

Nätverksteknik B - Dynamisk Routing

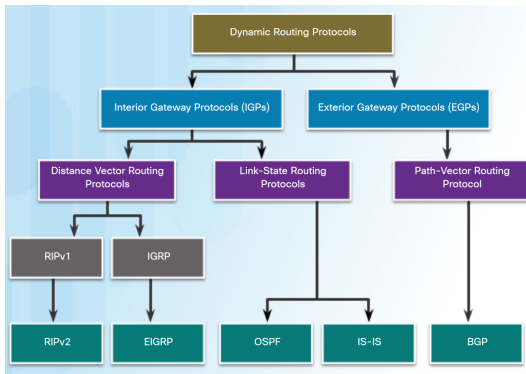
Lennart Franked

Informationssystem och -teknologier (IST)
Mittuniversitetet

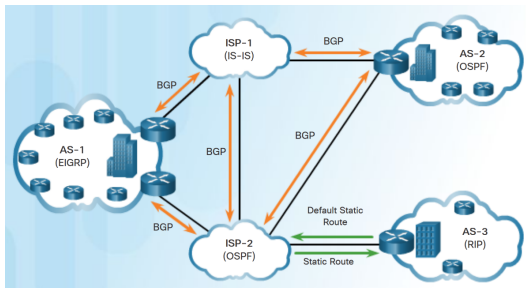
28 januari 2021

- Förklara funktionalitet och karaktärsdrag för dynamiska routingprotokoll
- Jämföra routingprotokoll
- Förklara hur distance-vector routing fungerar.
 - ▶ Bellman-Ford
- Förklara hur link-state routing fungerar.
 - ▶ Dijkstras algoritm.

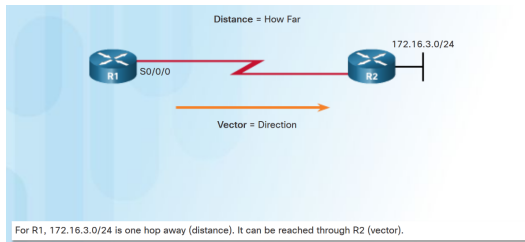
Klassificering av Routingprotokoll



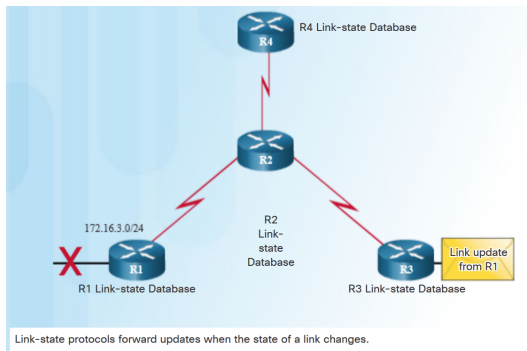
Figur: Klassificering av routingprotokoll [2017ScN]



Figur: IGP och EGP [2017ScN]



Figur: Distance Vector [2017ScN]



Figur: Link State [2017ScN]

	Distance Vector				Link State	
	RIPv1	RIPv2	IGRP	EIGRP	OSPF	IS-IS
Speed of Convergence	Slow	Slow	Slow	Fast	Fast	Fast
Scalability - Size of Network	Small	Small	Small	Large	Large	Large
Use of VLSM	No	Yes	No	Yes	Yes	Yes
Resource Usage	Low	Low	Low	Medium	High	High
Implementation and Maintenance	Simple	Simple	Simple	Complex	Complex	Complex

Figur: Jämnförelse IGP [2017ScN]

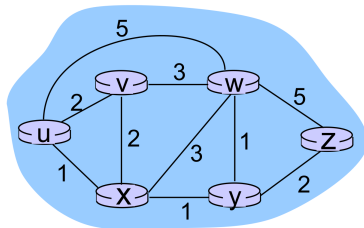
- Mekanism för att skicka och ta emot routing information.
- Mekanism för att beräkna bästa väg och installera en route.
- Mekanism för att upptäcka förändringar i topologin.
- RIP använder Bellman-Ford
- EIGRP använder DUAL.

$d_x(y) :=$ kostnad för den väg som har lägst kostnad mellan x och y

$$d_x(y) = \min_v \{c(x, v) + d_v(y)\}$$

$\min_v$ kollar alla grannar(v) till x
 $c(x, v)$ kostnaden mellan x och v
 $d_v(y)$ kostnaden mellan v och y

$$\begin{aligned}d_u(z) &= \min\{c(u, v) + d_v(z), \\&\quad c(u, x) + d_x(z), \\&\quad c(u, w) + d_w(z)\} \\&= \min\{2 + 5, \\&\quad 1 + 3, \\&\quad 5 + 3\} = 4\end{aligned}$$



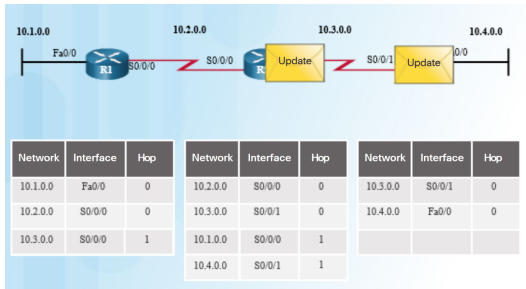
Figur: Bellman-Ford
example [Kurose2013cna]

$D_x(y) :=$ Uppskattad lägsta kostnad från x till y

- Noden x underhåller en lista $D_x = [D_x(y) : y \in N]$
- Detta innebär att:
 - ▶ x måste känna till kostnaden till alla sina grannar v : $c(x, v)$
 - ▶ För varje granne v måste x känna till dess distansvektor $D_v = [D_v(y) : y \in N]$

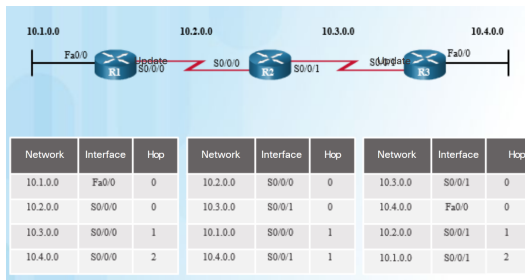
- Alla noder periodiskt utbyta sina distansvektorer
- Varje gång en nod tar emot en ny distansvektor så uppdaterar den sin routingtabell.
- $D_x(y) \leftarrow \min_v \{c(x, v) + D_v(y)\}$ för varje nod $y \in N$

RIP Exempel



Figur: RIP exempel [2017ScN]

RIP Exempel



Figur: RIP exempel [2017ScN]

- Bellman-Ford är lämplig att använda på grund utav att hela routingtabeller utbyts.
- Känslig för loopar.
- Vet ej hur topologin ser ut.
- EIGRP använder sig utav DUAL (Återkommer till denna algoritm)

Link State Dynamic Routing

- Kostnaden mellan alla noder är känd.
- Beräknar lägsta kostnaden till att nå en destination utifrån sig själv.
- Iterativ algoritm: Efter k iterationer så vet den lägsta kostnaden till k destinationer.

- $c(x, y)$ kostnaden från x till y . default = ∞
- $D(v)$ nuvarande värde för att nå nod v .
- $p(v)$ Föregående nod på vägen till v
- N' Set av noder vars lägsta kostnad för att nå är känd

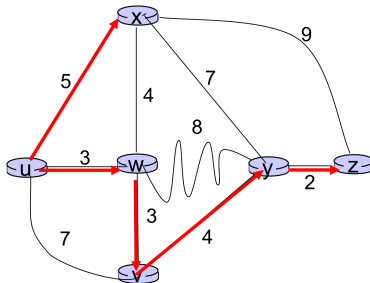
Dijkstras Algorithm

```
Initialization:
N' = {x}
For all nodes v:
    if v adjacent to x:
        D(v) = c(x,v)
    else:
        D(v) =  $\infty$ 

Loop until all nodes are added to N':
    Find w that not in N', such that D(w) is a minimum
    Add w to N'
    Update D(v) for all v adjacent to w and not in N':
        D(v) = min(D(v), D(w) + c(w,v))
```

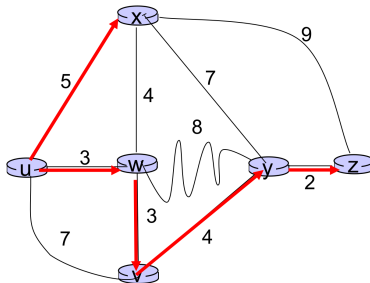
Listing 1: Dijkstras algorithm [Kurose2013cna]

Steg	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvzy					



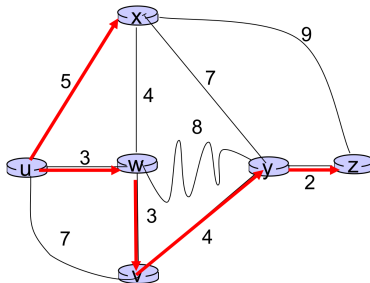
Figur: Topologi dijkstra exempel [Kurose2013cna]

Steg	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					



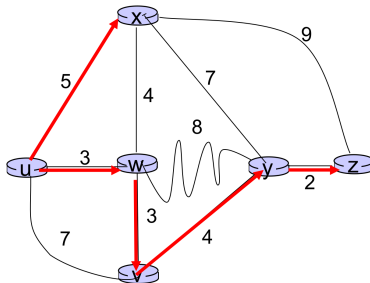
Figur: Topologi dijkstra exempel [Kurose2013cna]

Steg	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwvx	6,w			11,w	14,x
3	uwxcv				10,v	14,x
4	uwxcvy					12,y
5	uwxcvyz					



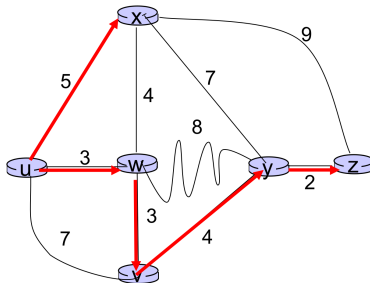
Figur: Topologi dijkstra exempel [Kurose2013cna]

Steg	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwvx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					



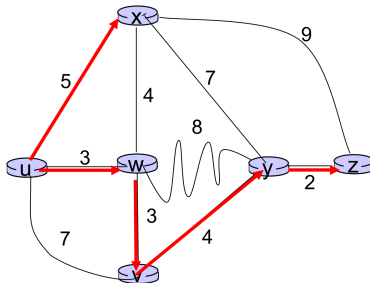
Figur: Topologi dijkstra exempel [Kurose2013cna]

Steg	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwvx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					



Figur: Topologi dijkstra exempel [Kurose2013cna]

Steg	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uw x	6,w			11,w	14,x
3	uw xv				10,v	14,x
4	uw xvy					12,y
5	uw xvzy					



Figur: Topologi dijkstra exempel [Kurose2013cna]

Link-state

- Varje router lär sig om sina direkt anslutna nätverk.
- Söker kontakt med andra routrar med 'Hello-paket'
- Konstruerar ett Link State Paket (LSP) innehållandes information om direkt anslutna nätverk.
- Skickar ut detta paket till sina grannar, som lagrar informationen i deras Link State Database (LSDB)
- Utifrån informationen i databasen skapas en karta över nätverkets topologi.
- Bästa route till varje subnät beräknas och installeras i routingtabellen.

Link-state

- Varje router lär sig om sina direkt anslutna nätverk.
- Söker kontakt med andra routrar med 'Hello-paket'
- Konstruerar ett Link State Paket (LSP) innehållandes information om direkt anslutna nätverk.
- Skickar ut detta paket till sina grannar, som lagrar informationen i deras Link State Database (LSDB)
- Utifrån informationen i databasen skapas en karta över nätverkets topologi.
- Bästa route till varje subnät beräknas och installeras i routingtabellen.

Link-state

- Varje router lär sig om sina direkt anslutna nätverk.
- Söker kontakt med andra routrar med 'Hello-paket'
- **Konstruerar ett Link State Paket (LSP) innehållandes information om direkt anslutna nätverk.**
- Skickar ut detta paket till sina grannar, som lagrar informationen i deras Link State Database (LSDB)
- Utifrån informationen i databasen skapas en karta över nätverkets topologi.
- Bästa route till varje subnät beräknas och installeras i routingtabellen.

Link-state

- Varje router lär sig om sina direkt anslutna nätverk.
- Söker kontakt med andra routrar med 'Hello-paket'
- Konstruerar ett Link State Paket (LSP) innehållandes information om direkt anslutna nätverk.
- Skickar ut detta paket till sina grannar, som lagrar informationen i deras Link State Database (LSDB)
- Utifrån informationen i databasen skapas en karta över nätverkets topologi.
- Bästa route till varje subnät beräknas och installeras i routingtabellen.

Link-state

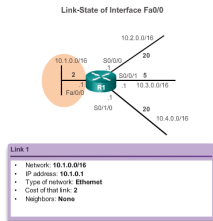
- Varje router lär sig om sina direkt anslutna nätverk.
- Söker kontakt med andra routrar med 'Hello-paket'
- Konstruerar ett Link State Paket (LSP) innehållandes information om direkt anslutna nätverk.
- Skickar ut detta paket till sina grannar, som lagrar informationen i deras Link State Database (LSDB)
- Utifrån informationen i databasen skapas en karta över nätverkets topologi.
- Bästa route till varje subnät beräknas och installeras i routingtabellen.

Link-state

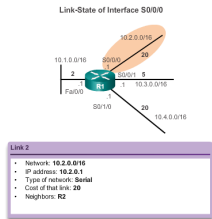
- Varje router lär sig om sina direkt anslutna nätverk.
- Söker kontakt med andra routrar med 'Hello-paket'
- Konstruerar ett Link State Paket (LSP) innehållandes information om direkt anslutna nätverk.
- Skickar ut detta paket till sina grannar, som lagrar informationen i deras Link State Database (LSDB)
- Utifrån informationen i databasen skapas en karta över nätverkets topologi.
- Bästa route till varje subnät beräknas och installeras i routingtabellen.

Direkt anslutna nätverk

Link State Dynamic Routing



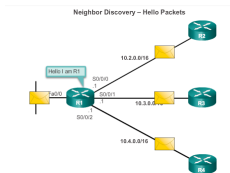
Figur: Länk 1 [2014RSE]



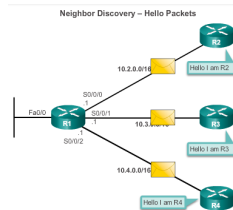
Figur: Länk 2 [2014RSE]

Hello

Link State Dynamic Routing



Figur: Fråga [2014RSE]

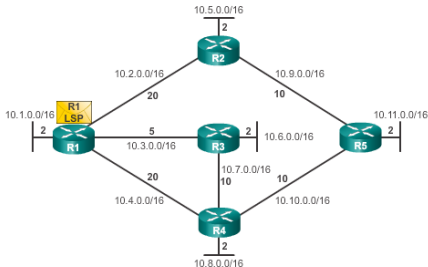


Figur: Svar 2 [2014RSE]

Generera LSP

Link State Dynamic Routing

Building the LSP



Figur: Generera LSP [2014RSE]

R1; Ethernet network 10.1.0.0/16; cost 2

R1 → R2; Serial point-to-point network; 10.2.0.0/16; Cost 20

R1 → R3; Serial point-to-point network; 10.3.0.0/16; Cost 5

R1 → R4; Serial point-to-point network; 10.4.0.0/16; Cost 20

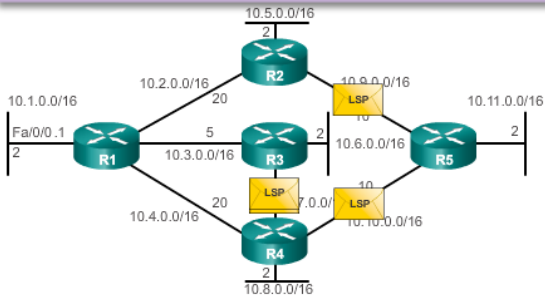
Skicka ut LSP till alla grannar

Link State Dynamic Routing

Flooding the LSP

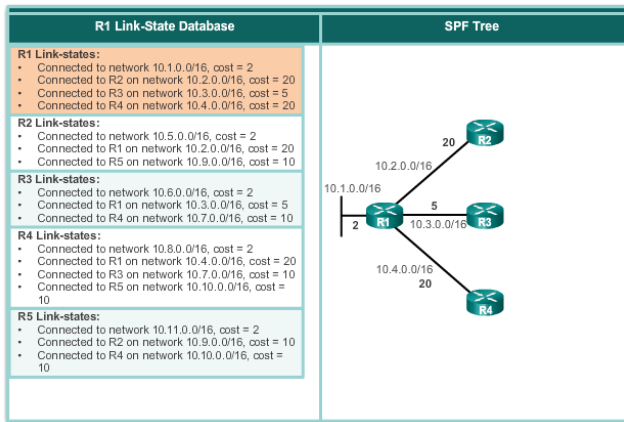
R1 Link State Contents

- R1; Ethernet network; 10.1.0.0/16; Cost 2
- R1 -> R2; Serial point-to-point network; 10.2.0.0/16; Cost 20
- R1 -> R3; Serial point-to-point network; 10.3.0.0/16; Cost 5
- R1 -> R4; Serial point-to-point network; 10.4.0.0/16; Cost 20



Figur: Flood LSP [2014RSE]

Identify the Directly Connected Networks



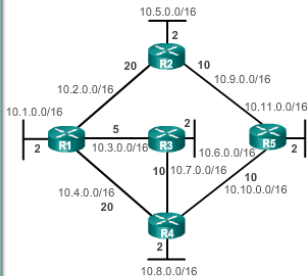
Figur: SPF-träd [2014RSE]

Skapa ett SPF träd 2

Link State Dynamic Routing

Resulting SPF Tree of R1

Destination	Shortest Path	Cost
10.5.0.0/16	R1 → R2	22
10.6.0.0/16	R1 → R3	7
10.7.0.0/16	R1 → R3	15
10.8.0.0/16	R1 → R3 → R4	17
10.9.0.0/16	R1 → R2	30
10.10.0.0/16	R1 → R3 → R4	25
10.11.0.0/16	R1 → R3 → R4 → R5	27



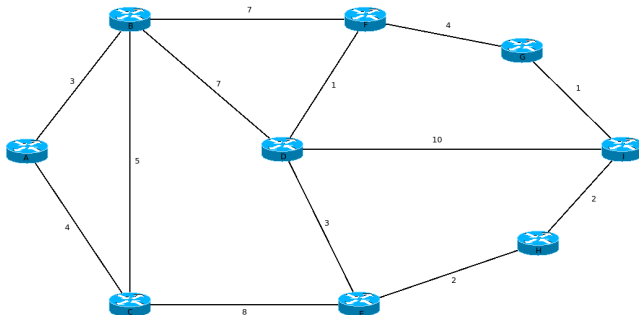
Figur: SPF-träd [2014RSE]

- Då varje nod bygger upp topologin uppstår inga loopar.
- Förändring skickas ut till samtliga noder, vilket ger en snabb konvergering.
- Hierarkisk design minskar skalbarhetsproblem.

- Resurskrävande
 - ▶ Minne
 - ▶ Processorkraft
 - ▶ Bandbredd
- Komplicerad design krävs vid större nätverk.

Dijkstra's SPF Algorithm

Dijkstra's Shortest Path First (SPF) Algorithm



Figur: Dijkstras SPF - Topologi exempel

- När är det lämpligt att använda statisk routing?
- När är det lämpligt att använda dynamisk routing?
- För- och nackdelar med distance-vector och link-state?
- Fördelar med flera metrics för att beräkna kostnaden på en länk?
- Varför vill vi ha en kort konvergeringstid?
- Vad påverkar konvergeringstid?
- Vilket routingprotokoll har kortast konvergeringstid? Varför?

- 5.2.1.6 — Investigating Convergence
- 5.2.3.4 — Comparing RIP and EIGRP Path Selection

