5 Weitverkehrsnetze und Routing

5.1 Das Prinzip der Paketvermittlung

5.2 Virtuelle Verbindungen vs. Datagramme

5.3 Wegewahl (Routing) für Punkt-zu-Punkt-Netze

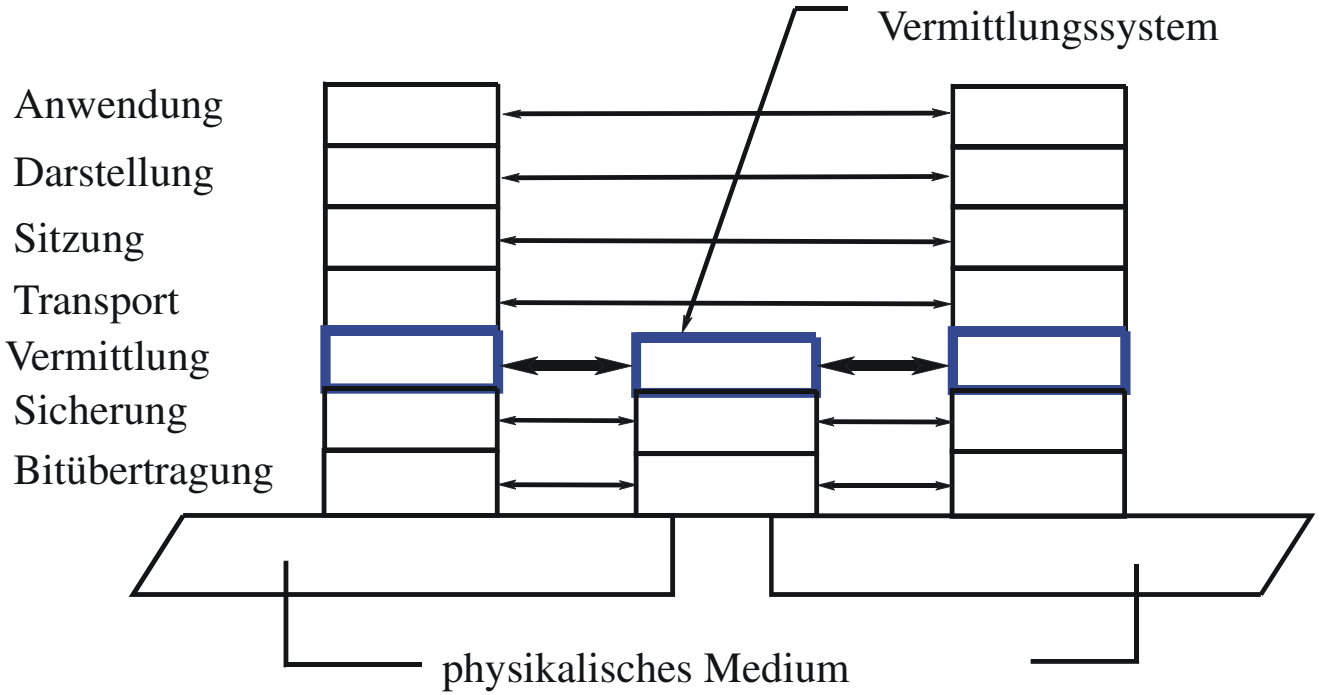
5.4 Wegewahl (Routing) für Multicast-Netze

5.5 Überlastkontrolle in der Vermittlungsschicht

5.6 Beispiele: IP, IPv6, ATM

5.1 Das Prinzip der Paketvermittlung

Die Vermittlungsschicht im OSI-Referenzmodell



ISO-Definition für die Vermittlungsschicht

Die Vermittlungsschicht stellt die Fähigkeit bereit, Netzverbindungen zwischen offenen Systemen über Zwischensysteme hinweg aufzubauen, zu betreiben und abzubauen.

Die Vermittlungsschicht bietet den Transportinstanzen Unabhängigkeit von Wegewahl- und Vermittlungsentscheidungen, die mit dem Aufbau und Betrieb einer Netzverbindung verbunden sind.

Aufgaben der Vermittlungsschicht

- Wegewahl und Vermittlung von Paketen
- Multiplexen von Ende-zu-Ende-Verbindungen über Schicht-2-Verbindungen
- Segmentierung („Fragmentierung“)
- Zusätzlich in verbindungsorientierten Vermittlungsschichten:
  - Verbindungsaufbau und -abbau
  - Fehlererkennung und Fehlerbehebung (Ende-zu-Ende)
  - Sicherstellung der Paketreihenfolge
  - Flusskontrolle (Ende-zu-Ende)

Dabei ist es wichtig, dass heterogene Teilnetze verbunden werden können („Internetworking“).

5.2 Virtuelle Verbindung vs. Datagramm

Virtuelle Verbindung

Der Weg durch das Netz wird beim Aufbau der virtuellen Verbindung ausgewählt, d. h., für jede neue virtuelle Verbindung findet in jedem Netzknoten nur einmal eine Wegewahlentscheidung statt. Der gesamte über diese virtuelle Verbindung fließende Verkehr nimmt denselben Weg durch das Netz.

Datagramm

Jedes Paket enthält die volle Adresse des Ziel-Hosts. Die Zieladresse bestimmt in jedem Netzknoten auf dem Pfad stets neu die ausgehende Leitung.

Die Virtuelle Verbindung

„Perfekter" Kanal durch das Netz

- Ordnung der Nachrichten (Sicherstellung der Reihenfolge)
- Fehlerüberwachung (verlorene und duplizierte Pakete)
- Flusskontrolle

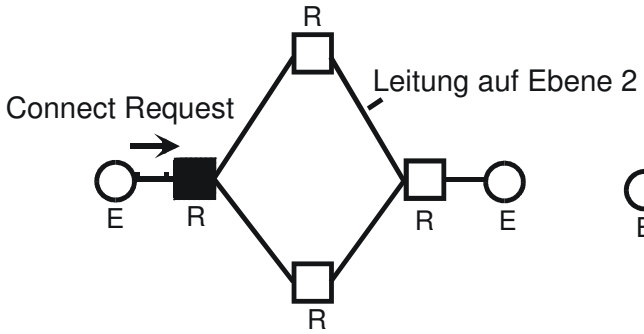
Phasen

- Verbindungsaufbau
- Datenübertragung
- Verbindungsabbau

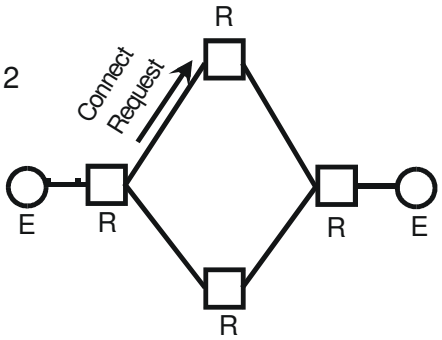
Vorteile

- Niedriger Mehraufwand für die Adressierung während der Datenübertragung
- Hohe Qualität des ankommenden Paketstroms: keine Neusortierung oder Fehlerüberwachung in den höheren Schichten nötig

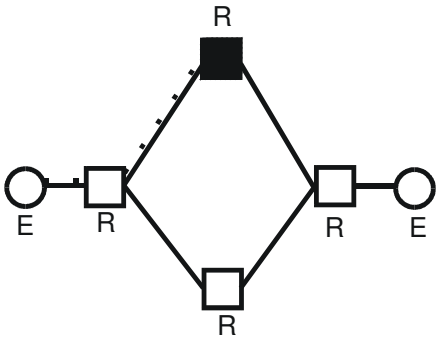
Aufbau einer virtuellen Verbindung



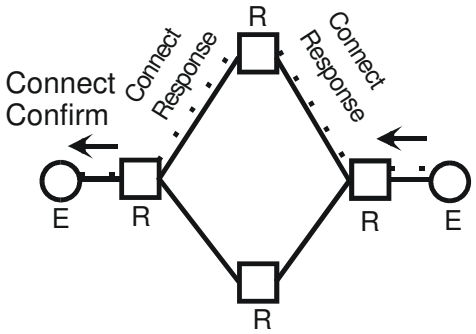
a) Festlegen des Weges



b) Aufbauphase der 1. Teilstrecke



c) Virtueller Verbindungsabschnitt existiert, Festlegung der Wegefortsetzung

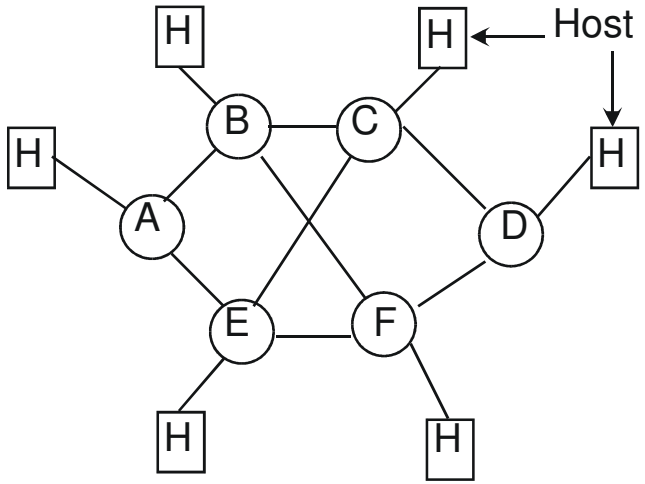


d) nach weiteren Schritten virtuelle Verbindung fertigestellt

Implementierung von virtuellen Verbindungen

In jedem Netzknoten werden Tabellen mit Zustandsinformationen über bestehende virtuelle Verbindungen verwaltet.

(a) Beispiel-Subnetz:

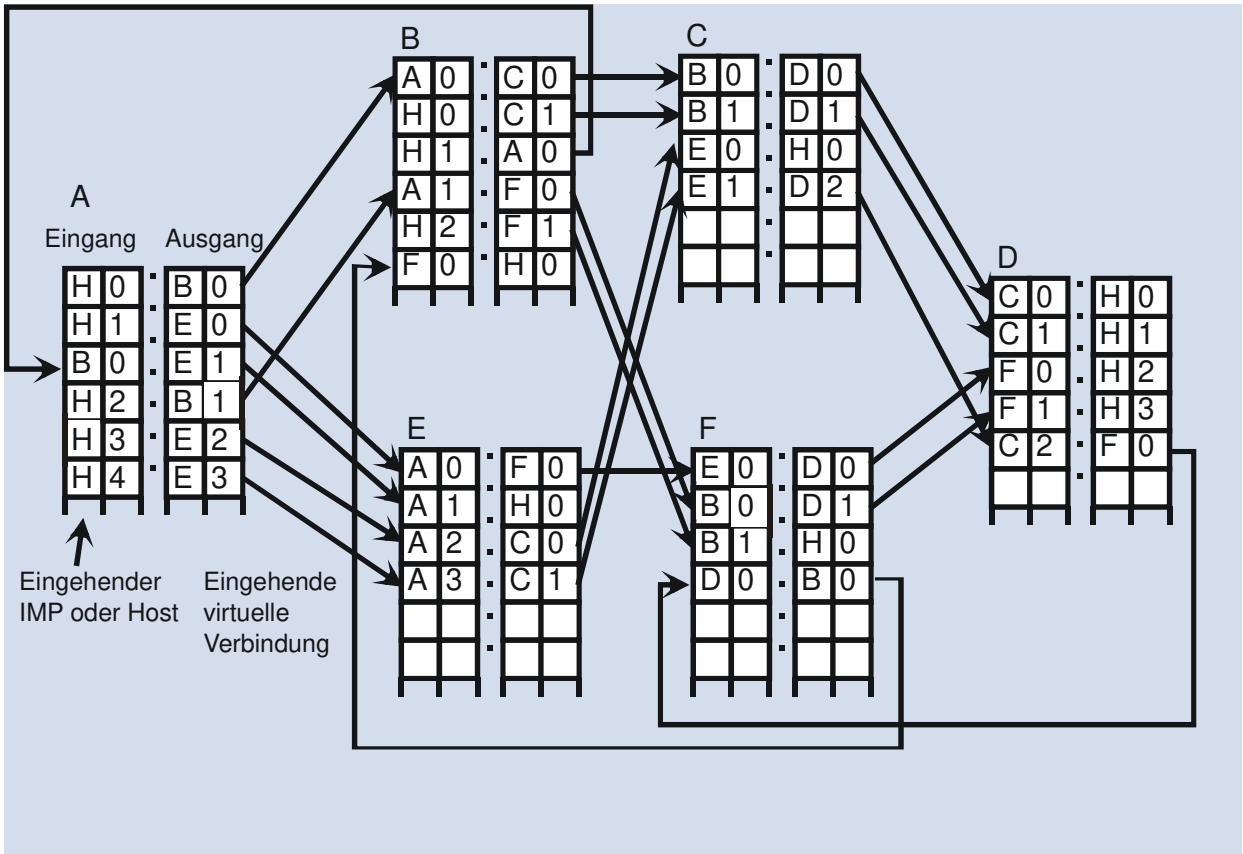


(b) Acht virtuelle Verbindungen durch dieses Subnetz:

Ausgehend von A	Ausgehend von B
0 – ABCD 1 – AEFD 2 – ABFD 3 – AEC 4 – AECDFB	0 – BCD 1 – BAE 2 – BF

Zustandsinformation in den Netzknoten

(c) Router-Tabellen für die virtuellen Verbindungen in (b)



Das Datagramm

Jedes Paket (Datagramm) wird als isolierte Einheit betrachtet (wie ein Telegramm im Postverkehr):

- Volle Zieladresse in jedem Paket
- Pakete können außerhalb der Reihenfolge eintreffen
- Keine Fehlerüberwachung, keine Flusskontrolle in Schicht 3

Vorteile

- Primitiver als virtuelle Verbindungen, daher viel einfacher zu implementieren
- Kein Verbindungsaufbau und -abbau, deshalb geringer Overhead für kurzlebige Verbindungen
- flexibler und zuverlässiger
- besser geeignet für Internetworking heterogener Subnetze

5.3 Wegewahl für Punkt-zu-Punkt-Netze

Vorbemerkung: Besondere Netztopologien

Wegfall des Wegewahlproblems auf Broadcast-Medien, zum Beispiel in einem Segment eines LANs (Bus- oder Ring-Topologie): hier ist keine Wegewahl erforderlich, da jede Nachricht wegen der Topologie des physikalischen Mediums alle Empfänger erreicht.

Routing-Algorithmen

Aufgabe: Leitwegbestimmung für Pakete durch das Netzwerk vom Quellsystem zum Zielsystem

Der Leitwegbestimmungsalgorithmus eines Vermittlungsrechners (Routers, Knotens) entscheidet, auf welcher Ausgangsleitung ein eingegangenes Paket weiter geleitet wird.

- Bei virtuellen Verbindungen: Leitwegbestimmung nur beim Verbindungsaufbau.
- Bei der Datagrammtechnik: individuelle Entscheidung für jedes Paket

Wünschenswerte Eigenschaften eines Routing-Algorithmus

- Korrekt
- Einfach
- Robust bei Rechner- oder Leitungsausfällen
- Fair
- Optimal

Algorithmen für die Leitwegbestimmung

Die genannten Kriterien stehen im Zielkonflikt. In der Praxis hat sich als Ziel bewährt: **Minimierung der Teilstrecken** (hops) vom Sender zum Empfänger.

Leitwegbestimmung

Klassifikation der Verfahren

1. Statische (nicht-adaptive) Verfahren

- keine Berücksichtigung des aktuellen Netzzustands
- gehen von Mittelwerten aus
- Leitweg zwischen i und j wird für alle i, j vor der Inbetriebnahme des Netzwerks bestimmt
- keine Änderung während des Betriebs

2. Adaptive Verfahren

- Entscheidungen basieren auf dem aktuellen Netzzustand
- laufend Messungen/Schätzungen der Topologie und des Verkehrsaufkommens
- Weitere Unterteilung der adaptiven Verfahren in
  - zentralisierte Verfahren
  - isolierte Verfahren
  - verteilte Verfahren

Statische Leitwegbestimmung

Beim **statischen Routing** ist die gesamte Topologie des Netzes einer zentralen Stelle bekannt. Sie berechnet die optimalen Pfade für jedes Paar (i,j) von Knoten, erstellt daraus die Routing-Tabellen für die einzelnen Knoten und versendet diese.

Die statische Leitwegbestimmung ist sinnvoll, wenn das Netz relativ klein und relativ statisch ist.

Mehrfach-Leitwegbestimmung (multipath routing)

Benutzung alternativer Leitwege zwischen jedem Knotenpaar (i,j)

- Häufigkeit der Nutzung abhängig von der Güte der Alternative
- höherer Durchsatz durch Verteilung des Datenverkehrs auf mehrere Pfade
- höhere Zuverlässigkeit, da der Ausfall eines Links nicht so schnell zur Unerreichbarkeit von Knoten führt

Realisierung (1)

Jeder Knoten enthält eine Routing-Tabelle mit je einer Spalte für jeden möglichen Zielknoten

Z	A1	G1	A2	G2		An	Gn

Z Ziel

A_i i-beste Ausgangsleitung

G_i Gewicht für A_i

(G_i bestimmt die Wahrscheinlichkeit, mit der A_i benutzt wird)

$$\left(\sum_{i=1}^{n} G_i = 1 \right)$$

Realisierung (2)

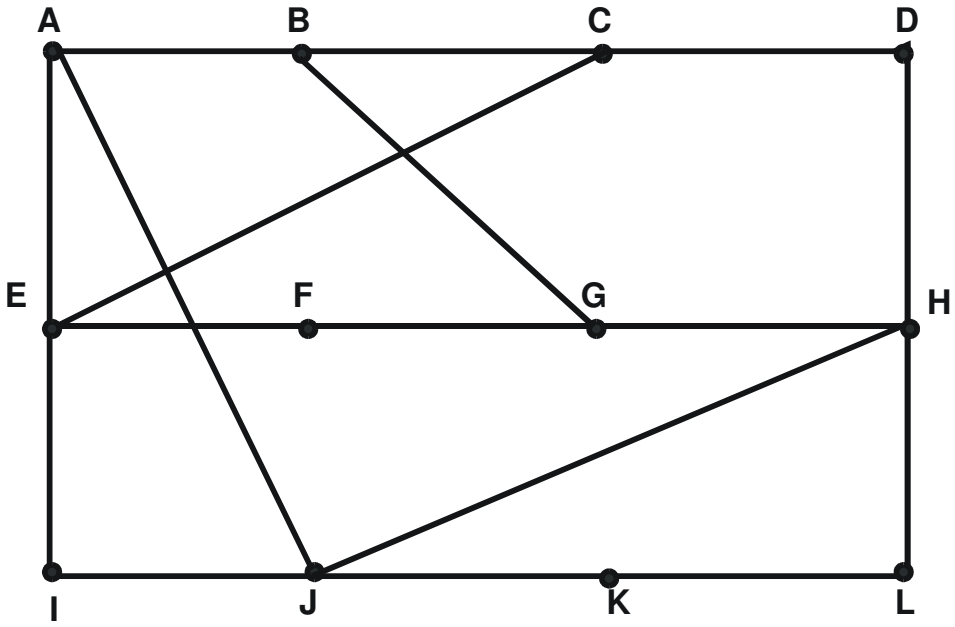
Auswahl der Alternativen:

```
Generiere eine Zufallszahl z (0 <= z <= 1)
Wähle A1, falls 0  <= z <  G1
Wähle A2, falls G1 <= z <  G1 + G2
...
Wähle An, falls G1 + G2 + .. + G(n-1) <= z <= 1
```

Beispiel : Ziel B, Quelle J

Statische Leitwegbestimmung: Beispieltopologie

Topologie des Beispielnetzes



Wir betrachten die Pfade vom Knoten J aus.

Statische Leitwegbestimmung: Beispieltabelle

Statische Routing-Tabelle mit alternativen Pfaden im Knoten j

Ziel	Erste Wahl		Zweite Wahl		Dritte Wahl	
A	A	0.63	I	0.21	H	0.16
B	A	0.46	H	0.31	I	0.23
C	A	0.34	I	0.33	H	0.33
D	H	0.50	A	0.25	I	0.25
E	A	0.40	I	0.40	H	0.20
F	A	0.34	H	0.33	I	0.33
G	H	0.46	A	0.31	K	0.23
H	H	0.63	K	0.21	A	0.16
I	I	0.65	A	0.22	H	0.13
-						
K	K	0.67	H	0.22	A	0.11
L	K	0.42	H	0.42	A	0.16

Bestimmung der Leitwegtabellen

Die Routing-Tabellen werden beim statischen Routing vom Netzwerkoperator zentral erstellt. Sie werden vor Inbetriebnahme des Netzes in die Knoten geladen und dann nicht mehr verändert.

Eigenschaften

- einfach
- gute Ergebnisse bei relativ konstanter Topologie und konstantem Verkehr

Aber:

- schlecht bei stark variierendem Verkehrsaufkommen und bei Topologieänderungen
- schlecht bei großen Netzen (skaliert nicht).

In der Praxis noch immer gelegentlich benutzt, zum Beispiel in SNA-Netzen.

Der Netzoperator kennt stets die gesamte Topologie. Er verwendet beispielsweise den Algorithmus „kürzeste Wege“ von Dijkstra einmal für jeden Knoten zur Konstruktion der Routing-Tabellen.

Zentralisierte adaptive Leitwegbestimmung (1)

Prinzip

- Im Netz gibt es ein **Routing Control Center (RCC,** (Leitwegsteuer-zentrum).
- Jeder Knoten sendet periodisch Status-Information zum RCC, beispielsweise
  - die Liste der verfügbaren Nachbarn
  - aktuelle Warteschlangenlängen
  - Auslastung der Leitungen, etc.
- Das RCC sammelt die Informationen und berechnet den optimalen Pfad für jedes Knotenpaar, berechnet die einzelnen Leitwegtabellen und verteilt diese dann an die Knoten.

Zentralisierte adaptive Leitwegbestimmung (2)

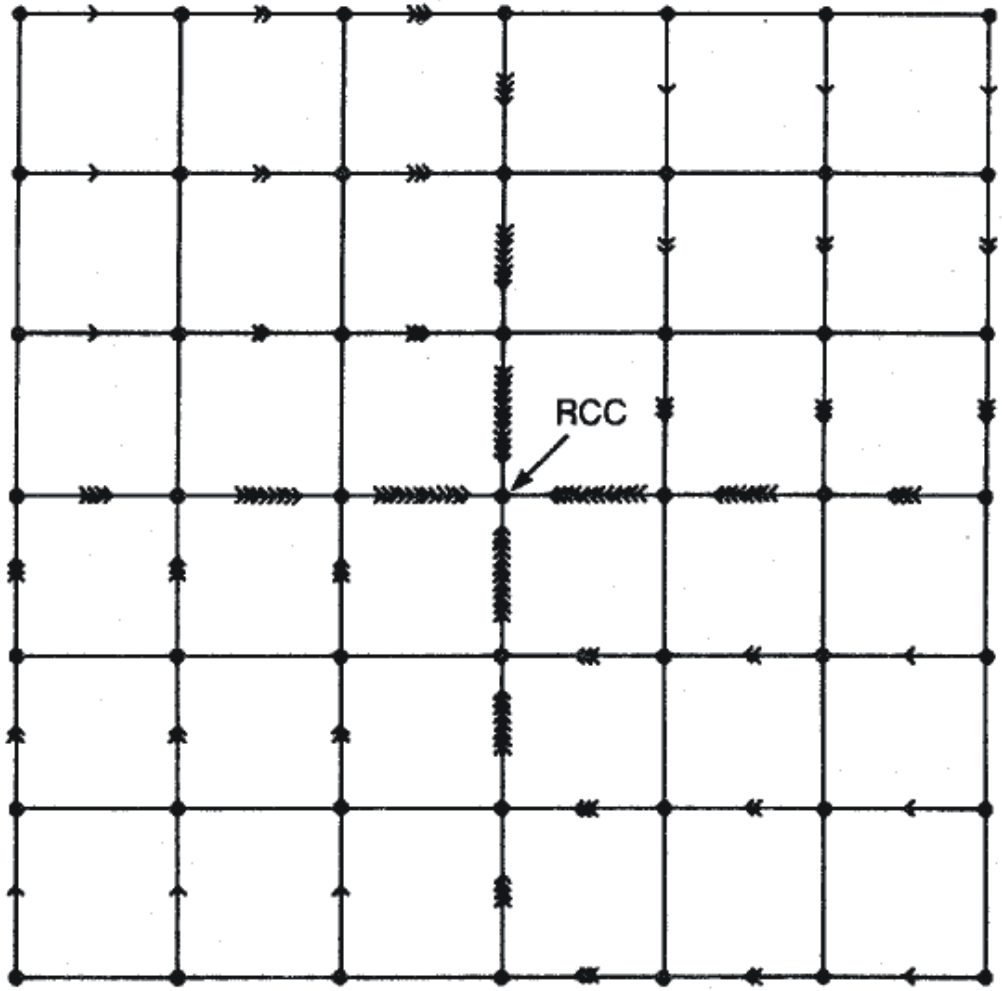
Eigenschaften

- Das RCC hat vollständige Information => die Entscheidungen sind optimal
- Die Einzelknoten sind von der Leitwegberechnung befreit.

Aber:

- Die Berechnung muss oft durchgeführt werden (ca. einmal pro Minute oder öfter)
- Es entsteht eine Verkehrskonzentration in der Nähe des RCC ("performance bottleneck")
- Die Technik ist nicht robust: das RCC ist ein “single point of failure“.
- Das Verfahren funktioniert nicht bei Netzpartitionierung.
- Die einzelnen Knoten erhalten ihre neuen Tabellen jeweils zu unterschiedlichen Zeiten => Inkonsistenzen und damit “routing loops“ sind möglich.

Das Routing Control Center



Isolierte adaptive Leitwegbestimmung

Prinzip

- Kein Austausch von Routing-Information zwischen Knoten
- Entscheidungen basieren ausschließlich auf lokal verfügbaren Informationen

Beispiele für Verfahren

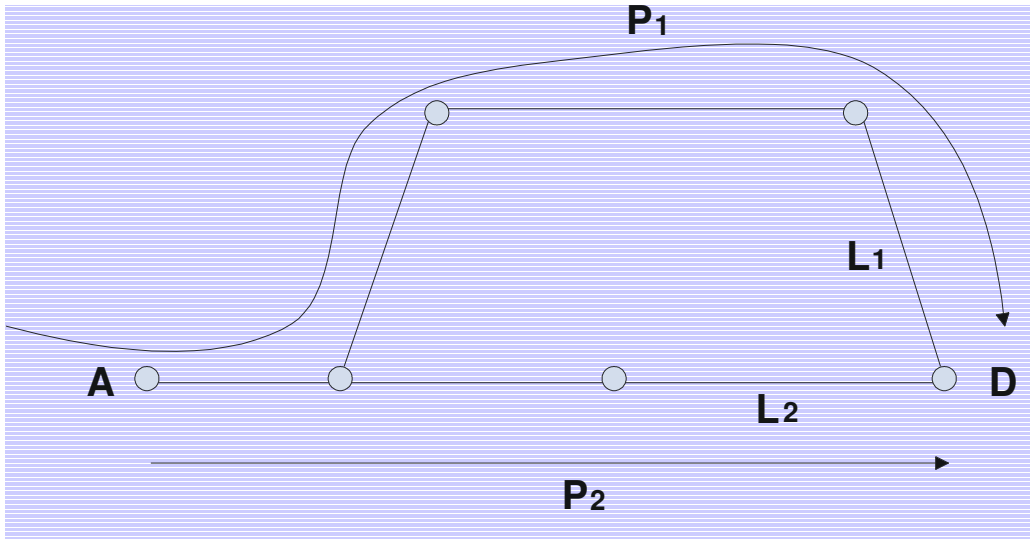
- Backward Learning
- Flooding
- Delta-Routing (Rudin, 1976)

Algorithmus "Backward Learning"

- Knoten "lernt" von eintreffenden Paketen: Paket(..., Q, Z, ...)
  Q = Quell–Knoten
  Z = Teilstreckenzähler (hop counter)
  Paket wird auf Leitung L empfangen => Q ist über L in Z Teilstrecken erreichbar
- Leitwegtabelle im Knoten: Jeder Eintrag ist ein Tripel
  (Zielknoten, Ausgangsleitung, Z_{min})
- Aktualisierung der Leitwegtabelle:
  Knoten empfängt Paket (..., Q, Z, ...) auf Leitung L

```
if not (Q in Tabelle)
    then  add(Q,L,Z)
    else  if Z < Zmin
          then update(Q,L,Z)
```

Backward Learning: Beispiel




```
P1( ..., A,4,...) -> add(A,L1,4)
P2( ..., A,3,...) -> update(A,L2,3)
```

Pfadverschlechterung beim Backward Learning

Problem

Algorithmus registriert keine Verschlechterungen!

Lösung

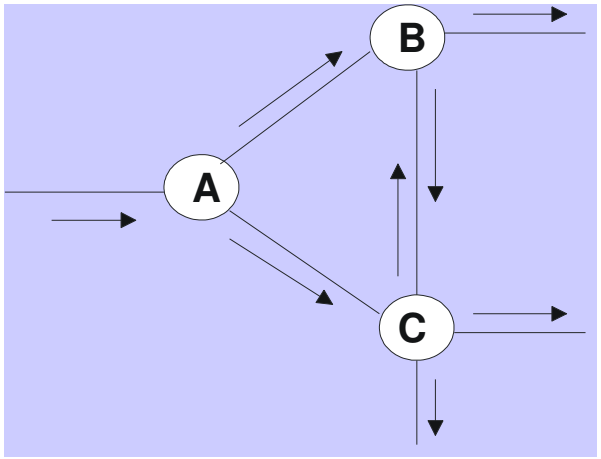
Periodisches Löschen der Leitwegtabellen (neue Lernperiode).

Aber: Löschzeitpunkte kritisch:

- zu häufig: Netz ist überwiegend in der Lernphase
- zu selten: zu langsame Reaktion auf Verschlechterungen

Algorithmus „Flooding“

Ein empfangenes Paket wird auf allen Leitungen weiter geleitet außer auf derjenigen, auf der es angekommen ist.



Flooding: Abklingen des Paketflusses

Problem: Unendliche Anzahl von Duplikaten

Begrenzung des Prozesses: Streckenzähler ("hop counter") im Paketkopf

- Initialisierung mit dem Durchmesser des Netzes = längstem Pfad im Netz (worst case)
- wird auf jeder Teilstrecke um 1 dekrementiert
- Duplikate erhalten den Streckenzähler des Originals
- Zähler = 0: Paket wird vom Router weggeworfen

Eigenschaften von Flooding

- sehr robust, sehr einfach, aber
- große Anzahl von Duplikaten, große Netzbelastung

  => Einsatz nur für sehr spezielle Anwendungen

Algorithmus "Delta-Routing" (1)

Prinzip

Kombination von isoliertem und zentralisiertem Verfahren.

- Jeder Knoten berechnet periodisch die "Kosten" seiner Leitungen und sendet diese zum RCC (Kosten = Funktion von Verzögerung, Warteschlangenlänge, ...).
- RCC berechnet
  - die k besten Pfade von Knoten i nach Knoten j (für alle i, j)
  - eine Liste der zum besten Pfad "äquivalenten" Pfade mit

$$c_{ij}^{n} - c_{ij}^{1} < \delta \qquad \text{mit} \quad c_{ij}^{m} \quad = \text{Gesamtlänge des m-besten Pfades}$$

- RCC sendet jedem Knoten für jedes mögliche Ziel eine Liste von äquivalenten Pfaden.
- Jeder Knoten darf zwischen den äquivalenten Pfaden frei wählen.

Algorithmus "Delta Routing" (2)

Die Wahl von δ entspricht dem Verschieben der „Macht“ zwischen Knoten und RCC:

$\delta \rightarrow 0$: RCC trifft die Entscheidung allein

$\delta \rightarrow \infty$: der Knoten trifft die Entscheidung allein

Bei geeigneter Wahl von δ kann eine bessere Leistung als bei rein isolierten oder rein zentralisierten Verfahren erreicht werden.

Verteilte Leitwegbestimmung

Prinzip

Die Knoten tauschen mit ihren Nachbarn Leitweginformationen aus:

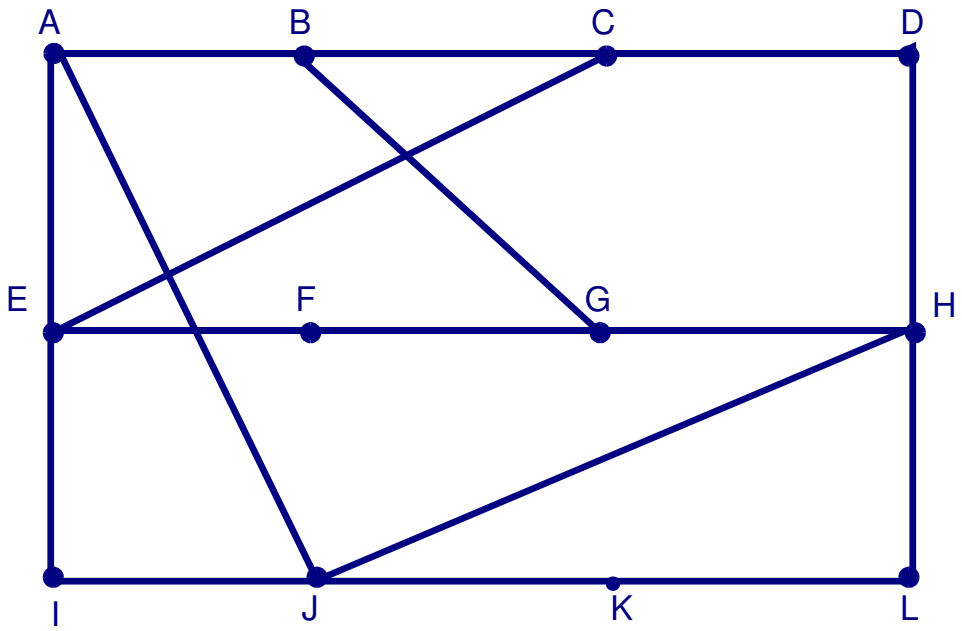
- Jeder Knoten kennt die "Entfernung" zu jedem Nachbarn
  - Anzahl der Teilstrecken (= 1)
  - Verzögerungszeit (Echo-Pakete)
  - Warteschlangenlänge, etc.
- Jeder Knoten sendet periodisch seinen Nachbarn eine Liste mit seinen geschätzten Entfernungen zu jedem Ziel.
- X empfängt eine Liste E vom Nachbarn Y
- Entfernung (X, Y) = e
- Entfernung (Y, Z) = E(Z)

  => Entfernung(X, Z) über Y : E(Z) + e

Die Tabelle mit den einem Knoten bekannten Distanzen heißt **Distanzvektor**. Das Verfahren heißt deshalb auch "**distance vector routing**".

Verteilte Leitwegbestimmung (1)

Beispiel



Wir betrachten die Distanzen vom Knoten J aus.

Verteilte Leitwegbestimmung (2)

	A	I	H	K		
A	0	24	20	21	8	A
B	12	36	31	28	20	A
C	25	18	19	36	28	I
D	40	27	8	24	20	H
E	14	7	30	22	17	I
F	23	20	19	40	30	I
G	18	31	6	31	18	H
H	17	20	0	19	12	H
I	21	0	14	22	10	I
J	9	11	7	10	0	-
K	24	22	22	0	6	K
L	29	33	9	9	15	K
	JA Ver-zöge-rung=8	JI Ver-zöge-rung=10	JH Ver-zöge-rung=12	JK Ver-zöge-rung=6		

Rechte Spalte: nach dem Eintreffen der Distanzvektoren neu ermittelte Distanzen von J aus

Hierarchische Leitwegbestimmung

Die Größe der Routing-Tabellen ist proportional zur Größe des Netzwerks:

- großer Speicherbedarf in den Knoten
- viel CPU-Zeit zum Durchsuchen der Tabellen
- viel Bandbreite zum Austausch von Routing-Informationen.

Eine **hierarchische** Leitwegbestimmung wird ab einer bestimmten Netzgröße notwendig:

- Die Knoten werden in Regionen gruppiert
- Jeder Knoten kennt
  - alle Details seiner Region
  - seine Leitwege zu allen anderen Regionen

Nachteil: nicht immer sind global optimale Entscheidungen möglich

Beispiel für die hierarchische Leitwegbestimmung

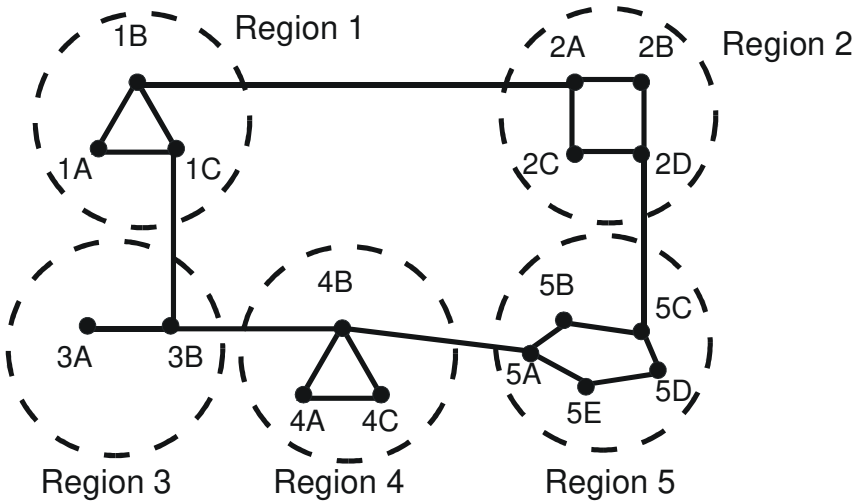


Tabelle für 1 A

Ziel	Leitung	TStrecke
1A	-	-
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

Hierarchische Tabelle

Ziel	Leitung	TStrecke
1A	-	-
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

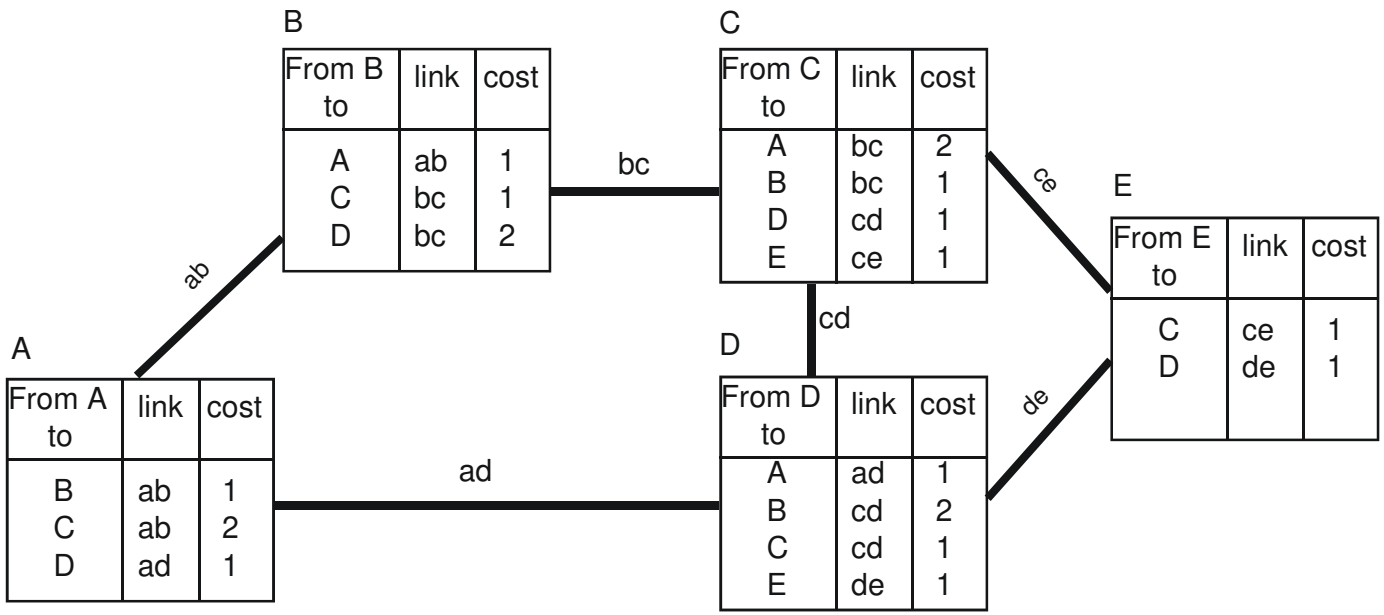
Routing im Internet

Distance Vector Routing

Das in den frühen Jahren im Internet am meisten verwendete Verfahren ist ein adaptives verteiltes Verfahren auf der Basis von Distanzvektoren (distance vector routing). Das eingesetzte Protokoll heißt **RIP** (Routing Information Protocol).

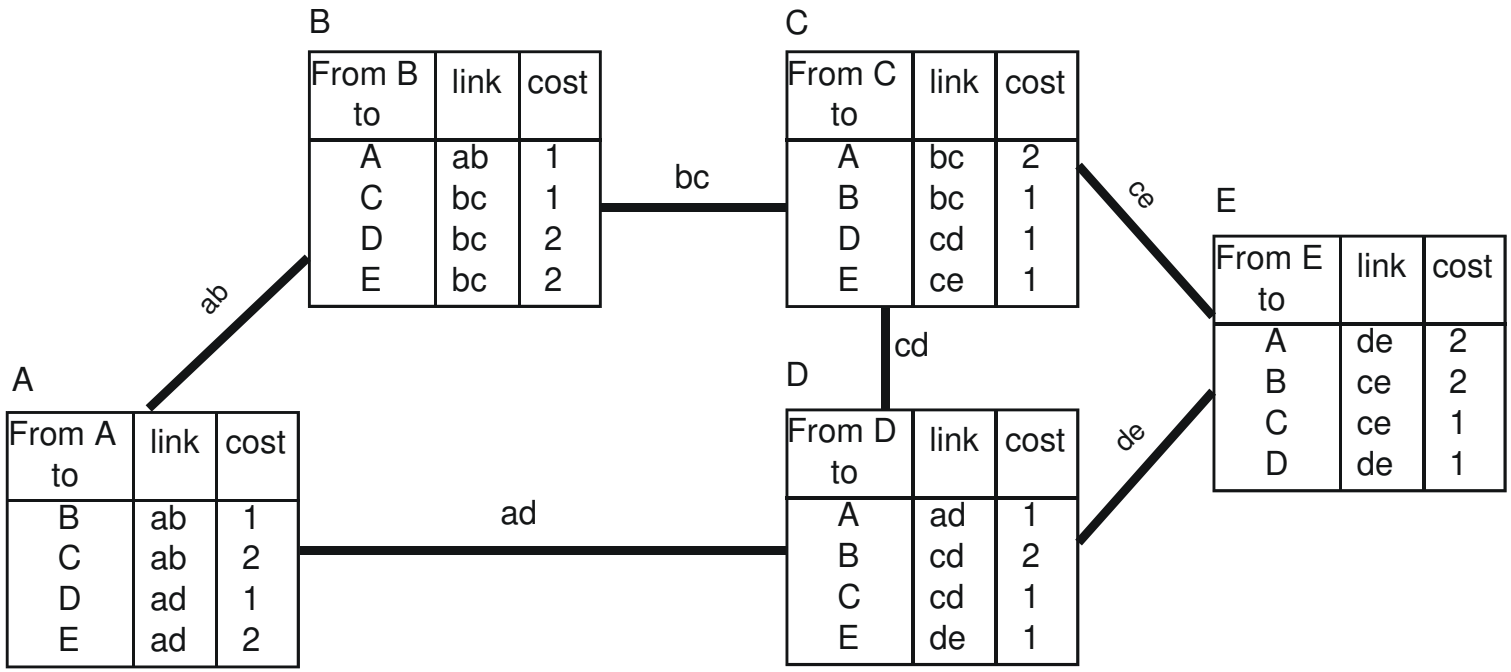
Alle Internet-Router tauschen dabei periodisch RIP-Nachrichten aus und aktualisieren ihre Routing-Tabellen beim Eintreffen von RIP-Nachrichten von ihren Nachbarn.

Beispiel für Routing mit Distanzvektoren (1)



(a) Knoten E ist gerade hinzugekommen

Beispiel für Routing mit Distanzvektoren (2)



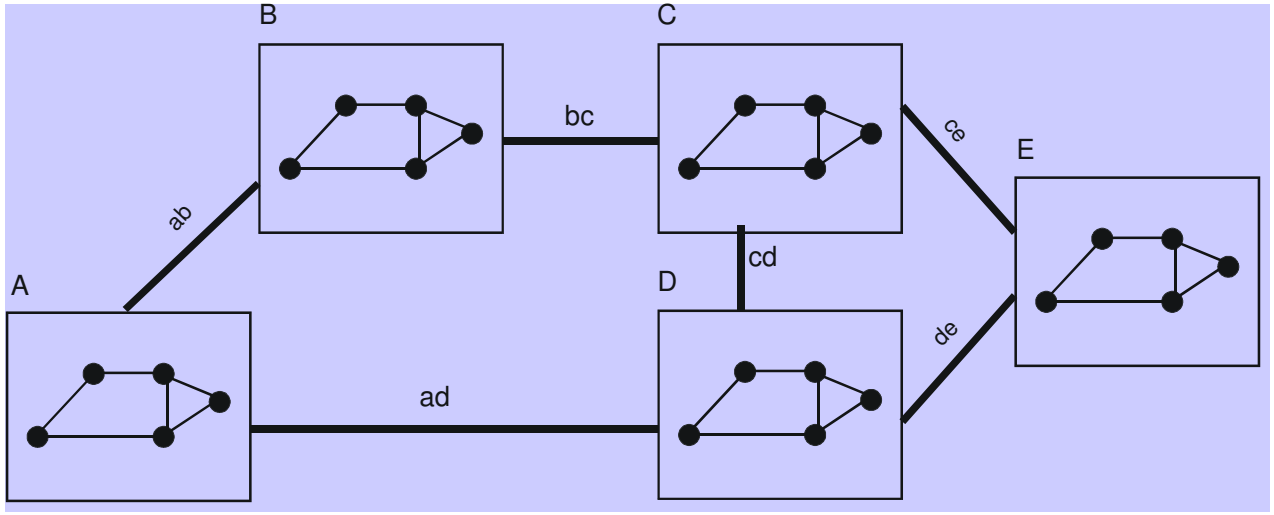
(b) nach einer Runde von RIP-Nachrichten

OSPF-Routing

Ein zweiter wichtiger und heute meist verwendeter Routing-Algorithmus im Internet ist **OSPF** (**Open Shortest Path First**). Die Idee ist, dass alle Knoten jederzeit die gesamte Netztopologie kennen und lokal alle optimalen Pfade berechnen können. Wenn sich die Topologie ändert, tauschen die Knoten Änderungsnachrichten aus. Jeder Knoten unterhält lokal eine Datenbank über die gesamte Topologie.

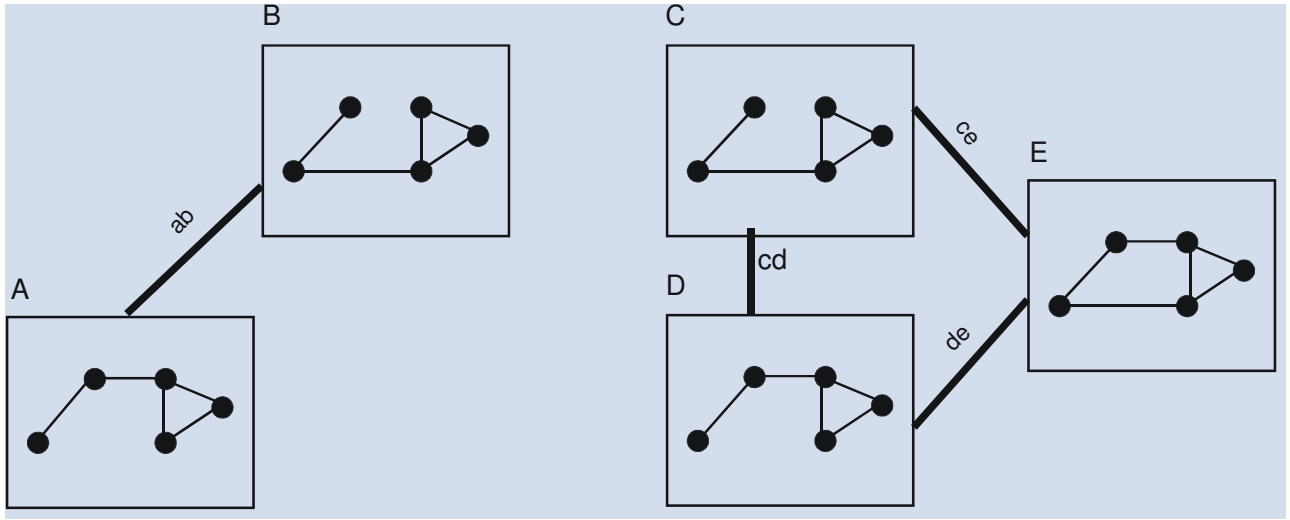
Auf der Basis der vollen Topologie werden die optimalen Pfade zu allen anderen Knoten mit dem Algorithmus für kürzeste Wege von Dijkstra (Shortest Path First = SPF) berechnet. Im Internet-Slang heißt der Algorithmus deshalb auch Open Shortest Path First.

Beispiel für OSPF-Routing (1)



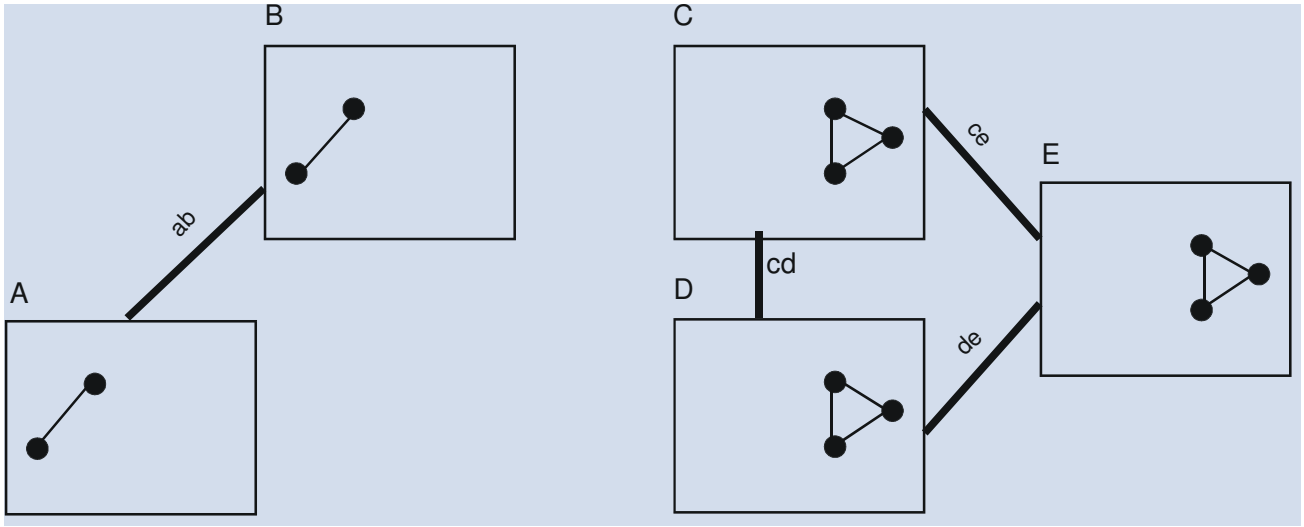
(a) Netzwerk im stabilen Zustand

Beispiel für OSPF-Routing (2)



(b) Die Links bc und ad sind ausgefallen

Beispiel für OSPF-Routing (3)



(c) Nach einer Runde von OSPF-Nachrichten