CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG

Systeme II

4. Die Vermittlungsschicht

Christian Schindelhauer
Technische Fakultät
Rechnernetze und Telematik
Albert-Ludwigs-Universität Freiburg

Version 03.06.2014

UNI FREIBURG

CoNe Freiburg

Circuit Switching oder Packet Switching

- Circuit Switching
  - Etablierung einer Verbindung zwischen lokalen Benutzern durch Schaltstellen
    - mit expliziter Zuordnung von realen Schaltkreisen
    - oder expliziter Zuordnung von virtuellen Ressourcen, z.B. Slots
  - Quality of Service einfach, außer bei
    - Leitungsaufbau
    - Leitungsdauer
  - Problem
    - Statische Zuordnung
    - Ineffiziente Ausnutzung des Kommunikationsmedium bei dynamischer Last
  - Anwendung
    - Telefon
    - Telegraf
    - Funkverbindung

CoNe Freiburg

Circuit Switching oder Packet Switching

- Packet Switching
  - Grundprinzip von IP
    - Daten werden in Pakete aufgeteilt und mit Absender/Ziel-Information unabhängig versandt
  - Problem: Quality of Service
    - Die Qualität der Verbindung hängt von einzelnen Paketen ab
    - Entweder Zwischenspeichern oder Paketverlust
  - Vorteil:
    - Effiziente Ausnutzung des Mediums bei dynamischer Last
- Resümee
  - Packet Switching hat Circuit Switching in praktisch allen Anwendungen abgelöst
  - Grund:
    - Effiziente Ausnutzung des Mediums

CoNe Freiburg

Taktik der Schichten

- Transport
  - muss gewisse Flusskontrolle gewährleisten
  - z.B. Fairness zwischen gleichzeiten Datenströmen
- Vermittlung
  - Quality of Service (virtuelles Circuit Switching)
- Sicherung
  - Flusskontrolle zur Auslastung des Kanals

Layer	Policies
Transport	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy • Timeout determination
Network	• Virtual circuits versus datagram inside the subnet • Packet queueing and service policy • Packet discard policy • Routing algorithm • Packet lifetime management
Data link	• Retransmission policy • Out-of-order caching policy • Acknowledgement policy • Flow control policy

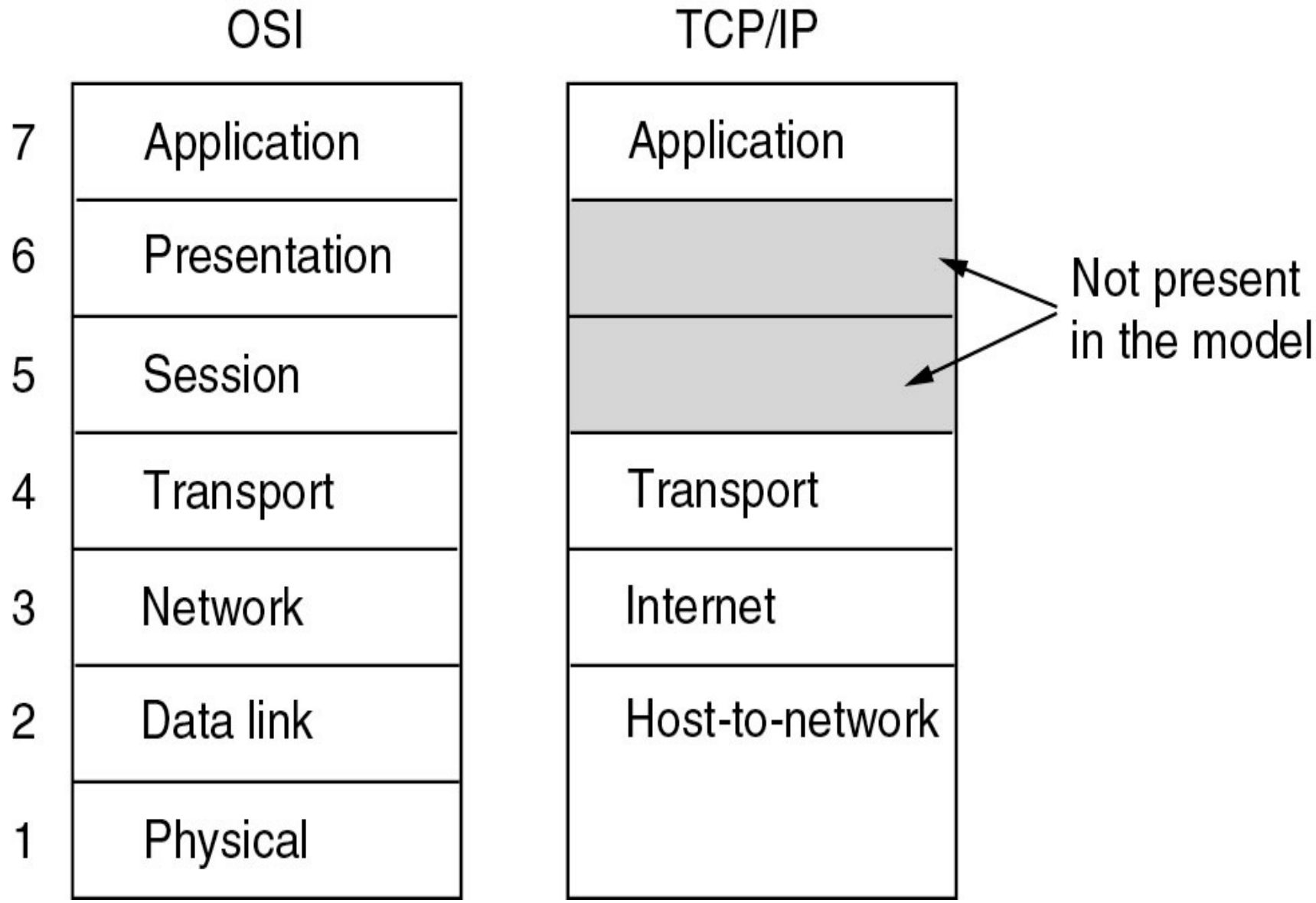
CoNe Freiburg

Die Schichtung des Internets - TCP/IP-Layer

Anwendung	Application	Telnet, FTP, HTTP, SMTP (E-Mail), ...
Transport	Transport	TCP (Transmission Control Protocol) UDP (User Datagram Protocol)
Vermittlung	Network	IP (Internet Protocol) + ICMP (Internet Control Message Protocol) + IGMP (Internet Group Management Protoccol)
Verbindung	Host-to-network	LAN (z.B. Ethernet, Token Ring etc.)

CoNe Freiburg

OSI versus TCP/IP



CoNe Freiburg

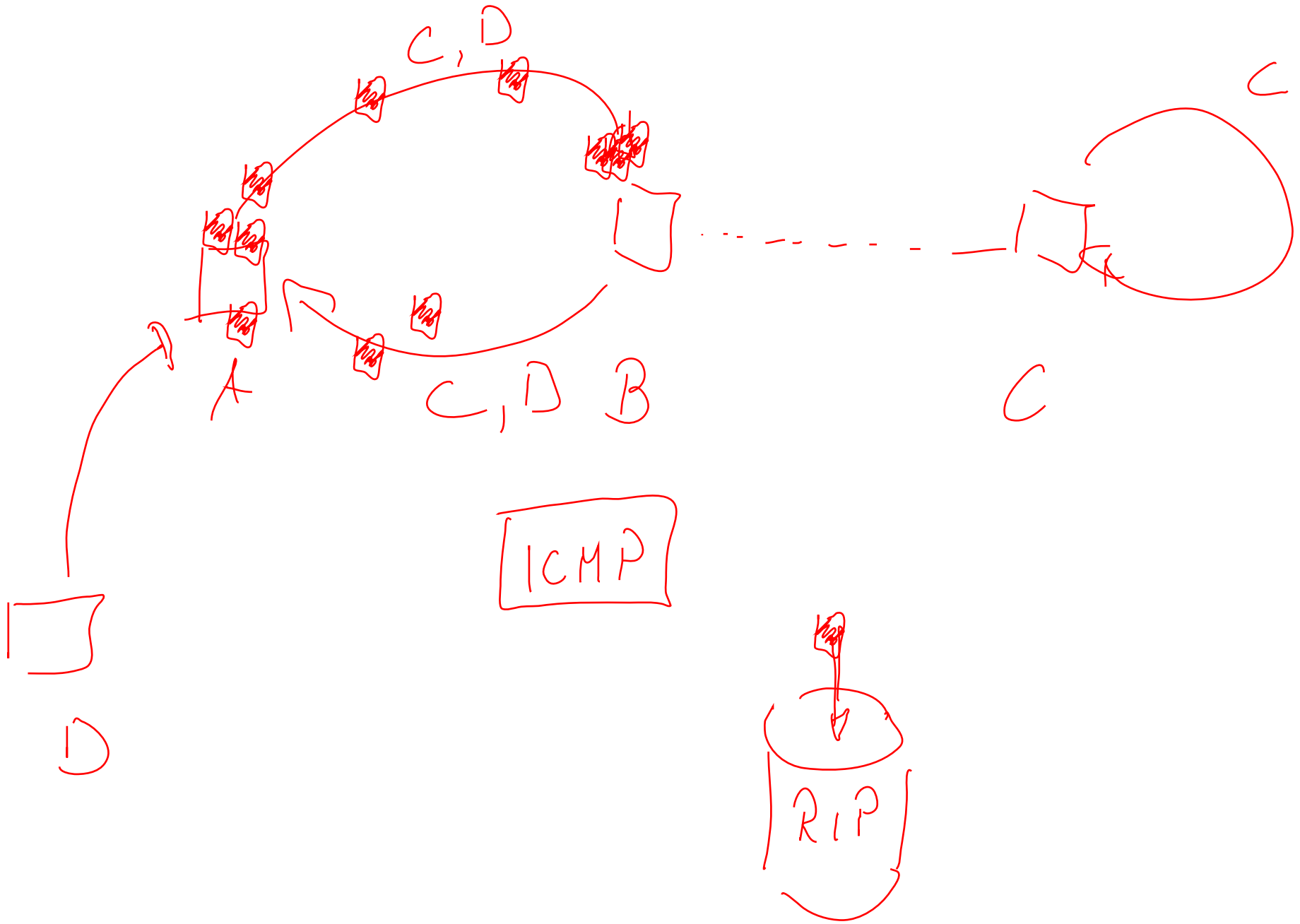
Warum eine Vermittlungsschicht

- Lokale Netzwerke können nicht nur über Hubs, Switches oder Bridges verknüpft werden
  - Hubs: Kollisionen nehmen überhand
  - Switches:
    - Routen-Information durch Beobachtung der Daten ineffizient
    - Broadcast aller Nachrichten schafft Probleme
  - Es gibt über 100 Mio. lokale Netzwerke im Internet...
- Zur Beförderung von Paketen in großen Netzwerken braucht man Routeninformationen
  - Wie baut man diese auf?
  - Wie leitet man Pakete weiter?
- Das Internet-Protokoll ist im wesentlich ein Vermittlungsschichtprotokoll

CoNe Freiburg

Routing-Tabelle und Paket-Weiterleitung

- IP-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 genannt
  - IP-Paket (datagram) enthält Start-IP-Adresse und Ziel-IP-Adresse
    - Ist Ziel-IP-Adresse = eigene Rechneradresse dann Nachricht ausgeliefert
    - 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 zum Default-Gateway


C, D
C
A
C, D
B
C
D
ICMP
RIP

CoNe Freiburg

Paket-Weiterleitung im Internet Protokoll

- IP-Paket (datagram) enthält unter anderen
  - TTL (Time-to-Live): Anzahl der Hops
  - Start-IP-Adresse
  - Ziel-IP-Adresse
- Behandlung eines Pakets
  - Verringere TTL (Time to Live) um 1
  - Falls TTL ≠ 0 dann Packet-Forwarding aufgrund der Routing-Tabelle
  - Falls TTL = 0 oder bei Problemen in Packet-Forwarding:
    - Lösche Paket
    - Falls Paket ist kein ICMP-Paket dann
      - Sende ICMP-Paket mit
        - Start= aktuelle IP-Adresse und
        - Ziel = alte Start-IP-Adresse

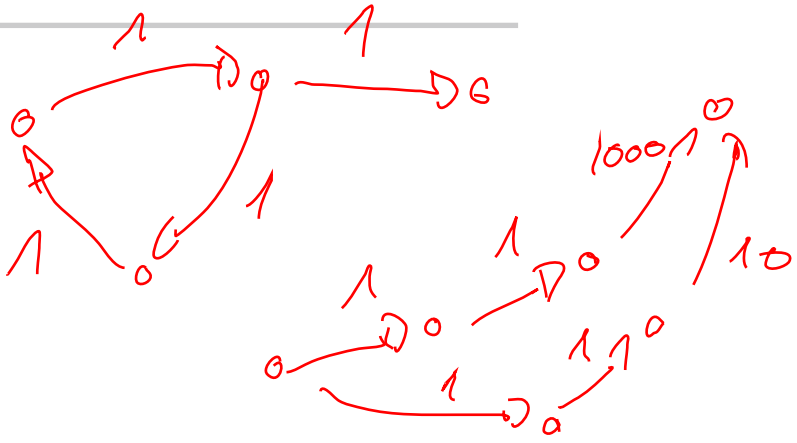
CoNe Freiburg

Statisches und Dynamisches Routing

- Forwarding:
  - Weiterleiten von Paketen
- Routing:
  - Erstellen Routen, d.h.
    - Erstellen der Routing-Tabelle
- Statisches Routing
  - Tabelle wird manuell erstellt
  - sinnvoll für kleine und stabile LANs
- Dynamisches Routing
  - Tabellen werden durch Routing-Algorithmus erstellt
  - Zentraler Algorithmus, z.B. Link State
    - Einer/jeder kennt alle Information, muss diese erfahren
  - Dezentraler Algorithmus, z.B. Distance Vector
    - arbeitet lokal in jedem Router
    - verbreitet lokale Information im Netzwerk

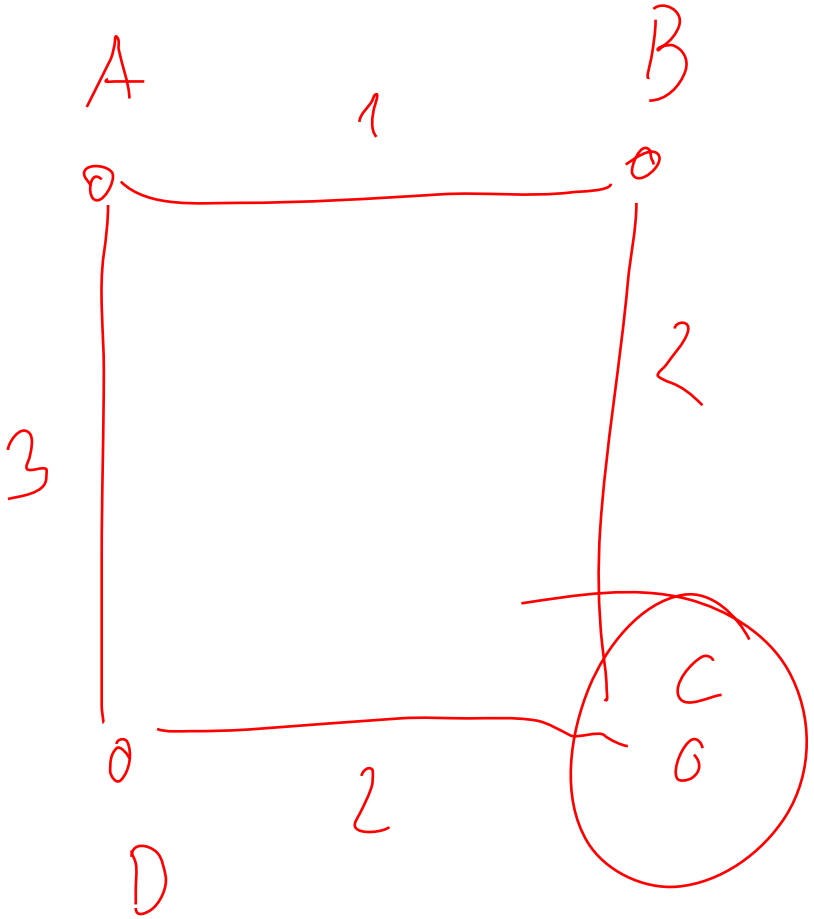
CoNe Freiburg

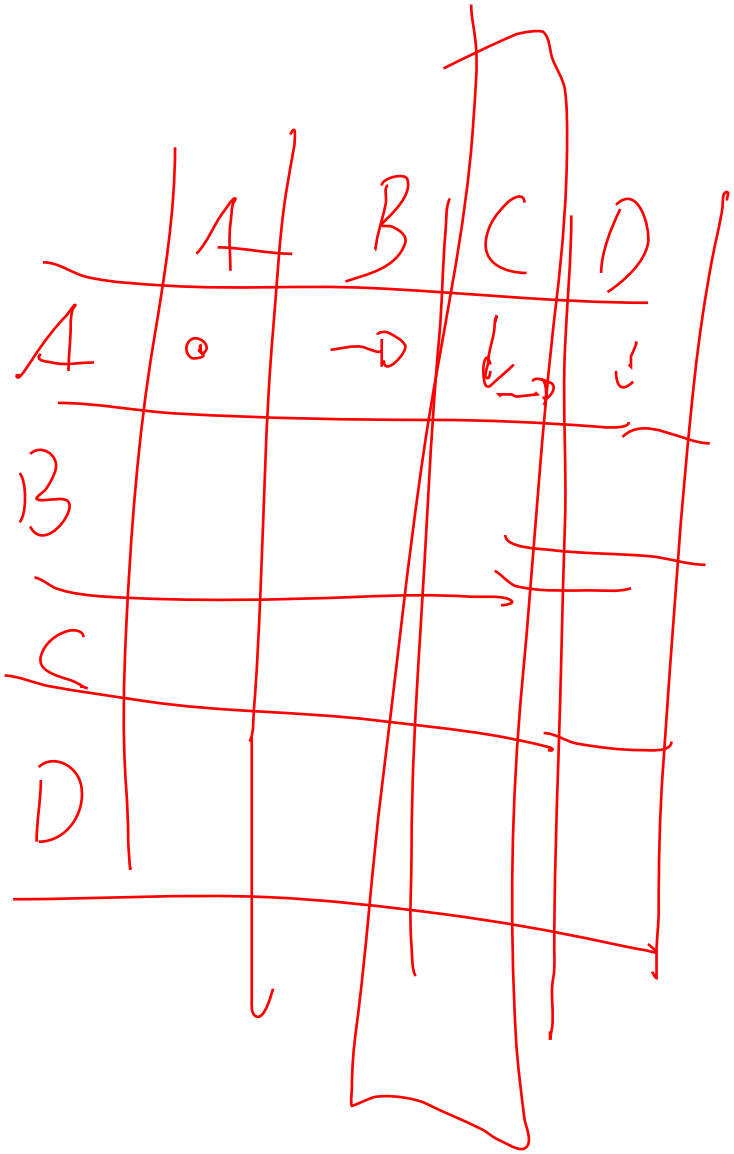
Das Kürzeste-Wege-Problem



- Gegeben:
  - Ein gerichteter Graph G=(V,E)
  - Startknoten
  - mit Kantengewichtungen $w : E \to \mathbb{R}$
- 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
    - also jeweils ein Pfad mit dem geringsten Gewicht zu jedem anderen Knoten
- Lösungsmenge:
  - wird beschrieben durch einen Baum mit Wurzel s
  - Jeder Knoten zeigt in Richtung der Wurzel

UNI FREIBURG


A
B
1
2
3
C
D
2


A
B
C
D
A
B
C
D

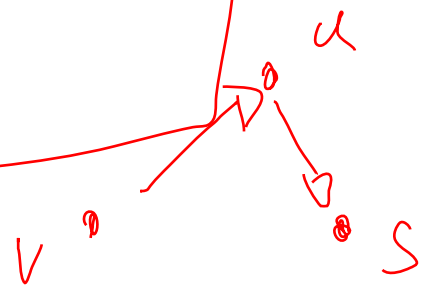
CoNe Freiburg

Kürzeste Wege mit Edsger Wybe Dijkstra

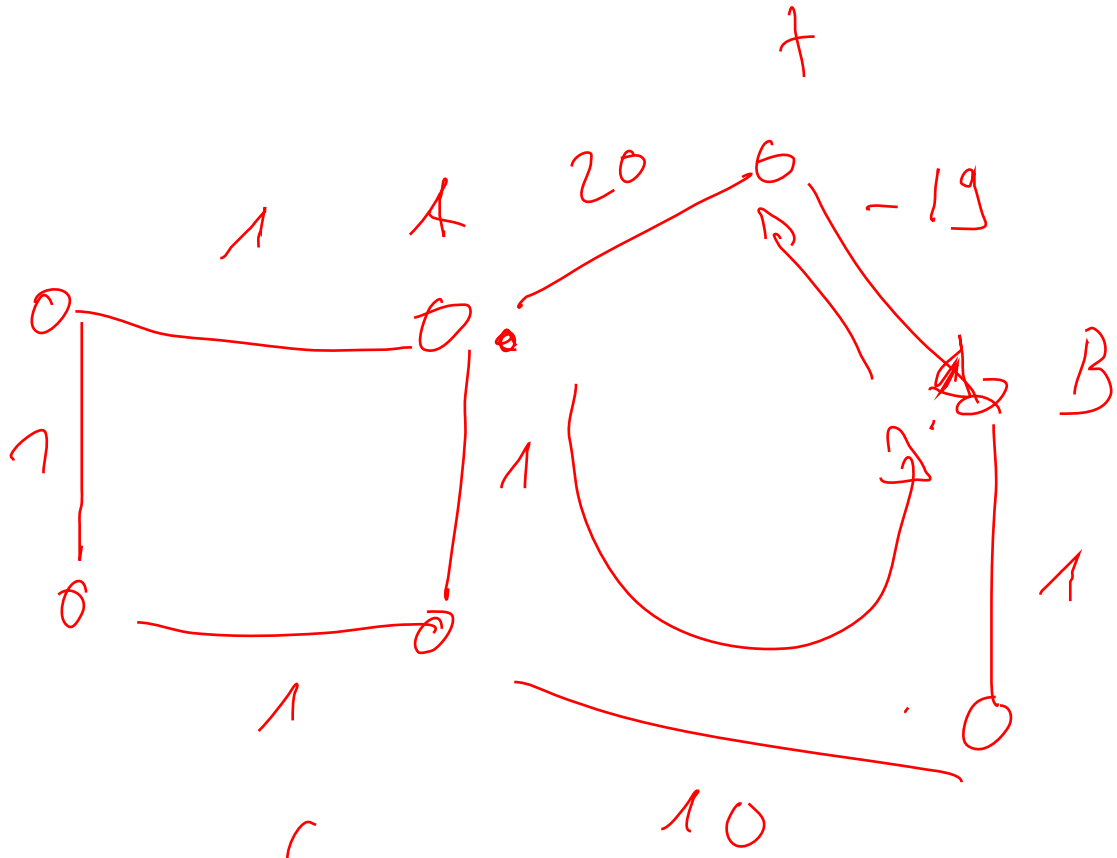
- Dijkstras Kürzeste-Wege-Algorithmus kann mit Laufzeit Θ(|E| + |V| log |V|) implementiert werden.

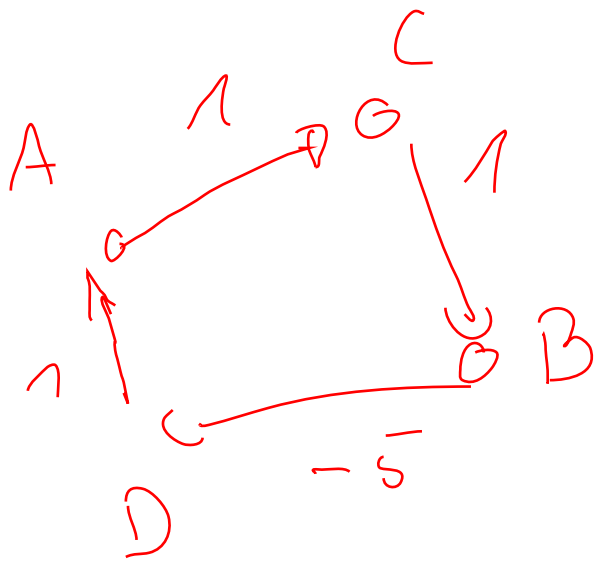
Dijkstra(G, w, s)
begin
  Init-Single-Source(G, w)
  $S \leftarrow \emptyset$
  $Q \leftarrow V$
  while $Q \neq \emptyset$ do
    $u \leftarrow$ Element aus Q mit minimalen Wert $d(u)$
    $S \leftarrow S \cup \{u\}$
    $Q \leftarrow Q \setminus \{u\}$
    for all $v \in$ Adj(u) do
      Relax(u, v)
    od
  od
end

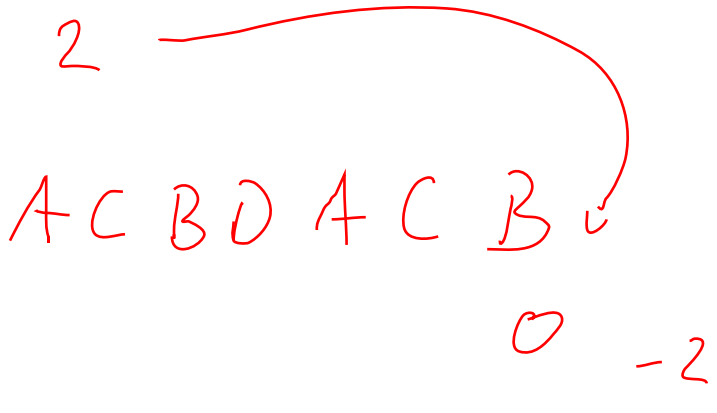
Init-Single-Source(G, w, s)
begin
  for all $v \in V$ do
    $d(v) \leftarrow \infty$
    $\pi(v) \leftarrow v$
  od
  $d(s) \leftarrow 0$
end

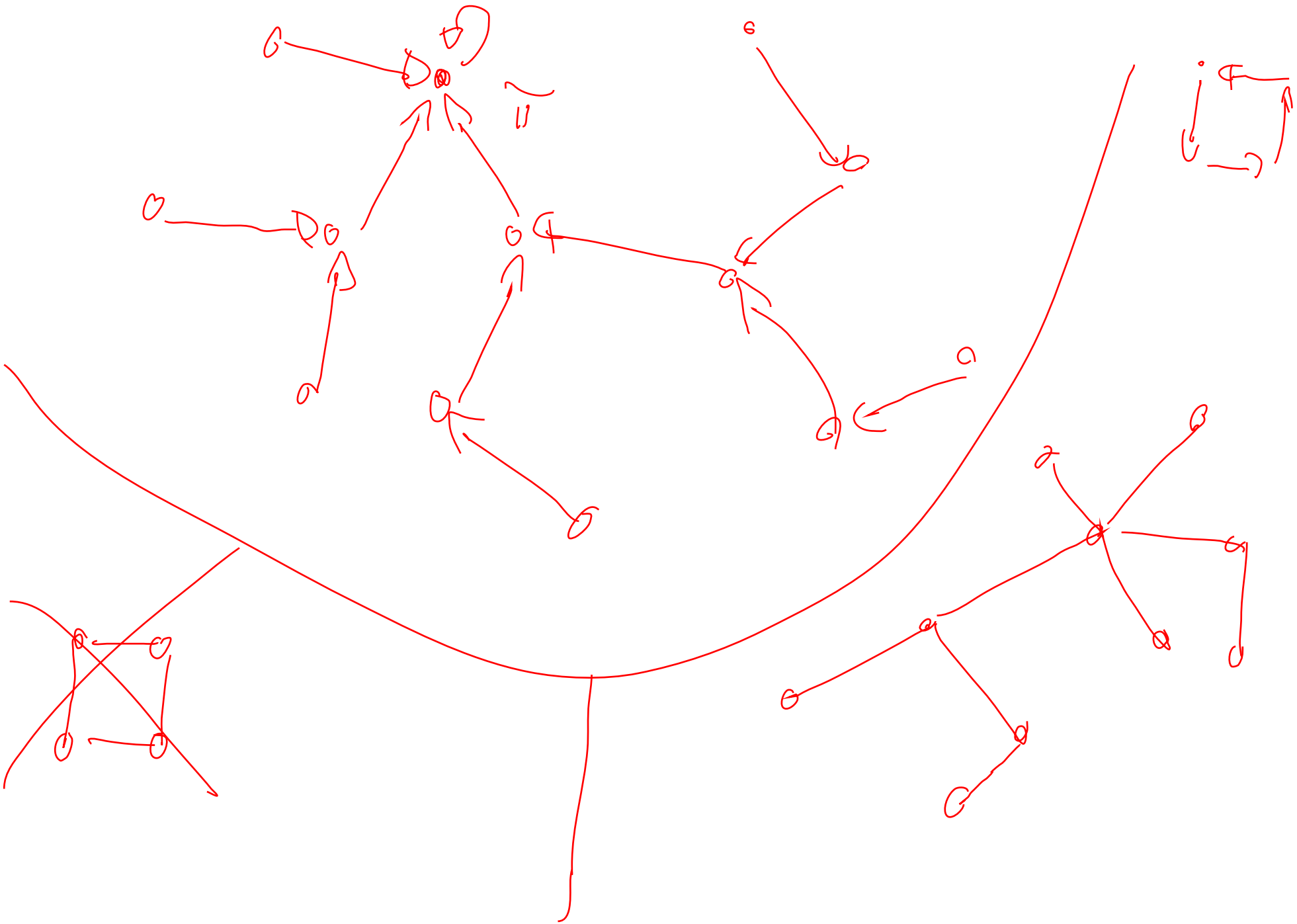


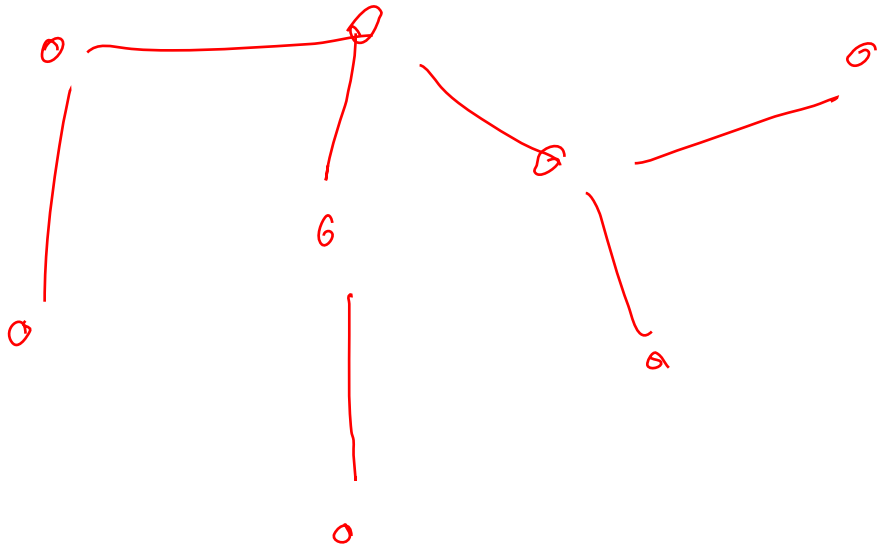
Relax(u, v)
begin
  if $d(v) > d(u) + w(u, v)$ then
    $d(v) \leftarrow d(u) + w(u, v)$
    $\pi(v) \leftarrow u$
  fi
end


1
A
20
-19
B
1
1
1
1
10

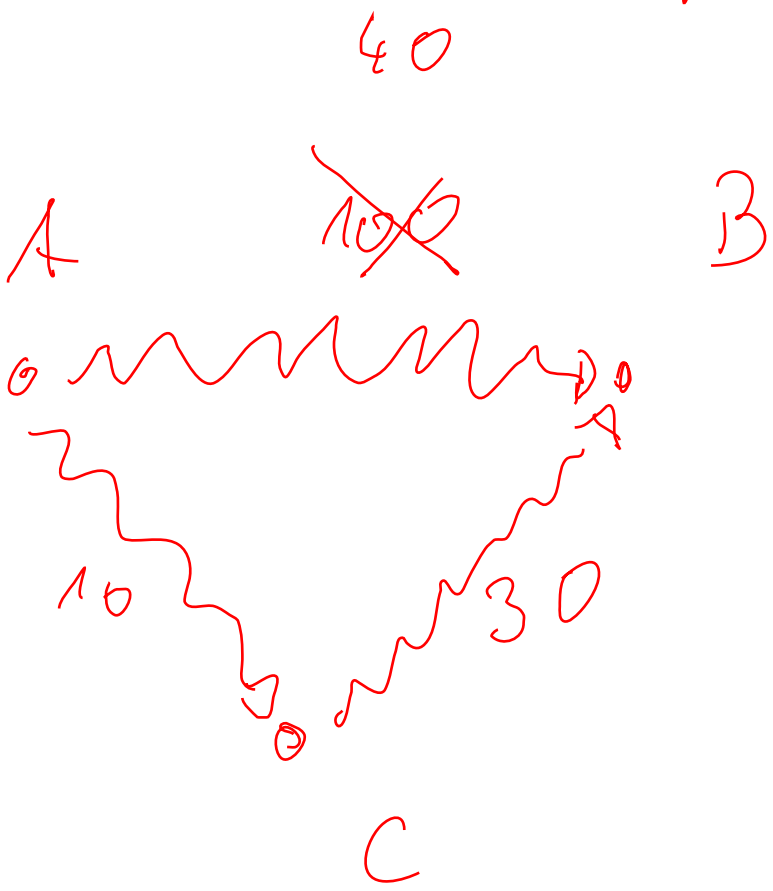

C
1
A
1
B
1
-5
D


2
ACBDACB
-2

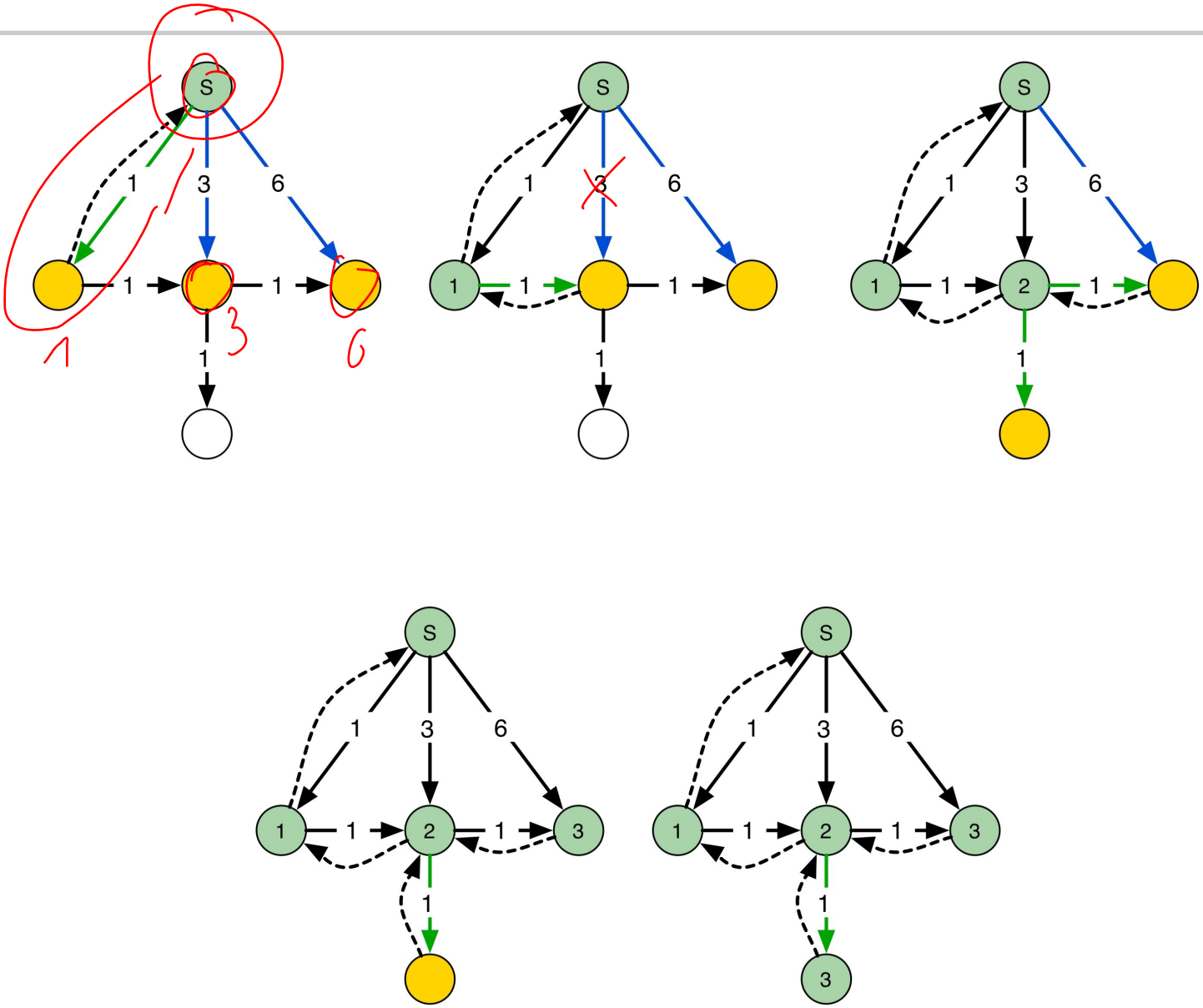




Relax



Dijkstra: Beispiel



CoNe Freiburg

Bellman-Ford

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

Bellman-Ford(G, w, s)

- **Init-Target**(G, w)
- loop $|V| - 1$ times:
  - for all $(u, v) \in E$ do
    - **Relax**(u, v)
- for all $(u, v) \in E$ do
  - if $d(u) > d(v) + w(u, v)$ then return false

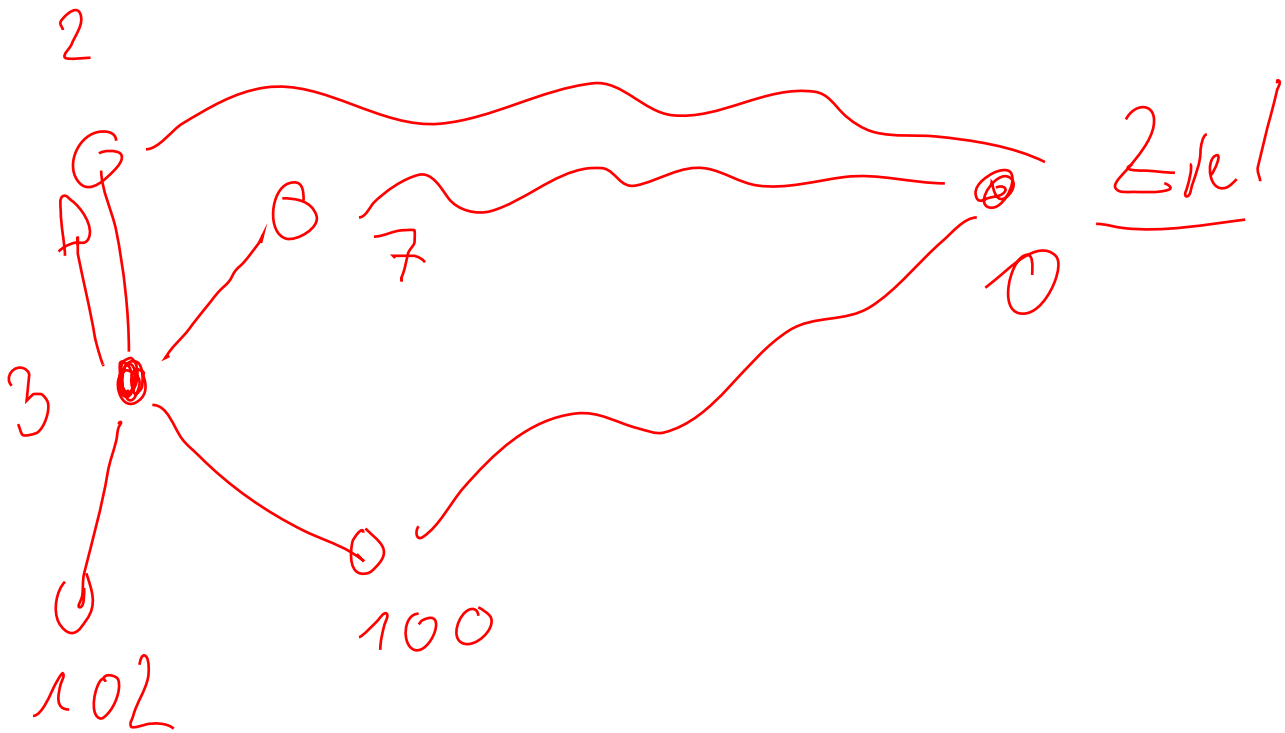
Init-Target(G, w, t)

- **Init-Target**(G, w)
- for all $v \in V$ do
  - $d(v) \leftarrow \infty$
  - $\pi(v) \leftarrow v$
- $d(t) \leftarrow 0$

Relax(u, v)

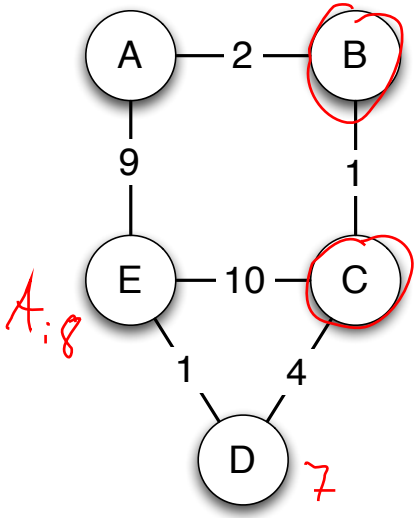
- **Relax**(u, v)
- if $d(u) > w(u, v) + d(v)$ then
  - $d(u) \leftarrow w(u, v) + d(v)$
  - $\pi(u) \leftarrow v$

UNI FREIBURG


2
G
A
B
7
0
Zvl
3
0
100
102

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
- Selbst Terminierend
  - läuft bis die Knoten keine Informationen mehr austauschen



Distance Table für A

von A	über B	über E	Routing Tabellen Eintrag
nach B	**2**	15	B
C	**3**	14	B
D	**7**	10	B
E	**8**	9	E

Distance Table für C

von C	über B	über D	über E	Routing Tabellen Eintrag
nach A	**3**	11	18	B
B	**1**	9	16	B
D	6	**4**	11	D
E	7	**5**	10	D

CoNe Freiburg

Distance Vector für ein Ziel t

Distributed Bellman Ford for target t (Distance-Vector Routing)

- If node is t then $d(t) \leftarrow 0$; $\pi(t) \leftarrow t$
- If a message from u to $\pi(u)$ fails then
  - $d(u) \leftarrow \infty$
- If u detects a new neighbor v then
  - send $(u, d(u))$ to v
- If u receives $(v, d(v))$ from v
  - if $d(u) > d(v) + w(u, v)$ or $v = \pi(u)$ then
    - $d(u) \leftarrow d(v) + w(u, v)$
    - $\pi(u) \leftarrow v$

Relax

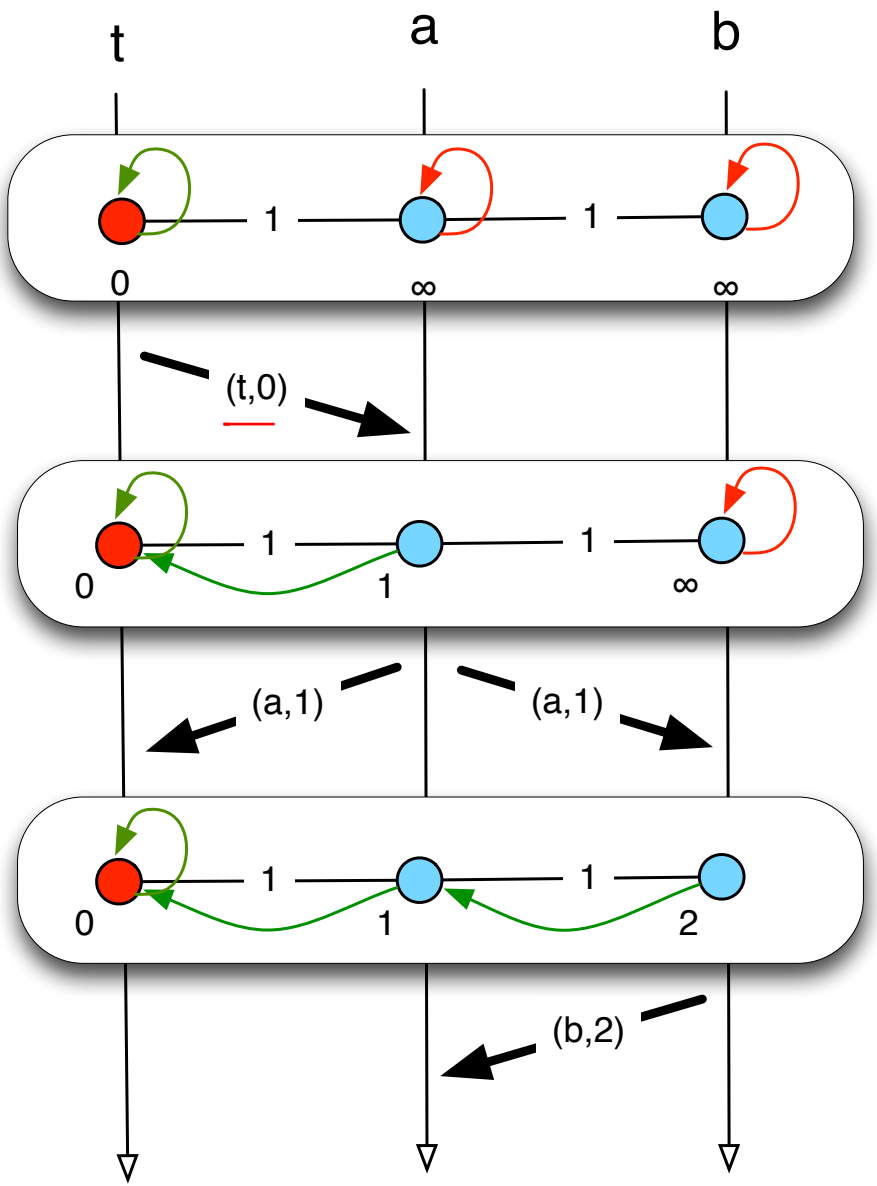
- if $d(u) = \infty$ then $\pi(u) \leftarrow u$
- Every time $d(u)$ or $\pi(u)$ has changed u sends $(u, d(u))$ to all neighbors

UNI FREIBURG

CoNe Freiburg

Beispiel für Distance-Vector für Ziel t

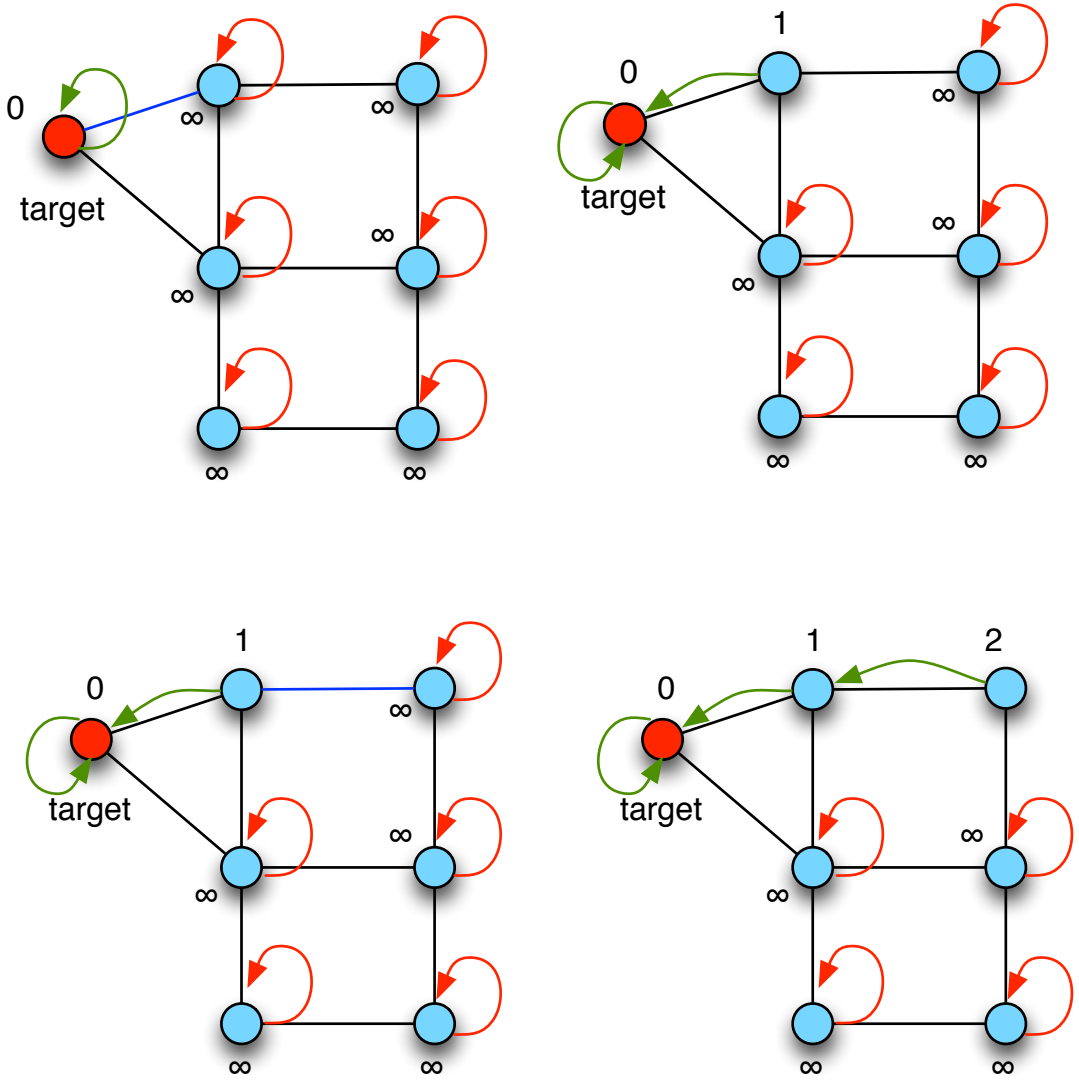
Adv.



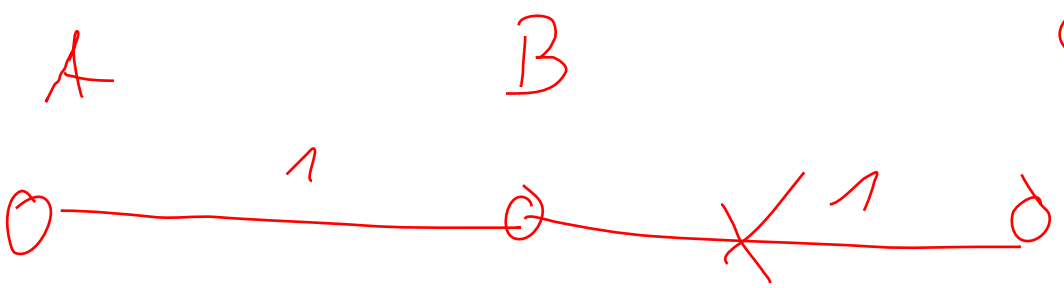
UNI FREIBURG

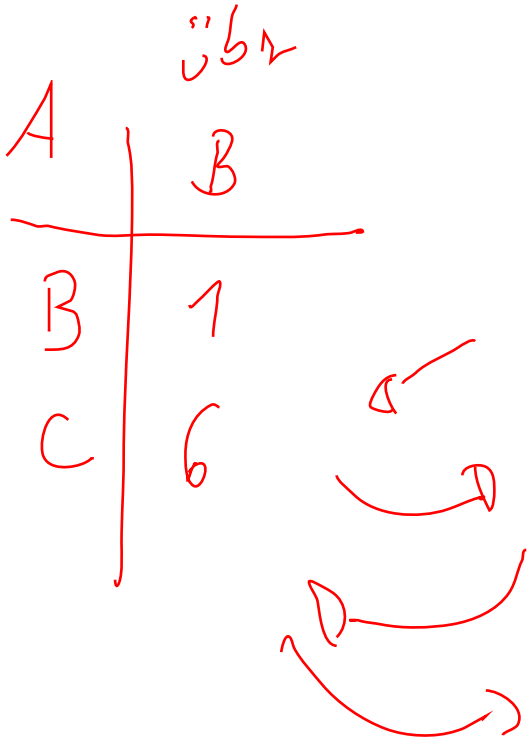
CoNe Freiburg

Distance-Vector für ein Ziel

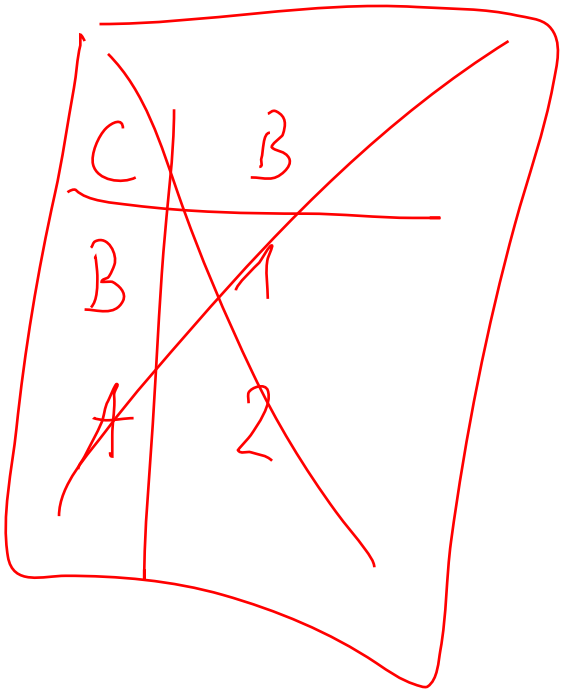


UNI FREIBURG


A
B
C
1
1

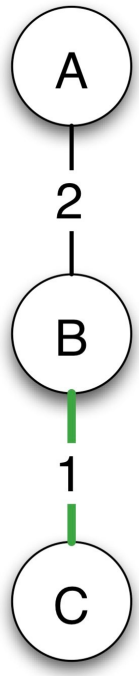

A
B
B
1
C
6


B
A
A
1
C
5


C
B
B
1
A
2

Das “Count to Infinity” - Problem

- Gute Nachrichten verbreiten sich schnell
  - Neue Verbindung wird schnell veröffentlicht



Distance Table für A

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

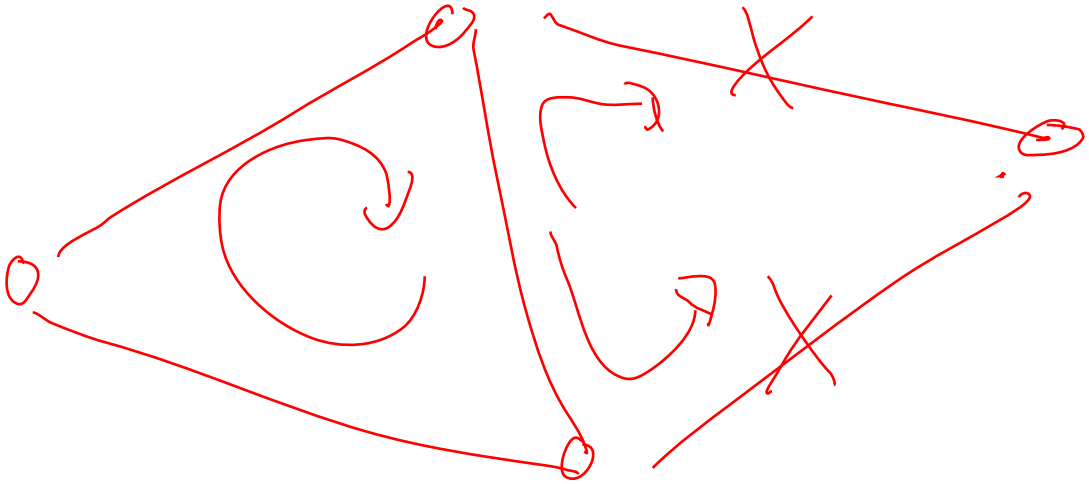
Distance Table für B

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

nach kurzer Zeit

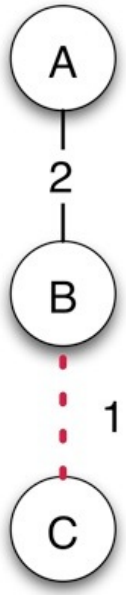
von A	über B	Routing Tabellen Eintrag
nach B	**2**	B
C	**3**	B

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



Das “Count to Infinity” - Problem

- Schlechte Nachrichten verbreiten sich langsam
  - Verbindung fällt aus
  - Nachbarn erhöhen wechselseitig ihre Entfernung
  - “Count to Infinity”-Problem



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

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

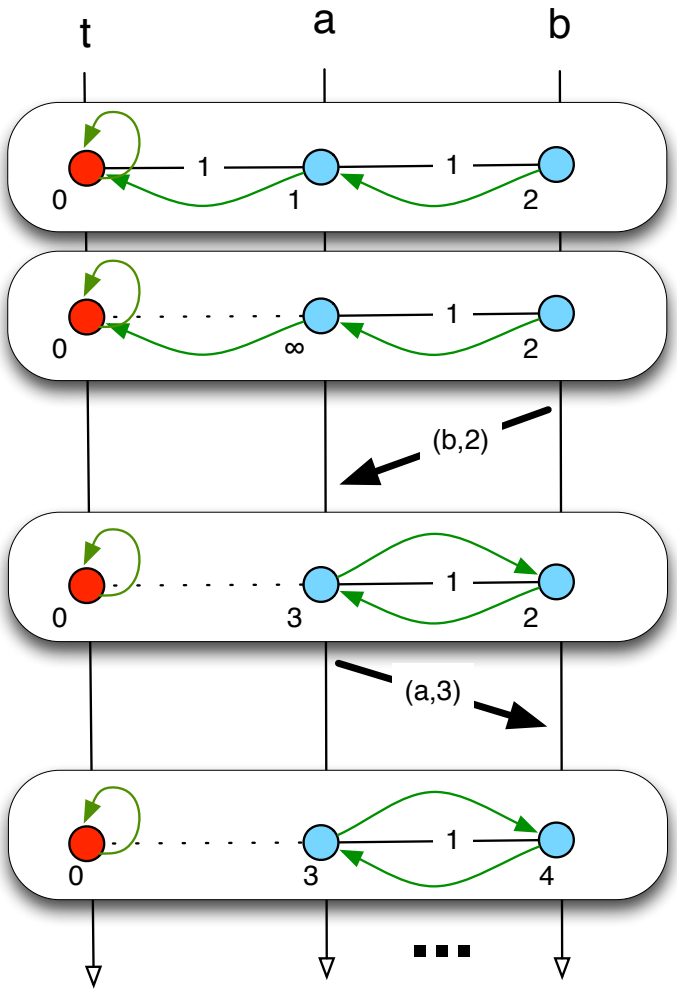
von A	nach	über B	Routing Tabellen Eintrag
	B	2	B
	C	7	B

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

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

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

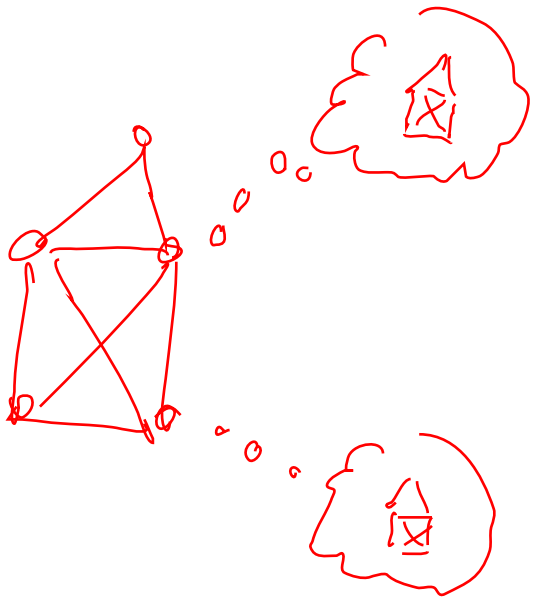
Das “Count to Infinity” - Problem für Ziel t



CoNe Freiburg

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



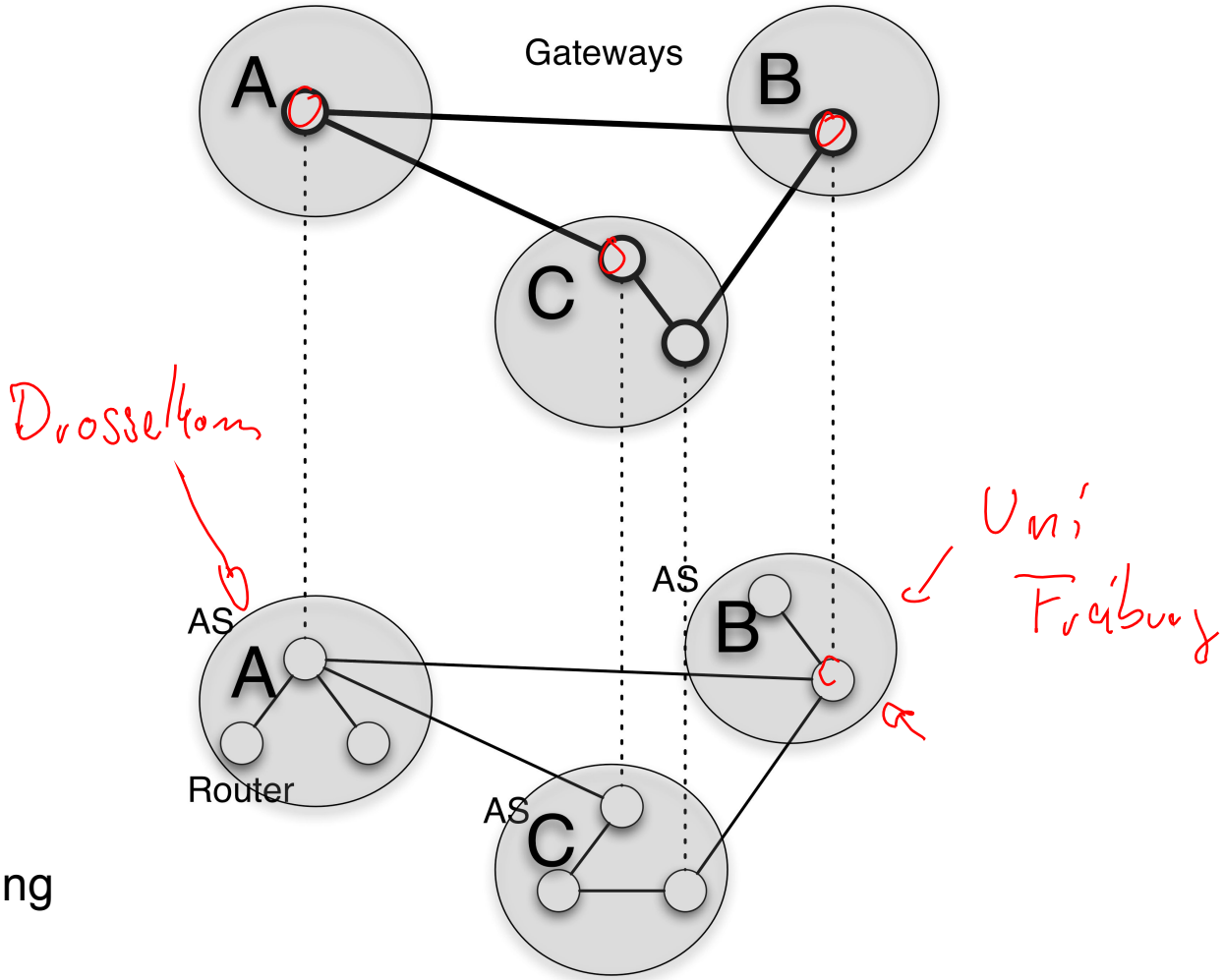
CoNe Freiburg

Die Grenzen des flachen Routing

- Link State Routing
  - benötigt O(g n) Einträge für n Router mit maximalen Grad g
  - Jeder Knoten muss an jeden anderen seine Informationen senden
- Distance Vector
  - benötigt O(g n) Einträge
  - kann Schleifen einrichten
  - Konvergenzzeit steigt mit Netzwerkgröße
- Im Internet gibt es mehr als 10^7 Router
  - damit sind diese so genannten flachen Verfahren nicht einsetzbar
- Lösung:
  - Hierarchisches Routing

AS, Intra-AS und Inter-AS

- Autonomous System (AS)
  - liefert ein zwei Schichten-Modell des Routing im Internet
  - Beispiele für AS:
    - uni-freiburg.de
- Intra-AS-Routing (Interior Gateway Protocol)
  - ist Routing innerhalb der AS
  - z.B. RIP, OSPF, IGRP, ...
- Inter-AS-Routing (Exterior Gateway Protocol)
  - Übergabepunkte sind Gateways
  - ist vollkommen dezentrales Routing
  - Jeder kann seine Optimierungskriterien vorgeben
  - z.B. EGP (früher), BGP



CoNe Freiburg

Typen autonomer Systeme

- Stub-AS
  - Nur eine Verbindung zu anderen AS
- Multihomed AS
  - Verbindungen zu anderen ASen
  - weigert sich aber Verkehr für andere zu befördern
- Transit AS
  - Mehrere Verbindungen
  - Leitet fremde Nachrichten durch (z.B. ISP)