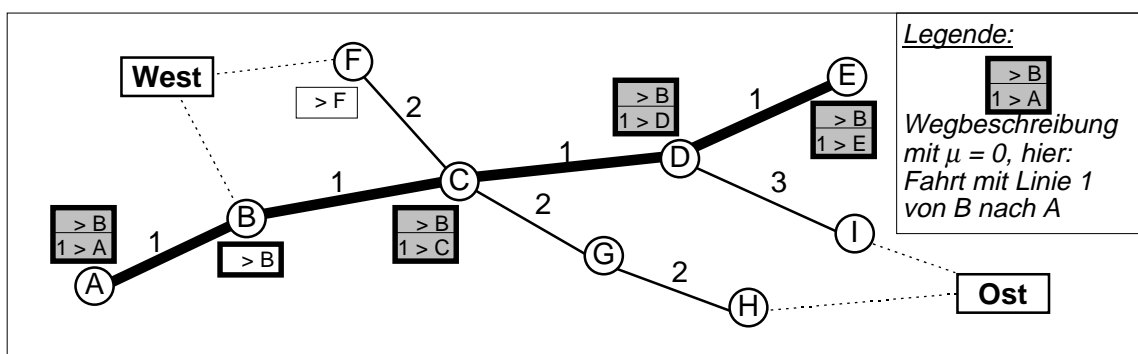
Gegeben sei ein ÖV-Netzmodell wie in Abbildung 4.7 dargestellt. Bezirk West sei Startbezirk. Gesucht sind alle sinnvollen Wege von West nach Ost. Aus Gründen der Übersichtlichkeit wird im Folgenden auf die Berücksichtigung von Widerständen verzichtet.



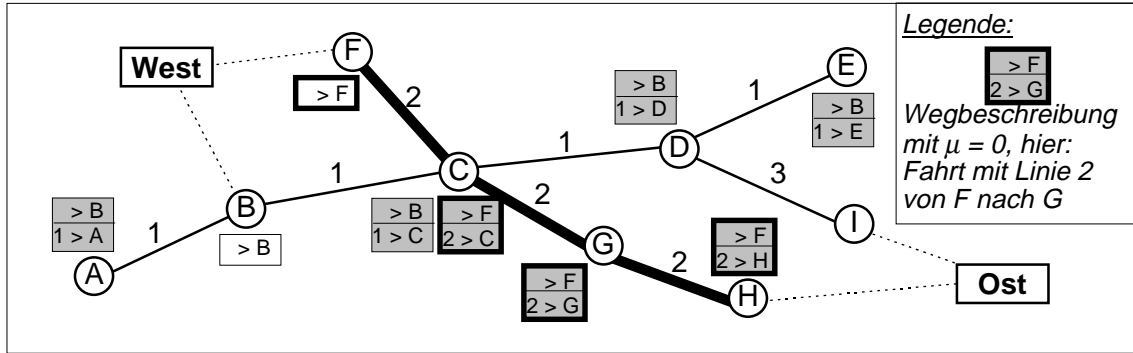
**Abbildung 4.7:** exemplarisches ÖV-Netzmodell mit 2 Bezirken, 9 Haltestellen und 3 Linien (im Prinzip 6 Linien). Bezirk West ist Ausgangspunkt der Wegsuche.



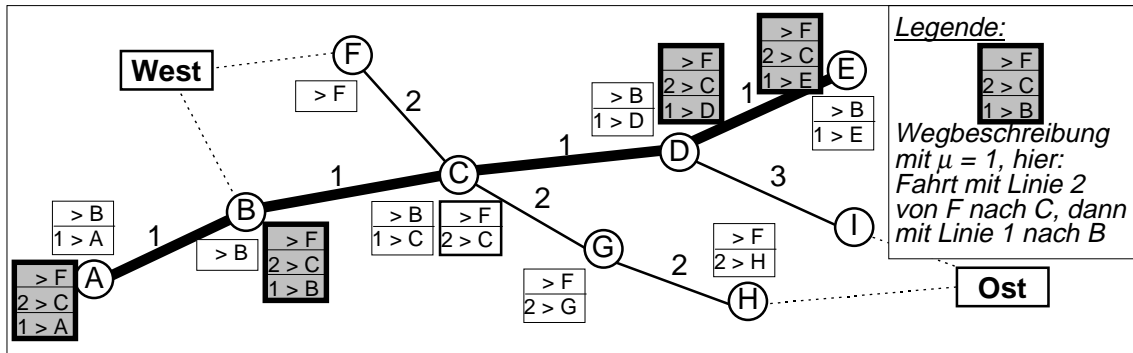
**Abbildung 4.8:** alle möglichen **Anmarschwege** verfolgen: vom Startbezirk aus sind B und F direkt erreichbar. An den eingebundenen Haltestellen werden Wegbeschreibungen (Suchpfade) protokolliert.



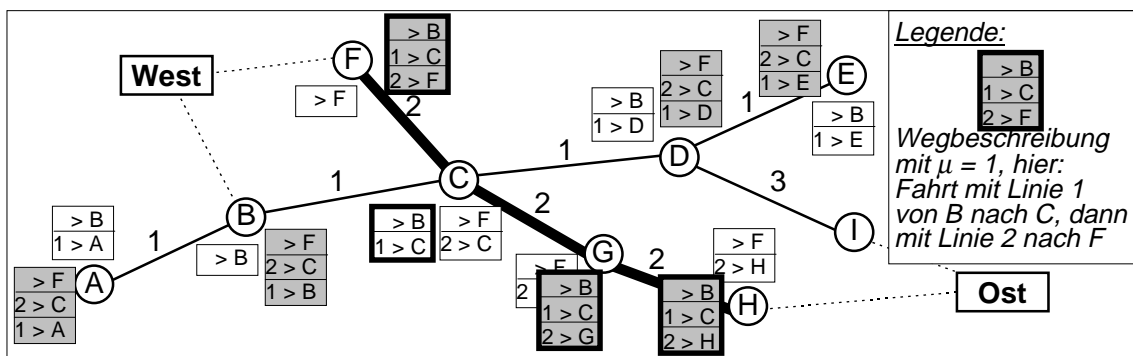
**Abbildung 4.9:** Schleifendurchlauf  $u = 0$ : Verfolgen von **Linie 1** bis zu den Endhaltestellen: Bei B Einsteigende werden aufgenommen und nach A, C, D und E befördert, was dort jeweils protokolliert wird.



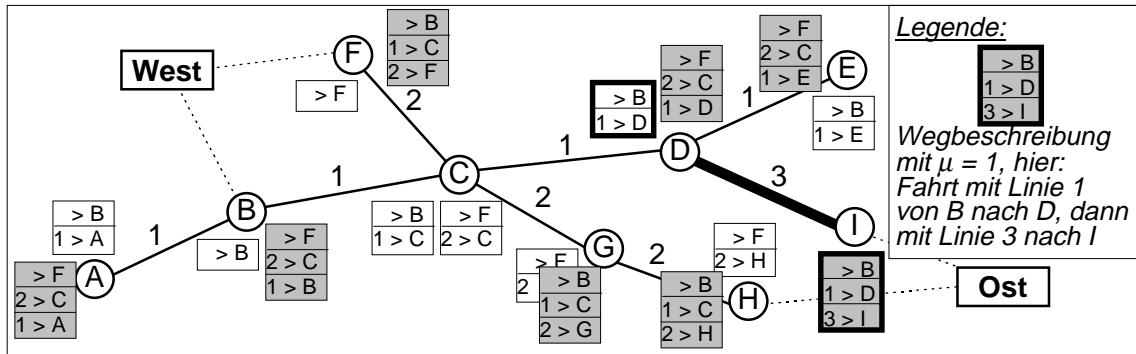
**Abbildung 4.10:** Schleifendurchlauf  $u = 0$ : Verfolgen von **Linie 2**: Bei F Einsteigende werden aufgenommen und nach C, G und H befördert. Verfolgen von **Linie 3** ändert die Situation nicht, da bei D und I keine Einsteiger ( $\mu = -1$ ) vorhanden sind. Damit stehen alle Haltestellen fest, welche ohne Umsteigevorgänge von Bezirk West aus erreichbar sind (graue Wegbeschreibungen).



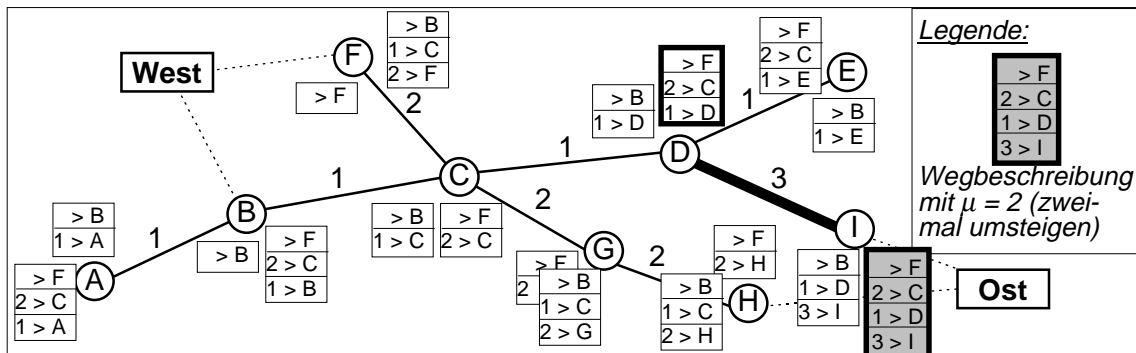
**Abbildung 4.11:** Schleifendurchlauf  $u = 1$ : Verfolgen von **Linie 1**: Bei C Umsteigende von Linie 2 werden aufgenommen und nach A, B, D und E befördert. Andere Suchpfade mit  $\mu=0$  sind Fahrten mit Linie 1. Sie werden übergangen, weil Umsteigen von Linie 1 nach Linie 1 verboten ist.



**Abbildung 4.12:** Schleifendurchlauf  $u = 1$ : Verfolgen von **Linie 2**: Bei C Umsteigende von Linie 1 werden aufgenommen und nach A, B, D und E befördert.



**Abbildung 4.13:** Schleifendurchlauf  $u = 1$ : Verfolgen von **Linie 3**: Bei D Umsteigende von Linie 1 werden aufgenommen und nach I befördert. Damit stehen alle Haltestellen fest, welche mit einmaligem Umsteigen vom Startbezirk aus erreichbar sind (graue Wegbeschreibungen).



**Abbildung 4.14:** Schleifendurchlauf  $u = 2$ : Verfolgen von **Linie 1** ändert die Situation nicht, weil in allen Suchpfaden bei A, B, D und E mit  $\mu = 1$  zuletzt Linie 1 benutzt wurde. Für **Linie 2** gilt dasselbe in F, G und H. Mit **Linie 3** können bei D Umsteigende ( $\mu = 1$ ) aufgenommen und nach I befördert werden. Damit steht fest, daß nur I mit zweimaligem Umsteigen vom Startbezirk aus erreichbar ist. Bei allen anderen Haltestellen würde die Schleifenbedingung verletzt.

Die Schleifendurchläufe 3, 4 und 5 ( $= u_{max}$ ) ändern die Situation nicht, d.h. die maximale Umsteigehäufigkeit im Netz ist gleich 2.

Zum Schluß werden bei jedem Bezirk außer dem Startbezirk alle möglichen Abgangsfußwege untersucht, d.h. von allen eingebundenen Haltestellen des Bezirks werden Wegbeschreibungen aufgegriffen. Für Bezirk Ost ergeben sich folgende vier Wegbeschreibungen:

- |       |
|-------|
| > F   |
| 2 > H |
- |       |
|-------|
| > B   |
| 1 > C |
| 2 > H |
- |       |
|-------|
| > B   |
| 1 > D |
| 3 > I |
- |       |
|-------|
| > F   |
| 2 > C |
| 1 > D |
| 3 > I |

oder gemäß Def. 4.5 formuliert:

$$(F,2,H), (B,1,C,2,H), (B,1,D,3,I), (F,2,C,1,D,3,I).$$

## 4.5 Ansätze zur Parallelisierung der Wegsuche

### 4.5.1 Parallelisierung des Verfahrens nach Minieka

Das modifizierte Verfahren nach Minieka läßt sich auf zwei verschiedenen Ebenen parallelisieren:

1. Innerhalb der äußeren Schleife (Laufindex  $m$ ) kann man für jede Relation  $(i,j)$  mit  $i \neq j$  einen Prozeß  $S_{i,j}$  starten, der Wege  $w_{i,m}$  und  $w_{m,j}$  kombiniert und für das Protokoll  $P_{i,j}$  selektiert. Die äußerste Schleife muß die Prozesse synchronisieren, da die Reihenfolge, in der  $m$  die Werte zwischen 1 und  $N$  annimmt, für alle Relationen  $(i,j)$  verbindlich eingehalten werden muß: Die Prozesse greifen auf verschiedene Ergebnisse des vorherigen Berechnungsschrittes  $P_{i,j}^{(m-1)}$  zu (lesender Zugriff ist gleichzeitig möglich). Beim schreibenden Zugriff von  $S_{i,j}$  auf  $P_{i,j}^{(m)}$  gibt es keine Kollisionen mit anderen Prozessen.
2. Die Kombination von Wegen aus  $P_{i,m}^{(m-1)}$  und  $P_{m,j}^{(m-1)}$  kann ebenfalls parallelisiert werden, indem man für jede Kombination einen Prozeß startet, der die Schleifenprüfung durchführt und den neuen Weg erzeugt, falls dessen Länge die Grenze  $D_{i,j}$  nicht überschreitet. Die reinen Lesezugriffe auf  $P_{i,m}^{(m-1)}$  und  $P_{m,j}^{(m-1)}$  sind unproblematisch, aber der gemeinsame Schreibzugriff auf  $P_{i,j}^{(m)}$  muß synchronisiert werden. Das Ergebnis ist auf alle Fälle korrekt, da kein Prozeß gleichzeitig aus  $P_{i,j}^{(m)}$  liest.

Da der zweite Ansatz die Komplexität des Algorithmus nicht verbessert, wird er nicht weiter untersucht.

#### Ablauf:

```

FOR m:= 1 TO N DO PARBEGIN
  FOR i:= 1 TO N DO
    FOR j:= 1 TO N DO
      IF i ≠ m AND j ≠ m AND i ≠ j THEN
        (*) Pi,j(m) := Pi,j(m-1) ∪ ( Pi,m(m-1) • Pm,j(m-1) );
      END;
    END;
  END;
PAREND;

```

⋮ äußere Schleife synchronisiert  
 ⋮ den Ablauf;  
  
 ⋮ paralleles Ausführen der Berech-  
 ⋮ nungen für jedes Knotenpaar (i,j);

Die Schlüsselworte „PARBEGIN“ und „PAREND“ umrahmen den Block, innerhalb welchem Aktionen parallel ablaufen können. Die FOR-Schleifen (Iteratoren) arbeiten zwar sequentiell, stoßen aber die Berechnung (\*) bei jedem Schleifendurchlauf als unabhängiger Prozeß an.

**Laufzeitverhalten:**

Unter der Annahme, daß alle nicht-sequentiellen Vorgänge echt parallel ausgeführt werden, ergeben sich folgende Werte:

Vorbereitung und Initialisierung:	$\in O(N^2 \cdot \log(N) + 2 \cdot N^2)$ (wie bisher)
Konkatenation zweier Wegeprotokolle $P_{i,m}, P_{m,j}$ :	$\in O(k^2 \cdot (N \cdot \log(N) + N))$ (wie bisher)
Anzahl der Konkatenationen (von Protokollen):	$N^2$ Prozesse aufrufen
Konkatenationen insgesamt:	$N^2 + k^2 \cdot (N \cdot \log(N) + N)$
Anzahl Schleifendurchläufe:	$N$ (äußere Schleife)
Korrektur der oberen Schranken:	$N^2$ Schritte
Beseitigen überflüssiger Wege:	$N^2 \cdot k$ Schritte
<b>Gesamt:</b>	$N^2 \cdot \log(N) + 2 \cdot N^2 + N \cdot (N^2 + k^2 \cdot (N \cdot \log(N) + N)) + N^2 + N^2 \cdot k$ $\in O(N^3)$

**Speicherplatzbedarf:**

Die Parallelisierung ändert nichts am Speicherplatzbedarf des Algorithmus, da die Protokolle  $P_{i,j}$  von allen Prozessen gemeinsam genutzt werden. Der Aufwand liegt also in  $O(N^3 \cdot k)$  (vgl. Abschnitt 4.2.7).

#### **4.5.2 Parallelisierung der Tiefensuche und des Algorithmus nach Dijkstra**

- Der Algorithmus nach Dijkstra läßt sich nicht wirksam parallelisieren: Die Bedingung, daß in jedem Schritt der kürzeste Weg aus dem Rand entfernt werden muß, erzwingt eine sequentielle Ausführung, weil jeder Berechnungsschritt vom Ergebnis des vorherigen abhängig sein kann.
- Bei der Tiefensuche kann die Verfolgung der Kanten von einem bestimmten Knoten aus parallel erfolgen. Da die Anzahl der Kanten, die von einem Knoten ausgehen, bei planaren Graphen nur von konstanter Größenordnung ist, führt die Parallelisierung nicht zu einer Verringerung der Komplexität des Verfahrens.
- Parallelisierung ist nur im Sinne der Problemstellung sinnvoll: Gesucht sind Wege für alle Knotenpaare  $(i,j) \in V^2$ ,  $i \neq j$ . Weil der Dijkstra-Algorithmus und die Tiefensuche nur Wege von einem festen Startknoten aus untersuchen (1:N-Wegsuche), müssen die Verfahren für jeden Knoten wiederholt werden. An dieser Stelle setzt die Parallelisierung an: Für alle Knoten werden Prozesse gestartet, die gleichzeitig und unabhängig voneinander eine 1:N-Wegsuche durchführen. Nach dem Terminieren der Prozesse stehen für alle Knotenpaare die gesuchten Wege fest.

#### **Laufzeitverhalten:**

Die parallele N:N-Wegsuche entspricht bezüglich der Laufzeit einer sequentiellen 1:N-Wegsuche, welche von der Größenordnung  $O(N^2 \cdot k)$  ist.

#### **Speicherplatzbedarf:**

Während beim sequentiellen Verfahren nur Wege von einem festen Startknoten aus betrachtet werden, müssen bei der Parallelisierung die gesuchten Wege für alle Knotenpaare  $(i,j) \in V^2$ ,  $i \neq j$  gleichzeitig im Speicher gehalten werden, was einen um Faktor N größeren Speicheraufwand verursacht. Wie beim Lösungsansatz nach Minieka liegt er in  $O(N^3 \cdot k)$ .

### **4.5.3 Parallele Linienverfolgung**

- Innerhalb der äußeren Schleife (Laufindex  $u$ ) kann man im Prinzip für jede Linie  $l \in L$  einen Prozeß  $S_l$  starten, welcher entlang des Verlaufs von  $l$  Suchpfade aufnimmt und befördert. Die äußerste Schleife muß die Prozesse synchronisieren, weil sich der Aufbau aller Suchpfade an der Umsteigehäufigkeit  $u$  orientiert: Die Prozesse greifen auf verschiedene Ergebnisse des vorherigen Berechnungsschrittes  $Q_h^{(u)}$  zu. Beim schreibenden Zugriff von  $S_l$  auf ein Haltestellenprotokoll  $Q_h^{(u+1)}$  sind Kollisionen mit anderen Prozessen möglich, weil einige Haltestellen Verknüpfungspunkte von mehreren Linien darstellen. Der Zugriff muß also synchronisiert werden. Im schlimmsten Fall (Haltestelle wird von allen Linien bedient) kann die Synchronisation einen sequentiellen Ablauf der Linienverfolgung erzwingen. Das Ergebnis ist auf alle Fälle korrekt, da kein Prozeß gleichzeitig aus  $Q_h^{(u+1)}$  liest. Weil aber das Konzept der parallelen Linienverfolgung keine Verbesserung der Komplexität des Algorithmus ergibt, wird es nicht weiter untersucht.
- Parallelisierung ist nur im Sinne der Problemstellung sinnvoll: Wie bei allen 1:N-Suchverfahren muß die Linienverfolgung für jeden Bezirk  $b \in B$  als Startpunkt wiederholt werden. Daher kann man für jeden Bezirk einen Prozeß  $S_b$  starten, welcher alle gesuchten Wege von  $b$  aus ermittelt. Die Prozesse können unabhängig voneinander arbeiten, weil die Ergebnisse in getrennten Speicherbereichen erzeugt werden können. Nach dem Terminieren der Prozesse stehen für alle Bezirkspaare die gesuchten Wege fest.

#### **Laufzeitverhalten:**

Die parallele N:N-Wegsuche entspricht bezüglich der Laufzeit der sequentiellen Linienverfolgung für einen Startbezirk, also  $O(N^2 \cdot k)$ .

#### **Speicherplatzbedarf:**

Während bei der sequentiellen Linienverfolgung nur Wege von einem festen Startbezirk aus betrachtet werden, müssen bei der Parallelisierung die gesuchten Wege für alle Bezirkspaare gleichzeitig im Speicher gehalten werden, was einen um Faktor  $N$  größeren Speicheraufwand verursacht, nämlich  $O(N^3 \cdot k)$ .

## 5. Verkehrsumlegung

### Allgemeines:

Unter einer Verkehrsumlegung in einem ÖV-Netz versteht man die rechnerische Verteilung der Verkehrsnachfrage auf alle geeigneten Routen, iteriert über alle Verkehrsbeziehungen. Ziel ist die Berechnung der zukünftigen Verkehrsbelastung auf allen Linien und Strecken des ÖV-Netzes. Durch einen Vergleich der Nachfrage mit dem Verkehrsangebot können rechtzeitig Mängel aufgedeckt und Maßnahmen zu deren Behebung eingeleitet werden.

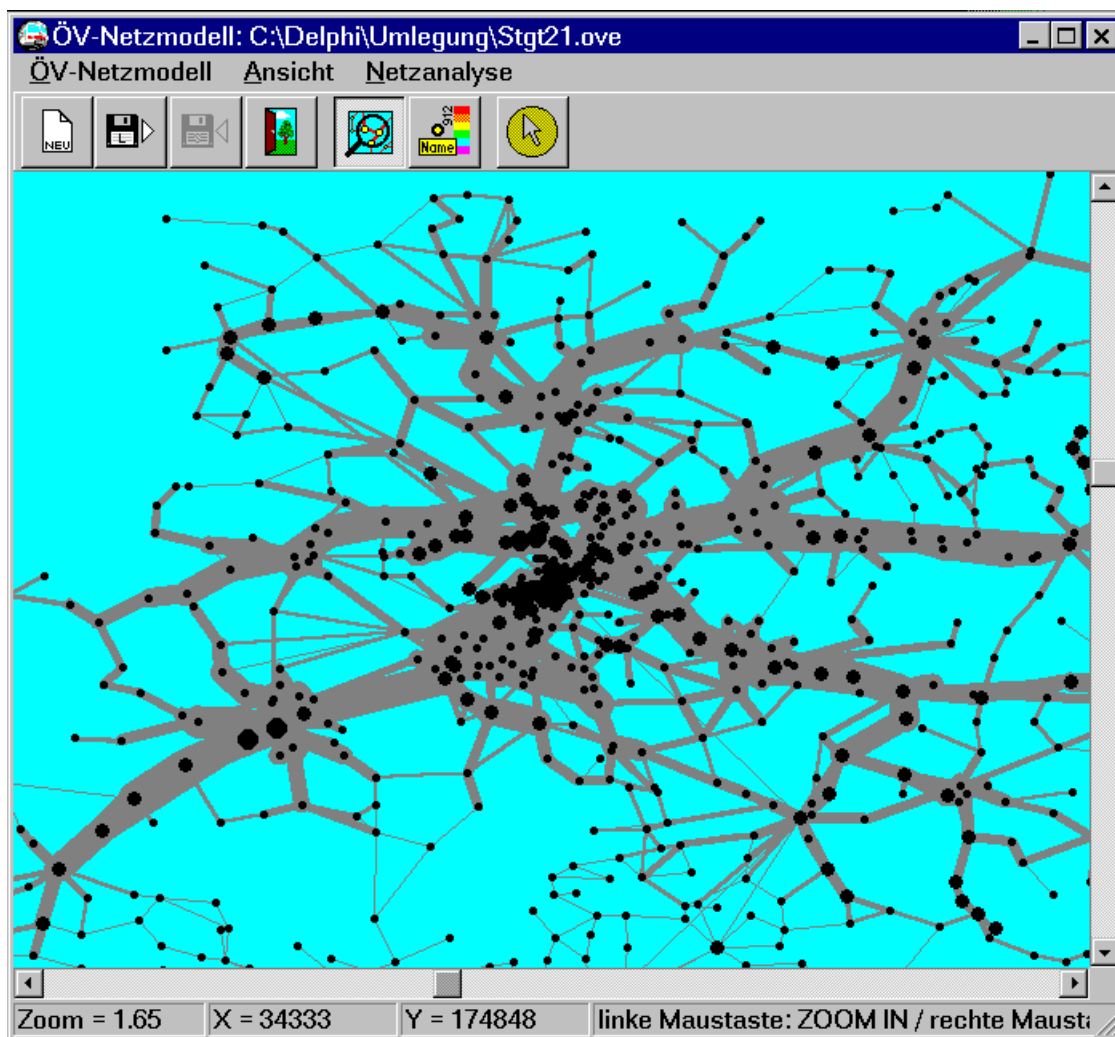


Abbildung 5.1: Streckenbelastung im ÖV-Netz der Region Stuttgart: Auffällig ist die radiale Ausrichtung und Konzentration des Verkehrs auf die Stadtmitte von Stuttgart, was einerseits zu Engpässen im Zentrum, andererseits zur Vernachlässigung von Tangentialverbindungen führt. Das Programm, welches das Bild erzeugte, dient dem Test der in Kapitel 4 erläuterten Wegsuche-Verfahren und läuft unter Windows 95 bzw. Windows NT.

**Gegeben:** Die Verkehrsnachfrage  $F_{i,j}$  zwischen zwei Verkehrszellen  $i, j \in B$  ( $i \neq j$ ) lässt sich mit verkehrswissenschaftlichen Methoden berechnen (siehe [2]) und sei hier als gegeben vorausgesetzt. Weiterhin sei die Menge  $L_{i,j}$  aller gemäß Def. 3.4 in Betracht kommenden ÖV-Wege zwischen  $i$  und  $j$  bekannt.

**Gesucht:** Funktion  $f_{i,j}: L_{i,j} \rightarrow \mathbb{R}_0^+$ , die für jede Route  $w \in L_{i,j}$  die rechnerische Verkehrsnachfrage ermittelt, welche auf  $w$  entfällt.

**Vorgehensweise:**

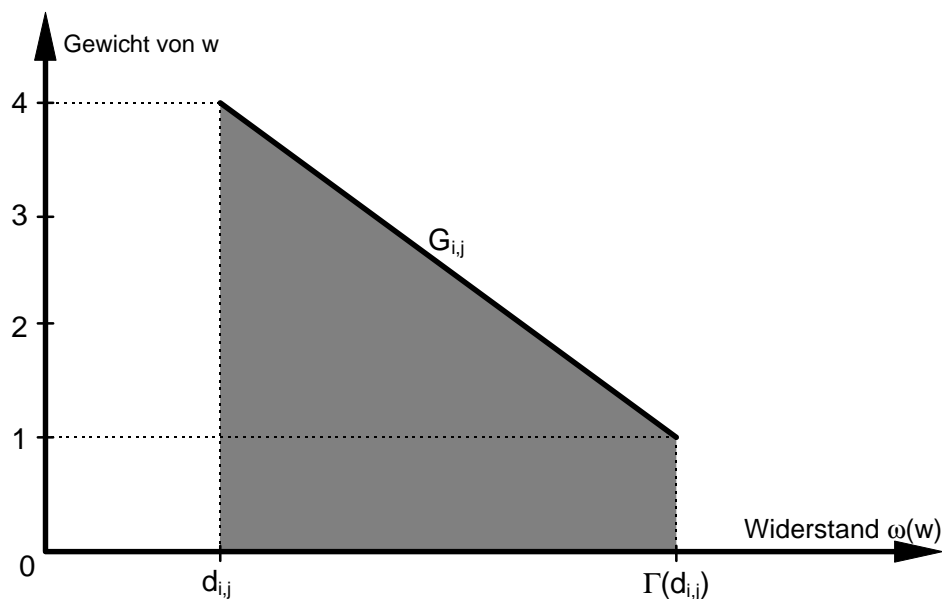
1.) Zunächst wird jedem ÖV-Weg  $w \in L_{i,j}$  ein positives Gewicht in Abhängigkeit von dessen Widerstand zugeordnet. Der gesuchte Anteil  $p_{i,j}(w)$  ergibt sich dann aus dem Verhältnis des Weggewichts zur Summe aller Weggewichte. Was die Gewichtungsfunktion  $G$  betrifft, gibt es unterschiedliche Modelle, wie z.B. das Kirchhoffsche Gesetz oder Exponentialfunktionen. Am IEUV hat sich folgender lineare Ansatz durchgesetzt:

**Def. 5.1:** Gewichtungsfunktion für Wege:

$$G_{i,j}: L_{i,j} \rightarrow \mathbb{R}^+ \text{ für } i, j \in B, i \neq j$$

$$\forall w \in L_{i,j}: G_{i,j}(w) := 3 \cdot \frac{\Gamma(d_{i,j}) - \omega(w)}{\Gamma(d_{i,j}) - d_{i,j}} + 1$$

wobei  $d_{i,j}$  der Widerstand des Bestwegs und  $\Gamma(d_{i,j})$  die obere Schranke für Weglängen bezogen auf die Verkehrsbeziehung  $(i, j) \in B^2$ ,  $i \neq j$  ist.



**Abbildung 5.2:** Abhängigkeit des Gewichts für Wege  $w \in L_{i,j}$  von deren Widerstand

Dies bedeutet anschaulich, daß die Nachfrage auf dem Bestweg viermal so groß ist wie auf der Route, deren Widerstand auf Sicht der Fahrgäste gerade noch akzeptabel ist. Der Fall  $d_{i,j} = \Gamma(d_{i,j}) = 0$  tritt in der Praxis nicht auf, weil jeder ÖV-Weg  $w \in L_{i,j}$  Fußwege mit positiven Widerständen enthält.

2.) Die Verkehrsnachfrage  $f_{i,j}$  für Routen berechnet sich wie folgt:

$$\forall w \in L_{i,j}: f_{i,j}(w) = F_{i,j} \cdot \frac{G_{i,j}(w)}{\sum_{w' \in L_{i,j}} G_{i,j}(w')}, \quad i, j \in B, i \neq j.$$

## 6. Implementierung und Tests

### 6.1 Voraussetzungen

Um die Wegsuche-Algorithmen auf den Rechnern des IEUV testen zu können, müssen sie auf einer **Microsoft-Windows-Plattform**, einem Betriebssystem mit einer graphischen Benutzeroberfläche, implementiert werden. Als Programmiersprache für Windows-Systeme bietet sich **Borland-Delphi** an ([18]). Delphi ist eine visuelle, objekt-orientierte Programmierumgebung mit einem Compiler, der windows-basierte Programme mit optimiertem 32-bit-Maschinencode erzeugt. Delphi bietet die Möglichkeit, einzelne Programm-Routinen direkt in Assembler zu schreiben, um die Datenverarbeitung in besonders zeitkritischen Bereichen zu beschleunigen.

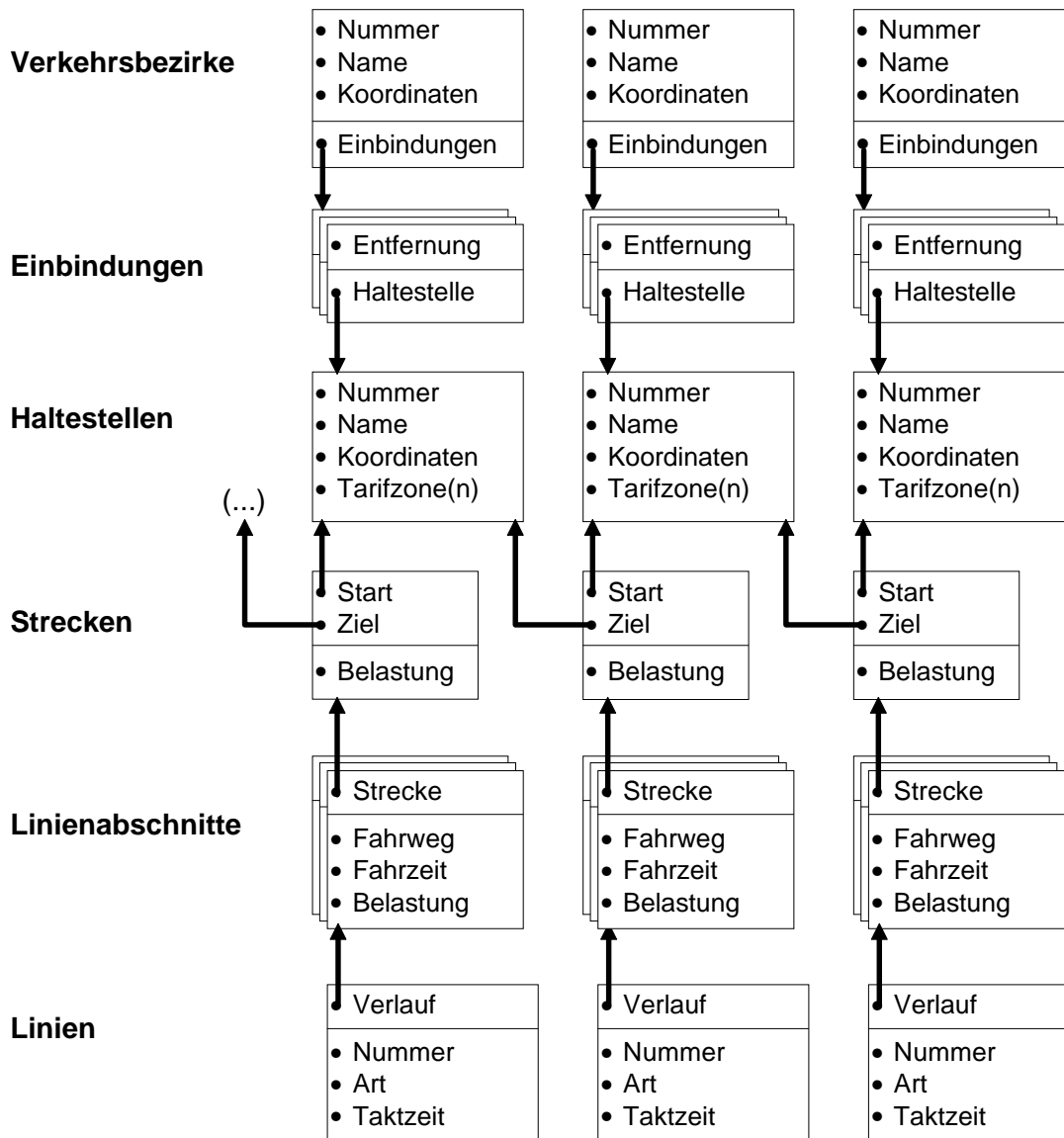
Am IEUV stehen umfangreiche **ÖV-Netzdaten** zur Verfügung, mit denen bereits Verkehrsuntersuchungen durchgeführt wurden. Darunter befindet sich ein ÖV-Netzmodell der Region Stuttgart, welches 785 Verkehrsbezirke, 784 Haltestellen und 1.064 Linien umfaßt. Es ist notwendig, die Größenordnungen solcher Modelle bei der Implementierung der Wegsuche-Verfahren zu berücksichtigen. Weitere **Vorgaben** des IEUV sind:

- Gewichtungsfaktoren zur Bewertung von Wegen,
- Ein- und Ausgabeformat von ÖV-Netzdaten, Fahrtenmatrizen und Ergebnissen der Verkehrsumlegung (ASCII-Dateien) sowie
- exemplarische Fahrtenmatrizen zur Durchführung einer Verkehrsumlegung.

## 6.2 Datenstrukturen im Rechner

### 6.2.1 Aufbau und Organisation von ÖV-Netzmodellen

Die Kommunikation zwischen Programmen und Dateien ist ineffizient, weil die Daten auf einem vergleichsweise langsamen Speichermedium (z.B. Festplatte) abgelegt sind. Da Wegsuche-Verfahren große Ansprüche an die Geschwindigkeit des Datenzugriffs stellen, muß das Programm über eine eigene, im Hauptspeicher angelegte Datenverwaltung verfügen. Die Datenstruktur spiegelt die Zusammenhänge zwischen den Netzelementen wider.



**Abbildung 6.1:** Struktur der ÖV-Netzdaten im Programm: Die Netzelemente (Haltestellen, usw.) sind Datensätze (Objekte), welche sich in 6 Klassen einteilen lassen. Bezirke besitzen Einbindungen (Listen), welche auf Haltestellen verweisen. Letztere sind über Strecken miteinander verbunden, die Strecken wiederum über Linienabschnitte (Listen). Die Pfeile stellen dabei einen Verweis auf ein Objekt einer anderen Klasse dar.

### Erzeugung der Datenstruktur:

Das Einlesen von **Netzdaten** aus einer Datei ist auch bei den größten Netzen eine Angelegenheit weniger Sekunden. Während dem Lesen werden die Daten auf der **Halde** (Hauptspeicher) angelegt und entsprechend der Abhängigkeiten zwischen den Objekten miteinander vernetzt. Verweise auf andere Objekte haben den Vorteil, daß man direkt auf alle Daten der verbundenen Objekte zugreifen kann. So sind z.B. für jede Einbindung alle Eigenschaften der eingebundenen Haltestelle über die zugehörigen Verweise direkt erreichbar. Änderungen dieser Eigenschaften haben keinerlei Auswirkungen auf die Verweise. Wird die Haltestelle komplett ausgetauscht, so muß an der Einbindung lediglich der Verweis geändert werden. Die Vernetzung hat jedoch den Nachteil, daß der Auf- und Abbau der Datenstruktur relativ aufwendig ist: Beim Entfernen eines Objekts müssen alle Verweise auf dasselbe beseitigt werden, was in vielen Fällen eine systematische Durchsuchung der Datenstruktur erfordert. Bei der Wegsuche spielt dieser Nachteil aber keinerlei Rolle, weil das Netz nicht modifiziert wird.

### Probleme beim Einlesen von Netzdaten:

Die Konvertierung von Dateistruktur in Programmdatenstruktur birgt Probleme in sich: Beim Einlesen von Netzdaten können **ungültige Werte** auftreten. Beispiel: In der Netzdatei sei ein Linienabschnitt von Haltestelle Nr. 101 zu Haltestelle Nr. 102 definiert. Letztere sei in der Datei nicht spezifiziert. Das Programm müßte in diesem Fall den Einlesevorgang abbrechen und dem Benutzer den Fehler melden. Im Interesse der Benutzerfreundlichkeit wird der Einlesevorgang jedoch fortgesetzt. Die **Fehler** werden nicht auf dem Bildschirm angezeigt, sondern in einer speziellen Datei **protokolliert**, welche zu einem späteren Zeitpunkt eingehend analysiert werden kann. An die Stelle eines fehlenden Objekts wird vom Programm eine Attrappe mit Standard-Eigenschaften gesetzt.

### Verwaltung von Einbindungen:

Einbindungen sind als Verknüpfung **von** einem **Verkehrsbezirk** **zu** einem **Knotenpunkt** gespeichert. Die Verknüpfung in umgekehrter Richtung wird hier nicht betrachtet, da sie redundant ist. Außerdem würde eine doppelte Verbindung zwischen Bezirk und Knotenpunkt unnötigen Speicherplatzbedarf und Verwaltungsaufwand mit sich bringen.

### Verwaltung von Linien:

Abweichend von Def. 2.1 ist eine Linie nicht als Folge von Haltestellen gespeichert, sondern als **Folge von Linienabschnitten**. Auf diese Weise lassen sich die Fahrzeiten, Fahrweglängen und Belastungsdaten (Umlegungsergebnisse) besser zuordnen. Die Linienabschnitte basieren auf Strecken, also Verbindungen zwischen jeweils zwei Haltestellen. Es werden nur Strecken verwaltet, über welche mindestens ein Linienabschnitt führt.

### Verwaltung von Strecken:

Bei Strecken werden **Richtung und Gegenrichtung gemeinsam** verwaltet, weil nahezu alle Strecken in beiden Richtungen befahren werden. Für richtungsbezogene Eigenschaften wie der Streckenbelastung sind jeweils zwei Einträge (Speicherplätze) vorgesehen. Durch die Zusammenlegung der Daten wird der Verwaltungsaufwand reduziert und der Speicherplatzbedarf für die Strecken nahezu halbiert.

## 6.2.2 Datenstruktur von Distanzgraphen

Der Aufbau von Distanzgraphen ist wesentlich einfacher als der Aufbau von ÖV-Netzmodellen, weil es nur zwei Objektklassen gibt, nämlich Knoten und Kanten. Andererseits sind Graphen umfangreicher, da Haltestellen und Bezirke in mehrere Knoten aufgespalten werden müssen (vgl. Abschnitt 2.4).

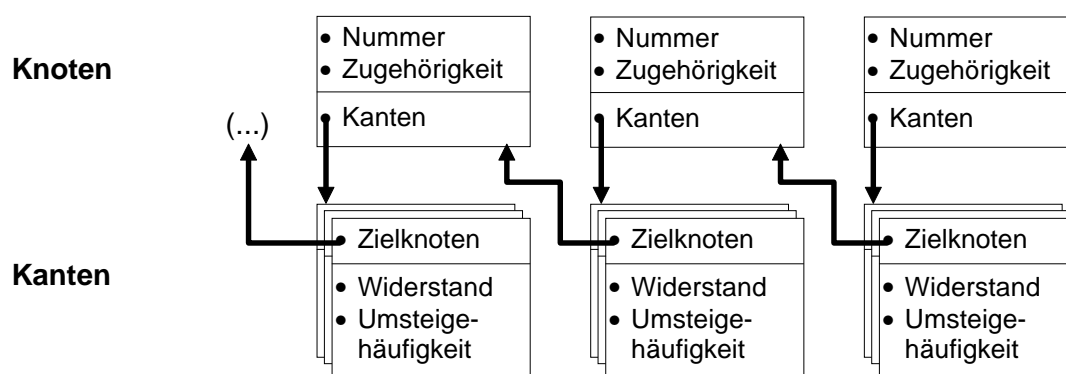


Abbildung 6.2: Struktur von Distanzgraphen im Programm: Knoten sind Datensätze, in denen alle ausgehenden Kanten als Listen gespeichert sind. Die gerichteten Kanten verweisen jeweils auf den Zielknoten. Um den Bezug zum äquivalenten ÖV-Netz zu verdeutlichen, wird die Zugehörigkeit der Knoten zu einer Haltestelle bzw. einem Bezirk festgehalten.

Fast die Hälfte aller Knoten besitzt nur eine einzige ausgehende Kante (Beginn eines Linienabschnitts). Andererseits kann es Knoten mit mehr als 20 ausgehenden Kanten geben (Beginn von Umsteigekanten und Fußwegen). Die Verwaltung der Kanten in **Listen auf der Halde** ermöglicht eine optimale Anpassung an die stark variierende Anzahl der Kanten je Knoten.

### 6.2.3 Konvertierung von ÖV-Netzmodellen in Distanzgraphen

Die Konvertierung erfolgt in vier Schritten:

1. Durchlaufen aller Linien zur Erzeugung von Kanten, welche **Linienabschnitten und Haltestellenaufenthalten** entsprechen. Gleichzeitig werden die den Kanten zugrundeliegenden Knoten gebildet
2. Bündeln von **Parallelverkehr** auf jeder Strecke (mehrere Linienabschnitte mit gleichem Widerstand), d.h. Erzeugung zusätzlicher Knoten und Kanten für alle Linienbündel.
3. Aufbau von **Umsteigebeziehungen** (Kanten) an jeder Haltestelle unter Berücksichtigung der Besonderheiten von Linienbündeln (vgl. Abschnitt 2.6).
4. Erzeugen von zwei Knoten je Bezirk (Ankunft, Abfahrt) sowie Aufbau der **Fußwegkanten** von allen Bezirken zu den eingebundenen Haltestellen und umgekehrt.

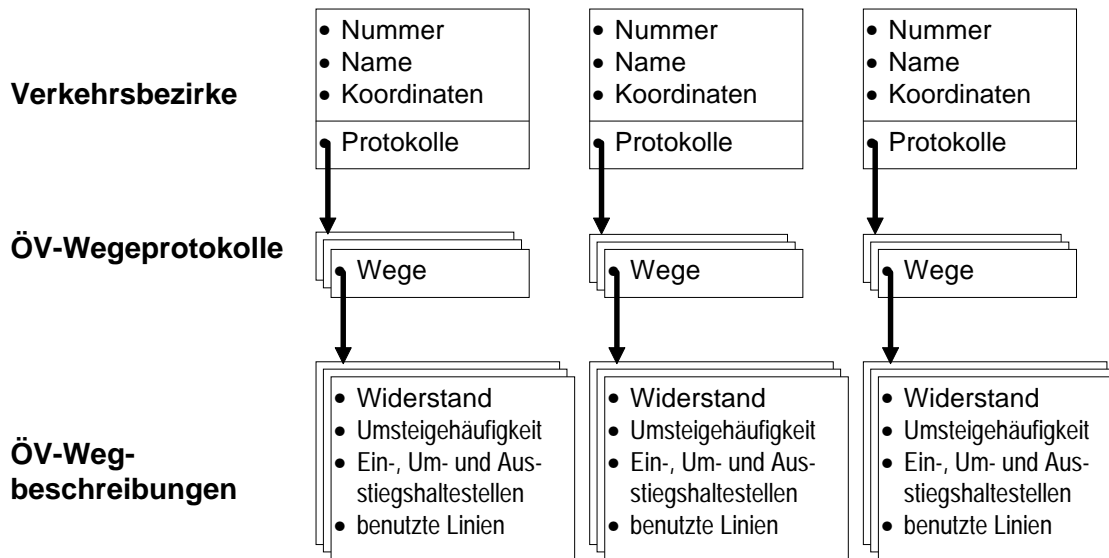
Um den Zugriff auf die **Knoten** zu beschleunigen, werden dieselben auf Listen verteilt, welche den Haltestellen bzw. Bezirken des ÖV-Netzes gehören. Für Ankunfts- und Abfahrtsknoten werden dabei in **getrennten Listen** verwaltet. Beispielsweise befindet sich ein Knoten  $h_l^{an} \in V$  ( $l \in L$ ) in der Liste für Ankunfts-knoten, welche im Datensatz von Haltestelle  $h \in H$  verankert ist. Nachdem alle erforderlichen Knoten erzeugt sind, erhalten dieselben eine **laufende Nummer**. Die Reihenfolge ist durch folgendes Verfahren festgelegt:

1. Aufzählen aller **Bezirke** (aufsteigend nach Nummern sortiert): Der Abfahrtsknoten folgt jeweils auf den Ankunfts-knoten.
2. Aufzählen aller **Haltestellen** (aufsteigend nach Nummern sortiert): Zuerst werden alle Ankunfts-knoten aufgezählt, danach alle Abfahrts-knoten der jeweiligen Haltestelle.

Mit Hilfe eines Feldes („Array“) von Zeigern wird ein **wahlfreier Zugriff** auf die Knoten hergestellt.

### 6.2.4 Datenstruktur von Wegeprotokollen

Die Protokolle für Wegbeschreibungen umfassen die mit Abstand größte und zugleich wichtigste Datenmenge des Programms. Sie enthalten die Beschreibungen aller gesuchten Wege. Für jedes Bezirkspaar ist ein Protokoll vorgesehen. Da deren Anzahl sehr groß sein kann (z.B. 615.440 für das Stuttgarter ÖV-Netz), steht beim Entwurf der Datensätze die Einsparung von Speicherplatz im Vordergrund.



**Abbildung 6.3:** Struktur von ÖV-Wegeprotokollen: Protokolle werden in Listen verwaltet, welche dem jeweiligen Startbezirk gehören. Der Zielbezirk ergibt sich implizit durch die Reihenfolge der Protokolle, welche mit der Reihenfolge der Bezirke (nach Nummern geordnet) übereinstimmt.

Die Verwaltung der Protokolle und Wege in **Listen auf der Halde** ermöglicht eine optimale Anpassung an deren Anzahl, welche stark variieren kann. Auf diese Weise ist der Speicherplatzbedarf relativ gering.

Für den Algorithmus nach Minieka, welcher auf Distanzgraphen abläuft, wird für jedes Knotenpaar ein Protokoll benötigt. Die Grundüberlegungen sind dieselben wie bei Protokollen für Bezirkspaare.

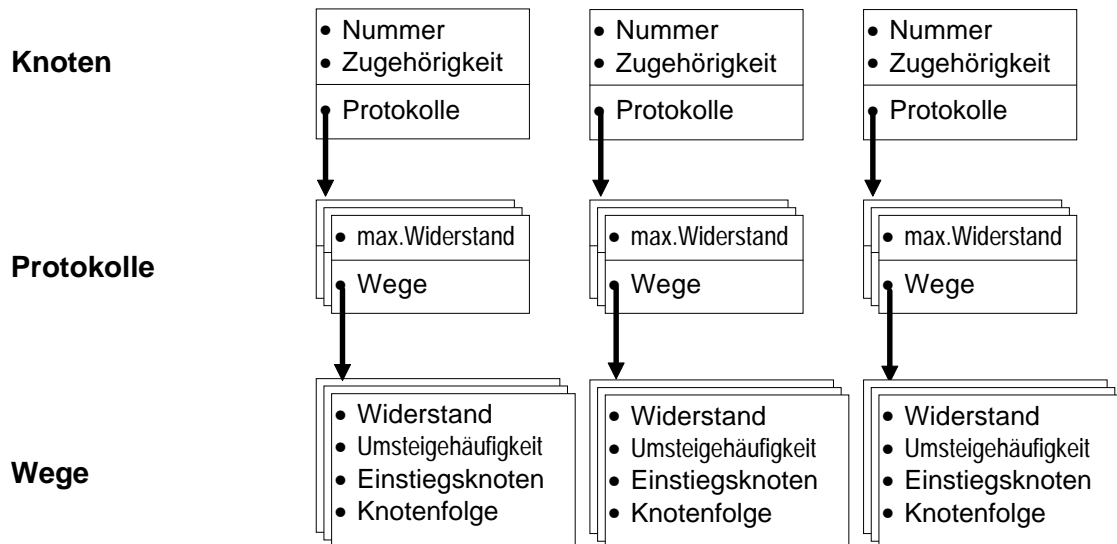


Abbildung 6.4: Struktur von Wegeprotokollen: Protokolle werden in Listen verwaltet, welche dem jeweiligen Startknoten gehören. Der Zielknoten ergibt sich implizit durch die Reihenfolge der Protokolle, welche mit der Reihenfolge der Knoten (laufende Nummer) übereinstimmt. Die Verwendung der Datensätze zur Wegbeschreibung ist im nächsten Abschnitt erläutert.

## **6.3 Implementierung der Wegsuche-Algorithmen**

### **6.3.1 Implementierung des Verfahrens nach Minieka**

Eine der Hauptursachen für den hohen Zeitaufwand ist die **Schleifenprüfung**, also der Test zweier Wege auf gemeinsame Knoten, was einen Aufwand von  $O(N \cdot \log(N))$  verursacht. Zur Beschleunigung dieses Vorgangs kann eine Besonderheit der hier verwendeten Distanzgraphen ausgenutzt werden:

- Auf jedem Weg des Distanzgraph wechseln sich Abfahrts- und Ankunftsknoten ab.
- Von einem Abfahrtspunkt geht nur eine Kante aus, so daß der nachfolgende Ankunftspunkt eindeutig bestimmt ist.
- Der auf einen Ankunftspunkt folgende Abfahrtspunkt gehört derselben Haltestelle an. Alle anderen Abfahrtsknoten des Weges gehören aufgrund der Zyklenfreiheit zu einer anderen Haltestelle.
- In der Aufzählung der Knoten (laufende Nummer) folgen alle Abfahrtspunkte einer Haltestelle unmittelbar auf den Ankunftspunkten derselben Haltestelle (vgl. Abschnitt 6.2.3).

Daraus läßt sich folgende **Methodik** ableiten:

Die **besuchten Knoten** werden nicht in der Reihenfolge ihres Auftretens entlang des Weges aufgelistet, sondern nach der laufenden **Nummer sortiert**. Hierfür steht in jeder Wegbeschreibung eine Knotenfolge zur Verfügung (vgl. Abschnitt 6.2.4). Start- und Endpunkt des Weges werden dabei aus Effizienzgründen nicht protokolliert. Sie ergeben sich implizit aus der Zugehörigkeit der Wegbeschreibung zu einem Protokoll, dessen Start- und Zielknoten bekannt ist. Der zweite Knoten des Weges muß allerdings in der Knotenfolge gekennzeichnet sein (Einstiegsknoten).

Aus der sortierten Knotenfolge läßt sich nun ableiten, in welcher Reihenfolge die Knoten besucht wurden. Der Einstiegsknoten ist bei Wegen, welche von einem Bezirk ausgehen, immer ein Abfahrtspunkt. Die **Nachfolger** von allen **Abfahrtsknoten** ist durch die einzige Kante **eindeutig** bestimmt. Der **Nachfolger** eines **Ankunftsknotens** ist der Knoten im Weg mit der **nächstgrößeren Nummer**, es sei denn, der Ankunftspunkt ist der letzte (Zwischen-)Knoten des Wegs. In diesem Fall gehört der Nachfolger nicht zu derselben Haltestelle.

Die Verkettung zweier Wege entspricht der **Verschmelzung** der zugehörigen **Knotenfolgen** („merge sort“), welche in linearer Zeit abläuft. Die Schleifenprüfung läuft gleichzeitig ab: Falls in zwei Wegen Knoten vorkommen, welche derselben Haltestelle angehören, dann werden diese aufgrund der Ordnung der Knoten in der sich ergebenden Folge unmittelbar nebeneinander liegen. Es genügt also, in der Knotenfolge des verketteten Wegs jeweils zwei benachbarte Knoten zu testen. Der **Gesamtaufwand** des Verfahrens nach Minieka liegt damit in  $O(N^4 \cdot k^2)$  statt  $O(N^4 \cdot \log(N) \cdot k^2)$ .

### 6.3.2 Implementierung des Verfahrens nach Dijkstra

Der Aufwand, einen Weg  $w$  in die Menge  $R$  der Suchpfade einzuordnen, hängt von der Anzahl der Suchpfade ab (schlimmstenfalls  $N$ ). Binäres Einfügen benötigt bis zu  $\log_2(N)$  Schritte. Mit dem „**Basket-Sort-Verfahren**“ (verteilen und sammeln) kann der Sortieraufwand auf einen Schritt pro Weg reduziert werden: Für jeden Wert  $i \in \{0, 1, \dots, n\}$ , den die Weglänge annehmen kann, ist eine Menge  $M_i$  vorgesehen. Der Mengenvektor  $(M_i)_{i \in \{0, 1, \dots, n\}}$  wird **Verteiler** genannt. Sortieren erfolgt durch **direktes Zuordnen**  $M_{\gamma(w)} := M_{\gamma(w)} \cup \{w\}$ . Der aktuell kürzeste Suchpfad ist dann ein Weg aus der ersten nicht-leeren Menge  $M_i$ , d.h.  $\forall 0 \leq j < i: M_j = \{ \}$ . Da der Widerstand in der Regel keine ganze Zahl ist ( $\gamma(w) \in \mathbb{R}^+$ ), müssen sehr große Verteiler, welche auf gerundete Weglängen ansprechen, zur Verfügung gestellt werden. Für die **Laufzeit** des Dijkstra-Algorithmus ergibt sich dank „basket sort“  **$O(N)$**  anstatt  $O(N \cdot \log(N))$ .

### 6.3.3 Implementierung des Verfahrens der Linienverfolgung

- Bei der **Schleifenprüfung** benötigt man  $\log(N)$  Schritte, um eine Haltestelle in einer Folge aufzufinden. Die Prüfung kann auch **in einem Schritt** erfolgen, wenn zusätzlich zu den Haltestellenfolgen binäre  $|H|$ -Tupel verwendet werden, mit denen jede besuchte Haltestelle durch Setzen des zugehörigen Bits registriert werden kann. Die Effizienzvorteile werden aber durch den Speicherplatzbedarf für die  $|H| \cdot k$  Tupel relativiert (insgesamt  $|H|^2 \cdot k$  [bit]).
- Der Aufwand, einen **Weg**  $w$  in ein Protokoll  $Q_h$  **einzuordnen**, kann mit dem „Basket-Sort-Verfahren“ auf **einen Schritt** reduziert werden (vgl. Abschnitt 6.3.2). Allerdings ist hierfür ein immenser Speicheraufwand notwendig, weil jede Haltestelle einen eigenen Verteiler besitzen muß, während beim Dijkstra-Algorithmus ein einziger Verteiler für alle Wege genügt.

## **6.4 Laufzeitverhalten und Speicherplatzbedarf in der Praxis**

Entscheidend für die Laufzeit in der Praxis ist die **Testumgebung**:

- PC mit Intel Pentium Prozessor, Taktfrequenz 133 MHz,
- 256 kB SRAM („pipelined burst cache“)
- 32 MB EDO-RAM (60 ns Zugriffszeit)
- 1 GB freier Plattenspeicher (12 ms Zugriffszeit, Durchsatz ca. 6 MB/sec)
- Betriebssystem: Microsoft Windows 95 (32-bit-Plattform)

Zum Testen der Algorithmen standen folgende **ÖV-Netzmodelle** zur Verfügung:

<b>ÖV-Netzmodell</b>	<b>Beispiel</b>	<b>Ulm</b>	<b>Stuttgart</b>
Anzahl der Haltestellen	15	271	784
Anzahl der Bezirke	6	96	785
Einbindungen je Bezirk, Durchschnitt	2,8	3,1	1,1
Einbindungen je Bezirk, Maximum	4	10	4
Anzahl der Linien (Richtung/Gegenri. getrennt)	20	78	1.064
Linienabschnitte je Linie, Durchschnitt	3,7	11,2	4,5
Linienabschnitte je Linie, Maximum	8	26	25
Anzahl der Strecken (ungerichtet)	22	302	1.136
Anzahl der Linienbündel	7	36	714

„Beispiel“ ist ein fiktives ÖV-Netz zu Testzwecken, „Ulm“ und „Stuttgart“ dagegen sind Netzmodelle, mit denen am IEUV bereits Verkehrsuntersuchungen durchgeführt wurden. Das reale ÖV-Netz der Region Stuttgart umfaßt rund 4.000 Haltestellen. Bei der Modellierung des Netzes wurden etwa 80% davon wegoptimiert, um Speicherplatz zu sparen bzw. um die Routensuche zu beschleunigen. Übrig geblieben sind nur Haltestellen, die als „Einspeisepunkte“ von Verkehrsbezirken oder als Verknüpfungspunkt mehrerer Linien dienen.

Bei der Konvertierung der ÖV-Netze entstehen folgende **Distanzgraphen**:

<b>zugehöriges ÖV-Netzmodell</b>	<b>Beispiel</b>	<b>Ulm</b>	<b>Stuttgart</b>
Anzahl der Knoten	224	2.746	17.034
Anzahl der Kanten	595	6.620	58.750

Die Anzahl der Knoten ist 20- bis 30-mal so groß wie die Anzahl der Bezirke. Die Anzahl der Kanten ist für alle Netze etwa dreimal so groß wie die Knotenzahl, eine Eigenschaft, die an planare Graphen erinnert.

Die **Wegsuche** mit Dijkstra bzw. Tiefensuche liefert folgende Ergebnisse:

ÖV-Netzmodell	Beispiel	Ulm	Stuttgart
Anzahl Verkehrsbeziehungen (ohne Binnenverkehr)	30	9.120	615.440
Anzahl relevanter* ÖV-Wege insgesamt	52	297.688	7.178.335
relevante* ÖV-Wege je Bezirkspaar, Durchschnitt	1,7	32,6	11,7
relevante* ÖV-Wege je Bezirkspaar, Maximum	3	1.142	302
Umsteigehäufigkeit, Durchschnitt	0,1	2,7	2,9
Umsteigehäufigkeit, Maximum	1	5	5
Haltestellen je ÖV-Weg, Durchschnitt	3,8	21,4	17,7
Haltestellen je ÖV-Weg, Maximum	7	43	44

*\* relevant im Sinne von Def.3.4*

Auffällig ist, daß die Anzahl der gesuchten Wege je Bezirkspaar in Ulmer Netz etwa dreimal so groß ist wie im vielfach größeren Stuttgarter Netz. Dieser Umstand ist vor allem auf die Tatsache zurückzuführen, daß die Anzahl der Einbindungen je Bezirk im Ulmer Netz ebenfalls dreimal so groß ist. Damit ergeben sich für die Verkehrsteilnehmer etwa neunmal so viele Möglichkeiten, eine Einstiegshaltestelle am Startbezirk mit einer Ausstiegshaltestelle am Zielbezirk zu kombinieren. In einem modifizierten Ulmer Netz mit 1,2 Einbindungen pro Bezirk (statt 3,1) ergaben sich durchschnittlich 14,5 Routen je Bezirkspaar (statt 35,3) bei einem Maximum von 203 Routen (statt 1.261).

Der **Speicherplatzbedarf für ÖV-Wegeprotokolle** (Lösungsmengen) ist bei großen Netzen problematisch:

ÖV-Netzmodell	Beispiel	Ulm	Stuttgart
Speicherplatzbedarf ÖV-Protokolle, leer	0,3 kB	0,1 MB	4,7 MB
Speicherplatzbedarf ÖV-Protokolle einschließlich Wegbeschreibungen	2,9 kB	15,4 MB	347,0 MB

Während das Verfahren Dijkstra/Tiefensuche mit dem genannten Speicherplatz auskommt, benötigt der Algorithmus nach Minieka ein Vielfaches davon, weil zusätzlich Wegeprotokolle für alle Knotenpaare verwaltet werden müssen, welche nach der Terminierung verworfen werden. Die folgende Werte sind zum Teil Hochrechnungen, basierend auf den oben genannten Werten für die Anzahl der gesuchten Wege.

**Speicherplatzbedarf für Wegeprotokolle** beim Verfahren nach Minieka:

ÖV-Netzmodell	Beispiel	Ulm	Stuttgart
Anzahl der Knotenpaare	50.176	7,54 Mio.	290 Mio.
Anzahl der zu berechnenden Wege insgesamt	88.160	*226 Mio.	*2,9 Mrd.
Wege je Knotenpaar, Durchschnitt	1,6	*30	*10
Speicherplatzbedarf Protokolle, leer	0,57 MB	90,2 MB	4,82 GB
Speicherplatzbedarf Protokolle inkl. Wege	6,63 MB	*20 GB	*250 TB

\* Schätzung

Für das Ulmer bzw. Stuttgarter Netz konnte der Minieka-Algorithmus mangels Speicherplatz ( $250 \text{ TB} \approx 2^{48} \text{ Byte} \approx 2,8 \cdot 10^{14} \text{ Byte}$ ) nicht durchgeführt werden. Damit ist besagter Algorithmus für die Praxis unbrauchbar. Die Tiefensuche dagegen schlägt alle Laufzeitrekorde, welche bisher bei der Wegsuche in ÖV-Netzen am IEUV aufgestellt wurden. Bei Rechnern mit wenig Hauptspeicher müssen allerdings deutliche Abstriche gemacht werden, was die Laufzeit betrifft, unter Umständen ist die Durchführung der Wegsuche überhaupt nicht möglich.

ÖV-Netzmodell	Beispiel	Ulm	Stuttgart
Laufzeit des Verfahrens nach Minieka	14 sec	*	*
Laufzeit Dijkstra/Tiefensuche	0,05 sec	35 sec	14,1 min
Laufzeit Linienverfolgung	0,5 sec	173 sec	36,7 min

\* Verfahren nicht ausführbar

Fazit: Konventionelle Verfahren gehen als klare Sieger hervor.

## 7. Zusammenfassung und Ausblick

Zur Prognose der Verkehrsnachfrage in Netzen des öffentlichen Verkehrs (ÖV) wird mit Hilfe von Verkehrsmodellen die Nachfrage  $F_{i,j}$  zwischen zwei Verkehrszellen  $i$  und  $j$  berechnet und auf alle Routen umgelegt, welche von  $i$  nach  $j$  führen und aus Sicht von Fahrgästen in Betracht kommen. Zur Bewertung und zum Vergleich von Routen ist eine Widerstandsfunktion definiert, in welche alle für die Routenwahl relevanten Einflußgrößen wie Reisezeit, Fahrtkosten und Reisekomfort eingehen. Gesucht sind alle Wege, deren Widerstand eine vom kürzesten Weg abhängige obere Schranke nicht überschreitet. Außerdem ist die Umsteigehäufigkeit begrenzt und das Fahren in Schleifen ausgeschlossen.

Die Problemstellung entspricht nicht dem „ $k$ -kürzeste-Wege-Problem“, sondern dem Problem „Wegsuche mit Zeitschranke“. Die Anzahl der gesuchten Wege kann gegenüber der Netzgröße exponentiell wachsen, was in der Praxis jedoch unwahrscheinlich ist. Im Interesse der Vergleichbarkeit von Wegsuche-Algorithmen wird ein festes  $k \in \mathbb{N}$  als obere Schranke für die Anzahl der Wege je Verkehrsbeziehung angenommen.

Die Wegsuche kann basierend auf ÖV-Netzmodellen oder Distanzgraphen definiert werden. Graphen entstehen aus ÖV-Netzen durch Verfeinerung der Struktur. Zur Berechnung der gesuchten Wege wurden drei Verfahren untersucht, implementiert und am Stuttgarter ÖV-Netz getestet (785 Verkehrszellen, 784 Haltestellen und 1.064 Linien). Der äquivalente Distanzgraph umfaßt 17.034 Knoten und 68.989 Kanten. Insgesamt sind etwa 7,2 Mio. Verkehrswege zu berechnen.

Algorithmus	nach Minieka	nach Dijkstra, Tiefensuche	Linienverfolgung
Suchprinzip	N:N-Suche	1:N-Suche	1:N-Suche
verwendete Datenstruktur	Distanzgraph	Distanzgraph	ÖV-Netzmodell
Laufzeitverhalten <b>N:N</b>	$O(N^4 \cdot \log(N) \cdot k^2)$	$O(N^3 \cdot k)$	$O(N^3 \cdot k)$
Speicherplatzbedarf	$O(N^3 \cdot k)$	$O(N^2 \cdot k)$	$O(N^2 \cdot k)$
Laufzeit, Testergebnis [min]	*	6	37

\* Der Minieka-Algorithmus funktioniert nur mit sehr kleinen Netzen (s.u.).

Das Verfahren nach **Minieka** scheitert in der Praxis an großen Netzen,

- weil der Speicherplatzbedarf alle Grenzen sprengt (Stuttgart: 290 Mio. Knotenpaare und etwa 40 GB für durchschnittlich zwei Wege je Knotenpaar),
- weil mehr als 99% der Laufzeit (und des Speichers) zur Berechnung von Wegen zwischen Haltestellenknoten benötigt werden, also Wege, die nicht gesucht sind. Man beachte, daß weniger als 10% aller Knoten zu Bezirken gehören. Die relevanten Knotenpaare (Start-, Zielbezirk) machen 0,2% aller Knotenpaare aus.
- weil der Aufwand für die Schleifenprüfung (Beseitigung von Wegen mit Schleifen) selbst bei Anwendung aller Tricks mindestens linear wächst,
- weil sich der Ablauf des Verfahrens an Knoten orientiert (dreifache Iteration über alle Knoten), was sich nur bei dichten Graphen lohnt. Weil die Kantenmenge bei ÖV-Netzen relativ klein ist, hat der Minieka-Algorithmus kaum Chancen gegenüber Verfahren, die sich an Kanten orientieren (z.B. Tiefen-/Breitensuche, Dijkstra).

**Tiefensuche** (kombiniert mit **Dijkstra**) wird der Problemstellung am besten gerecht:

- Die Suche beginnt nur bei Startknoten in Bezirken, nicht bei Haltestellenknoten.
- Es müssen keine Zwischenergebnisse gespeichert werden (Wege zu Haltestellen), weil zu jedem Zeitpunkt des Ablaufs nur ein Suchpfad untersucht wird (rekursiver Abstieg im Suchbaum). Somit müssen keine Wege verworfen werden.
- Die Schleifenprüfung erfolgt in einem Schritt, weil alle Knoten entlang des Suchpfads markiert sind.
- Mit Hilfe des Dijkstra-Algorithmus wird der relevante Teil des Suchbaums vor der Tiefensuche exakt abgesteckt. Damit ist Laufzeit-Overhead ausgeschlossen.

Die **Linienverfolgung** kann im Vergleich zur Tiefensuche einigermaßen mithalten, weil

- die Anzahl der Haltestellen gegenüber der Knotenzahl um Faktor 21 kleiner ist,
- die Anzahl der Linienabschnitte gegenüber der Kantenzahl um Faktor 14 kleiner ist,
- die Konvertierung des ÖV-Netzmodells in einen Distanzgraph entfällt (spart vor allem Speicherplatz)
- und damit auch die Rekonvertierung von Knotenfolgen auf ÖV-Wege entfällt.

Dem steht jedoch entgegen, daß

- an jeder Haltestelle Suchpfade zwischengespeichert werden müssen,
- Einstiegs- und Umstiegs widerstände beim Verfolgen jeder Linie neu berechnet werden müssen,
- zu Beginn der Wegsuche keine absoluten Zeitschranken  $D_{i,j}$  verfügbar sind und
- die Berücksichtigung von Parallelverkehr für die Linienverfolgung ein bislang ungelöstes Problem ist.

Es gibt vermutlich nur zwei Möglichkeiten, die Laufzeit der Tiefensuche zu unterbieten:

1. durch Anwendung von Heuristiken,
2. durch Parallelisierung der Wegsuche.

Durch Verwendung von **Heuristiken** kann die Wegsuche zwar beschleunigt werden, aber die Korrektheit der Ergebnisse ist nicht mehr garantiert. Deshalb muß bei Anwendung jeder Heuristik sorgfältig geprüft werden, ob die Laufzeitvorteile die Abweichungen vom korrekten Ergebnis rechtfertigen.

Ein weitaus größeres Einsparpotential bezüglich Laufzeit bieten **parallele Verfahren** gegenüber sequentiellen Verfahren: Auf Parallelrechnern oder verteilten Systemen können korrekte Ergebnisse theoretisch in einem Bruchteil der Zeit erzielt werden, wie folgende Tabelle zeigt:

paralleler Algorithmus	nach Minieka	nach Dijkstra, Tiefensuche	Linienvorfahrung
Laufzeitverhalten <b>N:N</b>	$O(N^3)$	$O(N^2 \cdot k)$	$O(N^2 \cdot k)$
Speicherplatzbedarf	$O(N^3 \cdot k)$	$O(N^3 \cdot k)$	$O(N^3 \cdot k)$

Wünschenswert und vor allem zukunftsweisend wäre die Implementierung der Algorithmen mit verteilter Verarbeitung auf geeigneten Rechnersystemen, um zu testen, welche Einsparungen in der Praxis erzielt werden. Wenn es gelänge, den Rechenaufwand optimal zu verteilen, dann wäre die Berechnung der gesuchten Wege selbst für die größten ÖV-Netze eine Angelegenheit weniger Sekunden.

## 8. Literaturquellen

### zu Kapiteln 1, 2, 3 und 5:

- [1] Bundesminister für Verkehr (Hrsg.):  
„**Standardisierte Bewertung von Verkehrsweegeinvestitionen des ÖPNV**“,  
erstellt im Auftrag des Bundesministers für Verkehr  
von G. Heimerl und Intraplan Consult GmbH, Stuttgart und München, 1988
- [2] H. Dobeschinsky, J. Hoffmann, F. Schedel:  
„**Verkehrsprognose im öffentlichen Verkehr**“,  
Skript zur Vorlesung „Planung und Entwurf von Anlagen des öffentlichen  
Verkehrs“, Fakultät Bauingenieurwesen, Universität Stuttgart, 1995
- [3] H. Dobeschinsky, U. Grote: „**Verkehrs- und Strukturanalyse**“,  
Skript zur Vorlesung „Planung und Entwurf von Anlagen des öffentlichen  
Verkehrs“, Fakultät Bauingenieurwesen, Universität Stuttgart, 1995
- [4] H. Hensel: „**Wörterbuch und Modellsammlung zum Algorithmus der  
Verkehrsprognose**“, Aachen, 1996
- [5] „**Regionalverkehrsplan Stuttgart, Analyse und Prognose**“, Stuttgart, 1976

### zu Kapitel 4:

- [6] H.-J. Appelrath, J. Ludewig: „**Skriptum Informatik**“,  
B. G. Teubner, Stuttgart, 1991
- [7] V. Diekert: Skript zur Vorlesung „**Algorithmen und Datenstrukturen**“  
Fakultät Informatik, Universität Stuttgart, 1993
- [8] E. W. Dijkstra: „**A Note on Two Problems in Connexion with Graphs**“,  
Numerische Mathematik 1, S.269-271
- [9] R. W. Floyd: „**Algorithm 97 - shortest path**“,  
Communications of the ACM 5, Nr. 6, S.345
- [10] J. Hoffmann: „**Entwicklung eines Programmsystems zur Wegsuche  
in Verkehrsnetzen des Öffentlichen und Individualverkehrs  
unter besonderer Berücksichtigung des P+R-Verkehrs.**“,  
Diplomarbeit Nr. 1094, Fakultät Informatik, Universität Stuttgart, 1993
- [11] M. Mack: „**Untersuchung von effizienten Algorithmen zur Bestimmung  
der k-kürzesten Wege innerhalb von ÖPNV-Verkehrsnetzen**“,  
Diplomarbeit Nr. 1374, Fakultät Informatik, Universität Stuttgart, 1996

- [12] E. Minieka: „**On computing sets of shortest paths in a graph**“,  
Communications of the ACM 17, Nr. 6, 1974, S.351-353
- [13] A. Perko: „**Implementation of Algorithms for K Shortest Loopless Paths**“,  
Networks 16, 1986, S.149-160
- [14] F. Schedel: „**Programmsystem zur Durchführung von Umlegungsrechnungen  
in Netzen des öffentlichen Personennahverkehrs und deren Darstellung  
auf postscriptfähigen Laserdruckern**“,  
Diplomarbeit Nr. 760, Fakultät Informatik, Universität Stuttgart, 1990
- [15] D. R. Shier: „**Iterative Methods for Determining the k Shortest Paths  
in a Network**“, Networks 6, 1976, S.205-229
- [16] J. Y. Yen: „**Finding the k shortest loopless paths in a network**“,  
Management Science 17, Nr. 11, S.712-716
- [17] Universität Karlsruhe:  
„<http://iinwww.ira.uka.de/searchbib/Theory/k-path.html>“

**zu Kapitel 6:**

- [18] Borland GmbH: Handbuchsatz „**Borland Delphi 2 für Windows 95  
und Windows NT**“, 1996
- [19] :J. Rensmann: „**Software Training / Delphi 2**“,  
DATA BECKER, Düsseldorf, 1996

## Erklärung

Hiermit versichere ich, daß ich diese Diplomarbeit selbständig und nur unter Verwendung der angegebenen Hilfsmittel verfaßt habe.

Stuttgart, den 13. Oktober 1997

---

Bearbeiter

cand. inform. Claus Lener