ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

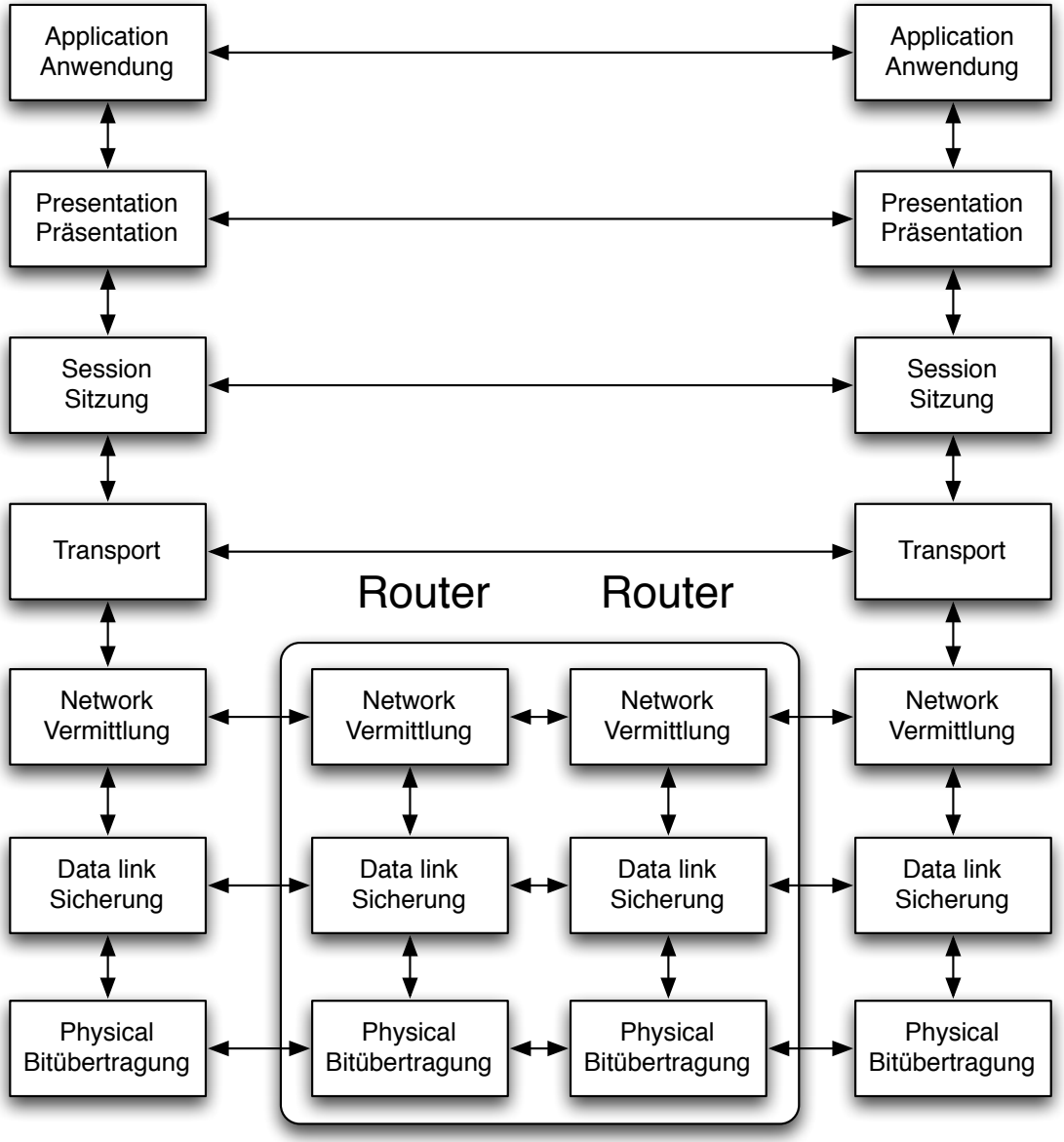
Algorithmen für drahtlose Netzwerke

Routing, Distance-Vector, Link-State

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

ISO/OSI Referenzmodel

- **7. Anwendung (Application)**
  - Datenübertragung, E-Mail, Terminal, Remote login
- **6. Darstellung (Presentation)**
  - Systemabhängige Darstellung der Daten (EBCDIC/ASCII)
- **5. Sitzung (Session)**
  - Aufbau, Ende, Wiederaufsetzpunkte
- **4. Transport (Transport)**
  - Segmentierung, Stauvermeidung
- **3. Vermittlung (Network)**
  - Routing
- **2. Sicherung (Data Link)**
  - Prüfsummen, Flusskontrolle
- **1. Bitübertragung (Physical)**
  - Mechanische, elektrische Hilfsmittel



Routing im Internet

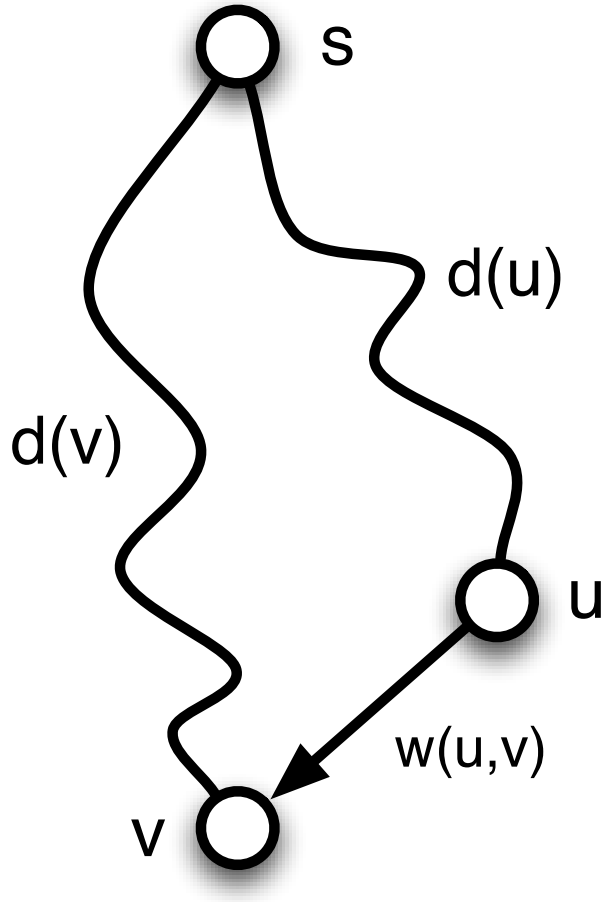
- **Routing-Tabelle**
  - enthält für Ziel (Destination) die Adresse des nächsten Rechners (Gateway)
  - Destination kann einen Rechner oder ganze Sub-nets beschreiben
  - Zusätzlich wird ein Default-Gateway angegeben
- **Packet Forwarding (früher Packet Routing)**
  - IP-Paket (datagram) enthält Start-Adresse und Ziel-Adresse
    - Ist Ziel-IP-Adresse = eigene Rechneradresse Nachricht angekommen
    - Ist Ziel-IP-Adresse in Routing-Tabelle dann leite Paket zum angegeben Gateway
    - Ist Ziel-IP-Subnetz in Routing-Tabelle dann leite Paket zum angegeben Gateway
    - Ansonsten leite Paket zum Default-Gateway

Das Kürzeste-Wege-Problem

- **Gegeben:**
  - Ein gerichteter Graph G=(V,E)
  - Startknoten
  - mit Kantengewichtungen: w: E $\rightarrow \mathbb{R}$
- **Definiere Gewicht des kürzesten Pfades**
  - δ(u,v) = minimales Gewicht w(p) eines Pfades p von u nach v
  - w(p) = Summe aller Kantengewichte w(e) der Kanten e des Pfades
- **Gesucht:**
  - Die kürzesten Wege vom Startknoten s zu allen Knoten in G
    - ein Pfad mit dem geringsten Gewicht zu jedem anderen Knoten
- **Lösungsmenge:**
  - einen Baum mit Wurzel s
  - Jeder Knoten zeigt in Richtung der Wurzel

Anwendung der Dreiecksungleichung

Relax(u, v)
begin
    if $d(v) > d(u) + w(u, v)$ then
        $d(v) \leftarrow d(u) + w(u, v)$
        $\pi(v) \leftarrow u$
    fi
end



Initialisierung

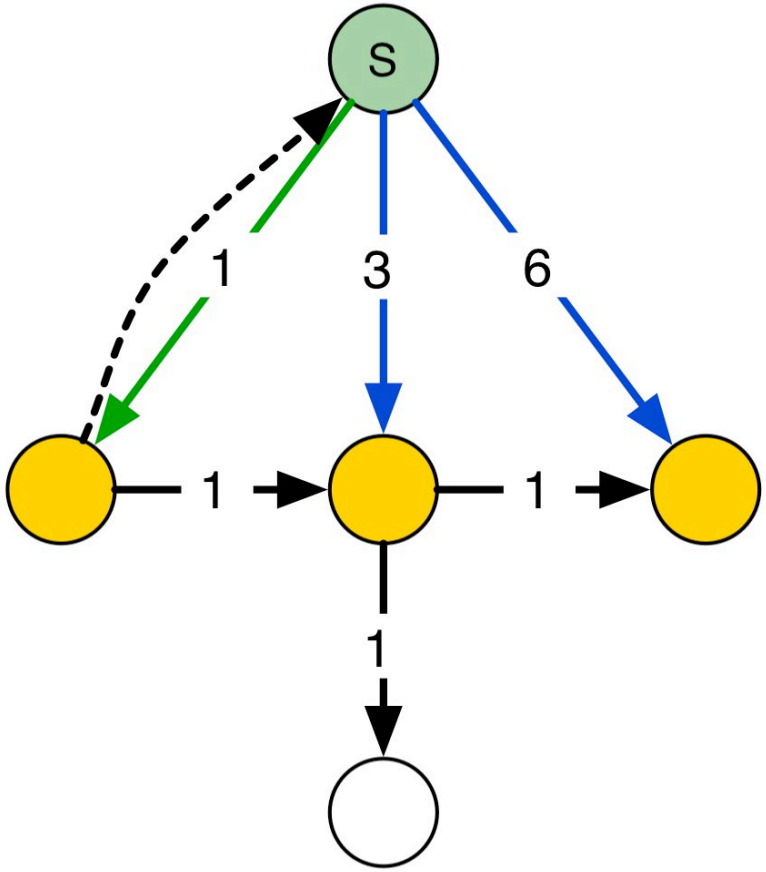
Init-Single-Source(G, w, s)
begin
    for all $v \in V$ do
        $d(v) \leftarrow \infty$
        $\pi(v) \leftarrow v$
    od
    $d(s) \leftarrow 0$
end

Kürzeste Wege Berechnung nach Dijkstra

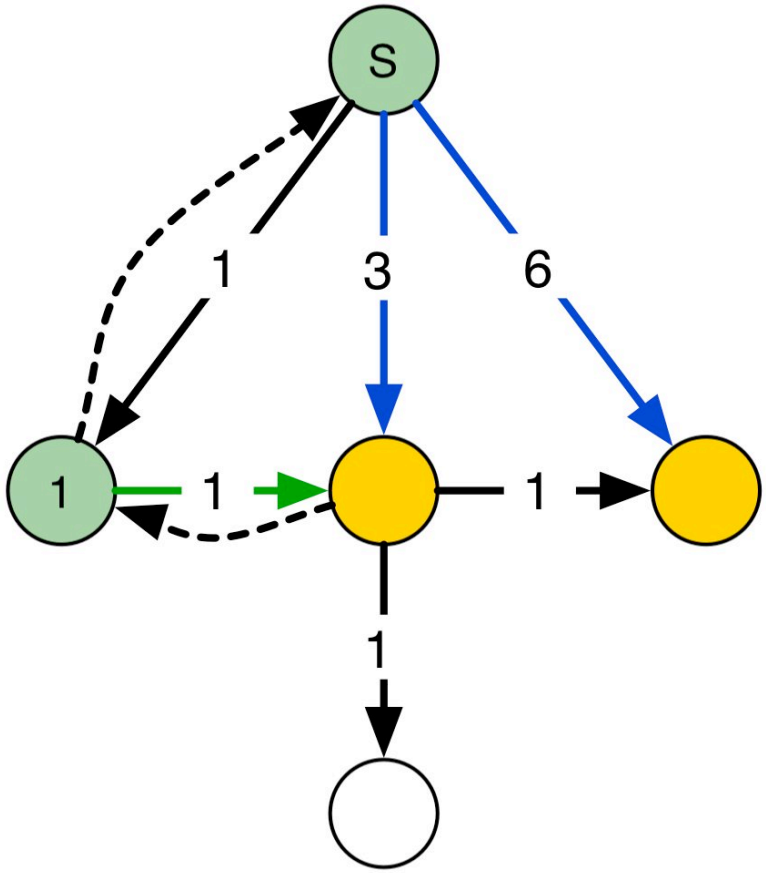
```
Dijkstra(G, w, s)
begin
    Init-Single-Source(G, w)
    S ← ∅
    Q ← V
    while Q ≠ ∅ do
        u ← Element aus Q mit minimalen Wert d(u)
        S ← S ∪ {u}
        Q ← Q \ {u}
        for all v ∈ Adj(u) do
            Relax(u, v)
        od
    od
end
```

- Laufzeit Θ(|E| + |V| log |V|)

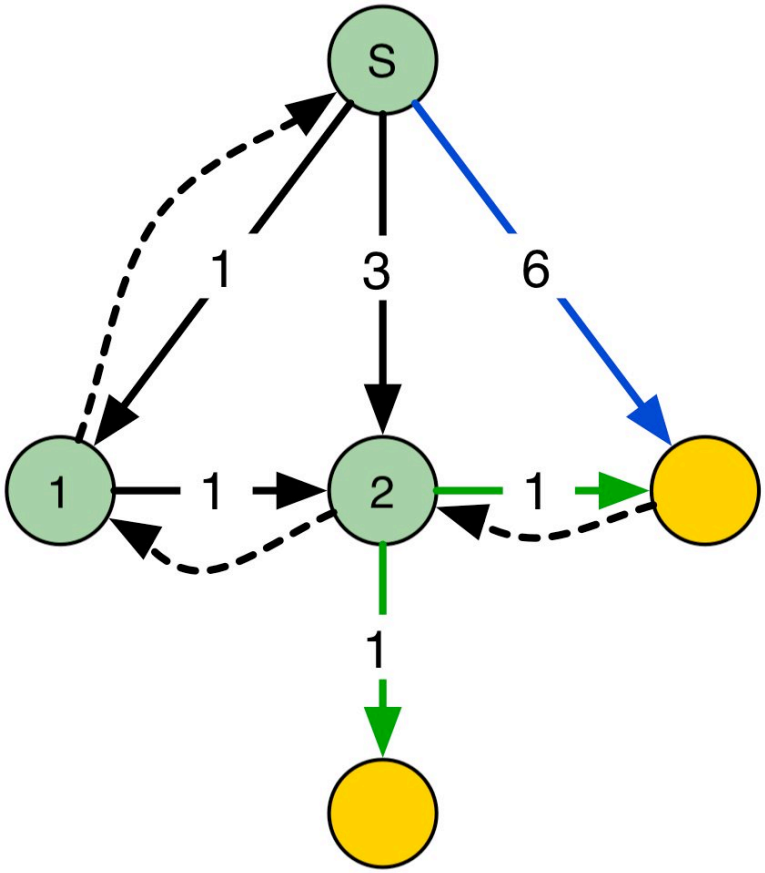
Dijkstra: Beispiel



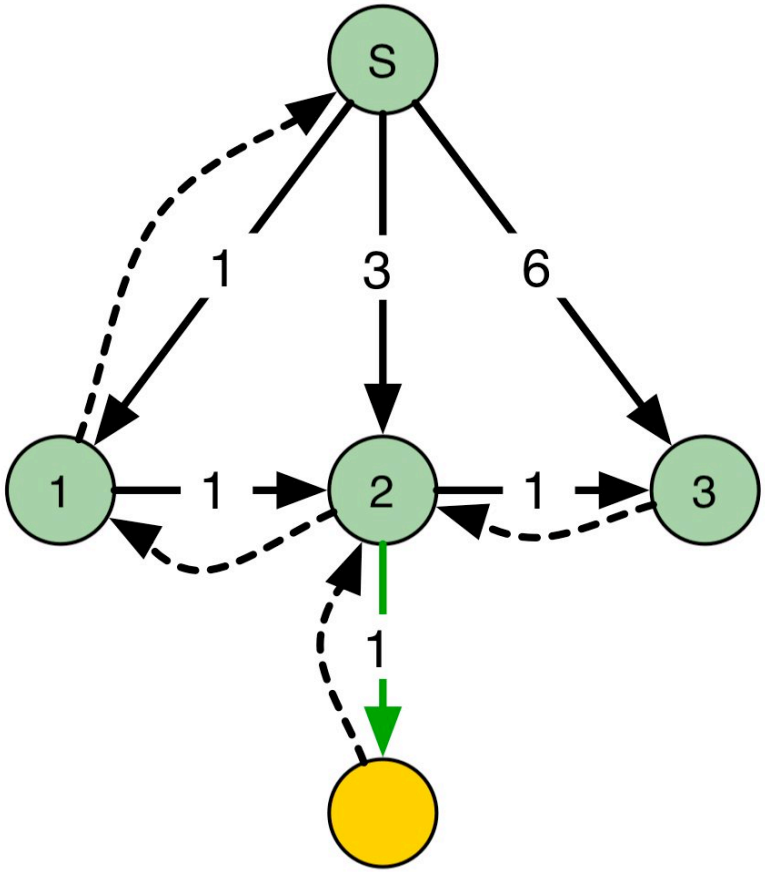
Dijkstra: Beispiel



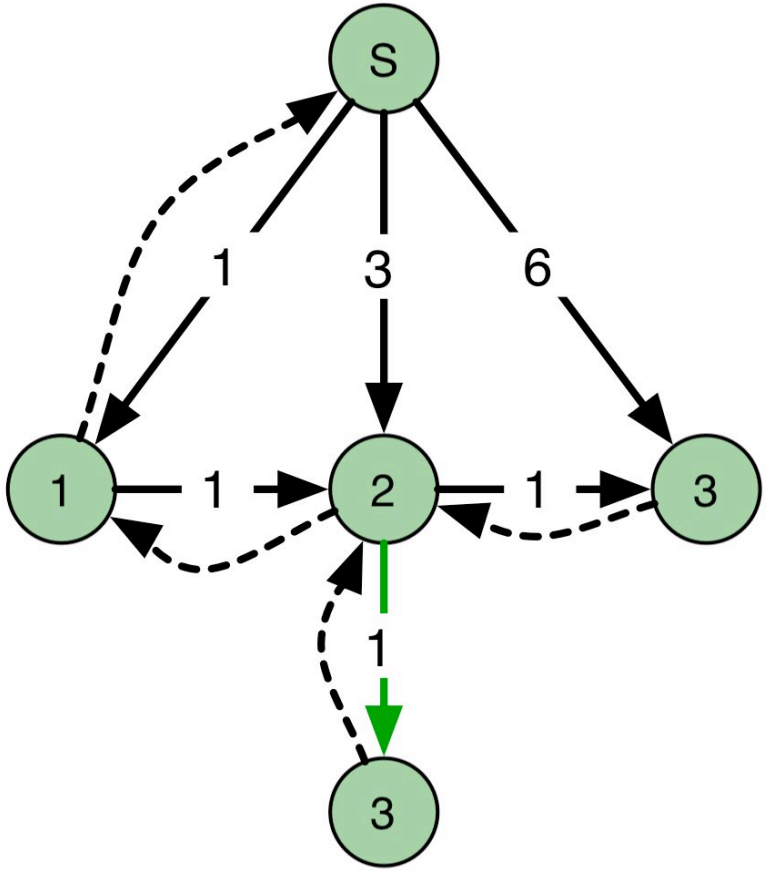
Dijkstra: Beispiel



Dijkstra: Beispiel



Dijkstra: Beispiel



Bellman-Ford

Bellman-Ford(G, w, s)
begin
    Init-Single-Source(G, w)
    loop $|V| - 1$ times do
        for all $(u, v) \in E$ do
            Relax(u, v)
        od
    od
    for all $(u, v) \in E$ do
        if $d(v) > d(u) + w(u, v)$ then return false
    od
    return true
end

- **Bei negativen Kantengewichten versagt Dijkstras Algorithmus**
- **Bellman-Ford**
  - löst dies in Laufzeit O(|V| |E|).

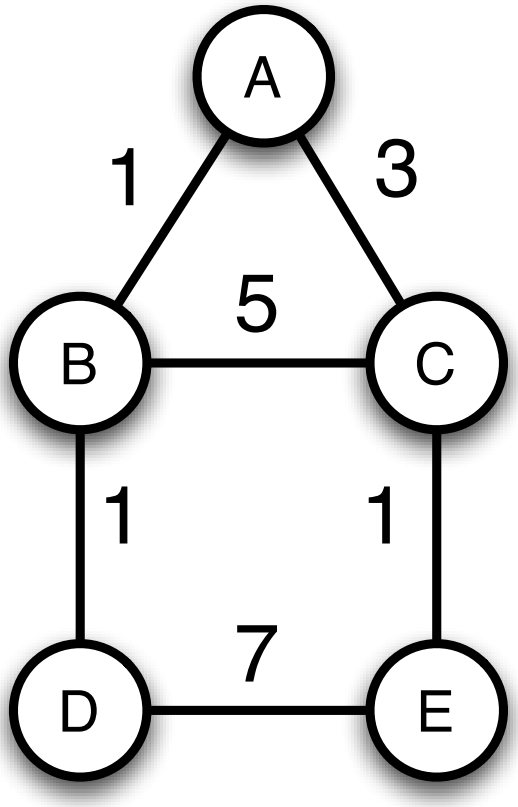
Distance Vector Routing Protocol

- **Distance Table Datenstruktur**
  - Jeder Knoten besitzt eine
    - Zeile für jedes mögliches Ziel
    - Spalte für jeden direkten Nachbarn
- **Verteilter Algorithmus**
  - Jeder Knoten kommuniziert nur mit seinem Nachbarn
- **Asynchroner Betrieb**
  - Knoten müssen nicht Informationen austauschen in einer Runde
- **Selbstterminierend**
  - läuft bis die Knoten keine Informationen mehr austauschen

Distance Table für A

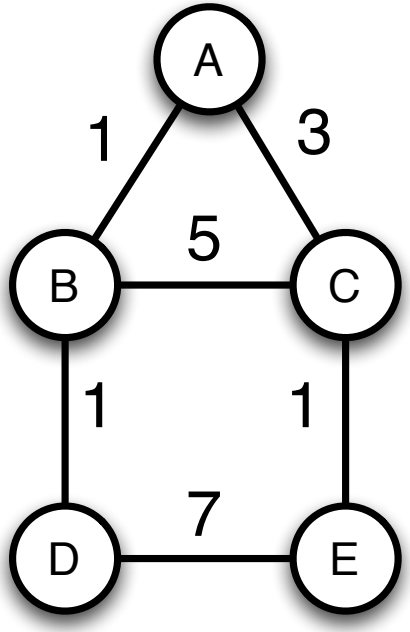
	über		Routing Tabellen
von A	B	E	Eintrag
nach B	**2**	11	B
C	**3**	19	B
D	**7**	10	B
E	**8**	9	E

Distance Vector Routing Beispiel



von A nach	über		Eintrag
	B	C	
B	**1**	7	**B**
C	5	**3**	**C**
D	**2**	8	**B**
E	6	**4**	**C**

Distance Vector Routing



von A nach	über		Eintrag
	B	C	
B	1	-	B
C	-	3	C
D	-	-	-
E	-	-	-

von B nach	über			Eintrag
	A	C	D	
A	1	-	-	A
C	-	3	-	C
D	-	-	1	C
E	-	-	8	D

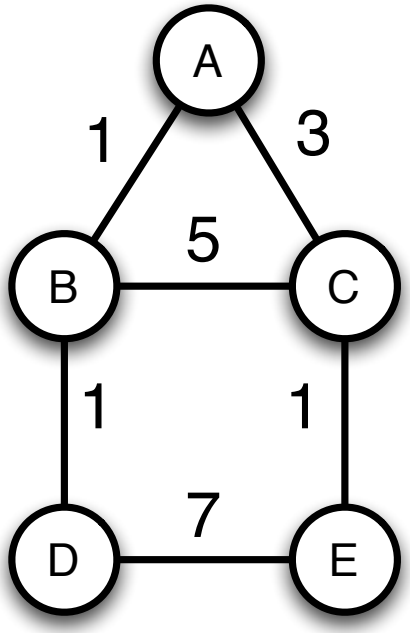
von C nach	über			Eintrag
	A	B	E	
A	3	-	-	A
B	-	5	-	B
D	-	-	8	E
E	-	-	1	E

Distance Vector Routing

von B nach	über A	über C	über D	Eintrag
A	1	-	-	A
C	-	5	-	C
D	-	-	1	D
E	-	-	8	D

von C nach	über A	über B	über E	Eintrag
A	3	-	-	A
B	-	5	-	B
D	-	-	8	E
E	-	-	1	E

von B nach	über A	über C	über D	Eintrag
A	1	8	-	A
C	-	5	-	C
D	-	13	1	D
E	-	6	8	C

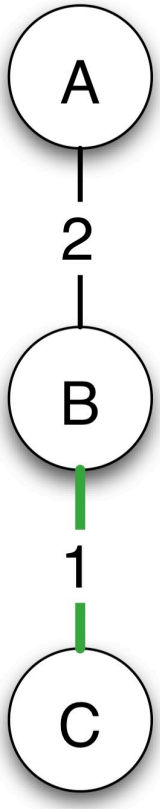


von C nach	über A	über B	über E	Eintrag
A	3	6	-	A
B	-	5	-	B
D	-	6	8	B
E	-	13	1	E

Das “Count to Infinity” - Problem

- **Gute Nachrichten verbreiten sich schnell**
  - Neue Verbindung wird schnell veröffentlicht
- **Schlechte Nachrichten verbreiten sich langsam**
  - Verbindung fällt aus
  - Nachbarn erhöhen wechselseitig ihre Entfernung
  - “Count to Infinity”-Problem

Das “Count to Infinity” - Problem



Distance Table für A

	von A	über B	Routing Tabellen Eintrag
nach	B	**2**	B
	C	**-**	B

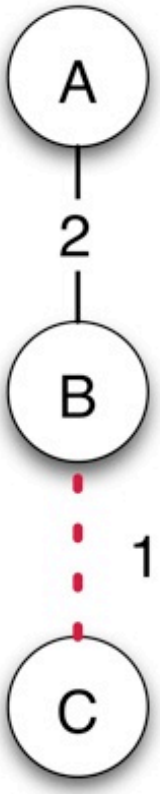
Distance Table für B

	von B	über A	über C	Routing Tabellen Eintrag
nach	A	2	-	A
	C	-	1	C

nach kurzer Zeit

	von A	über B	Routing Tabellen Eintrag
nach	B	**2**	B
	C	**3**	B

	von B	über A	über C	Routing Tabellen Eintrag
nach	A	2	4	A
	C	5	1	C



von A	über B	Routing Tabellen Eintrag
nach B	2	B
C	3	B

von B	über A	über C	Routing Tabellen Eintrag
nach A	2	-	A
C	5	-	A

von A	über B	Routing Tabellen Eintrag
nach B	2	B
C	7	B

von B	über A	über C	Routing Tabellen Eintrag
nach A	2	-	A
C	5	-	A

von A	über B	Routing Tabellen Eintrag
nach B	2	B
C	7	B

von B	über A	über C	Routing Tabellen Eintrag
nach A	2	-	A
C	9	-	A

Link-State Protocol

- **Link State Router**
  - tauschen Information mittels Link State Packets (LSP) aus
  - Jeder verwendet einen eigenen Kürzeste-Wege-Algorithmus zu Anpassung der Routing-Tabelle
- **LSP enthält**
  - ID des LSP erzeugenden Knotens
  - Kosten dieses Knotens zu jedem direkten Nachbarn
  - Sequenznr. (SEQNO)
  - TTL-Feld für dieses Feld (time to live)
- **Verlässliches Fluten (Reliable Flooding)**
  - Die aktuellen LSP jedes Knoten werden gespeichert
  - Weiterleitung der LSP zu allen Nachbarn
    - bis auf den Knoten der diese ausgeliefert hat
  - Periodisches Erzeugen neuer LSPs
    - mit steigender SEQNOs
  - Verringern der TTL bei jedem Weiterleiten

ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Algorithmen für drahtlose Netzwerke

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer