Vorlesung Kommunikationssysteme
Wintersemester 2026/27

LAN Komponenten
und WAN Technologien

Christoph Lindemann

Zeitplan

Nr.	Datum	Thema
01	16.10.26	Organisation, Internet Trends und Grundlagen von Android
02	23.10.26	Protokolldesign und das Internet
03	30.10.26	Anwendungen und Netzwerkprogrammierung
04	06.11.26	LAN und Medienzugriff
05	13.11.26	Ethernet und drahtlose Netze
06	20.11.26	LAN Komponenten und WAN Technologien
07	27.11.26	Routingverfahren und Protokolle
08	04.12.26	Internetworking und Adressierung mit IP
09	11.12.26	IP Datagramme
10	18.12.26	Zusätzliche Protokolle und Technologien
11	08.01.27	User Datagram Protocol und Transmission Control Protocol
12	15.01.27	TCP Überlastkontrolle / Internet Routing und Routingprotokolle
13	22.01.27	TCP für Hochgeschwindigkeitsnetze
14	29.01.27	Review der Vorlesung

Überblick

Ziele:

Vertiefen des Verständnisses für Routingverfahren und Protokolle

Themen:

- Dienste der Internetschicht
- Virtuelle Leitungen
- Routing und Weiterleitung
- Link State Verfahren
- Distanzvektor Verfahren

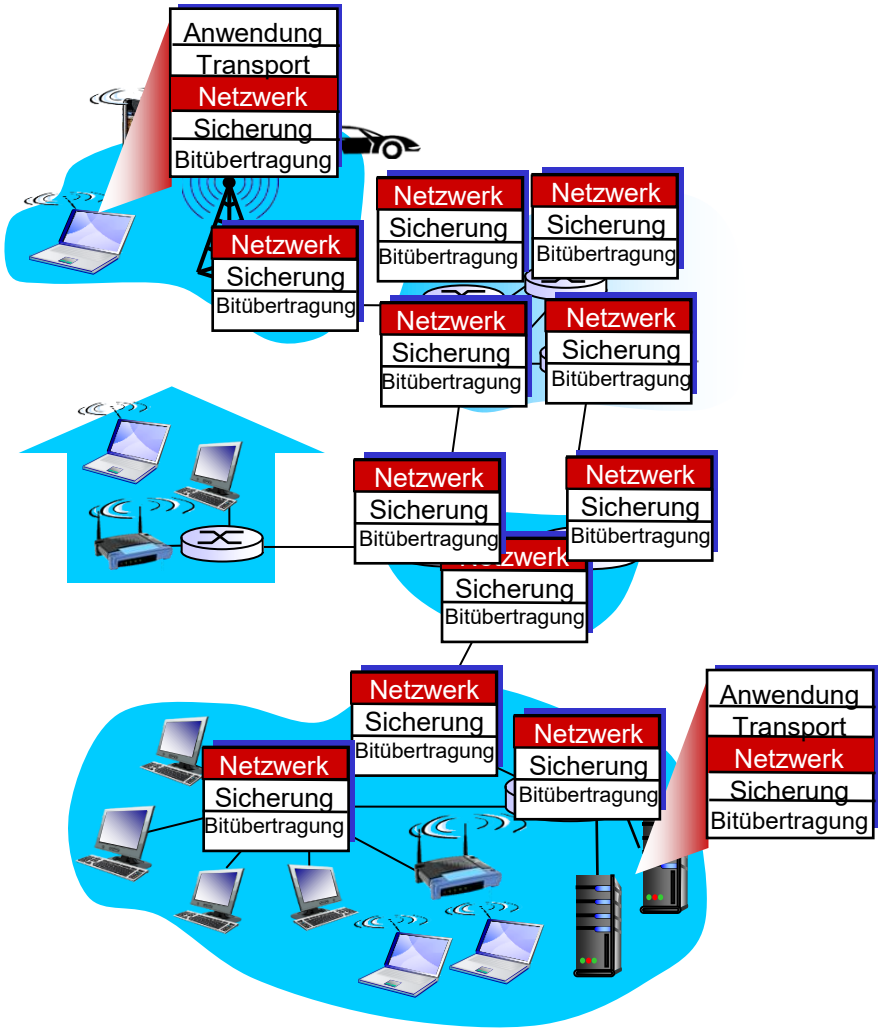
Dienste der Internetschicht

Aufgaben der Internetschicht

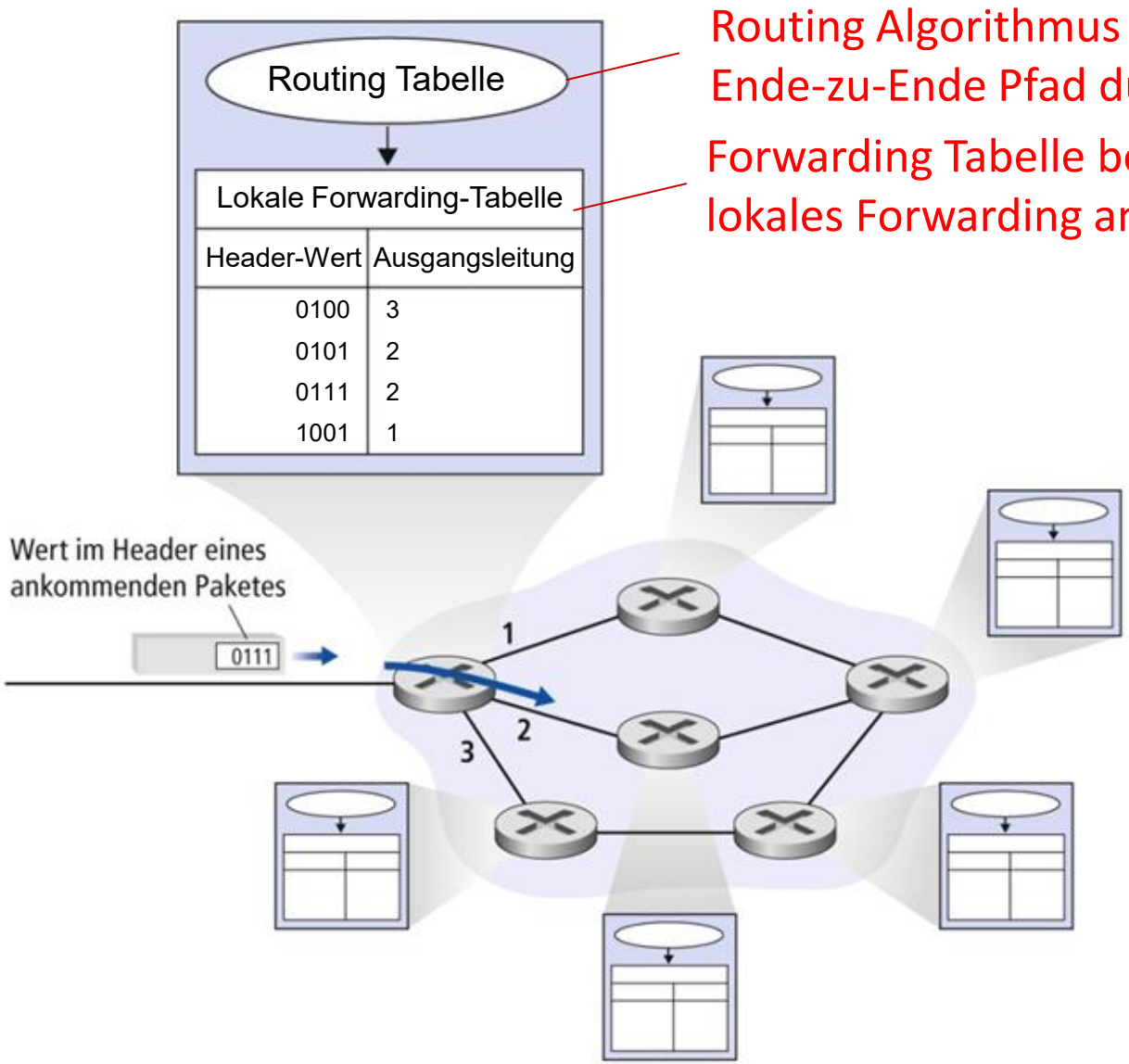
- Pakettransport vom sendenden Host zum empfangenden Host
- Protokolle der Internetschicht sind auf allen Routern und Hosts vorhanden

3 wichtige Aufgaben:

- *Pfadbestimmung:* Bestimmung des Pfades von der Quelle zum Ziel: *Routing Algorithmen*
- *Forwarding:* Pakete innerhalb des Routers vom Eingang zum Ausgang bewegen
- *Pfadaufbau:* Manche Netz-architekturen erfordern einen expliziten Verbindungsaufbau vor der Datenübertragung



Routing und Forwarding



Routing Algorithmus bestimmt Ende-zu-Ende Pfad durch Netzwerk

Forwarding Tabelle bestimmt lokales Forwarding an Router

Dienstmodelle

F: Welches *Dienstmodell* ist für den “Kanal”, der Paket vom Sender zum Empfänger transportiert, geeignet?

Beispiele für Dienstmodelle für Datagramme (Pakete):

- Garantierte Zustellung
- Garantierte Zustellung in weniger als 40 ms Verzögerung

Beispiele für Dienstmodelle für einen Paketfluss:

- Korrekte Reihenfolge der Zustellung der Datagramme
- Garantierte minimale Bandbreite für den Fluss
- Beibehaltung der Zeit-intervalle zwischen den Datagrammen

Dienstmodelle auf Internetschicht:

Netz Architektur	Dienst-modell	Garantien ?				Informationen zur Netzlast
		Bandbr.	Verlust	Ordnung	Timing	
Internet	best effort	nein	ja	nein	nein	Verzögerung Paketverlust
ATM	CBR	constant rate	ja	ja	ja	Keine Überlast möglich
ATM	VBR	guaranteed rate	ja	ja	ja	Keine Überlast möglich
ATM	ABR	guaranteed minimum	nein	ja	nein	ja
ATM	UBR	keine	nein	ja	nein	nein

Virtuelle Leitungen

Virtuelle Leitungen (VCs)

"Pfad zwische Quelle und Ziel verhält sich ähnlich wie eine Telefonleitung"

- Bezüglich Performance
- Bezüglich Netz-Verhalten entlang Pfad

- Expliziter Verbindungsaufbau vor Datenübertragung notwendig
- Jedes Paket trägt VC Kennung (Nicht die ID des Ziels)
- *Jeder* Router auf dem Pfad kennt die laufenden Verbindungen
  - Bei Verbindung auf Transportschicht: Nur die Endsysteme involviert
- Link und Router Resourcen (Bandbreite, Puffer) können auf Basis der VCs zugewiesen werden -> Leitungsähnlich

VC Implementierung

Eine VC besteht aus:

1. *Pfad* von Quelle zum Ziel
2. *VC Nummern*, eine Nummer für jeden Link auf dem Pfad
3. *Einträgen in den Forwarding Tabelle* in den Routern auf dem Pfad

- Zum VC zugehöriges Paket trägt VC Nummer (anstelle der Zieladresse)
- VC Nummer kann bei jedem Link geändert werden.
  - Neue VC Nummer erhalten aus der Forwarding Tabelle

VC Weiterleitungstabelle

VC Nummer

Interface Nummer

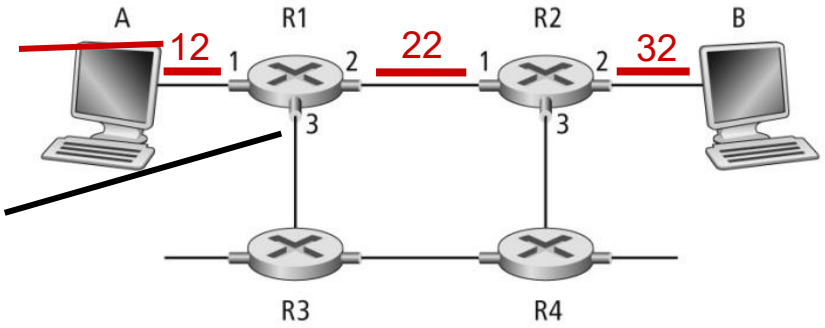


Abbildung 4.3: Ein einfaches Netzwerk mit virtuellen Verbindungen.

Forwarding Tabelle des Routers R1:

Eingehendes Interface	Eingehende VC	Ausgehendes Interface	Ausgehende VC
1	12	2	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

VC Router verwalten Statusinformationen der Verbindungen!

VC: Signalisierungsprotokolle

- Verwendet zur Verwaltung und zum Auf- und Abbau von VCs
- ATM, frame-relay, X.25

Im heutigen Internet nicht mehr vorhanden

Paketvermittelte Netze: das Netzmodell

- Kein Verbindungsaufbau auf Internetschicht
- Router: Kein Wissen über Ende-zu-Ende Verbindungen
  - Konzept “Verbindung” auf Internetschicht unbekannt
- Pakete werden anhand ihrer Zieladresse weitergeleitet
  - Pakete zwischen Host A und B können über unterschiedliche Pfade geleitet werden

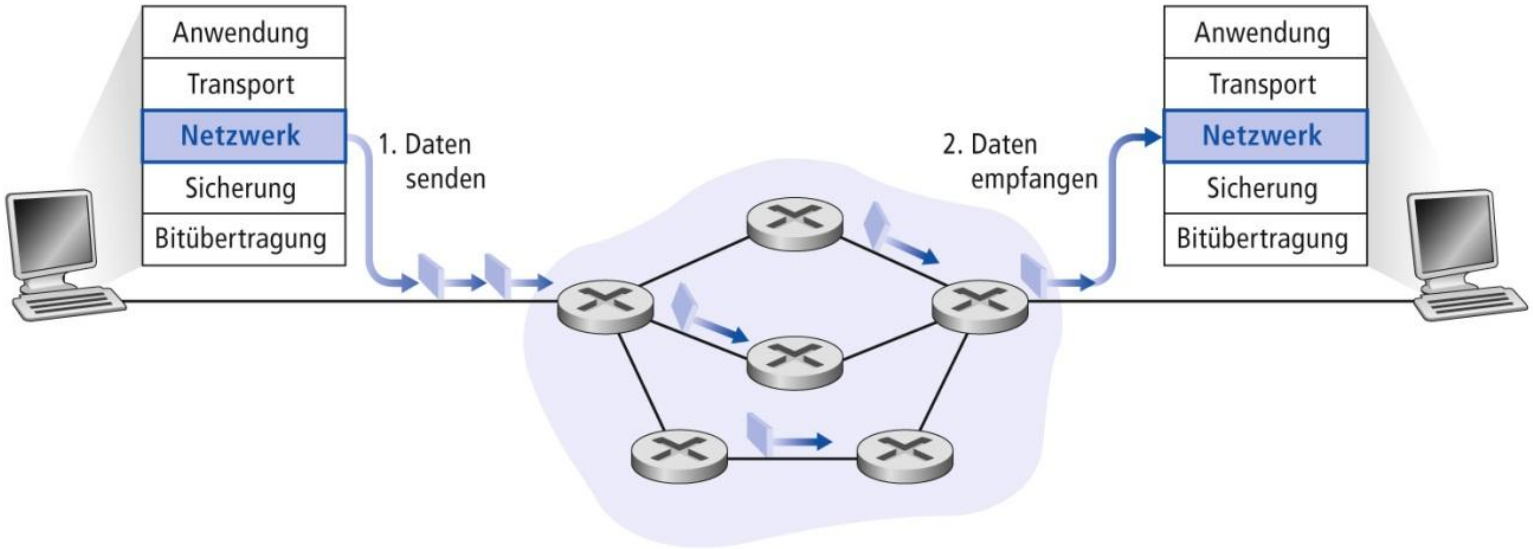
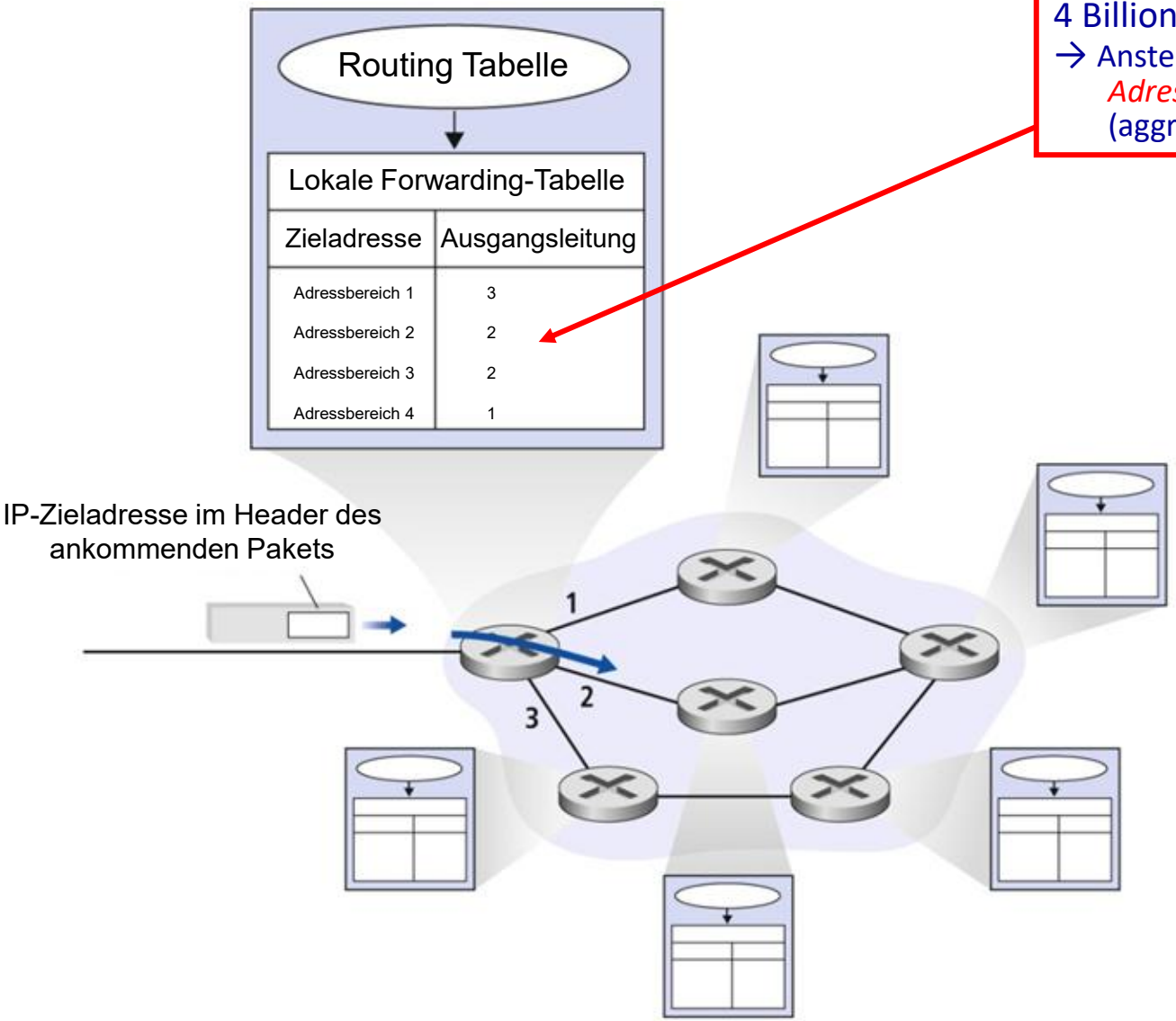


Abbildung 4.5: Datagrammnetzwerk.

Forwarding Tabelle (1)

4 Billionen IP Adressen
→ Anstelle individuelle Zieladressen ganze *Adressbereiche* auflisten (aggregierte Tabelleneinträge)



Forwarding Tabelle (2)

Zieladressbereich	Link Interface
11001000 00010111 00010000 00000000 bis 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 bis 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 bis 11001000 00010111 00011111 11111111	2
sonst	3

F: *Was passiert wenn die Adressbereiche sich nicht so leicht aufteilen lassen?*

Longest Prefix Matching

Longest Prefix Matching

Nutze beim Suchen nach einem Eintrag in der Weiterleitungstabelle den längsten Adresspräfix der der Zieladresse entspricht.

Zieladressbereich	Link Interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
sonst	3

Beispiele:

DA: 11001000 00010111 00010110 10100001 Welches Interface?

DA: 11001000 00010111 00011000 10101010 Welches Interface?

Paketvermittelt oder VC-Basis?

Internet (Paket)

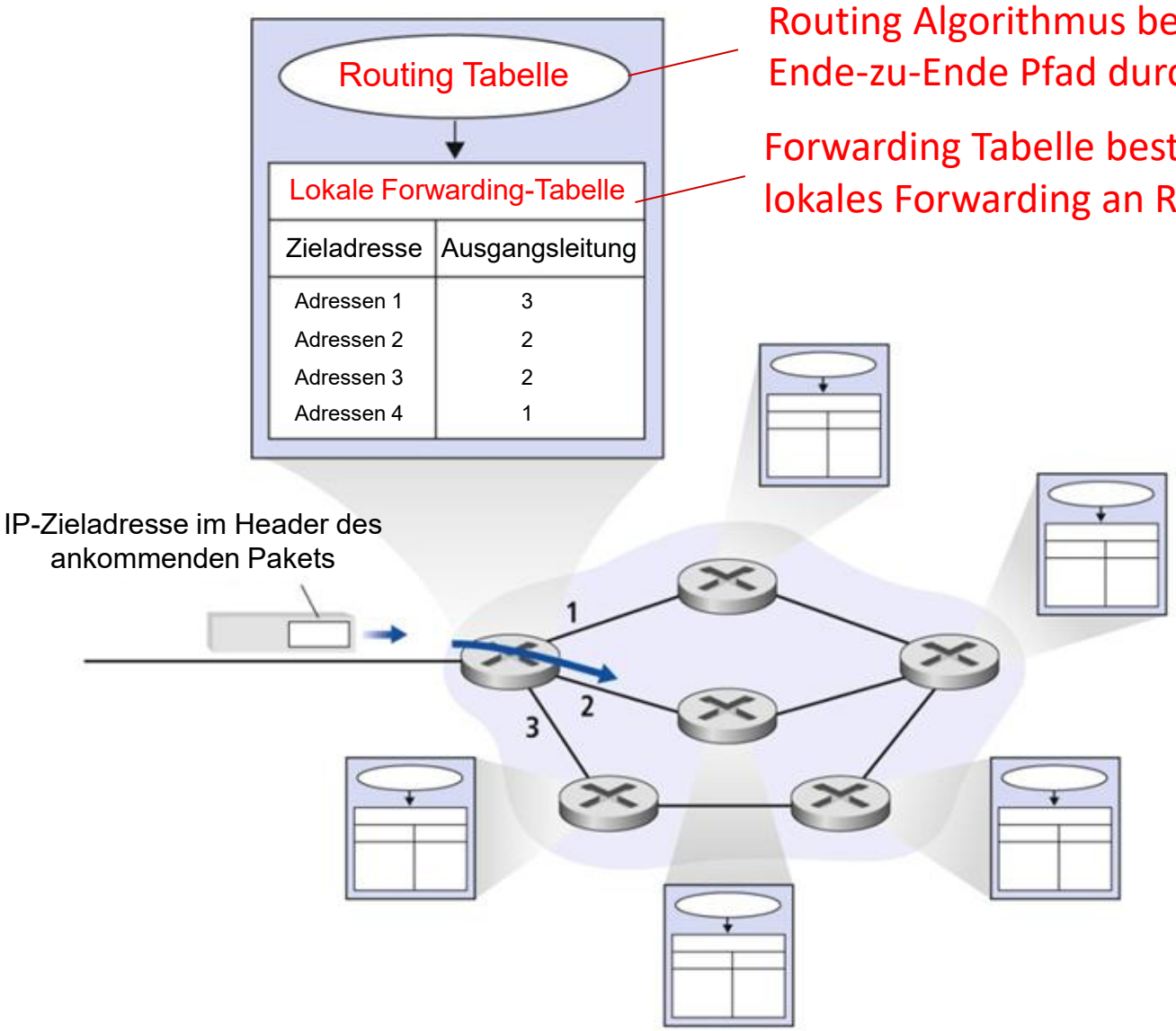
- Datenaustausch zwischen Computern
  - "elastischer" Dienst, keine strengen Forderungen bzgl. Timing ...
- Viele verschiedende Leitungstypen
  - Verschiedene Charakteristiken
  - Einheitlicher Dienst schwierig
- Leistungsfähige Endsysteme (Computer)
  - Können sich anpassen und Fehlerbehandlung durchführen
  - Netz einfach, Komplexe Endsysteme

ATM (VC)

- Stammt aus Telefonwelt
- Menschliches Gespräch:
  - Exaktes Timing
  - Anforderungen an Zuverlässigkeit
  - Benötigt garantierten Dienst
- "Einfache" Endsysteme
  - Telefone
  - Netz ist komplex

Routing und Weiterleitung

Routing und Forwarding



Netz als Graph

Graph:

Menge der Router:

Menge der Links mit Kosten:

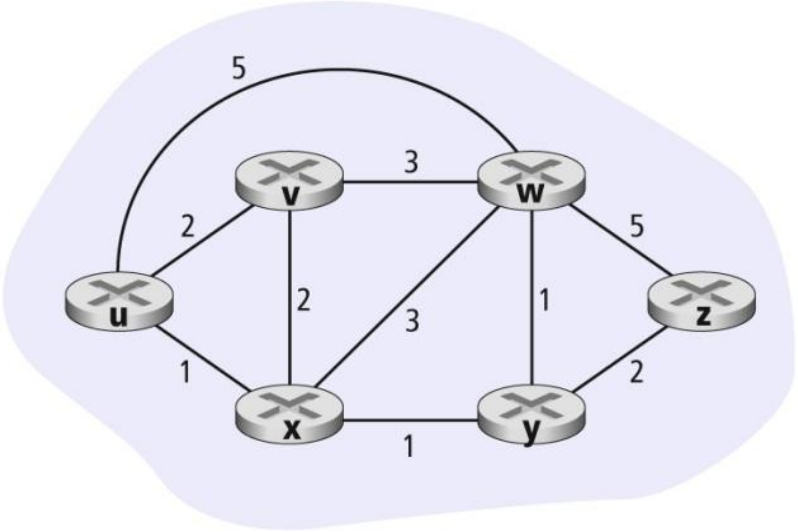


Abbildung 4.27: Abstraktes Graphenmodell eines Computernetzwerks.

Anmerkung:

Graphabstraktion ist auch in anderen Netzwerk Kontexten nützlich

z.B. P2P: Menge der Peers und Menge der TCP Verbindungen

Routing

Routing Protokoll

Ziel: Einen “guten” Pfad (Folge von Routern) von der Quelle zum Ziel finden

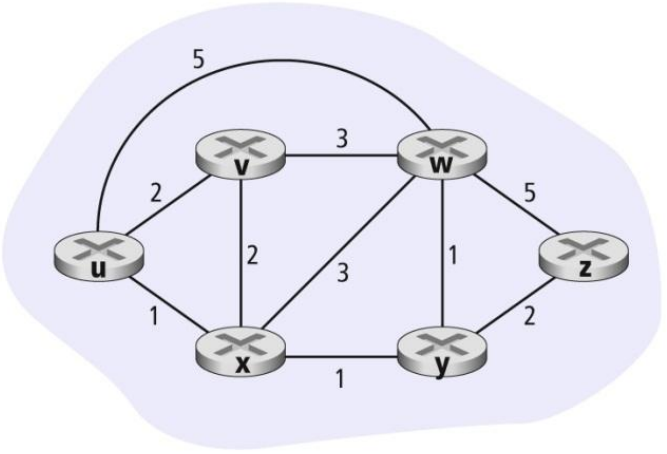


Abbildung 4.27: Abstraktes Graphenmodell eines Computernetzwerks.

Graph dient als Abstraktion des Netzes für Routing Algorithmen:

- Knoten = Router
- Kanten = Leitungen (physische Netzverbindungen)
- Kantenmarkierung = Kosten (z.B.: Verzögerung, Tarif, Auslastung)

- “guter” Pfad:
  - Typischerweise der Pfad mit den geringsten Kosten
  - Andere Definitionen denkbar

Routing Algorithmen: Klassifizierung

Globales oder lokales Wissen?

Global:

- Router kennen die gesamte Netztopologie (Graph mit allen Kanten, Knoten, Markierung)
- “link state” Algorithmen

Lokal:

- Router kennt seine physischen Nachbarn und Übertragungskosten
- Router tauschen Informationen über ihre Nachbarn aus
- ”Distance Vector” Algorithmen

Statisch oder dynamisch?

Statisch:

- Routen ändern sich nur langsam über die Zeit

Dynamisch:

- Routen ändern sich schneller
  - Regelmäßiger Abgleich
  - bei Änderung der Kosten einer Leitung

Link State Verfahren

Ein Link State Routing Algorithmus

Dijkstra Algorithmus

- Netztopologie und Leitungskosten sind allen Knoten bekannt
  - Wird durch “link state broadcast” erreicht
  - Alle Knoten habe gleiche Informationen
- Bestimmung der kostengünstigsten Pfade von einem Knoten A (“Quelle”) zu allen anderen Knoten
  - Ergibt Routing-Tabelle für A
- Iterativ: Nach k Iterationen sind die günstigsten Pfade bekannt

Notation:

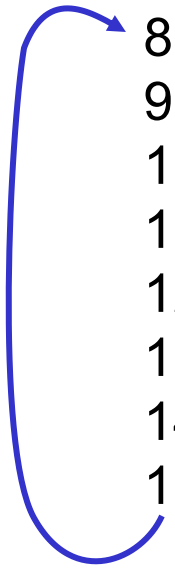
- Leitungskosten von Knoten i nach j. Unendlich, wenn keine direkte Verbindung

Für jede Quelle:

- Aktuelle Kosten von der Quelle zum Ziel v
- Nächster Knoten auf dem Weg von der Quelle zu v
- Menge von Knoten, für die der kostengünstigste Pfad bekannt ist

Dijkstra Algorithmus (Revisited)

```
1  Initialization:
2
3    for all nodes
4      if  adjacent to
5        then
6        else
7
8   Loop
9     find w not in N such that  is a minimum
10    add w to N
11    update for all v adjacent to w and not in N:
12
13    /* new cost to v is either old cost to v or known
14     shortest path cost to w plus cost from w to v */
15  until all nodes in N
```



Dijkstra Algorithmus: Beispiel

Step	start N	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u		
1	ux	2,u	4,x		2,x	
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					

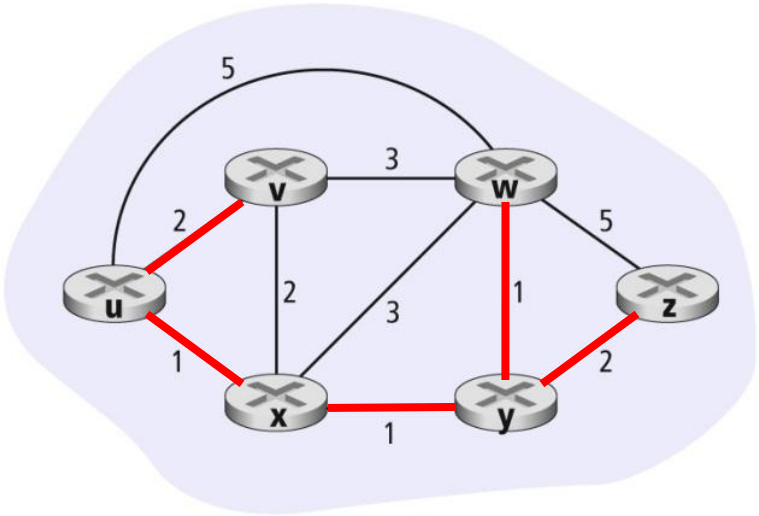


Abbildung 4.27: Abstraktes Graphenmodell eines Computernetzwerks.

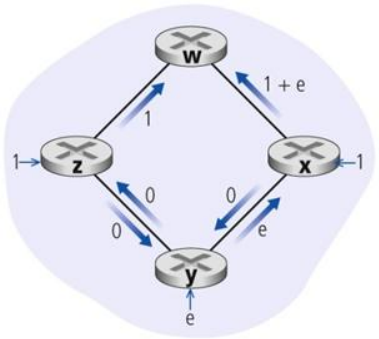
Dijkstra Algorithmus: Diskussion

Komplexität: n Iterationen

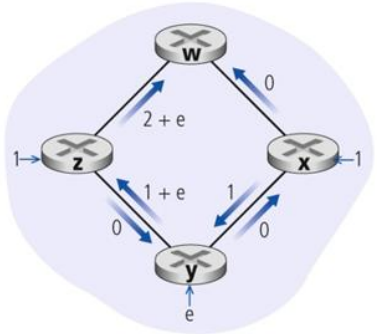
- Pro Iteration: Alle Knoten w, die nicht in N sind prüfen
- #Vergleiche: ??
- Effizientere Implementierung möglich: $O(nlogn)$

Oszillationen möglich:

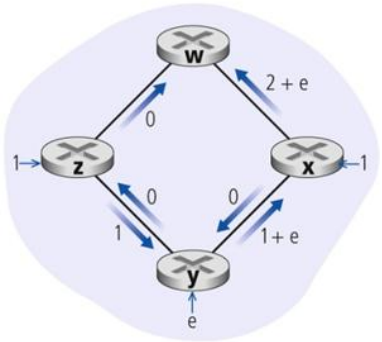
- Bsp.: Leitungskosten = Anzahl der beförderten Pakete



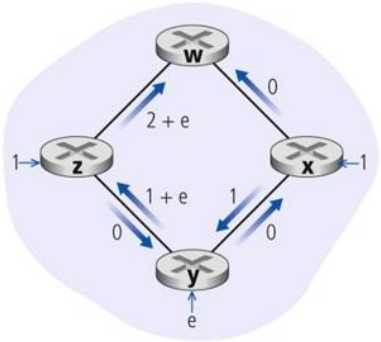
Start



… Erneut Berechnen



… Erneut Berechnen



… Erneut Berechnen

Distanzvektor Verfahren

Distance Vector Routing (revisited)

Iterativ:

- Läuft solange Knoten Informationen austauschen.
- *selbst-terminierend*: Kein Stop-Signal von außen

Asynchron:

- Knoten müssen sich nicht synchronisieren

Verteilt:

- Knoten kommunizieren mit ihren direkten Nachbarn, keine zentrale Instanz

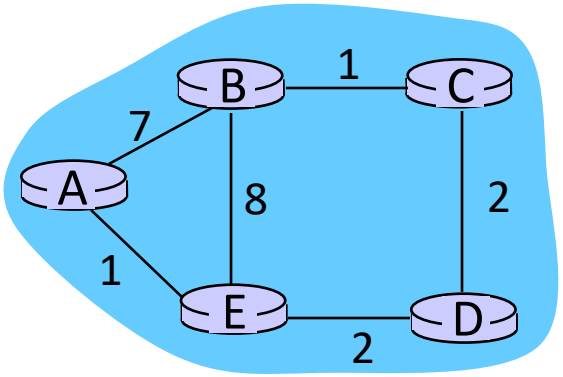
Datenstruktur *Distanz-Tabelle*

- Pro Knoten
- Eine Zeile pro mögliches Ziel
- Spalte für jeden direkten Nachbarn
- Bsp.: Auf Knoten X nach Ziel Y über Nachbar Z:

$$D^{X}(Y,Z) = \text{Entfernung von X zu Y, mit Z als Next-Hop}$$

$$= c(X,Z) + \min_{w}\{D^{Z}(Y,w)\}$$

Beispiel: Distanz-Tabelle



$$D^E(C,D) = c(E,D) + \min_w\{D^D(C,w)\}$$
$$= 2+2 = 4$$

$$D^E(A,D) = c(E,D) + \min_w\{D^D(A,w)\}$$
$$= 2+3 = 5$$

Schleife!

$$D^E(A,B) = c(E,B) + \min_w\{D^B(A,w)\}$$
$$= 8+6 = 14$$

Schleife!

Kosten zum Ziel über

Ziel	$D^E()$ A	B	D
A	1	14	5
B	7	8	5
C	6	9	4
D	4	11	2

Distance Vector Routing: Überblick

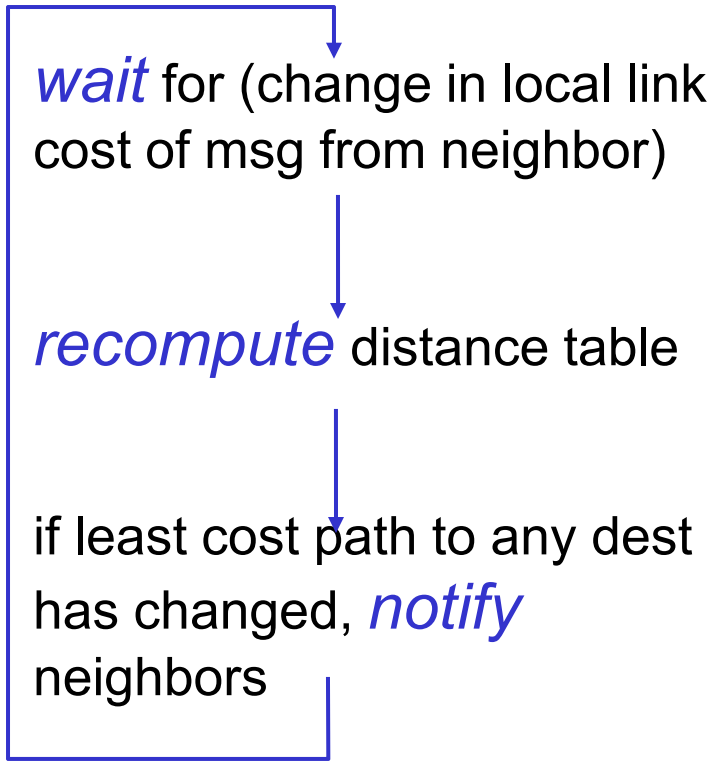
Iterativ, asynchron: Lokale Iteration hervorgerufen durch:

- Änderung lokaler Leitungskosten
- Nachricht von Nachbarknoten: Änderung bei günstigsten Pfaden

Verteilt:

- Ein Knoten benachrichtigt seine Nachbarn nur wenn sein kostengünstigster Pfad zu einem Ziel ändern
  - Nachbarn verbreiten Information wiederum, wenn sich dadurch ihr kostengünstigster Pfad ändert

Pro Knoten:

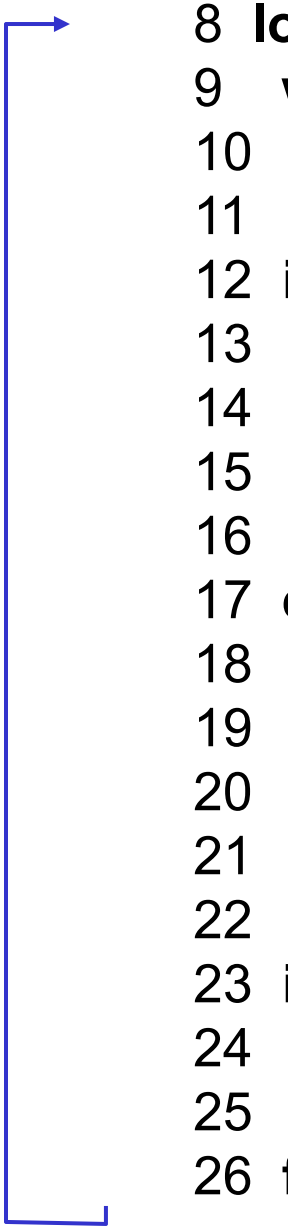


Distance Vector Algorithmus (1):

Auf jedem Knoten X:

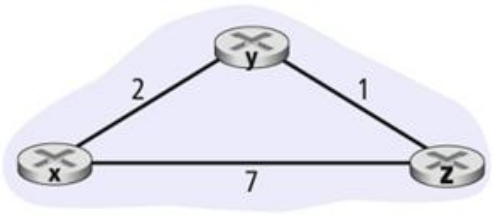
```
1  Initialization:
2    for all adjacent nodes v:
3      /* the * operator means "for all rows" */
4
5    for all destinations, y
6       send  to each neighbor  /* w over all X's neighbors */
```

Distance Vector Algorithmus (2):



```
8  loop
9    wait (until I see a link cost change to neighbor
10         or until I receive update from neighbor V)
11
12  if ( changes by d)
13     /* change cost to all dest's via neighbor v by d */
14     /* note: d could be positive or negative */
15     for all destinations y:
16
17  else if (update received from V wrt. destination Y)
18     /* shortest path from V to some Y has changed  */
19     /* V has sent a new value for its  */
20     /* call this received new value is NewVal     */
21     for the single destination y:
22
23  if we have a new  for any destination Y
24      send new value of  to all neighbors
25
26  forever
```

Distance Vector: Beispiel (1)



D^X

dest	via Y	via Z
Y	2	∞
Z	∞	7

D^X

dest	via Y	via Z
Y	2	8
Z	3	7

D^Y

dest	via X	via Z
X	2	∞
Z	∞	1

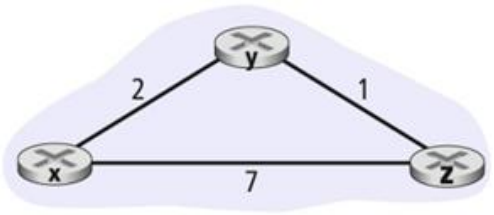
D^Z

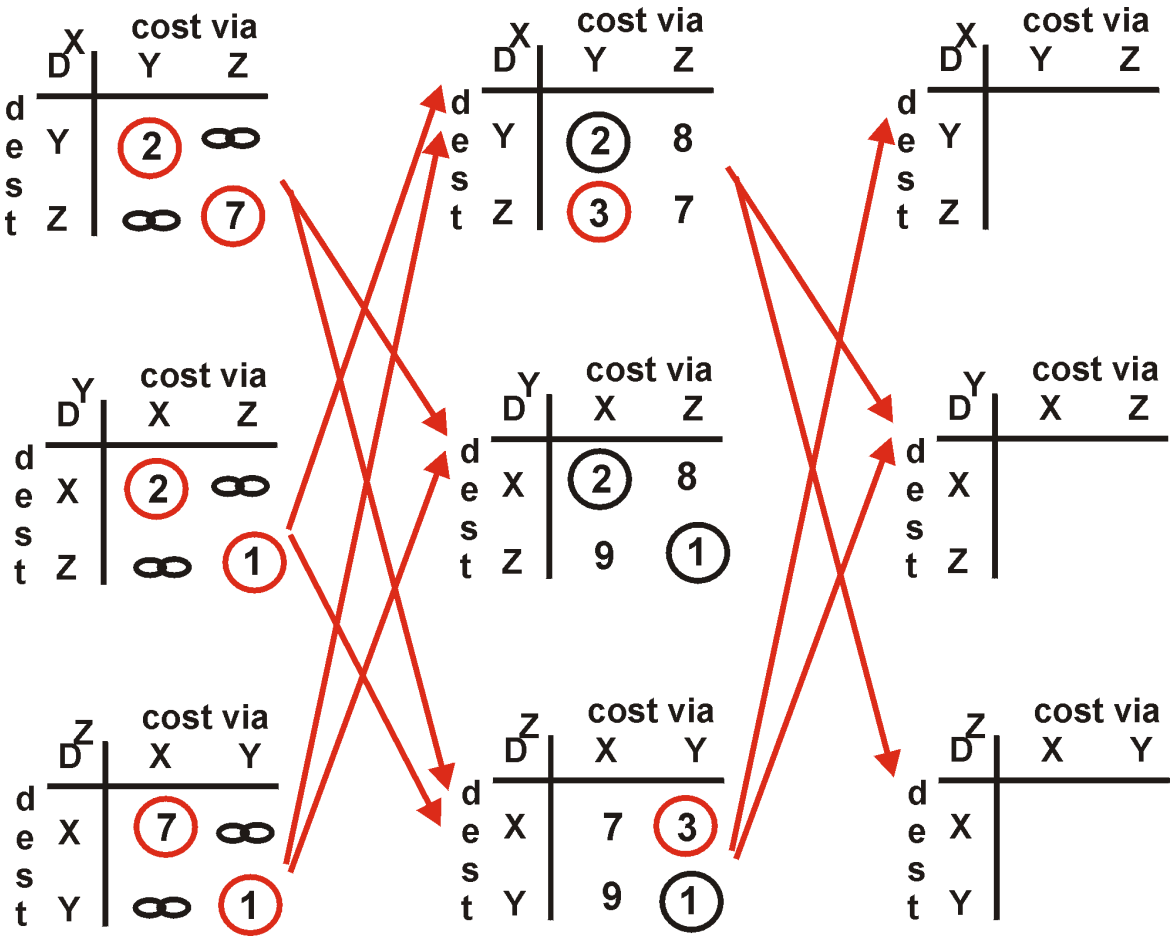
dest	via X	via Y
X	7	∞
Y	∞	1

$$D^X(Y,Z) = c(X,Z) + \min_w\{D^Z(Y,w)\}$$
$$= 7+1 = 8$$

$$D^X(Z,Y) = c(X,Y) + \min_w\{D^Y(Z,w)\}$$
$$= 2+1 = 3$$

Distance Vector: Beispiel (2)

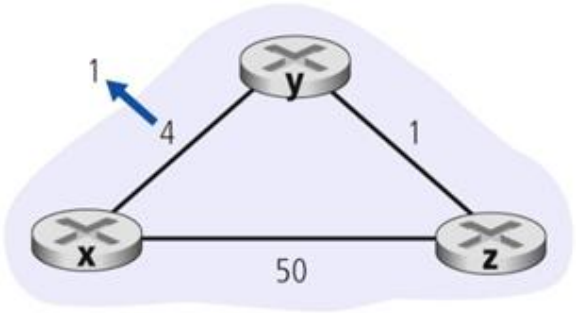




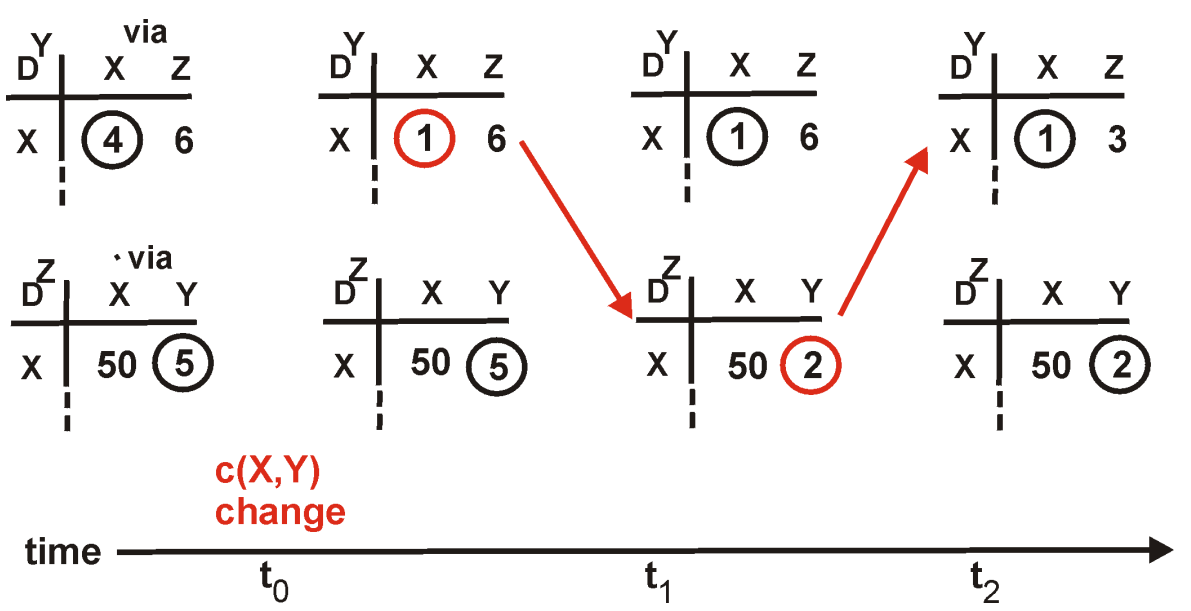
Distance Vector: Änderung von Leitungskosten (1)

Änderung von Linkkosten:

- Knoten erkennt Änderung der lokalen Leitungskosten
- Update Distanz-Tabelle (Zeile 15)
- Änderung der kostengünstigsten Pfade –> Nachbarn informieren (Zeile 23,24)



"Gute Nachrichten verbreiten sich schnell"

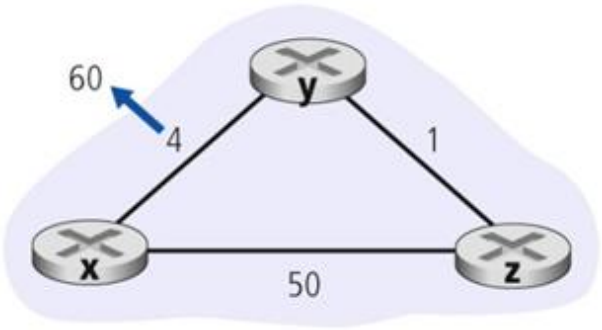


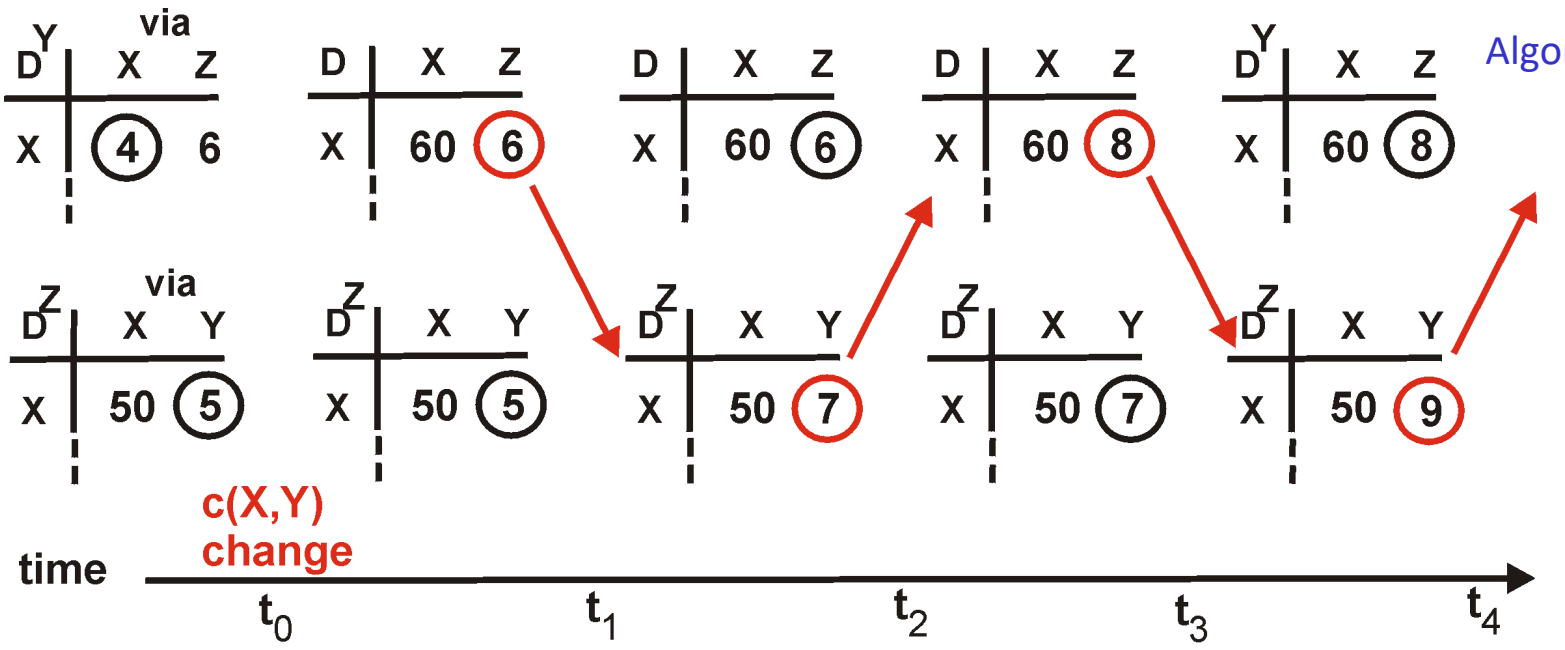
Algorithmus terminiert

Distance Vector: Änderung von Leitungskosten (2)

Änderung der Linkkosten:

- Schnelle Verbreitung guter Nachrichten
- Langsame Verbreitung schlechter Nachrichten - "count to infinity" Problem!

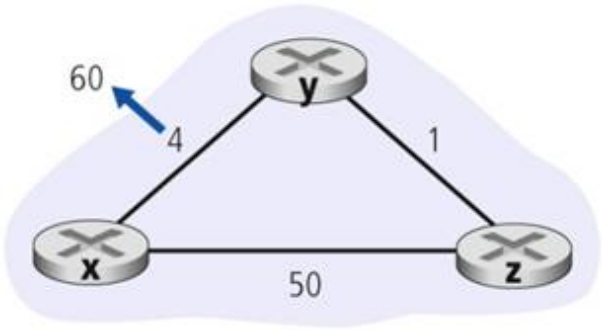


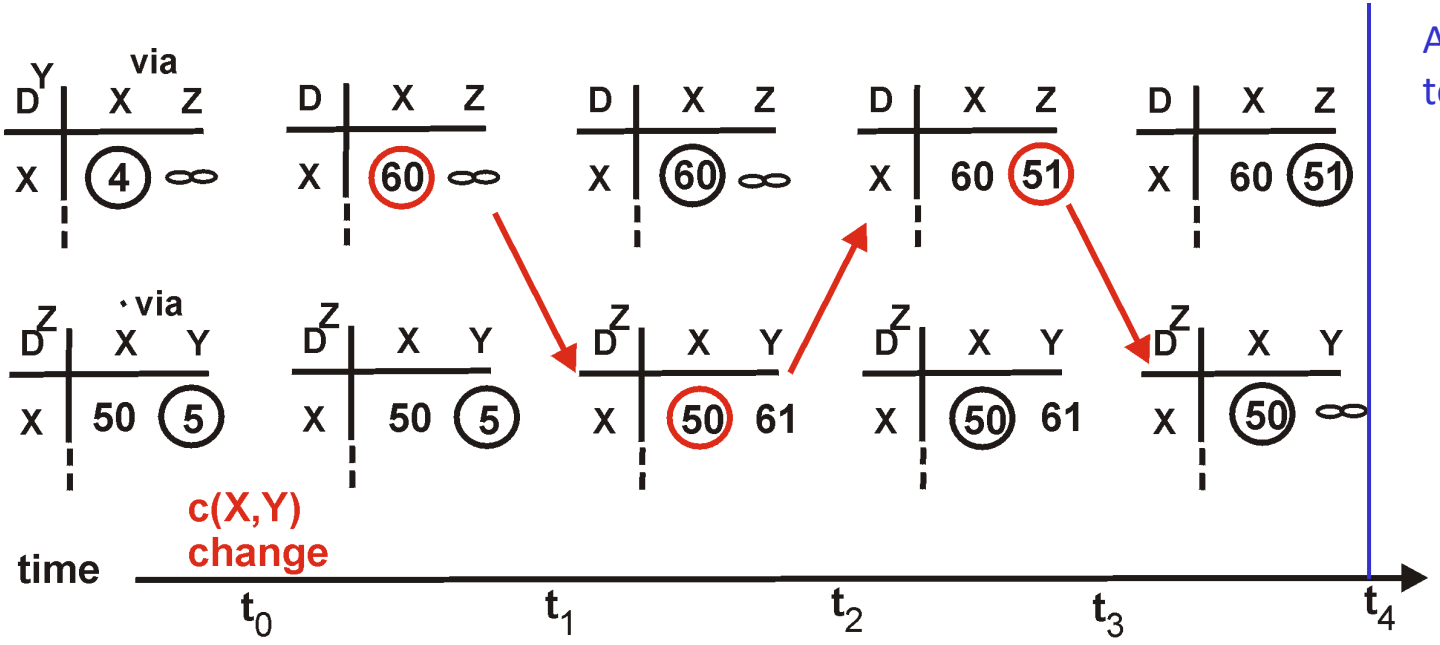


Distance Vector: Poisoned Reverse Verfahren

Wenn Z über Y zu X routet:

- Z teilt Y mit, dass er X nicht erreichen kann. (Somit routet Y nicht über X zu Z)
- Ist hier das “count to infinity” Problem vollständig gelöst ?





Algorithmus terminiert

Vergleich Link State (LS) vs. Distance Vector (DV)

Nachrichten Komplexität

- LS: N Knoten, E Links, O(nE) Nachrichten
- DV: Austausch nur zwischen benachbarten Knoten
  - Zeit bis Konvergenz variiert

Konvergenz-Geschwindigkeit

- LS: Algorithmus in $O(n^2)$, Anzahl Nachrichten in O(nE)
  - Oszillation möglich
- DV: Konvergenz-Zeit variiert
  - Schleifen möglich
  - Count-to-Infinity Problem

Robustheit: Was bei Router Fehlfunktionen?

LS:

- Knoten verbreitet falsche Leitungskosten
- Jeder Knoten berechnet eigene Tabelle

DV:

- Knoten kann falsche *Pfadkosten* verbreiten
- Knoten verwenden Routing-Tabellen anderer Knoten
  - Schnelle Verbreitung von Fehlern

Weiterführendes Lehrbuch zur Vorlesung

James Kurose, Keith Ross, Computer Networking: A Top-Down Approach, Global Edition, 8. Auflage, Pearson, 2021.

