Proseminar

Routing Information Protocol

Open Shortest Path First

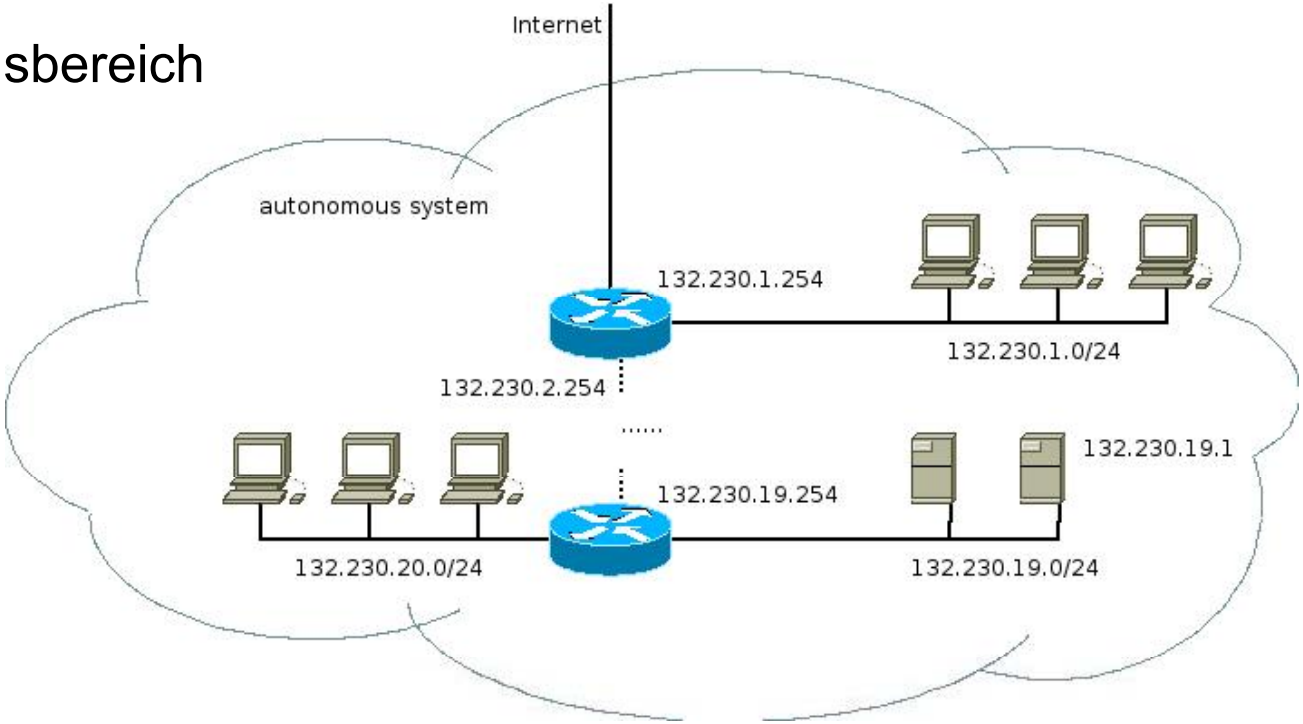
Martin Bauer

Gliederung

- Grundlagen
- Router Information Protocol (RIP)
  - Bellman-Gleichung
  - Bellman-Ford-Algorithmus
  - Spezifikation
  - Count-to-Infinity
- Open Shortest Path First (OSPF)
  - Dijkstra Algorithmus
  - Spezifikation
- Zusammenfassung

Autonomes System

- Zusammenschluss zu einem logischen Netzwerk
- Verwaltung durch **Interior Gateway Protocols** (IGP)
- Verantwortungsbereich einer einzigen Organisation



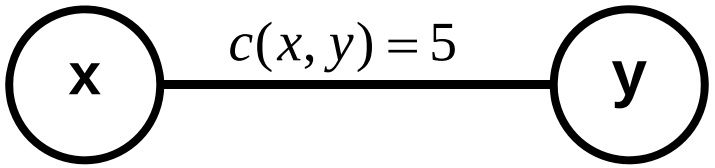
Routing Tabelle

- Ermittlung von Ziel-Host/-Router zu IP-Adresse
- dynamische vs. statische Einträge

Destination	Netmask	Gateway	Interface	Metric
132.230.1.0	255.255.255.0	132.230.18.254	eth1	12
...	...	...	…	...
132.230.18.0	255.255.255.0	132.230.18.254	eth1	1
132.230.19.0	255.255.255.0	*	eth0	0
132.230.20.0	255.255.255.0	*	eth0	0
0.0.0.0	0.0.0.0	132.230.18.254	eth2	0

Kürzeste Pfade in Graphen

- Netzwerke werden als Graphen interpretiert
- $G = (V, E)$ ist ein Graph mit $x, y \in V$ und $(x, y) \in E$
- Kanten sind ungerichtet $(x, y) = (y, x)$
- Kanten besitzten Kostenfunktionen: $c : E \rightarrow R^{+}$
- Kürzester Pfad ? Bellman-Ford-Algorithmus.

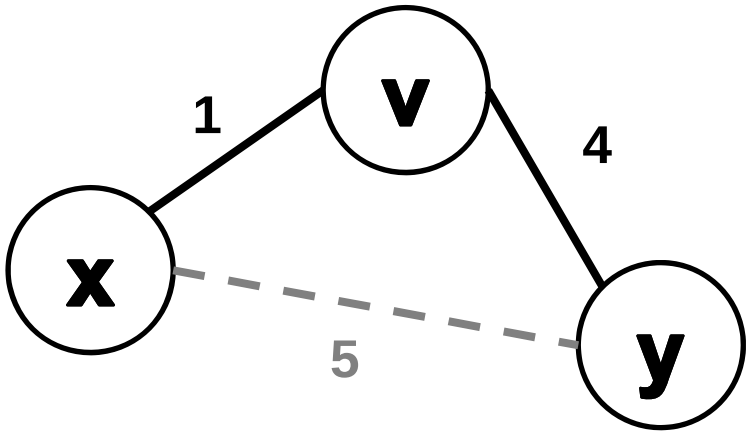


Bellman-Gleichung

- kürzeste Distanz von Startknoten x zu Ziel y über Nachbarknoten v
- Distanzkosten: $D_x(y) = \min_v \{c(x,v) + d_v(y)\}$

- Beispiel: $D_x(y) = \min\{1+4\}$

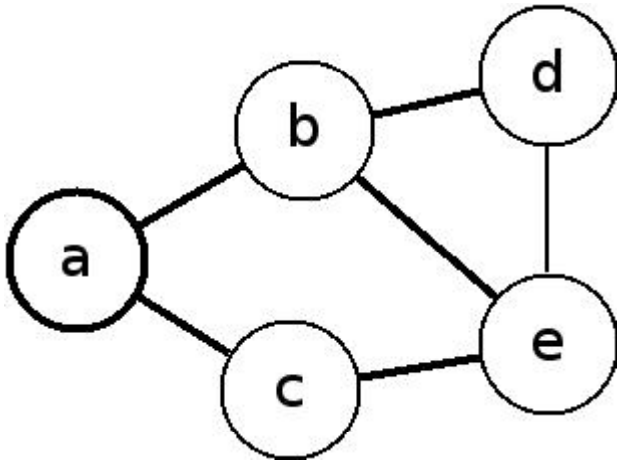
  $D_x(y) = 5$



Distanzvektoren

- Distanz-Vector Routing Algorithmen
  - Knoten kennen nur einen Teil des Netzwerks
  - Knoten lernen stetig hinzu
- Erweiterung der Bellman-Gleichung um Distanzvektor:

$$D_v = \lfloor D_v(y) : y \; in \; V \rfloor$$



Bellman-Ford-Algorithmus (0)

Initialization:

for all destination y in V:

$D_x(y) = c(x,y)$ /* if y is not neighbour than $c(x,y) = \infty$ /*

for each neighbour v

$D_v(y) = \infty$ for all destinations y in V

for each neighbour v

send distance vector $D_x = \lfloor D_x(y) : y \; in \; V \rfloor$ to v

Loop

wait (until I see link cost change to some neighbour v or
until I receive a distance vector from some neighbour v)

for each y in V:

$D_x(y) = \min_v \{c(x,v) + D_v(y)\}$

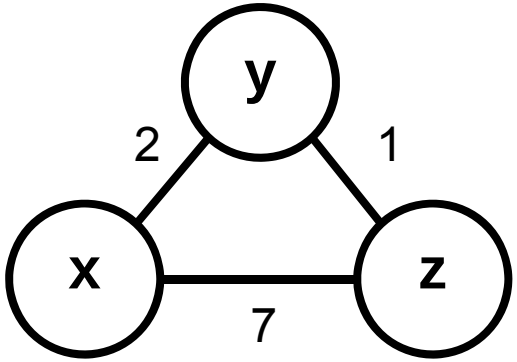
if D_x(v) changed for any destination y

send distance vector $D_x = \lfloor D_x(y) : y \; in \; V \rfloor$ to all neighbours

forever

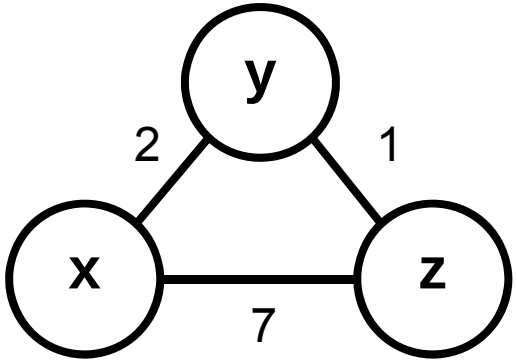
Bellman-Ford-Algorithmus (1)

Beispiel:



Bellman-Ford-Algorithmus (2)

Beispiel: Initialisierungsphase



x in t_0

von \ nach	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

y in t_0

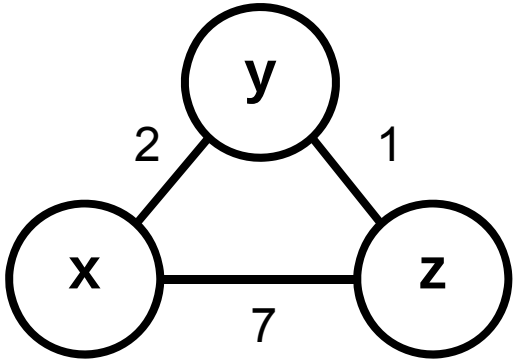
von \ nach	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

z in t_0

von \ nach	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

Bellman-Ford-Algorithmus (3)

Beispiel:



$$D_x(z) = \min\left\{\underbrace{2+1}_{3}, 7+0\right\}$$

$$D_z(x) = \min\left\{7+0, \underbrace{2+1}_{3}\right\}$$

x in t_0

von \ nach	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

y in t_0

von \ nach	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

z in t_0

von \ nach	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

x in t_1

von \ nach	x	y	z
x	0	2	**3**
y	2	0	1
z	7	1	0

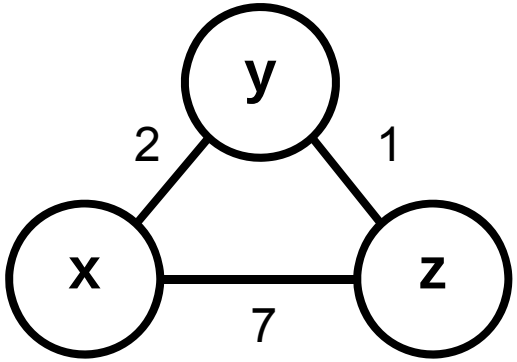
y in t_1

von \ nach	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

z in t_1

von \ nach	x	y	z
x	0	2	7
y	2	0	1
z	**3**	1	0

Beispiel:



x in t_0

von \ nach	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

y in t_0

von \ nach	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

z in t_0

von \ nach	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

x in t_1

von \ nach	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

y in t_1

von \ nach	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

z in t_1

von \ nach	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

x in t_2

von \ nach	x	y	z
x	0	2	3
y	2	0	1
z	**3**	1	0

y in t_2

von \ nach	x	y	z
x	0	2	**3**
y	2	0	1
z	**3**	1	0

z in t_2

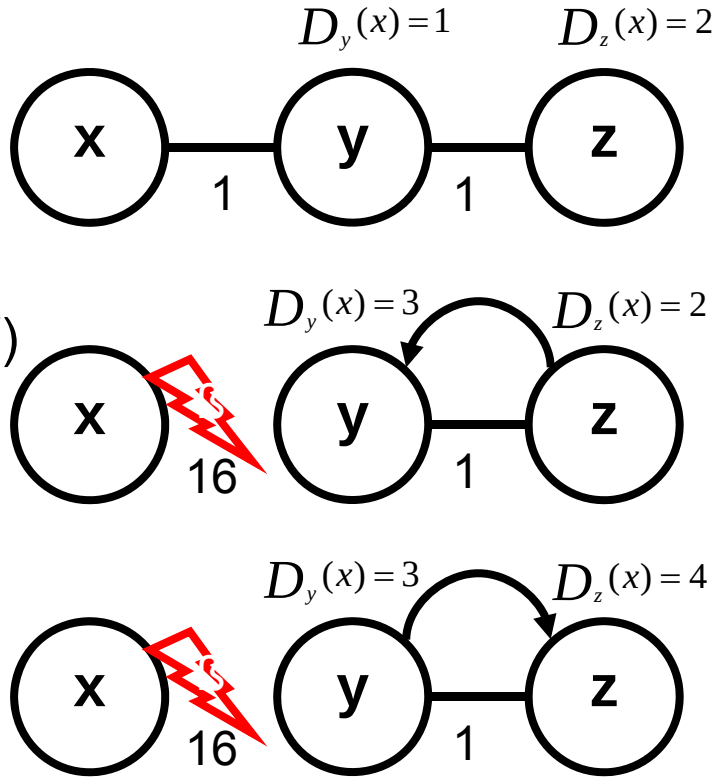
von \ nach	x	y	z
x	0	2	**3**
y	2	0	1
z	3	1	0

Routing Information Protocol

- Definition in RFC1058
- Kostenfunktion zu Nachbarn konstant $c(x, y) = 1$
- Maximal zulässige Kosten = 15 (16 Infinität)
  - Beschränkung in Netzgröße auf Durchmesser 15
- 30 sekündliche Austausch der Distanzvektoren
  - 7,5 Minuten für komplette Netzaktualisierung (=Konvergenz)

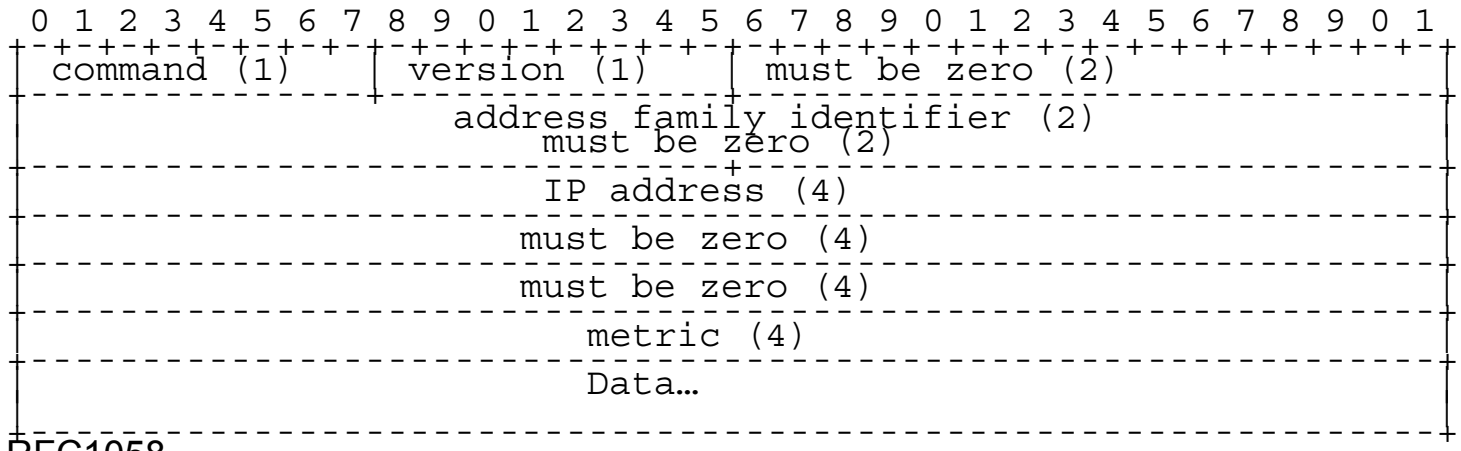
Count-to-Infinity

- Problem: Count-to-Infinity
- Lösung: Triggered Updates
  - Änderungen sofort mitteilen
- Lösung: Split-Horizont-Verfahren (SHV)
  - Pfadinformation darf nicht über das gleiche Interface gesendet werden, über dass es erlernt wurde.
- Lösung: SHV mit Poisoned Reverse
  - Reset der Verbindung auf beiden Knoten und lerne neu



Anatomie eines RIP-Datagram

- UDP/520
- Command = {request|response}
- IP Adresse = Zieladresse
- Version = {1|2}



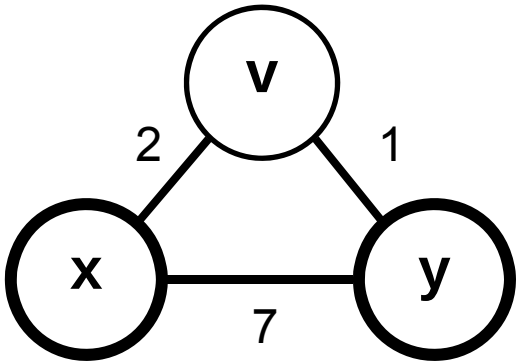
RFC1058

Eigenschaften von RIP

- geringer Rechenleistung notwendig
- interoperabel
- ungenügende Skalierbarkeit (max. Hops)
- mangelhafte Konvergenzeigenschaften
- unzureichende Authentifikation (optional Klartextpassworte)

Open Shortest Path First Protocol

- gehört zur Klasse der Link-State Routing Algorithmen
  - jeder Knoten kennt vollständiges Netzwerk
  - jeder Knoten sendet vollständige Routingtabelle an alle anderen
- Grundlage ist Algorithmus von Dijkstra
- Prinzip: $D_y = \min\left(D(y), D(v) + c(v,y)\right)$



Dijkstra-Algorithmus

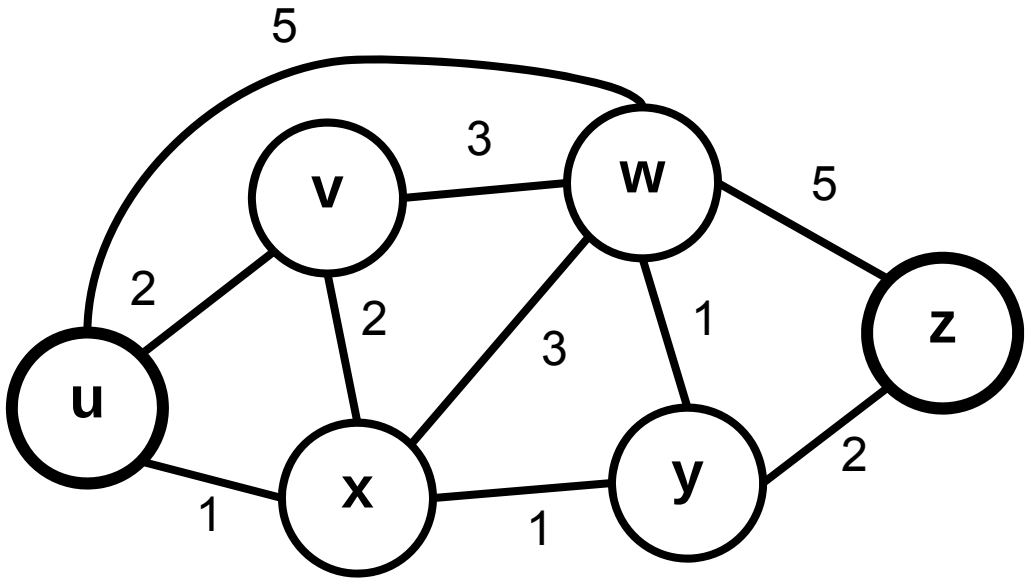
```
1.   Initialization
2.   N' = {x}
3.     for all nodes v
4.         if v is a neighbour of x
5.             then D(v) = c(x,v)
6.         else D(v) = infinity

7.   Loop
8.     find v not in N' such that D(v) is a minimum
9.     add v to N'
10.    update D(v) for each neighbour v of w and not in N':
11.        D(y) = min(D(y), D(v) + c(v,y))
12.    /* new cost to y is ether old cost to y or known
13.    least path cost to v plus cost from v to y */
14.  until N' = N
```

Dijkstra-Algorithmus (0)

Beispiel:

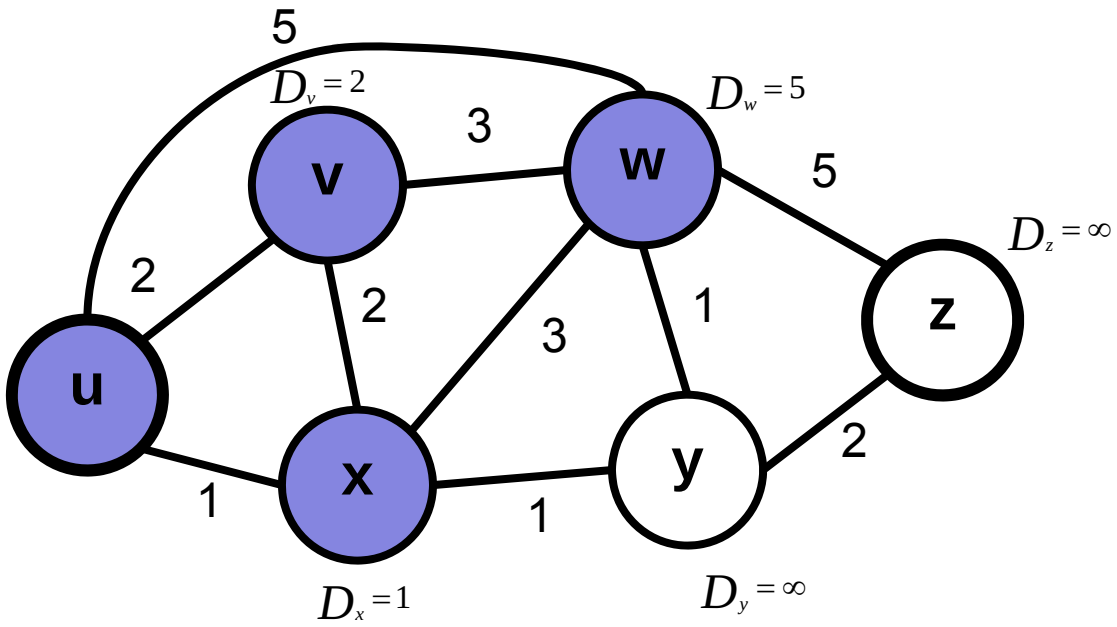
$$N = \{u, v, w, x, y, z\}$$



Dijkstra-Algorithmus (1)

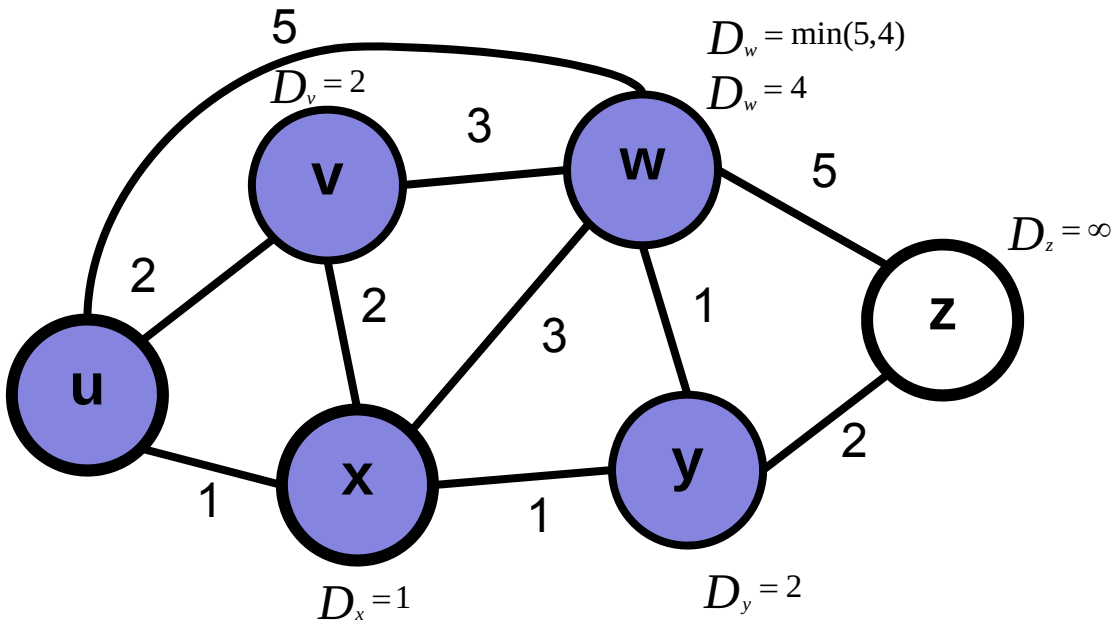
Beispiel: Initialisierungsphase

$$N = \{x, v, w, y, z\}$$
$$N' = \{u\}$$



Dijkstra-Algorithmus (2)

Beispiel:

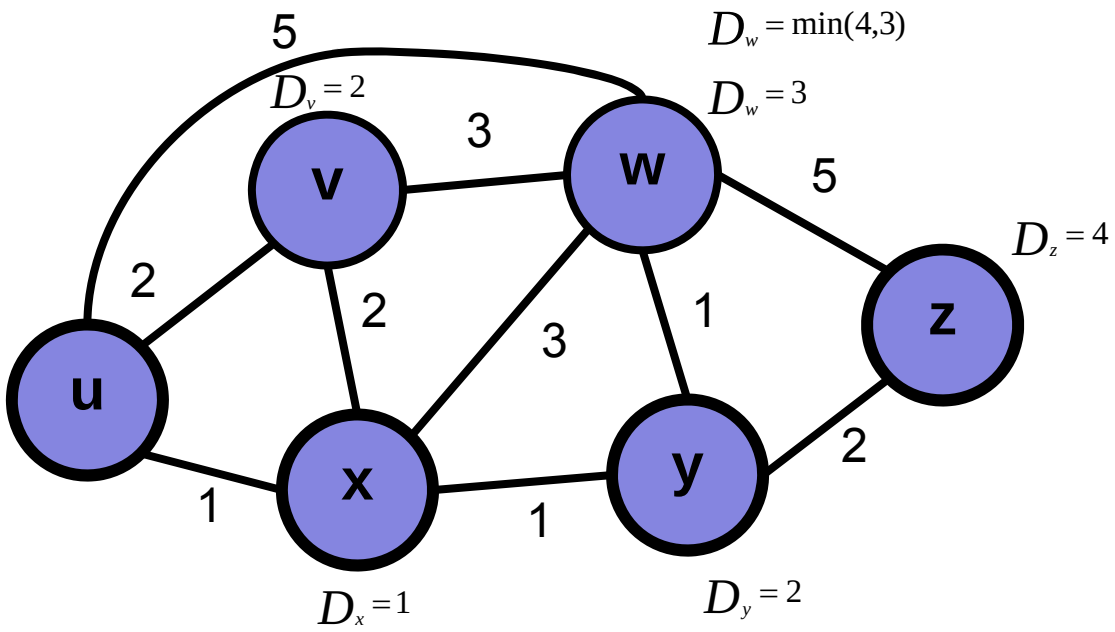


$N = \{v, w, y, z\}$

$N' = \{u, x\}$

Dijkstra-Algorithmus (3)

Beispiel:

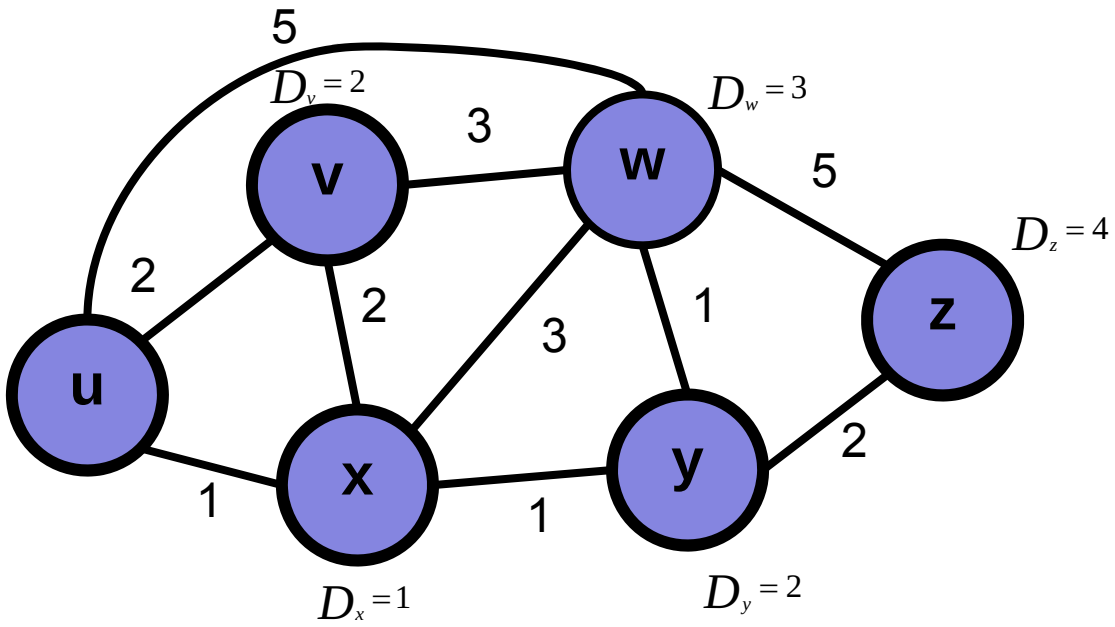


$$N = \{v, w, z\}$$

$$N' = \{u, x, y\}$$

Dijkstra-Algorithmus (4)

Beispiel:

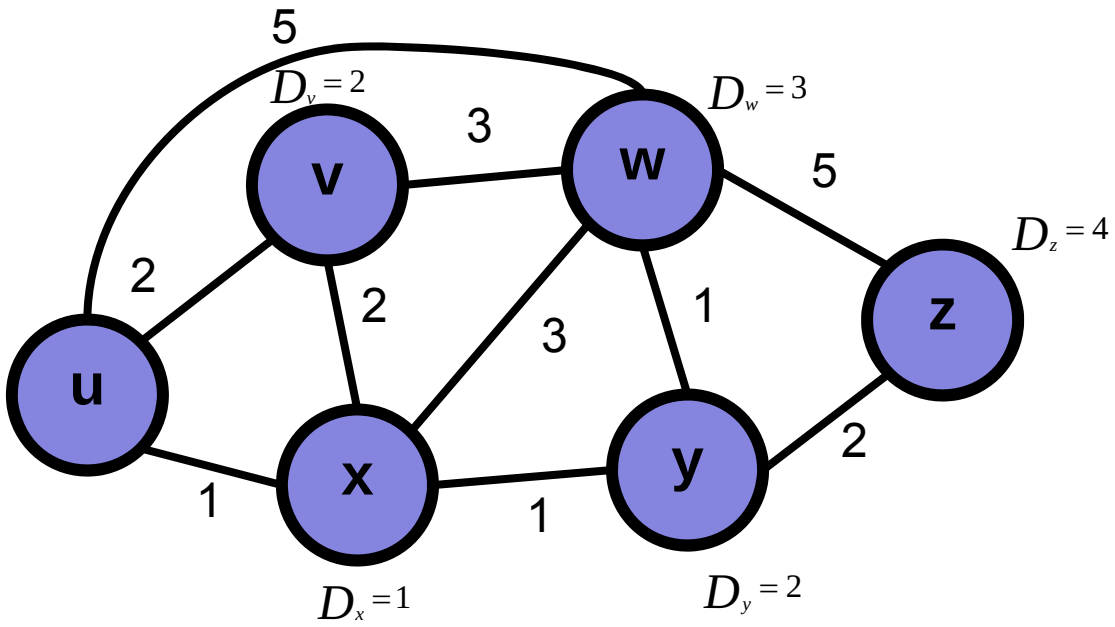


$N = \{w, z\}$

$N' = \{u, x, y, v\}$

Dijkstra-Algorithmus (5)

Beispiel:

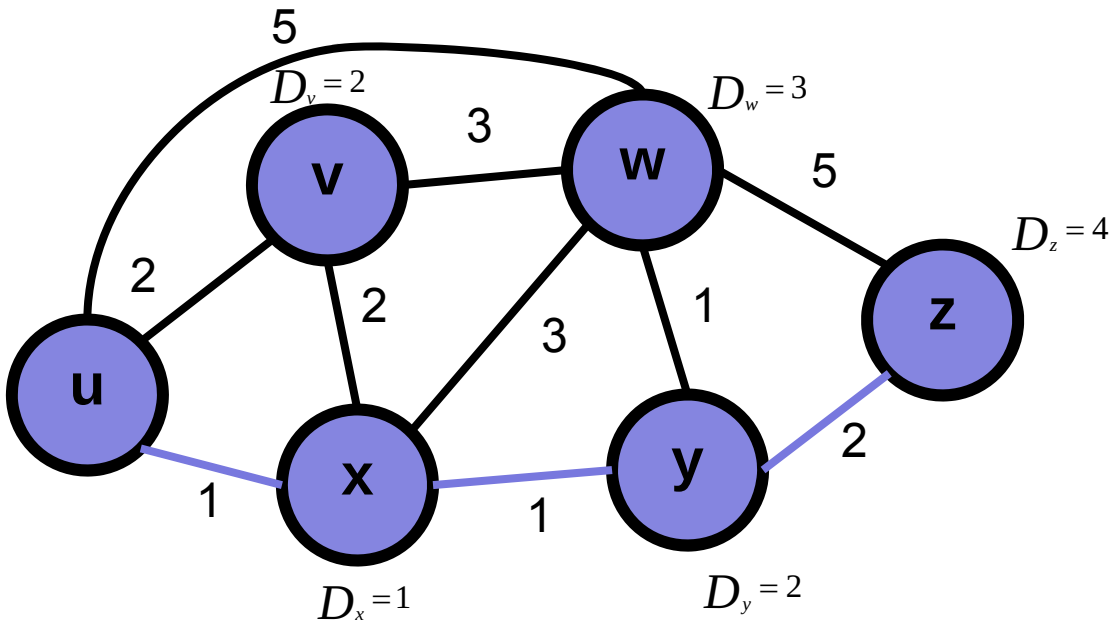


$N = \{z\}$

$N' = \{u, x, y, v, w\}$

Dijkstra-Algorithmus (6)

Beispiel:

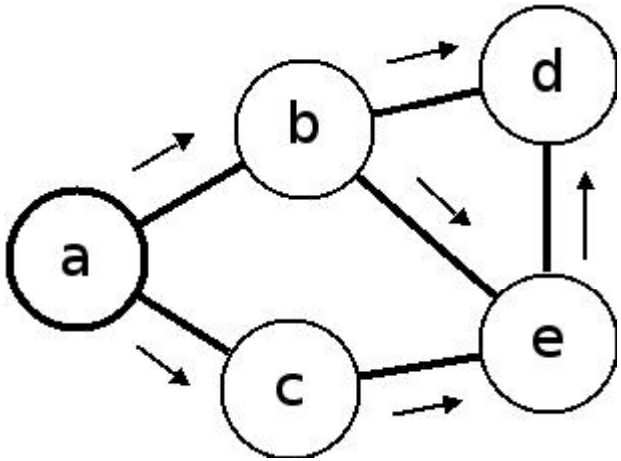


$N = \{\ \}$

$N' = \{u, x, y, v, w, z\}$

Open Shortest Path First

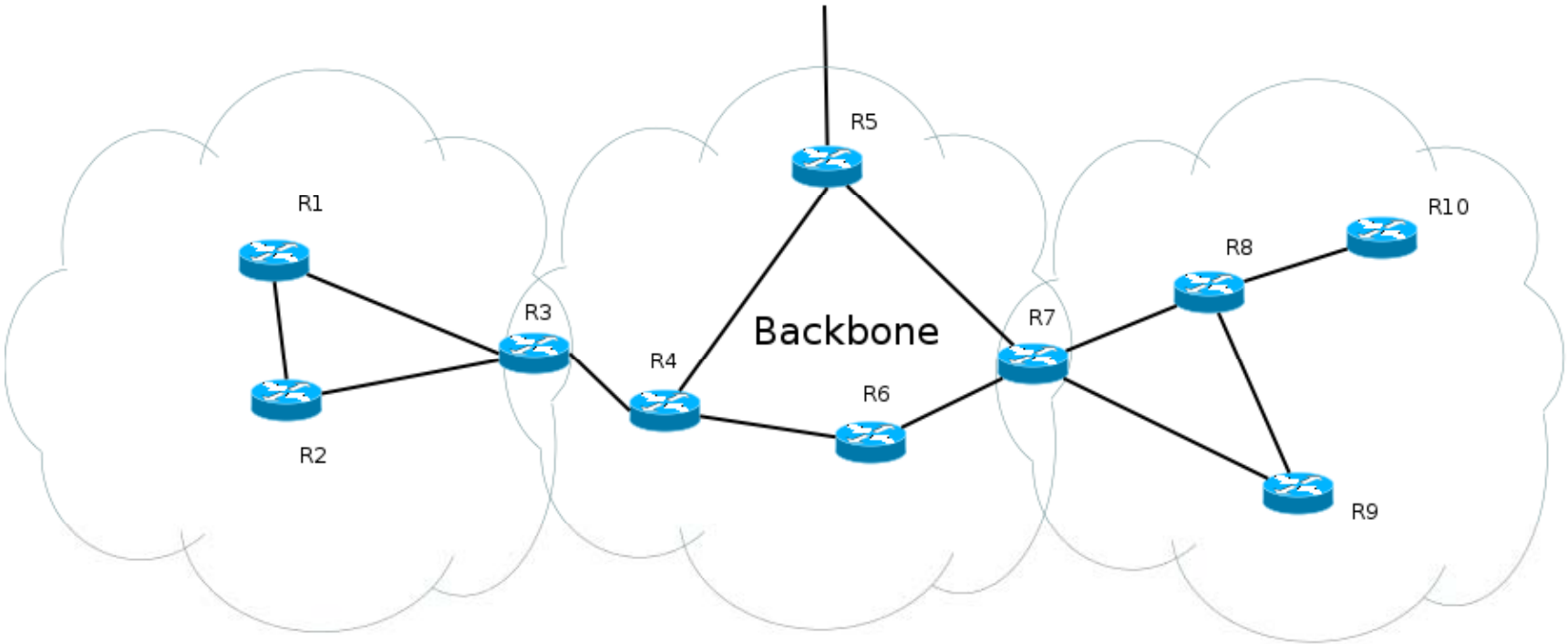
- variable Kostenfunktion
- Überwachung der Kosten zu Nachbarn (Link-State)
  durch HELLO Pakete
- Datenaustausch regelmässig
  durch (LSAdvertisment)
- Änderungen sofort übertragen
  (LSAnnounce)
- Implementierung durch eigenes
  Level-3 Protokoll (ID 89)



Features OSPF

- Multicast
  - eine Nachricht an mehrere Empfänger
- Sicherheit
  - Authentifizierung
- Designierter Router (DR)
  - ermöglicht zentrale Verteilung
- Topologie

Topologie



R1, R2, R8, R9, R10 - Internal Router
R3, R7 - Area Boarder Router (ABR)
R3, R4, R5, R6, R7 - Backbone Router (BR)
R5 - Autonomous System Boundary Router (ASBR)

Literaturhinweise

Computer Networking, A Top-Down Approach Featuring the Internet,
von James F. Kurose, Keith W. Ross

Algorithmen und Datenstrukturen, Thomas Ottmann, Peter Widmayer

RFC1058 RIP http://tools.ietf.org/html/rfc1058

RFC2453 RIPv2 http://tools.ietf.org/html/rfc2453

RFC2328 OSPF http://tools.ietf.org/html/rfc2328

On a Routing Problem in Quarterly of Applied Mathematics, R. E. Bellman
16(1)/1958. Brown University

Network flow theory, L. R. Ford
Paper P-923. The Rand Corporation, Santa Monica 1956

Zusammenfassung & Fragen

Vielen Dank für die Aufmerksamkeit