Routing in Nahverkehrsnetzen

Daniel Friesel
derf@chaosdorf.de — @derfnull
CCC Düsseldorf / Chaosdorf

12. Juni 2015

Einleitung

- Es gibt diverse Routingdienste

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...
- Aber wie funktionieren sie eigentlich?

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...
- Aber wie funktionieren sie eigentlich?
  - Vor allem: Wie schaffen sie es, auch komplexe Anfragen effizient zu beantworten?

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...
- Aber wie funktionieren sie eigentlich?
  - Vor allem: Wie schaffen sie es, auch komplexe Anfragen effizient zu beantworten?
  - Z.B. $\leq$1 Sekunde für Düsseldorf Luisenstr $\rightarrow$ Prag Hbf auf bahn.de

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...
- Aber wie funktionieren sie eigentlich?
  - Vor allem: Wie schaffen sie es, auch komplexe Anfragen effizient zu beantworten?
  - Z.B. $\leq$1 Sekunde für Düsseldorf Luisenstr $\rightarrow$ Prag Hbf auf bahn.de

Vortrag in zwei Teilen:

Einleitung

- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...
- Aber wie funktionieren sie eigentlich?
  - Vor allem: Wie schaffen sie es, auch komplexe Anfragen effizient zu beantworten?
  - Z.B. $\leq$1 Sekunde für Düsseldorf Luisenstr $\rightarrow$ Prag Hbf auf bahn.de

Vortrag in zwei Teilen:

1. Route überhaupt berechnen

Einleitung

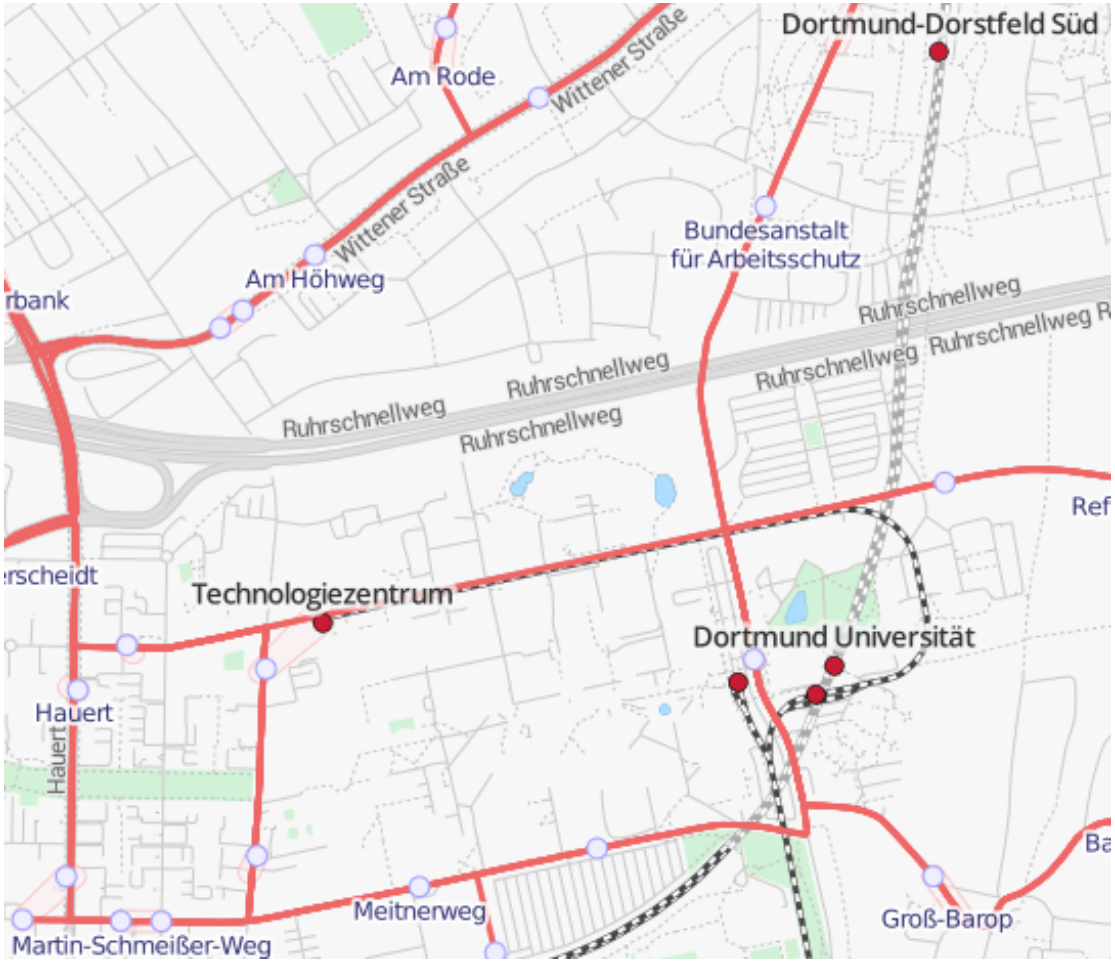
- Es gibt diverse Routingdienste
  - Google Maps / Google Transit
  - Deutsche Bahn
  - EFA / VRR
  - ...
- Aber wie funktionieren sie eigentlich?
  - Vor allem: Wie schaffen sie es, auch komplexe Anfragen effizient zu beantworten?
  - Z.B. $\leq$1 Sekunde für Düsseldorf Luisenstr $\rightarrow$ Prag Hbf auf bahn.de

Vortrag in zwei Teilen:

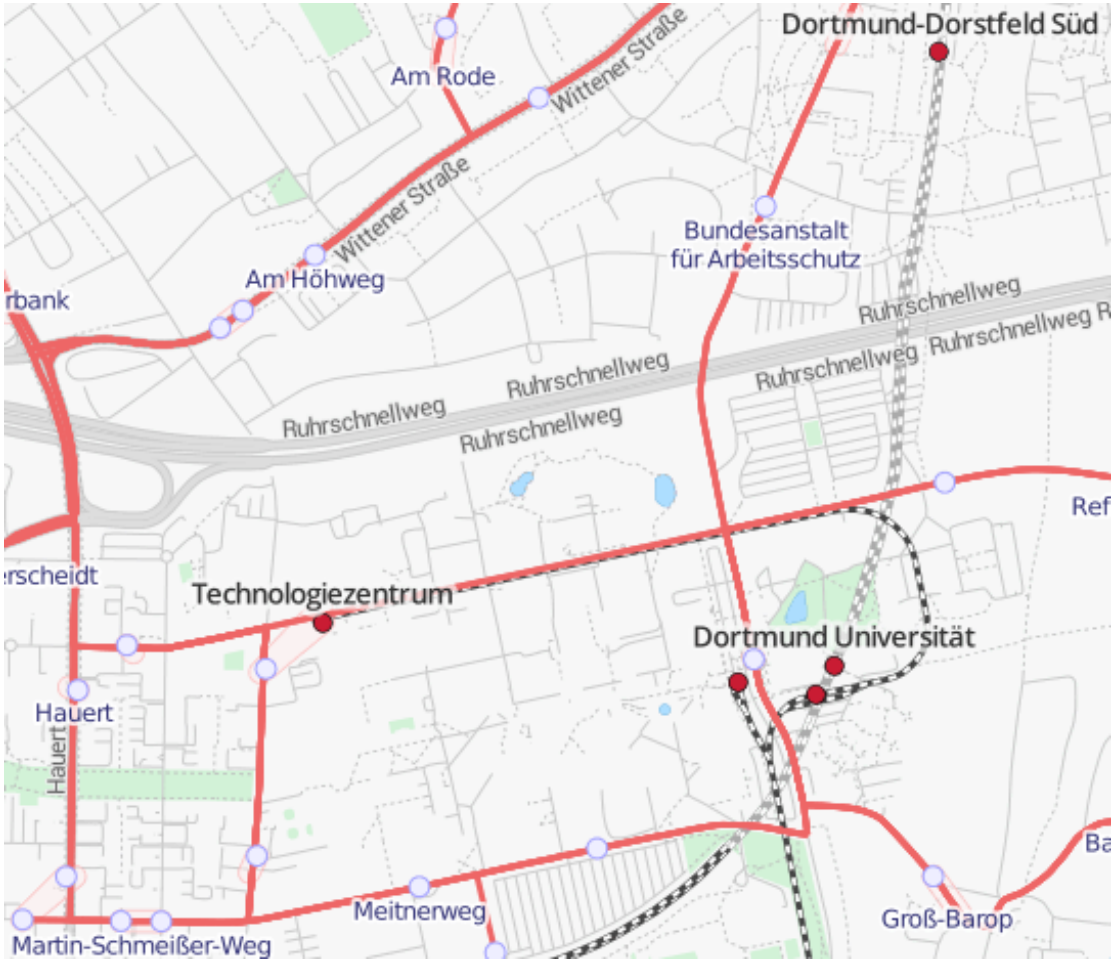
1. Route überhaupt berechnen
2. Route effizient berechnen

Themen

Modellierung

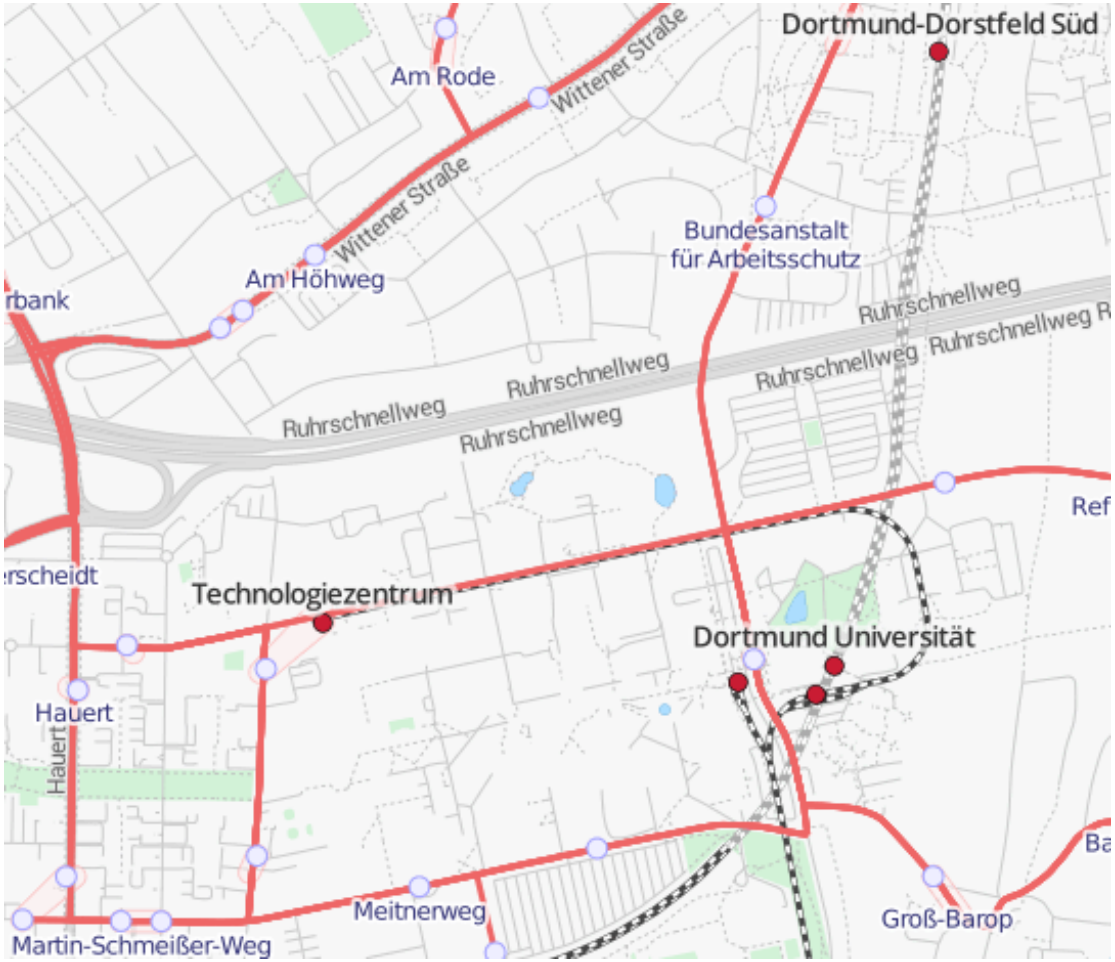


Modellierung



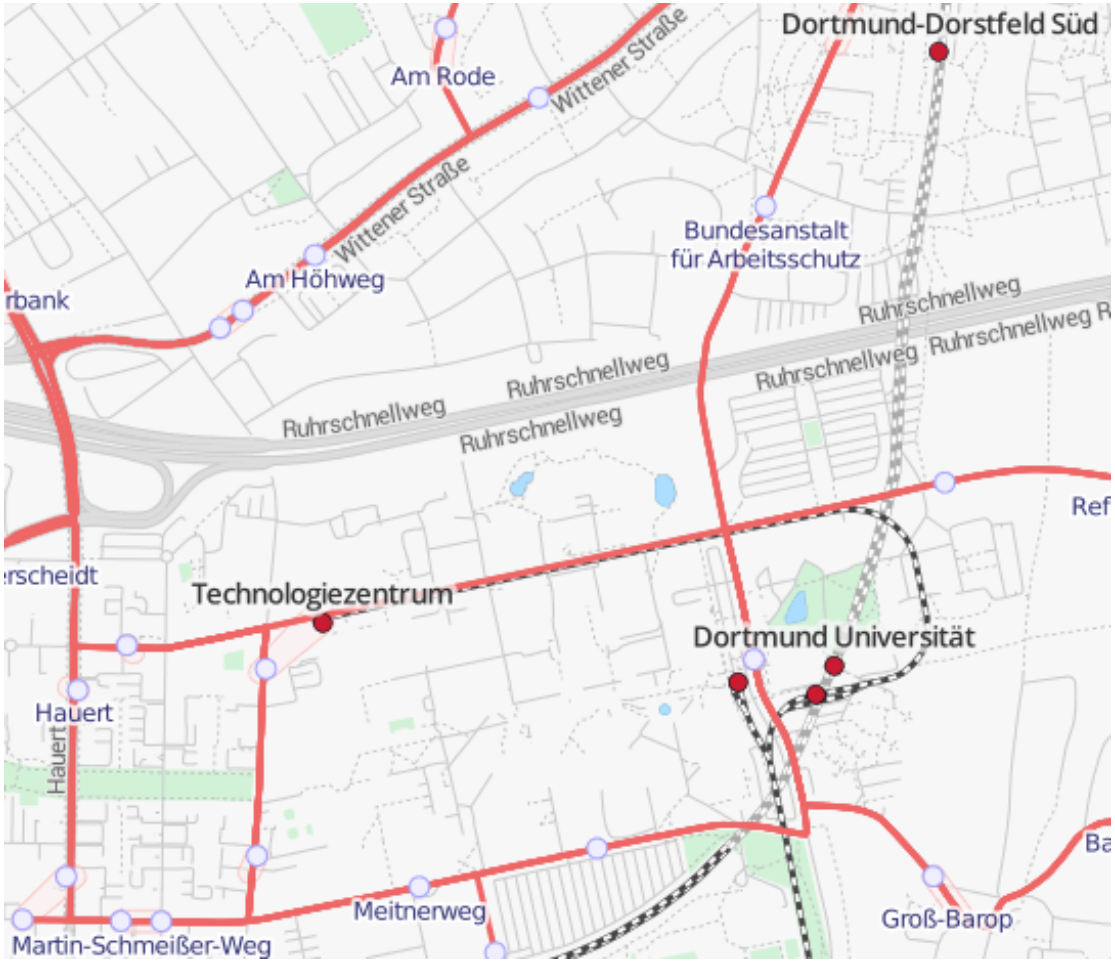
- Als Bild: Wohl kaum

Modellierung



- Als Bild: Wohl kaum
- Intern: Tabelle mit Fahrtzeiten

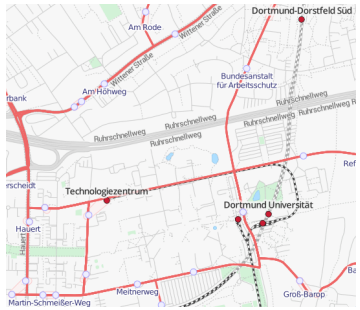
Modellierung



- Als Bild: Wohl kaum
- Intern: Tabelle mit Fahrtzeiten
- Anschaulicher: Graph

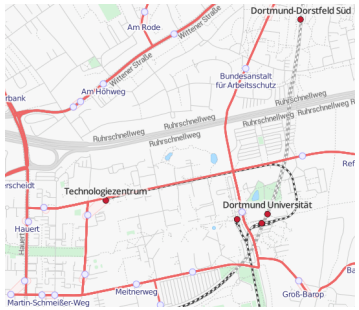
Nahverkehr als Graph

- Graph: Besteht aus Knoten und Kanten



Nahverkehr als Graph

- Graph: Besteht aus Knoten und Kanten
- Knoten $\widehat{=}$ Haltestelle



Technologiezentrum

Universität

Dorstfeld Süd

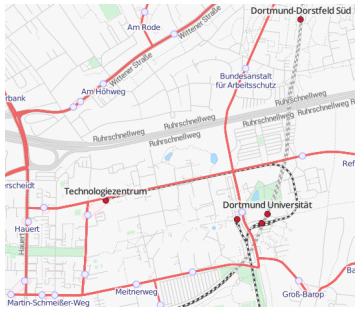
J-v-F Str.

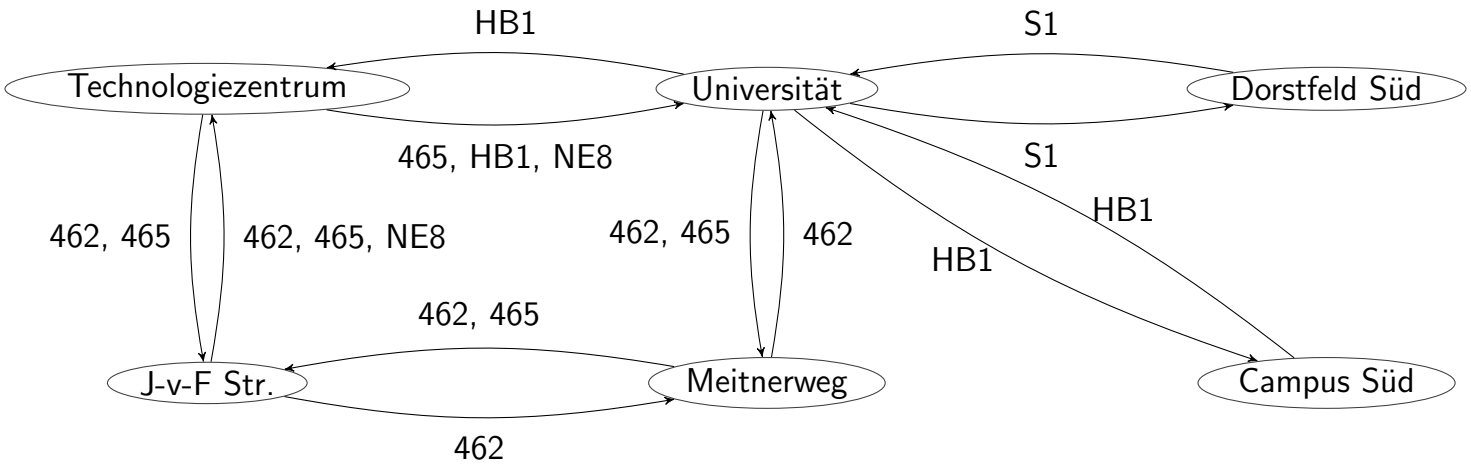
Meitnerweg

Campus Süd

Nahverkehr als Graph

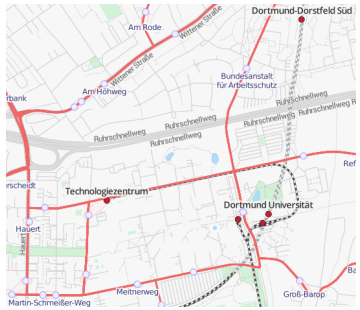
- Graph: Besteht aus Knoten und Kanten
- Knoten $\widehat{=}$ Haltestelle
- Kanten $\widehat{=}$ Verbindung

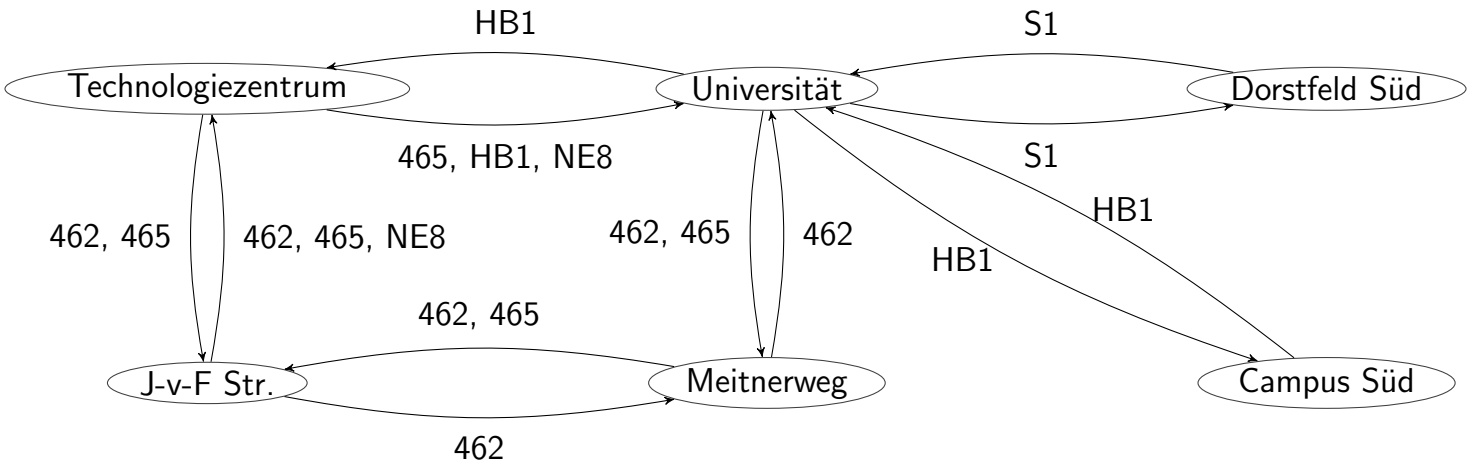




Nahverkehr als Graph

- Graph: Besteht aus Knoten und Kanten
- Knoten $\widehat{=}$ Haltestelle
- Kanten $\widehat{=}$ Verbindung

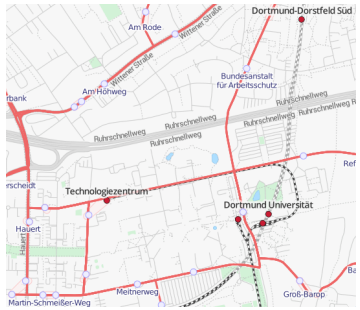


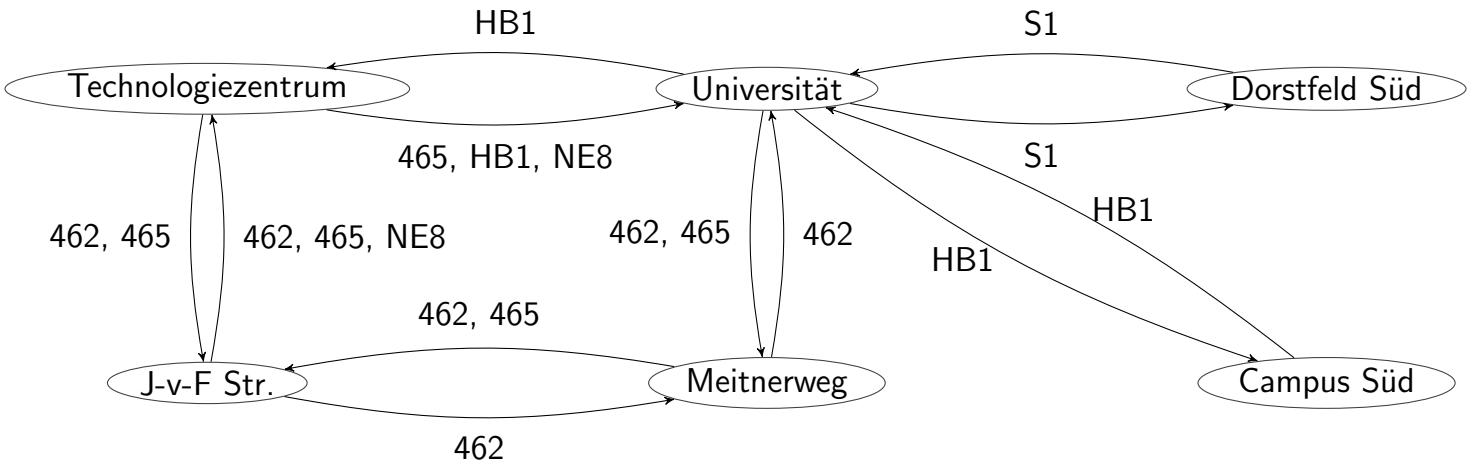


- Kürzesten Weg in Graphen finden: Sehr effizient (z.B. Straßennetz)
  - Stichwort: Dijkstra-Algorithmus

Nahverkehr als Graph

- Graph: Besteht aus Knoten und Kanten
- Knoten $\widehat{=}$ Haltestelle
- Kanten $\widehat{=}$ Verbindung





- Kürzesten Weg in Graphen finden: Sehr effizient (z.B. Straßennetz)
  - Stichwort: Dijkstra-Algorithmus
- Aber: Verkehrsmittel haben einen Fahrplan, Umsteigen dauert...

→ Reisedauer hängt von Tageszeit, Takt etc. ab

Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz

Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein

Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein

D:Uni 11:55,HB1	D:Uni 11:56,HB1	D:Uni 11:59,S1	D:Uni 12:00,S1	D:Uni 12:00,HB1	D:Uni 12:00,462	D:Uni 12:02,462

Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein

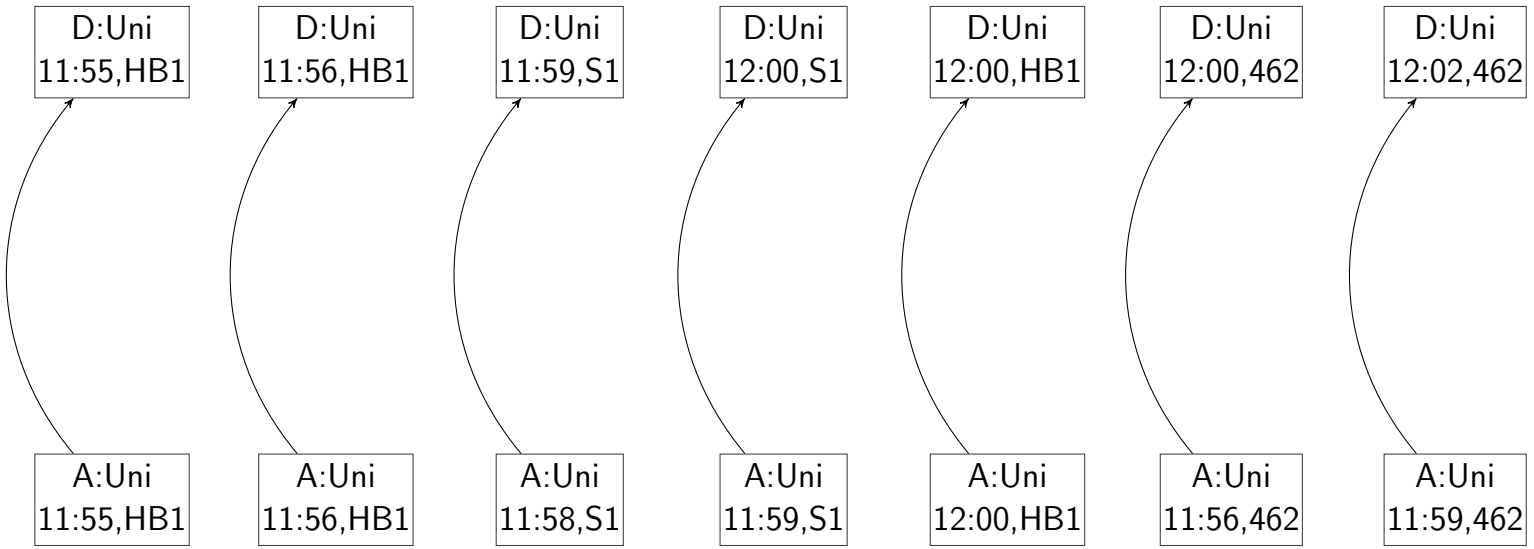
D:Uni 11:55,HB1	D:Uni 11:56,HB1	D:Uni 11:59,S1	D:Uni 12:00,S1	D:Uni 12:00,HB1	D:Uni 12:00,462	D:Uni 12:02,462

A:Uni 11:55,HB1	A:Uni 11:56,HB1	A:Uni 11:58,S1	A:Uni 11:59,S1	A:Uni 12:00,HB1	A:Uni 11:56,462	A:Uni 11:59,462

Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

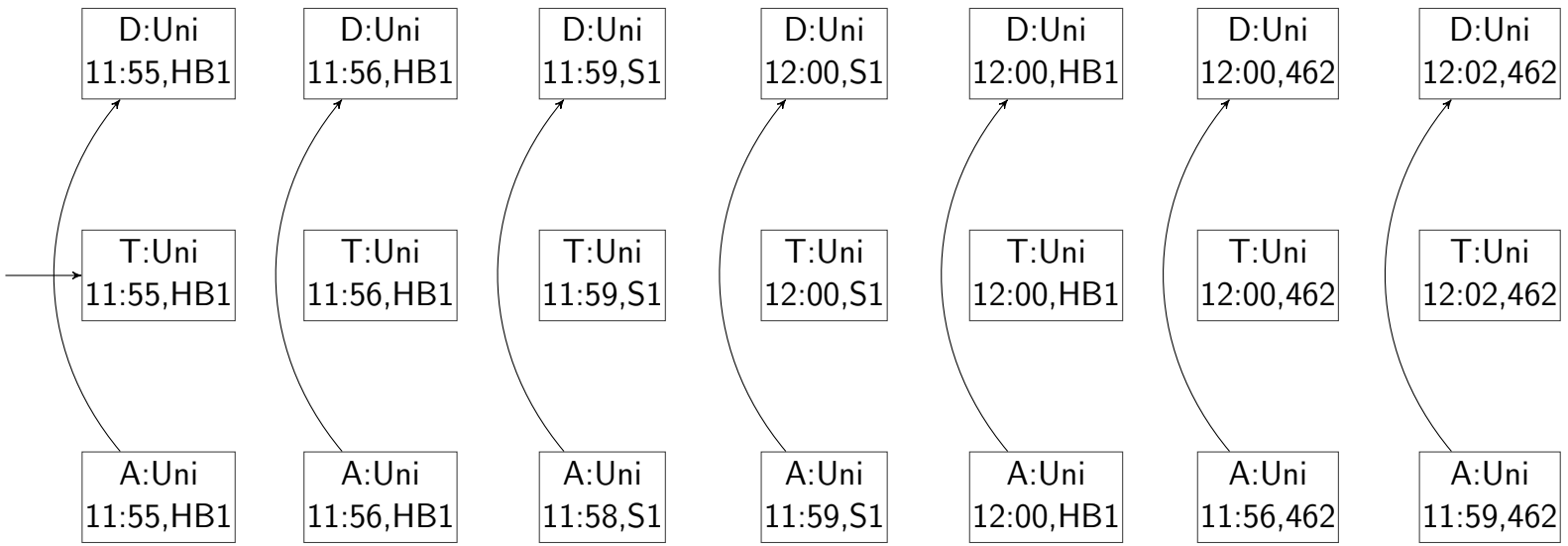
→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein



Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

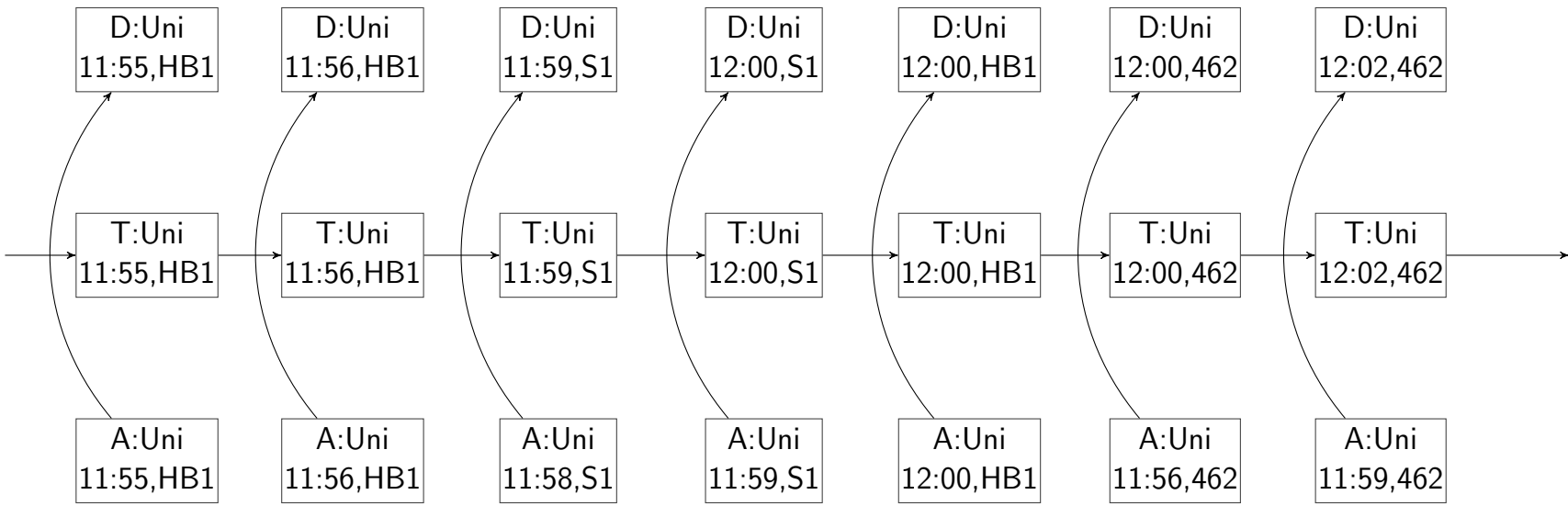
→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein



Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

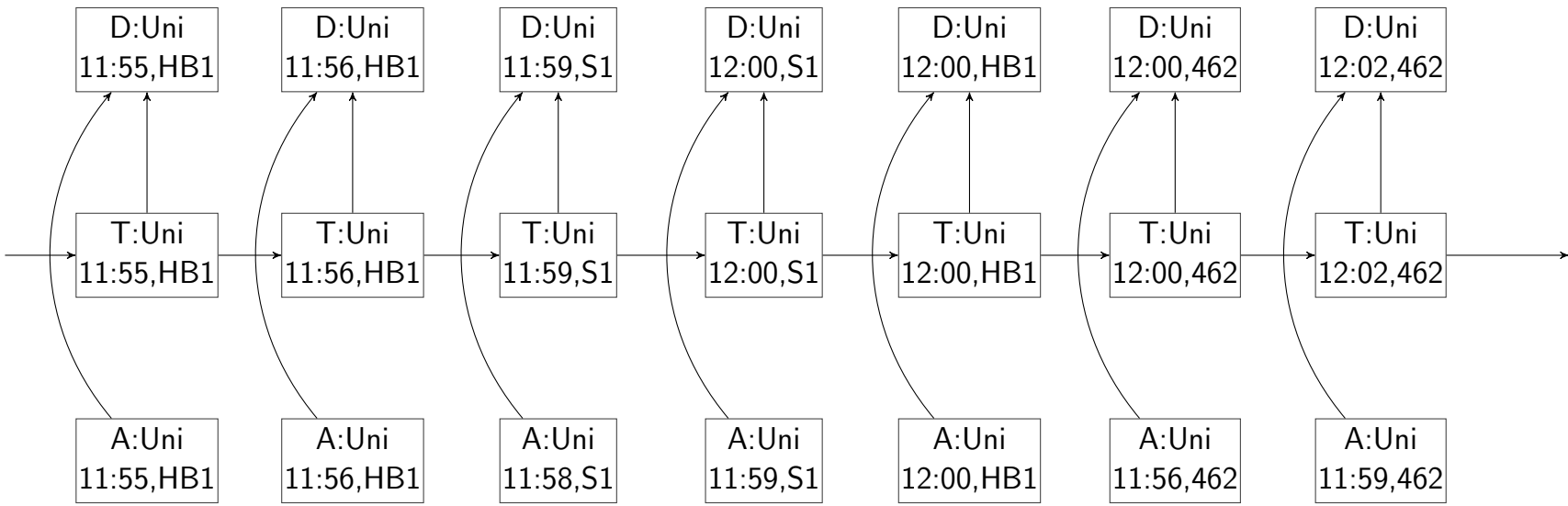
→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein



Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

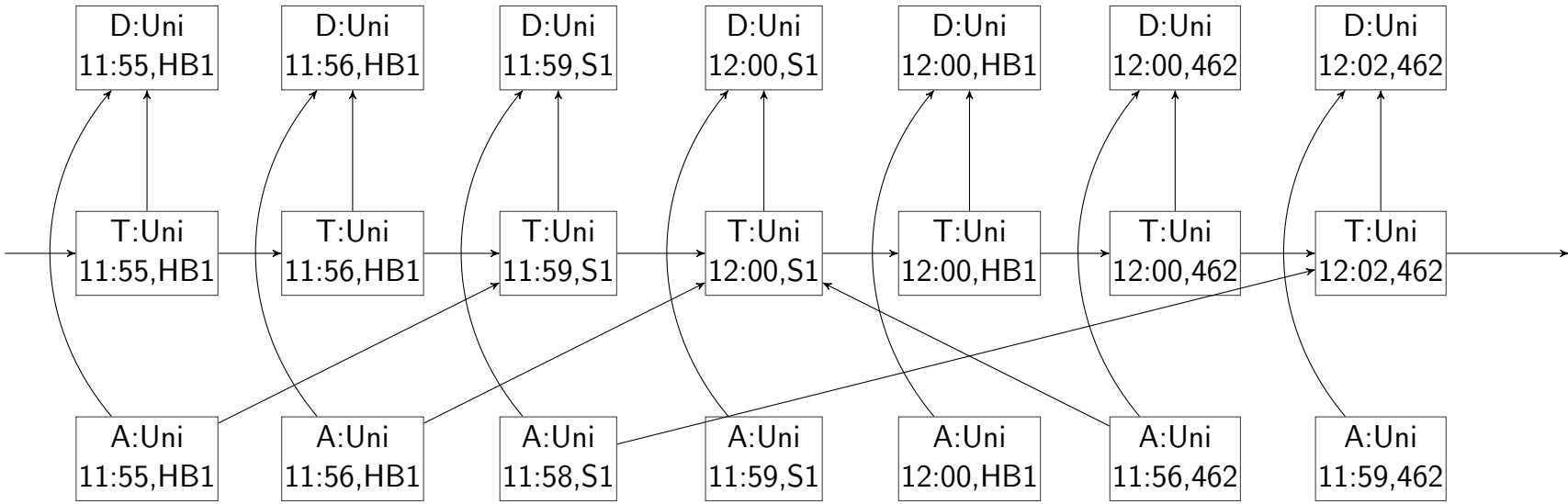
→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein



Nahverkehr als Graph – Mit Fahrplan

- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein

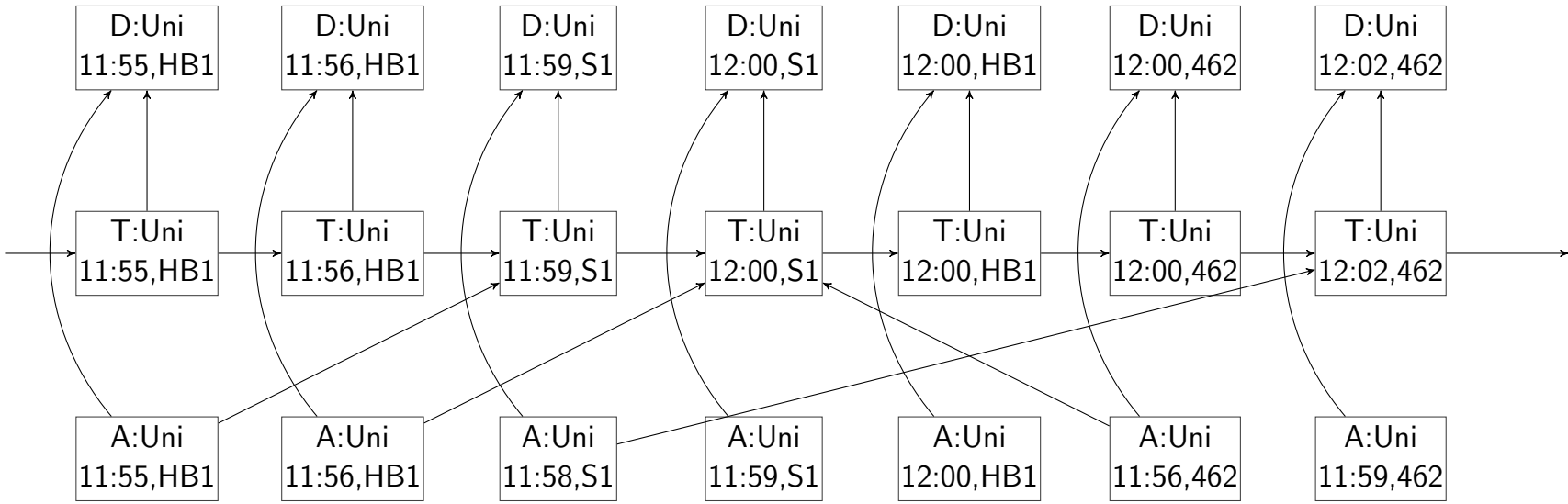


- Umsteigen: Hier nur mit mindestens 4 Minuten Puffer

Nahverkehr als Graph – Mit Fahrplan

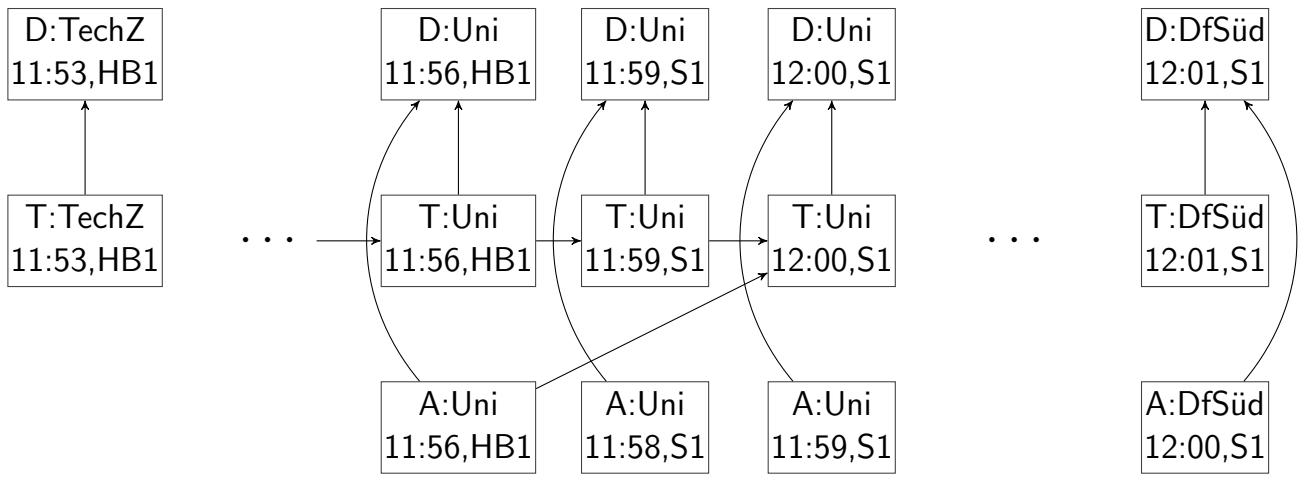
- Ein Graph ist ein guter Ansatz
- Ohne Fahrplandaten bringt er aber nichts

→ Jede Ankunft/Abfahrt (mit Zeit) muss eindeutig sein



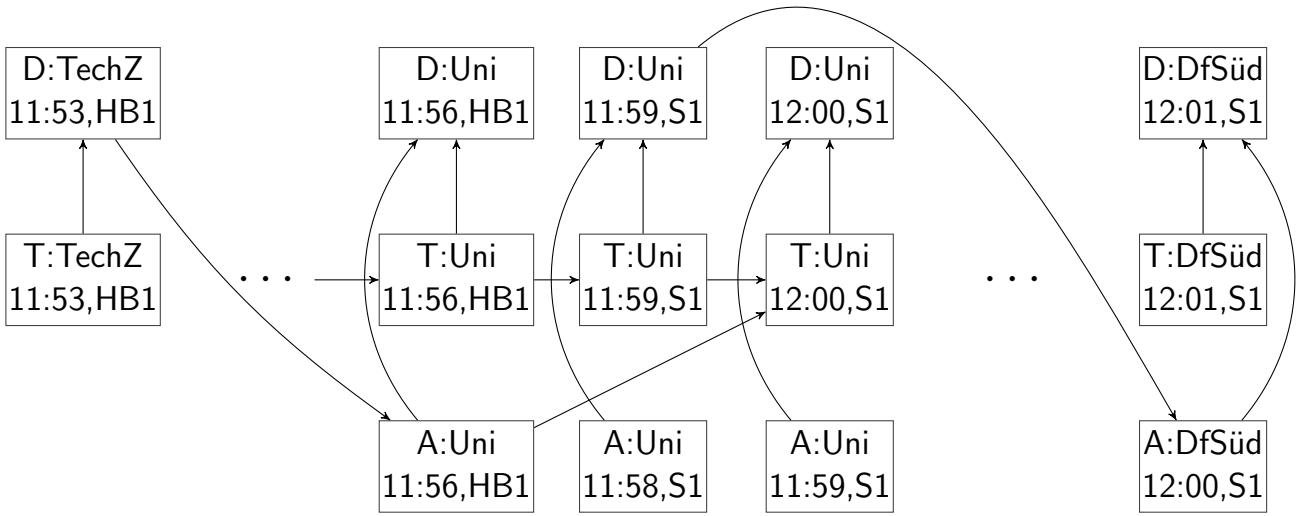
- Umsteigen: Hier nur mit mindestens 4 Minuten Puffer
- Eine Haltestelle ist damit vollständig bekannt

Nahverkehr als Graph – Mit Fahrplan (2)



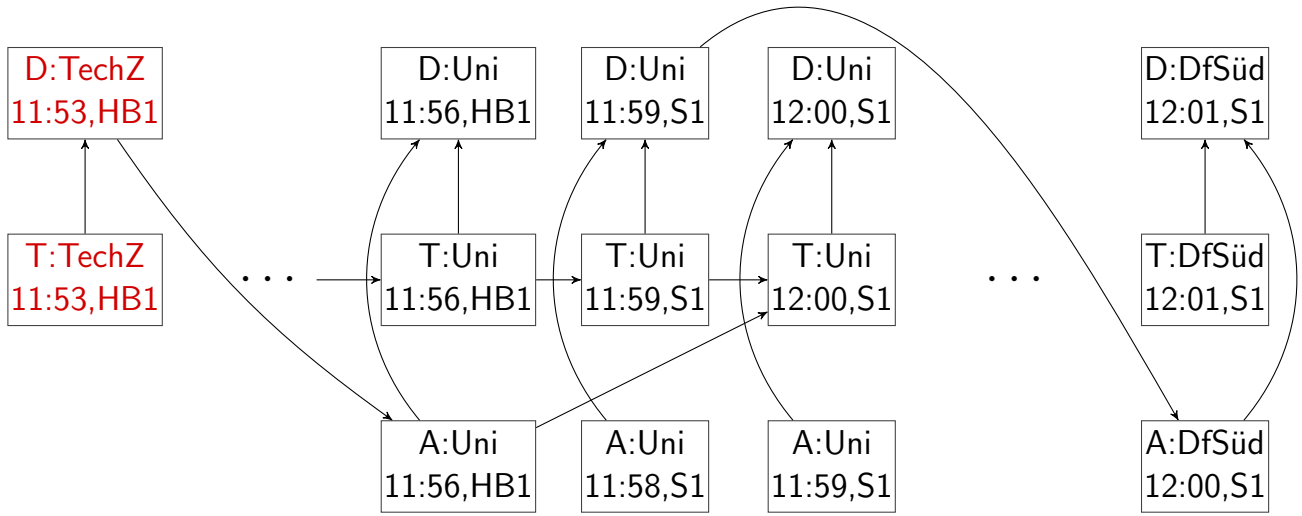
Nahverkehr als Graph – Mit Fahrplan (2)

- Zwischen Haltestellen: Direkte Verbindungen eintragen



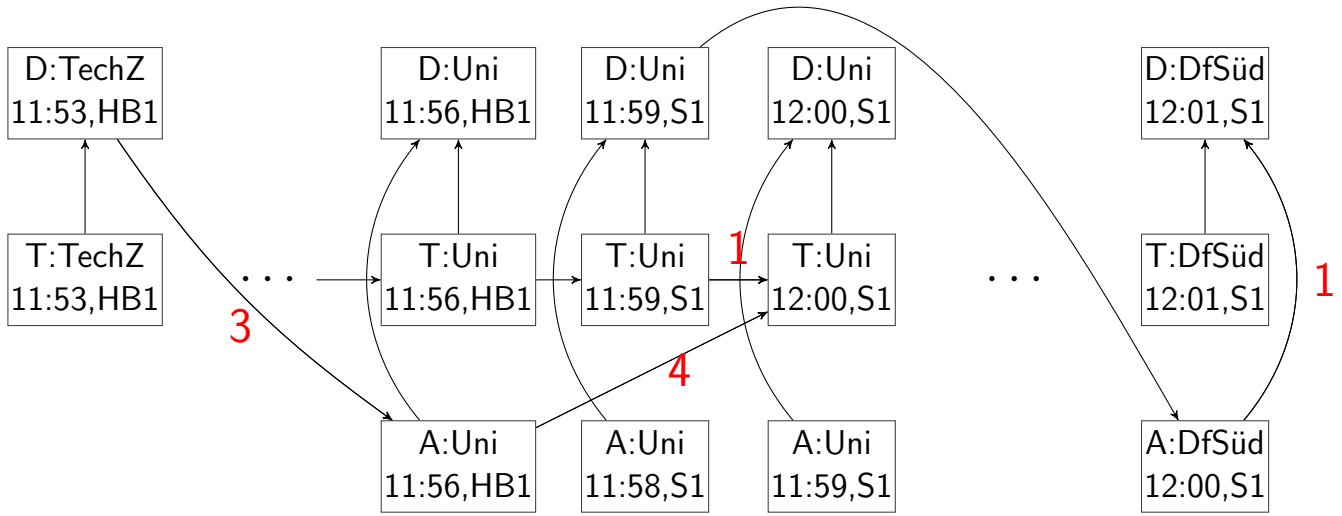
Nahverkehr als Graph – Mit Fahrplan (2)

- Zwischen Haltestellen: Direkte Verbindungen eintragen
- Bei Anfangs-/Endstation: Entsprechend Knoten weglassen



Nahverkehr als Graph – Mit Fahrplan (2)

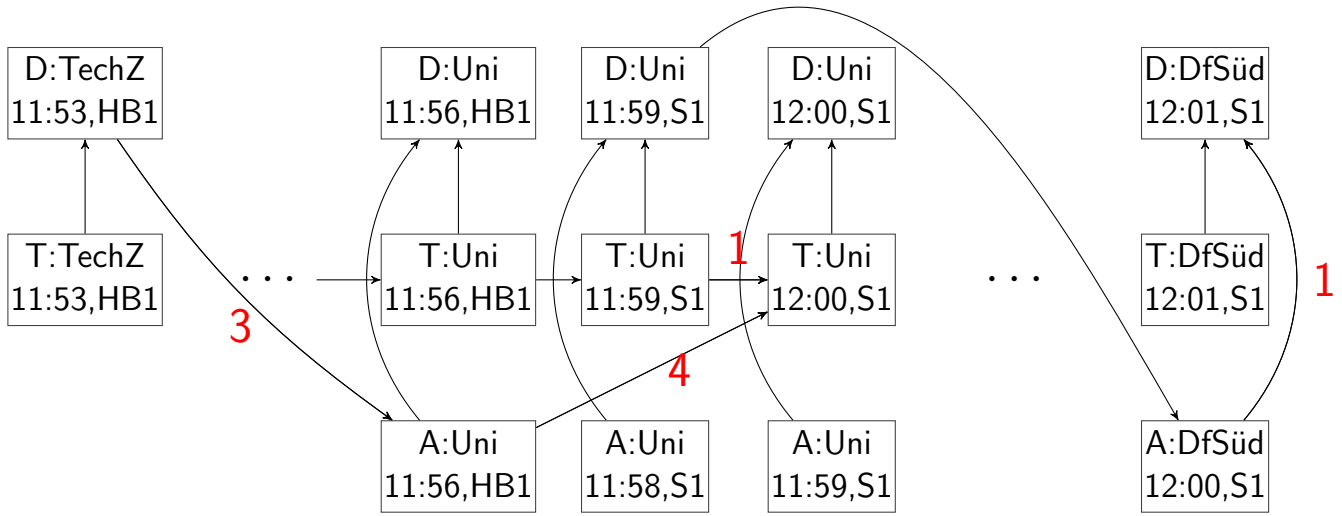
- Zwischen Haltestellen: Direkte Verbindungen eintragen
- Bei Anfangs-/Endstation: Entsprechend Knoten weglassen



- Jede Kante hat eine Reise-/Wartedauer

Nahverkehr als Graph – Mit Fahrplan (2)

- Zwischen Haltestellen: Direkte Verbindungen eintragen
- Bei Anfangs-/Endstation: Entsprechend Knoten weglassen

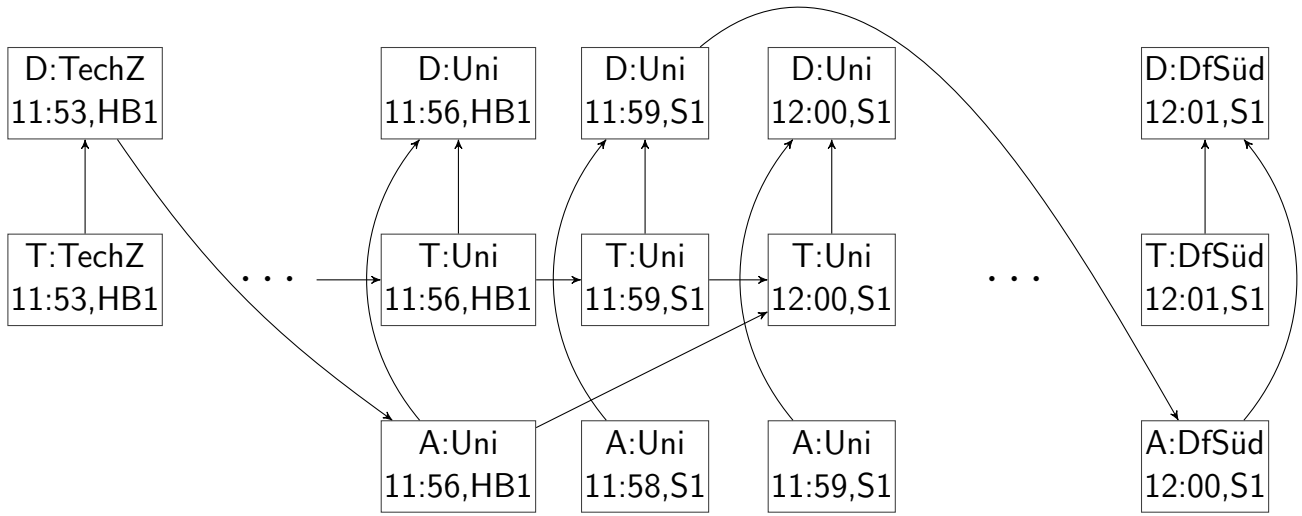


- Jede Kante hat eine Reise-/Wartedauer

→ Kürzester Weg kann gesucht werden

Nahverkehr als Graph – Mit Fahrplan (2)

- Zwischen Haltestellen: Direkte Verbindungen eintragen
- Bei Anfangs-/Endstation: Entsprechend Knoten weglassen



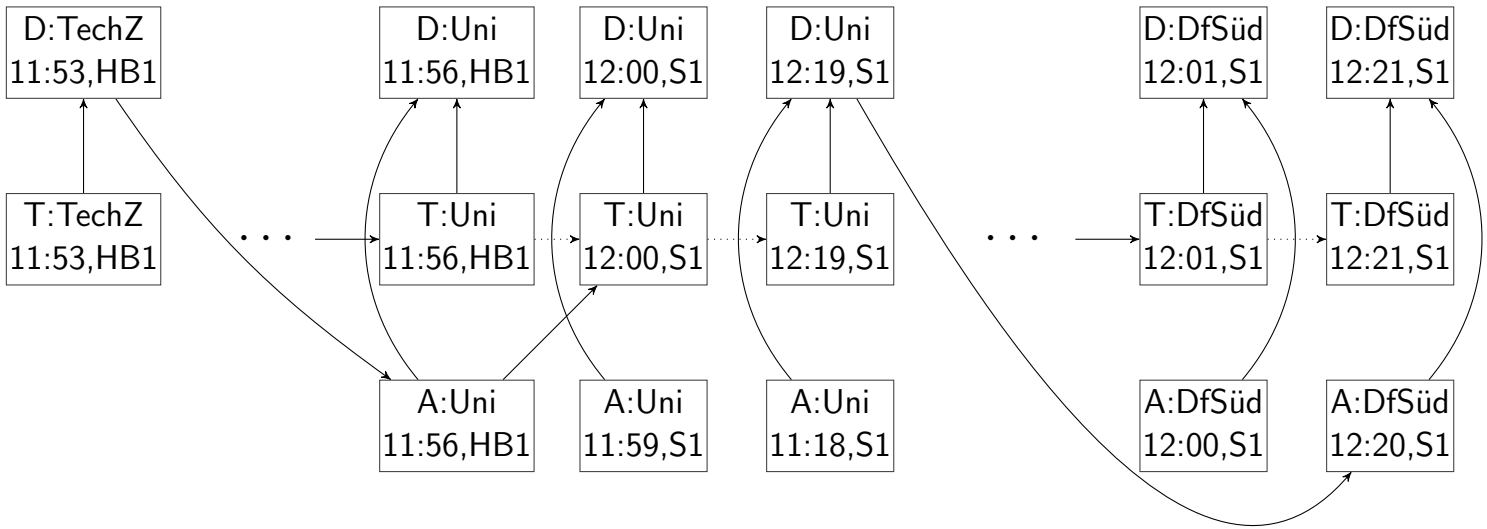
- Jede Kante hat eine Reise-/Wartedauer

→ Kürzester Weg kann gesucht werden

  - Fast gleiche Methode wie bei Straßennetzen u.ä.

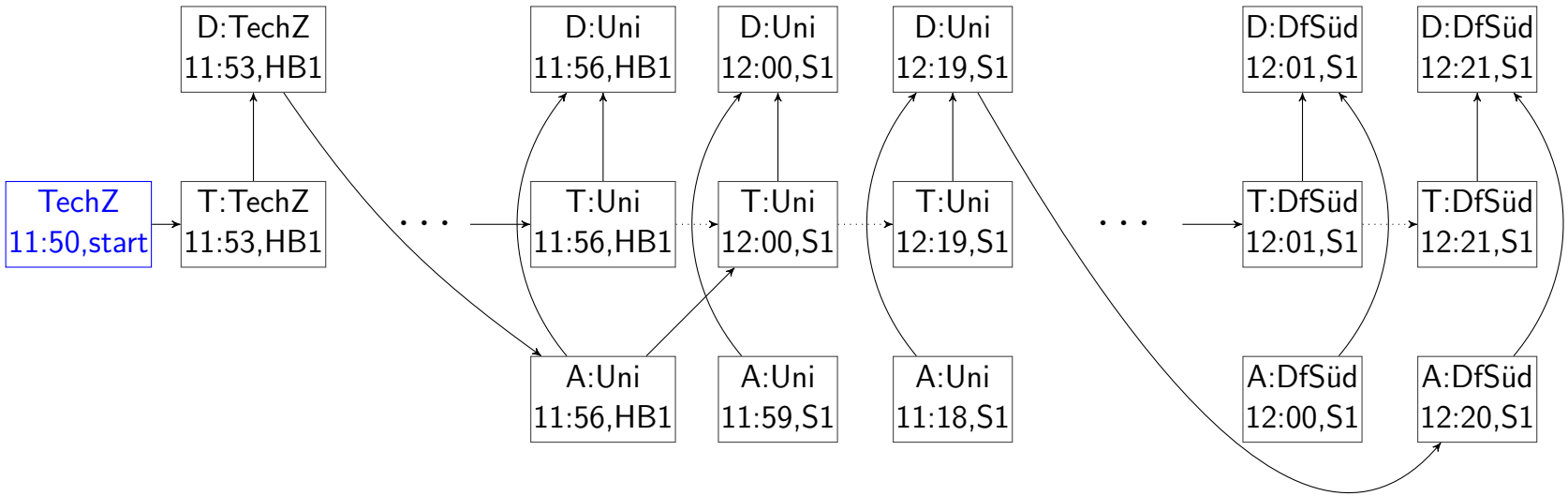
Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd



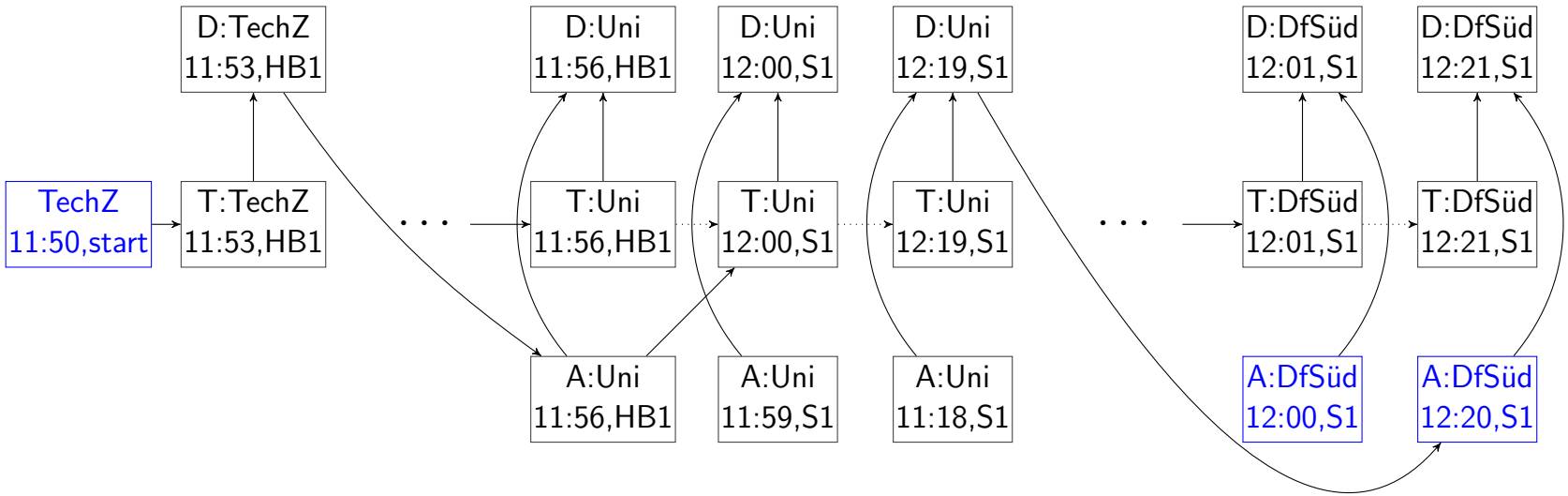
Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)



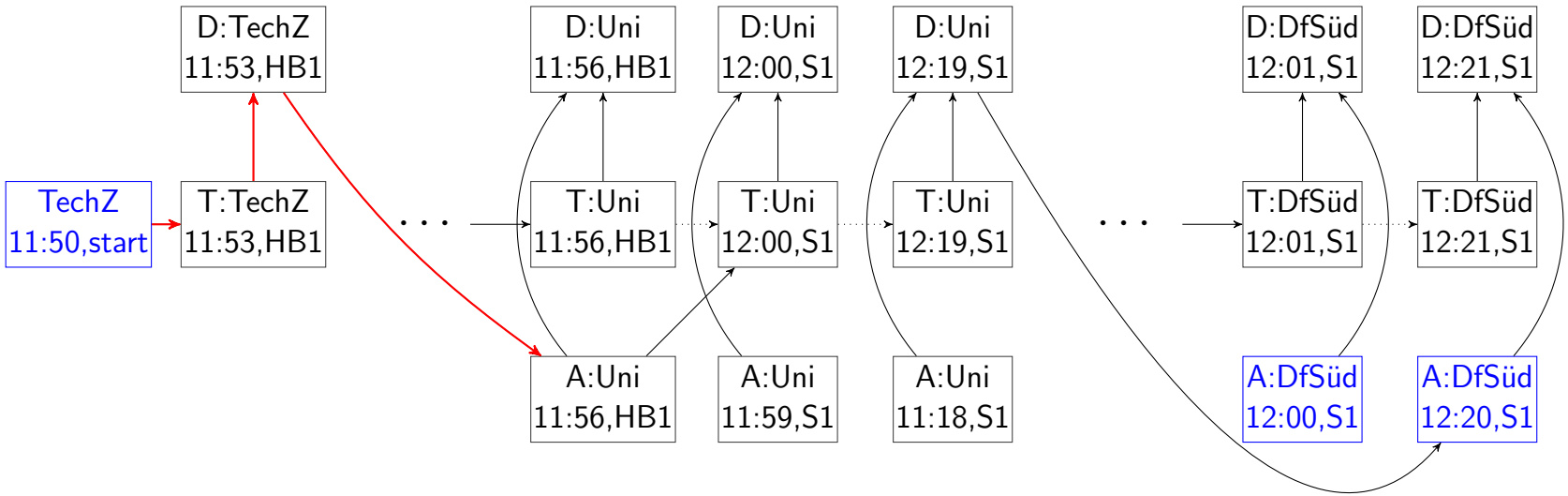
Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)
  - Ziel: Irgendein Ankunftsknoten



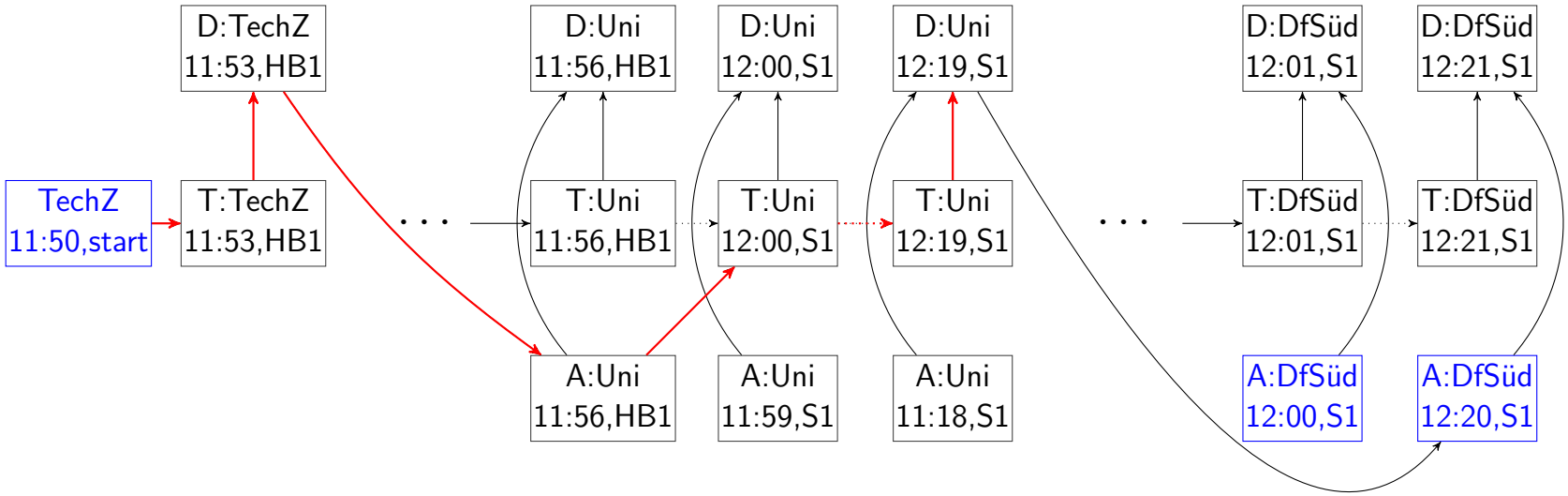
Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)
  - Ziel: Irgendein Ankunftsknoten



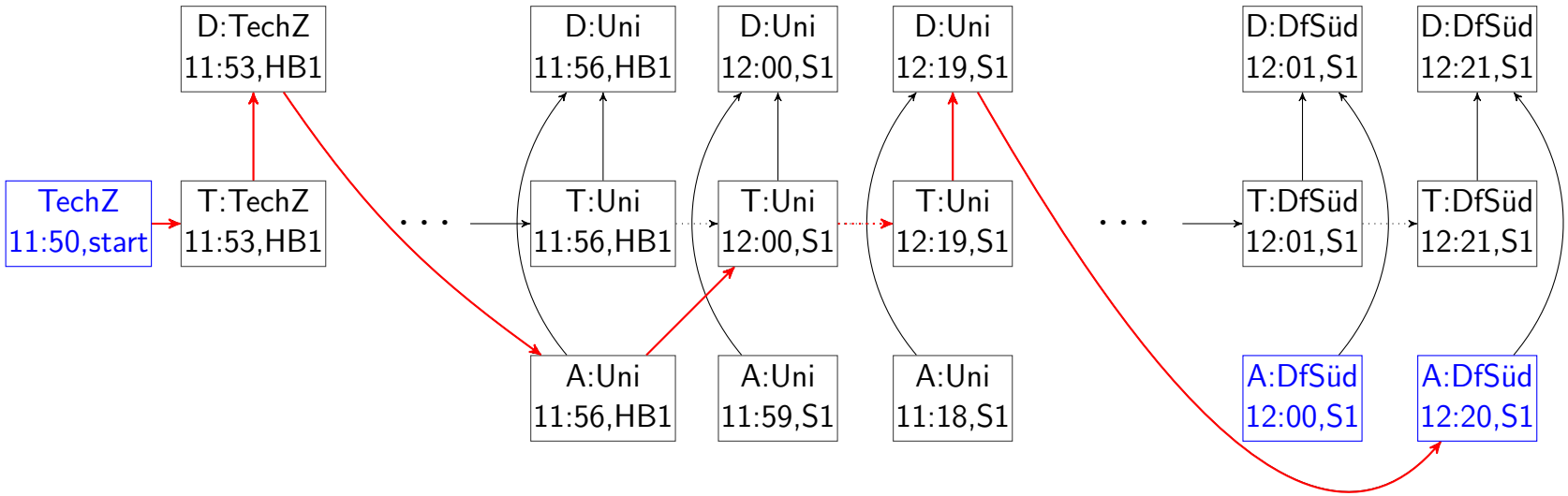
Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)
  - Ziel: Irgendein Ankunftsknoten



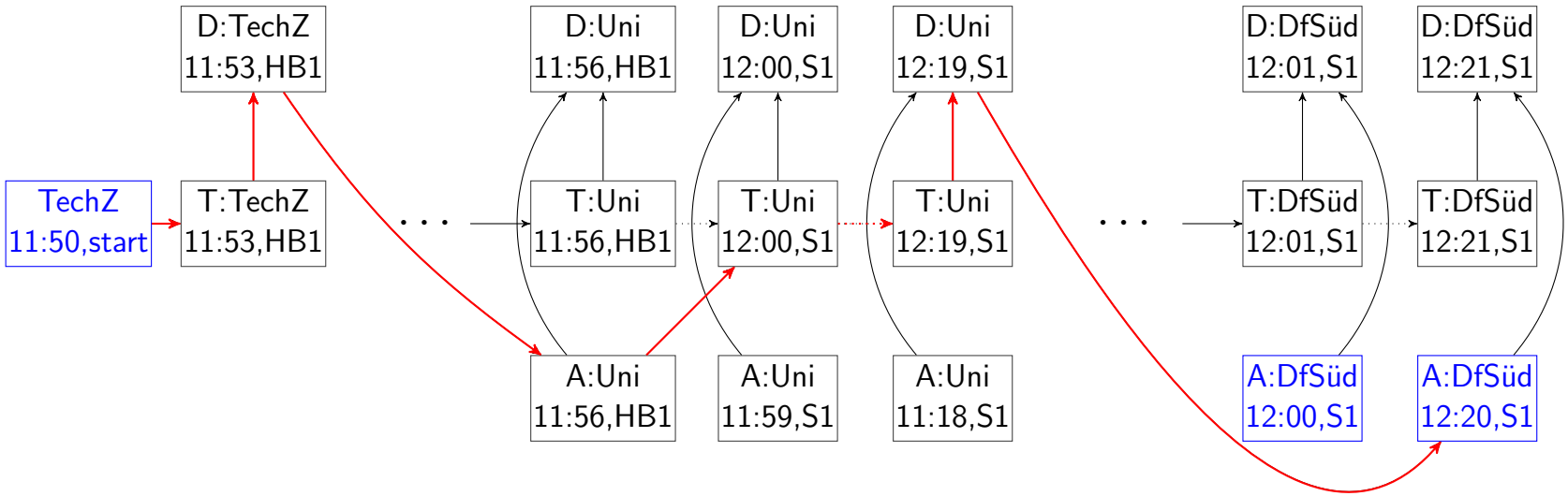
Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)
  - Ziel: Irgendein Ankunftsknoten



Verbindungssuche

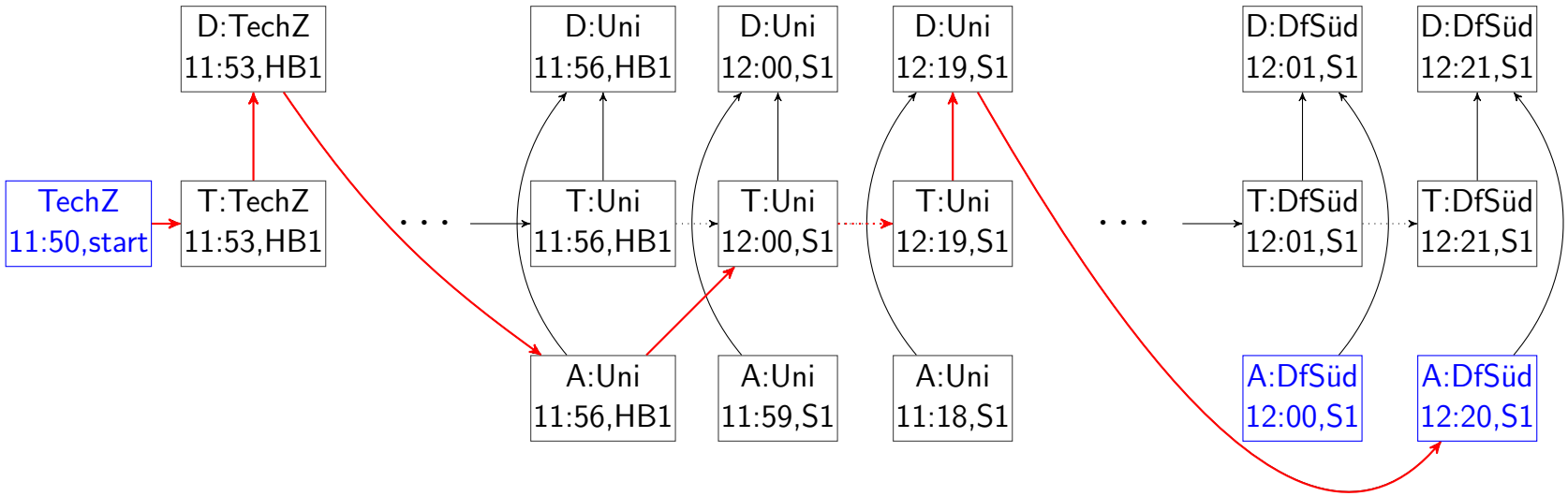
- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)
  - Ziel: Irgendein Ankunftsknoten



- Funktioniert – Aber skaliert nicht
  - Anfragen sind langsam, Graph passt u.U. noch nichtmal in den RAM

Verbindungssuche

- Beispiel: Technologiezentrum → Dorstfeld Süd
  - Neuer Startknoten mit Anfragezeit (erlaubt auch warten)
  - Ziel: Irgendein Ankunftsknoten



- Funktioniert – Aber skaliert nicht
  - Anfragen sind langsam, Graph passt u.U. noch nichtmal in den RAM

→ Preprocessing

Themen

1. Einleitung
2. Routenberechnung
3. Effiziente Routenberechnung
4. Fazit

Preprocessing: Idee

- Direktverbindungen sind einfach

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit
- Umsteigen ist dafür sehr schwierig

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit
- Umsteigen ist dafür sehr schwierig

→ Umsteigehalte für alle möglichen Anfragen vorberechnen

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit
- Umsteigen ist dafür sehr schwierig

→ Umsteigehalte für alle möglichen Anfragen vorberechnen

  - Unabhängig von Start/Ziel

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit
- Umsteigen ist dafür sehr schwierig

→ Umsteigehalte für alle möglichen Anfragen vorberechnen

  - Unabhängig von Start/Ziel
  - Unabhängig von der Uhrzeit(!)

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit
- Umsteigen ist dafür sehr schwierig

→ Umsteigehalte für alle möglichen Anfragen vorberechnen

- Unabhängig von Start/Ziel
- Unabhängig von der Uhrzeit(!)

→ *Transfer Patterns*

- Speichern zu jeder Starthaltestelle A optimale Verbindungen zu allen anderen Haltestellen

Preprocessing: Idee

- Direktverbindungen sind einfach
  - Fahrplan betrachten und schnellste Verbindung ausgeben
  - Im Computer: Kostet quasi keine Zeit
- Umsteigen ist dafür sehr schwierig

→ Umsteigehalte für alle möglichen Anfragen vorberechnen

- Unabhängig von Start/Ziel
- Unabhängig von der Uhrzeit(!)

→ *Transfer Patterns*

- Speichern zu jeder Starthaltestelle A optimale Verbindungen zu allen anderen Haltestellen
  - Als Folge von Umsteigehalten
  - Ohne Zeitangaben

Transfer Patterns: Berechnung

Für jede Haltestelle A:

Dortmund Hbf

Transfer Patterns: Berechnung

Für jede Haltestelle A:

- Betrachte alle Abfahrtszeiten an A

Dortmund Hbf

Transfer Patterns: Berechnung

Für jede Haltestelle A:

- Betrachte alle Abfahrtszeiten an A
- Suche jeweils kürzeste Verbindung zu jeder Haltestelle $B \neq A$

Düsseldorf Hbf

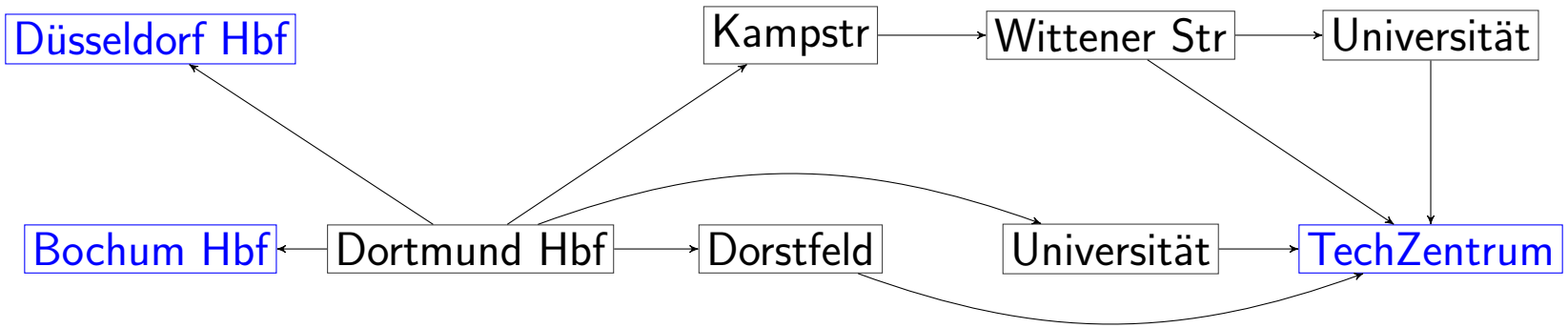
Bochum Hbf Dortmund Hbf TechZentrum

Transfer Patterns: Berechnung

Für jede Haltestelle A:

- Betrachte alle Abfahrtszeiten an A
- Suche jeweils kürzeste Verbindung zu jeder Haltestelle $B \neq A$
- Speichere Verbindung in einem neuen Graphen

Beispiel für Uni Dortmund (Ausschnitt):

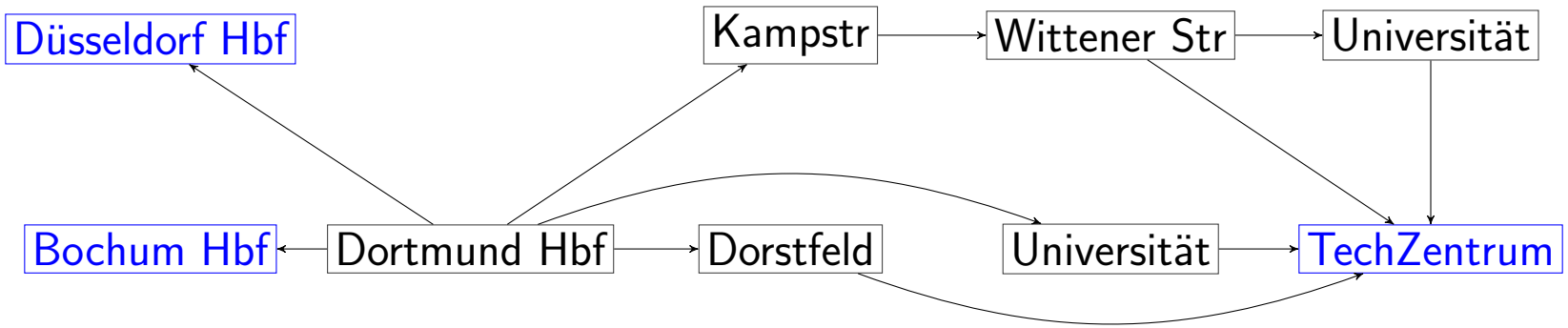


Transfer Patterns: Berechnung

Für jede Haltestelle A:

- Betrachte alle Abfahrtszeiten an A
- Suche jeweils kürzeste Verbindung zu jeder Haltestelle $B \neq A$
- Speichere Verbindung in einem neuen Graphen

Beispiel für Uni Dortmund (Ausschnitt):



- Knoten $\widehat{=}$ Umsteigehaltestelle
- Blaue Knoten $\widehat{=}$ Zielhaltestellen

Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

Routing mit Transfer Patterns

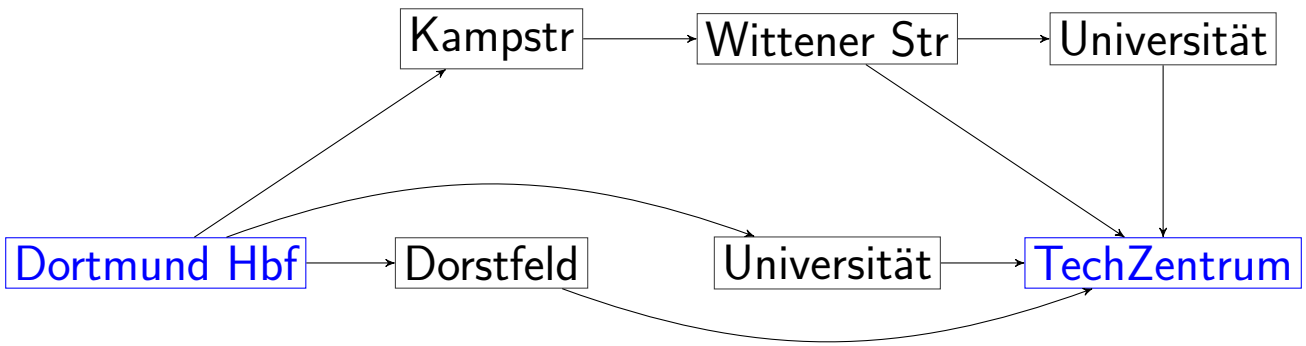
Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A

Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

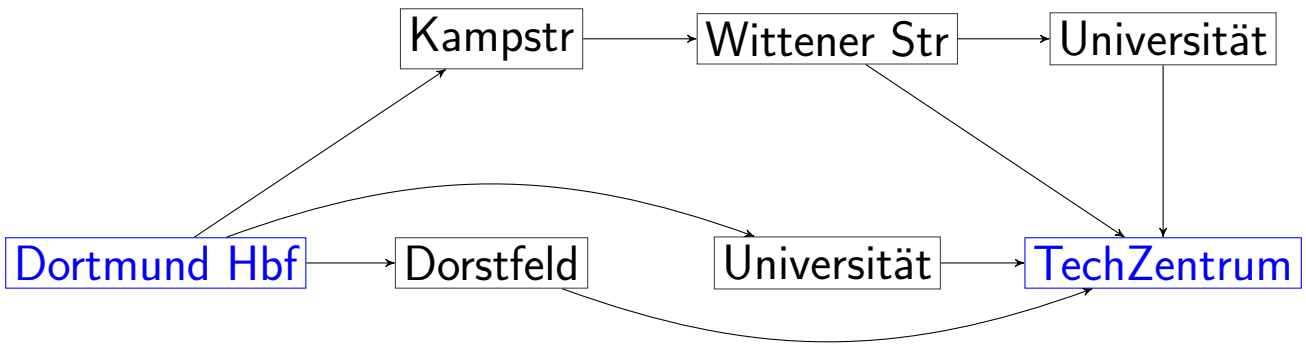
- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg



Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

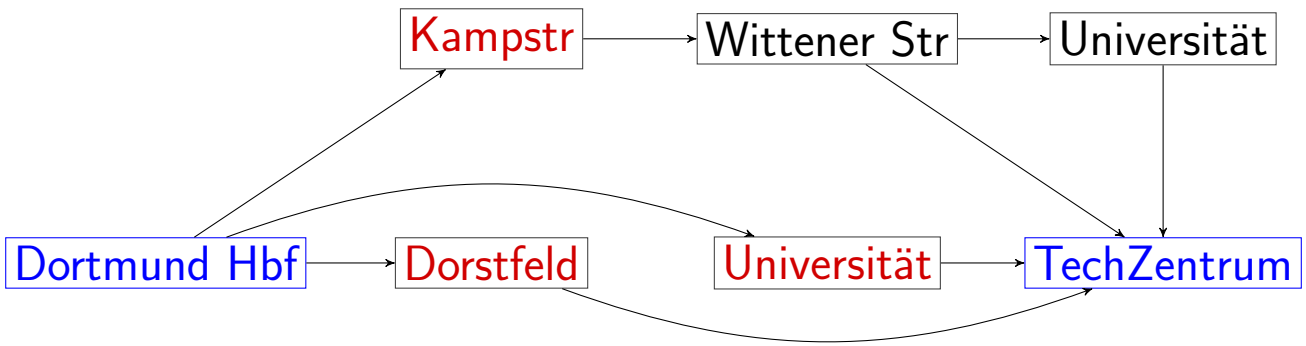
- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:



Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

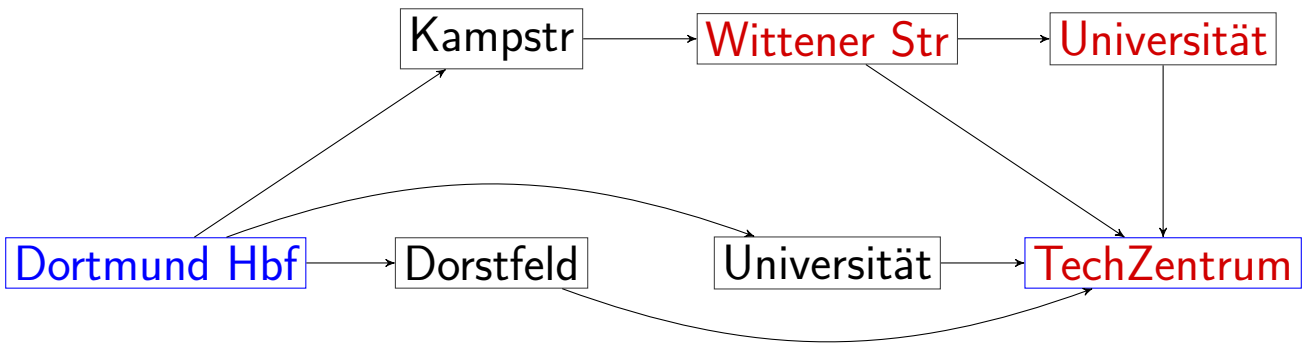
- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab



Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

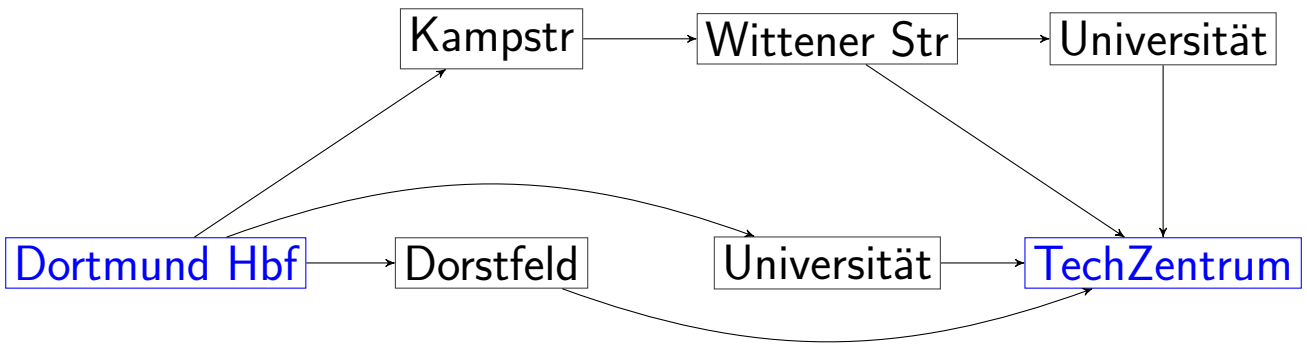
- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...



Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...
- Gebe Verbindung mit frühester Ankunft aus

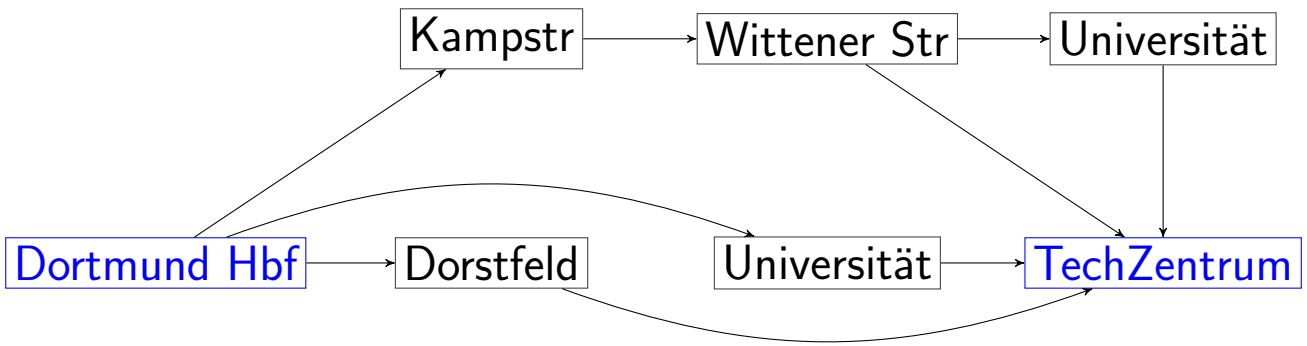


Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...
- Gebe Verbindung mit frühester Ankunft aus

Für Do Hbf $\rightarrow$ Technologiezentrum: Nur 4 Kandidaten

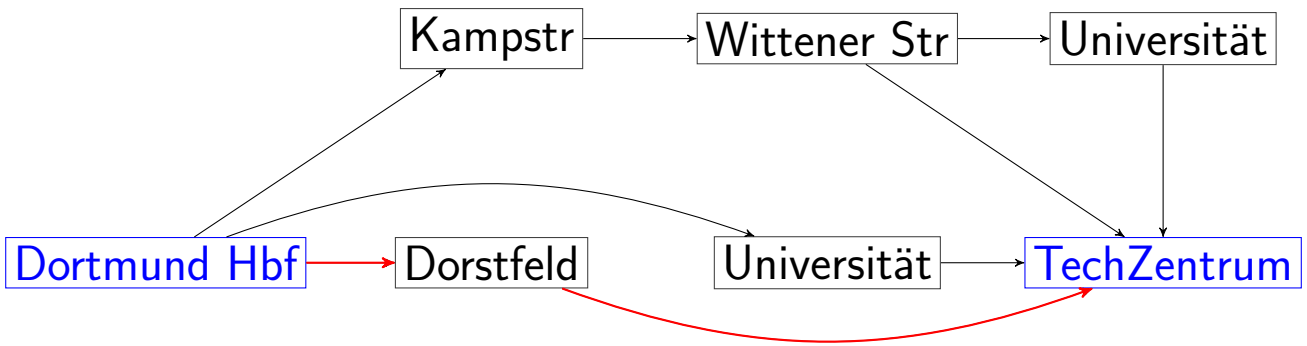


Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...
- Gebe Verbindung mit frühester Ankunft aus

Für Do Hbf $\rightarrow$ Technologiezentrum: Nur 4 Kandidaten

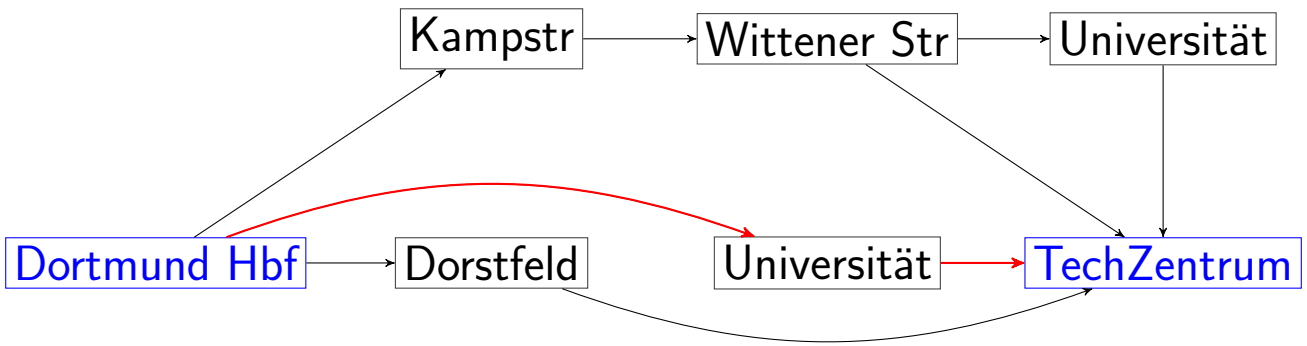


Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...
- Gebe Verbindung mit frühester Ankunft aus

Für Do Hbf $\rightarrow$ Technologiezentrum: Nur 4 Kandidaten

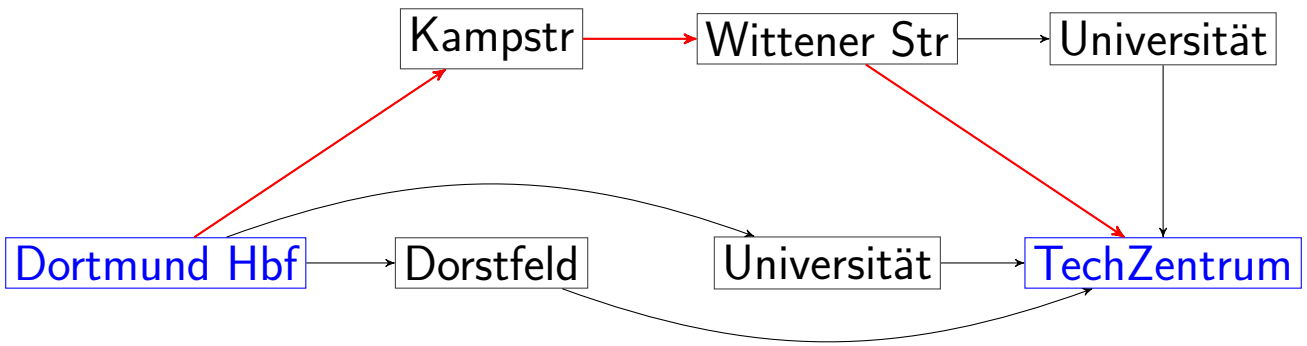


Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...
- Gebe Verbindung mit frühester Ankunft aus

Für Do Hbf $\rightarrow$ Technologiezentrum: Nur 4 Kandidaten

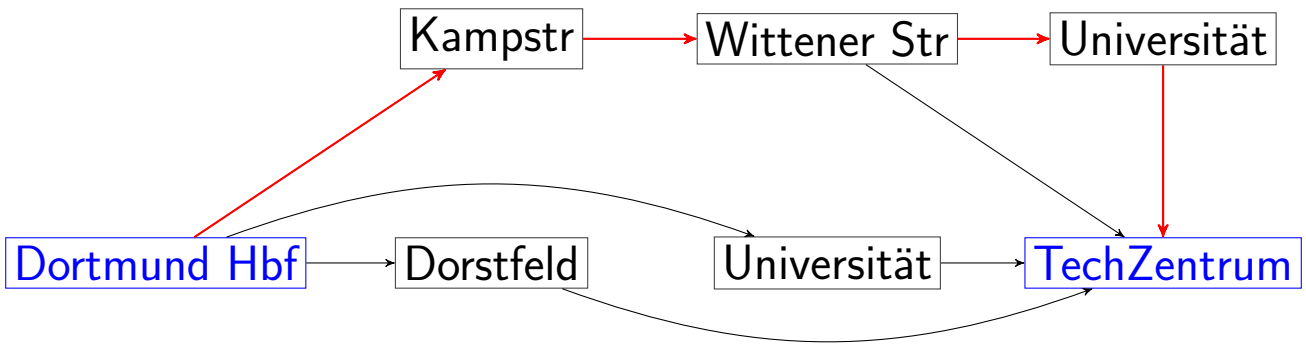


Routing mit Transfer Patterns

Für eine Anfrage $A \rightarrow B$ mit Startzeit t:

- Lade Transfer Pattern-Graph für A
- Suche Zielknoten B, werfe alle anderen weg
- Für jede Verbindung $A \rightarrow B$:
  - Lese früheste Ankunftszeit am ersten Umsteigebahnhof aus Fahrplan ab
  - Lese früheste Ankunftszeit am zweiten Umsteigebahnhof aus Fahrplan ab
  - ...
- Gebe Verbindung mit frühester Ankunft aus

Für Do Hbf $\rightarrow$ Technologiezentrum: Nur 4 Kandidaten



Benchmarks

Netz	Preprocessing	Output	Queryzeit
Schweiz			
New York			
Nordamerika			

Benchmarks

Netz	Preprocessing	Output	Queryzeit
Schweiz	4 h		
New York	64 h		
Nordamerika	571 h		

Benchmarks

Netz	Preprocessing	Output	Queryzeit
Schweiz	4 h	154 MB	
New York	64 h	786 MB	
Nordamerika	571 h	7151 MB	

Benchmarks

Netz	Preprocessing	Output	Queryzeit
Schweiz	4 h	154 MB	1 ms
New York	64 h	786 MB	6 ms
Nordamerika	571 h	7151 MB	10 ms

Benchmarks

Netz	Preprocessing	Output	Queryzeit
Schweiz	4 h	154 MB	1 ms
New York	64 h	786 MB	6 ms
Nordamerika	571 h	7151 MB	10 ms

- Quelle: [Bas+10]
  - Preprocessing auf 2010er Xeon/Opteron Cluster
  - Query-Messung auf einer Maschine davon, Daten im RAM

Benchmarks

Netz	Preprocessing	Output	Queryzeit
Schweiz	4 h	154 MB	1 ms
New York	64 h	786 MB	6 ms
Nordamerika	571 h	7151 MB	10 ms

- Quelle: [Bas+10]
  - Preprocessing auf 2010er Xeon/Opteron Cluster
  - Query-Messung auf einer Maschine davon, Daten im RAM

→ Sehr effizient

→ Alle Graphen passen in den RAM

Themen

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen
  - Teils notwendige, teils optionale Verbesserungen

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen
  - Teils notwendige, teils optionale Verbesserungen
  - Ändern nichts am grundlegenden Prinzip

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen
  - Teils notwendige, teils optionale Verbesserungen
  - Ändern nichts am grundlegenden Prinzip
- Preprocessing ist weitgehend Brute Force-Ansatz
- Nicht effizient, aber sehr effektiv

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen
  - Teils notwendige, teils optionale Verbesserungen
  - Ändern nichts am grundlegenden Prinzip
- Preprocessing ist weitgehend Brute Force-Ansatz
- Nicht effizient, aber sehr effektiv
- Und nur einmal pro Fahrplanwechsel nötig

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen
  - Teils notwendige, teils optionale Verbesserungen
  - Ändern nichts am grundlegenden Prinzip
- Preprocessing ist weitgehend Brute Force-Ansatz
- Nicht effizient, aber sehr effektiv
- Und nur einmal pro Fahrplanwechsel nötig
- Transfer Patterns 2010 schon bei Google Maps im Einsatz

Fazit / Realitätsabgleich

- Ein paar Details wurden unterschlagen
  - Teils notwendige, teils optionale Verbesserungen
  - Ändern nichts am grundlegenden Prinzip
- Preprocessing ist weitgehend Brute Force-Ansatz
- Nicht effizient, aber sehr effektiv
- Und nur einmal pro Fahrplanwechsel nötig
- Transfer Patterns 2010 schon bei Google Maps im Einsatz
- Es gibt auch ganz andere Methoden
  - Teils ohne Preprocessing
  - Queryzeiten im Sekundenbereich bei Großstädten

Literatur

[Bas+10] Hannah Bast u. a. “Fast Routing in Very Large Public Transportation Networks Using Transfer Patterns”. In: *ESA 2010*. Bd. 6346. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, S. 290–301. ISBN: 978-3-642-15774-5.