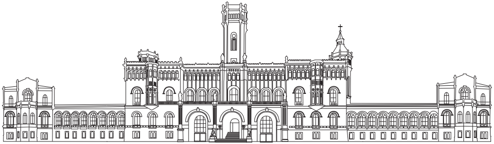
Rechnernetze - Computer Networks

Problem Set 8: Internetworking & Routing

Markus Fidler, Mark Akselrod, Lukas Prause



Institute of Communications Technology
Leibniz Universität Hannover

June 10, 2024

1.1 Difference between forwarding, switching and routing:

- **Forwarding:** router-local action of transferring a packet from an input link interface to the appropriate output link interface.
- **Switching:** moving packets between devices on the same network.
- **Routing:** a network-wide process that determine the end-to-end paths that packets take from source to destination.

1.2 Link-state vs. distance-vector routing

- **Link-state routing**
  - Nodes intercommunicate with all other nodes via broadcast ...
  - and provide the costs of the directly connected links
  - Requires knowledge of the entire network topology
- **Distance-vector routing**
  - Only neighbouring nodes exchange information ...
  - they provide each least-cost estimates from itself to all other (known) nodes
  - Requires only connections to the neighbours and no topology information

Institut für Kommunikations-Technik

1.3 Link cost selection

- Can be chosen arbitrarily
- Often associated with a network metric, e.g., bandwidth, delay
- Can force traffic to flow along a certain path

1.4 Drawbacks of dynamic link weights

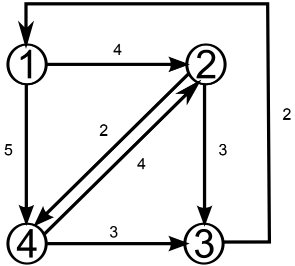
- More responsive to network changes
- Can cause routing instabilities such as routing loops and oscillations

1.5 Internet routing

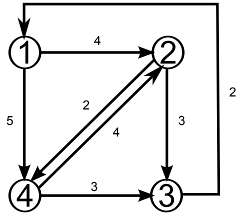
- Interconnected autonomous systems (ASes)
- Intra-domain (local) routing within each AS
- Inter-domain routing: edge routers forward packets to neighbouring ASes

2.1 Plot the topology:

$$E = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}, \quad C = \begin{pmatrix} 0 & 4 & 0 & 5 \\ 0 & 0 & 3 & 2 \\ 2 & 0 & 0 & 0 \\ 0 & 4 & 3 & 0 \end{pmatrix}$$



$$E = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

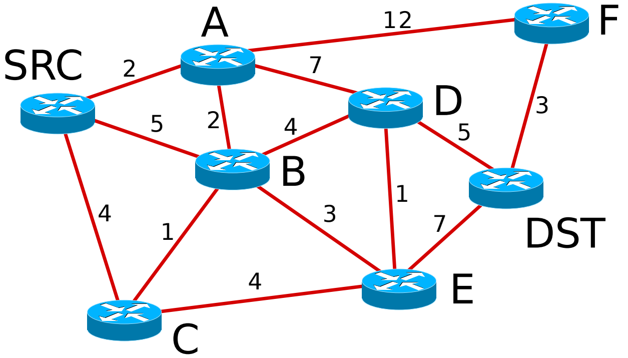


- E denotes the nodes that are reachable in one step
- E^2 denotes the number of two step paths
- E^3 denotes the number of three step paths ...

$$E^2 = \begin{pmatrix} 0 & 1 & 2 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix}, \quad E^3 = \begin{pmatrix} 2 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \\ 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 1 \end{pmatrix}, \quad E^4 = \begin{pmatrix} 2 & 3 & 2 & 3 \\ 1 & 3 & 3 & 2 \\ 2 & 1 & 2 & 1 \\ 1 & 2 & 3 & 3 \end{pmatrix}$$

$$E + E^2 + E^3 + E^4 = \begin{pmatrix} 4 & 6 & 6 & 6 \\ 3 & 5 & 6 & 5 \\ 3 & 3 & 4 & 3 \\ 3 & 5 & 6 & 5 \end{pmatrix}$$

Since all matrix elements are non-zero, all nodes are reachable.



Least-cost path SRC $\rightarrow$ DST ?

Denote

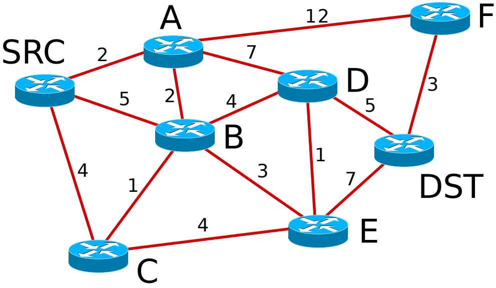
- $c(i,j)$ the cost of link $e_{i,j}$ from node i to node j
  - $c(i,j) = \infty$ if link $e_{i,j}$ does not exist
- $d(i)$ cost (distance) from the source to node i
- $p(i)$ predecessor of node i on the least-cost path from the source to node i
- M set of nodes for which least-cost paths are uniquely determined

3.1 Dijkstra's Algorithm

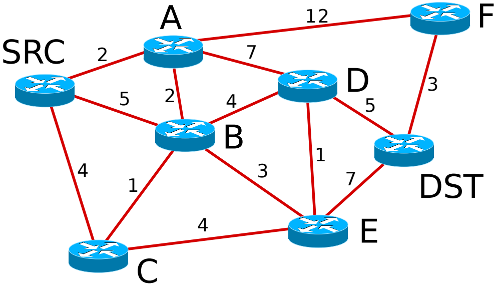
$M = \{a\}$
for all nodes i **do**
    if link $e_{a,i}$ exists **then**
        $d(i) = c(a, i)$
    else
        $d(i) = \infty$
    end if
end for
repeat
    find a node j not in M such that $d(j)$ is minimal
    add j to M
    for all nodes k not in M **do**
        if $d(k) > d(j) + c(j, k)$ **then**
            $d(k) = d(j) + c(j, k)$
            $p(k) = j$
        end if
    end for
until all nodes in M

3.1 Dijkstra's Algorithm - Example

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞

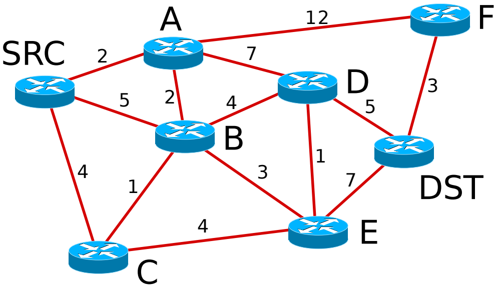


step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_{A}$	$\mathbf{4}_{SRC}$	9_{A}	∞	14_{A}	∞



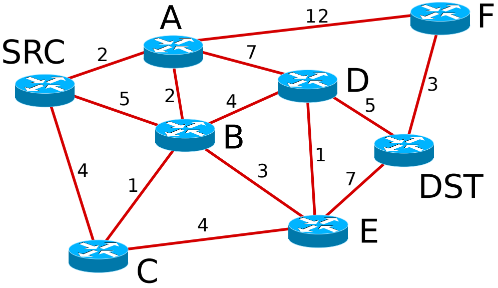
Institut für Kommunikations-Technik

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_{A}$	$\mathbf{4}_{SRC}$	9_{A}	∞	14_{A}	∞
2.1	SRC,A,B			$\mathbf{4}_{SRC}$	8_{B}	7_{B}	14_{A}	∞



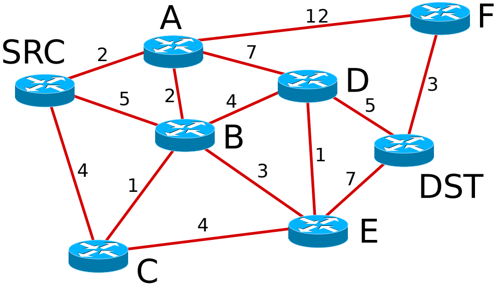
3.1 Dijkstra's Algorithm - Example

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_A$	$\mathbf{4}_{SRC}$	9_A	∞	14_A	∞
2.1	SRC,A,B			$\mathbf{4}_{SRC}$	8_B	7_B	14_A	∞
2.2	SRC,A,C		$\mathbf{4}_A$		9_A	8_C	14_A	∞



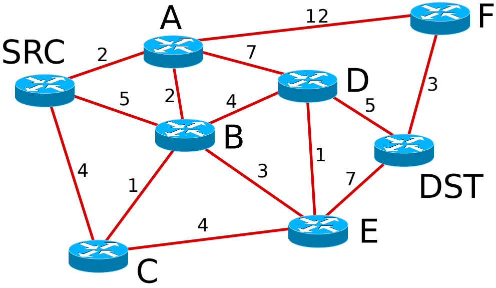
3.1 Dijkstra's Algorithm - Example

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_{A}$	$\mathbf{4}_{SRC}$	9_{A}	∞	14_{A}	∞
2	SRC,A,B			$\mathbf{4}_{SRC}$	8_{B}	7_{B}	14_{A}	∞
3	SRC,A,B,C				8_{B}	$\mathbf{7}_{B}$	14_{A}	∞



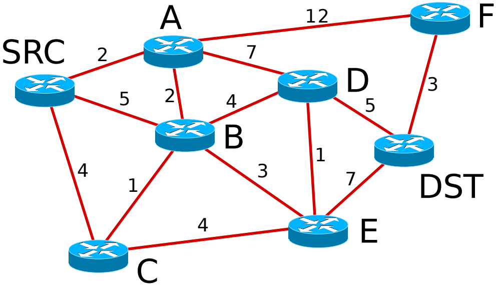
3.1 Dijkstra's Algorithm - Example

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_A$	$\mathbf{4}_{SRC}$	9_A	∞	14_A	∞
2	SRC,A,B			$\mathbf{4}_{SRC}$	8_B	7_B	14_A	∞
3	SRC,A,B,C				8_B	$\mathbf{7}_B$	14_A	∞
4	SRC,A,B,C,E				$\mathbf{8}_B$		14_A	14_E



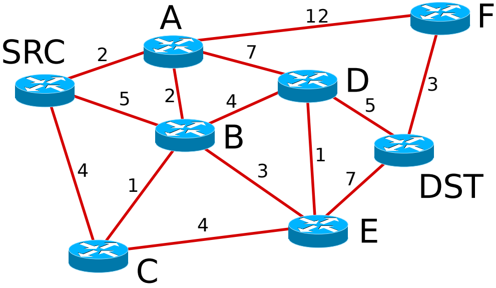
3.1 Dijkstra's Algorithm - Example

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_{A}$	$\mathbf{4}_{SRC}$	9_{A}	∞	14_{A}	∞
2	SRC,A,B			$\mathbf{4}_{SRC}$	8_{B}	7_{B}	14_{A}	∞
3	SRC,A,B,C				8_{B}	$\mathbf{7}_{B}$	14_{A}	∞
4	SRC,A,B,C,E				$\mathbf{8}_{B}$		14_{A}	14_{E}
5	SRC,A,B,C,E,D						14_{A}	$\mathbf{13}_{D}$



3.1 Dijkstra's Algorithm - Example

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_{A}$	$\mathbf{4}_{SRC}$	9_{A}	∞	14_{A}	∞
2	SRC,A,B			$\mathbf{4}_{SRC}$	8_{B}	7_{B}	14_{A}	∞
3	SRC,A,B,C				8_{B}	$\mathbf{7}_{B}$	14_{A}	∞
4	SRC,A,B,C,E				$\mathbf{8}_{B}$		14_{A}	14_{E}
5	SRC,A,B,C,E,D						14_{A}	$\mathbf{13}_{D}$
6	SRC,A,B,C,E,D,DST						$\mathbf{14}_{A}$	



Institut für Kommunikations-Technik

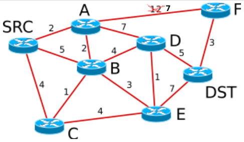
step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_A$	$\mathbf{4}_{SRC}$	9_A	∞	14_A	∞
2	SRC,A,B			$\mathbf{4}_{SRC}$	8_B	7_B	14_A	∞
3	SRC,A,B,C				8_B	$\mathbf{7}_B$	14_A	∞
4	SRC,A,B,C,E				$\mathbf{8}_B$		14_A	14_E
5	SRC,A,B,C,E,D						14_A	$\mathbf{13}_D$
6	SRC,A,B,C,E,D,DST						$\mathbf{14}_A$	
7	SRC,A,B,C,E,D,DST,F							

The least-cost path from SRC to DST has a cost of **13** over the path **SRC-A-B-D-DST**.

Institut für Kommunikations-Technik

Link A-F Cost: $12 \to 7$

step	M	d(A)	d(B)	d(C)	d(D)	d(E)	d(F)	d(DST)
init	SRC	$\mathbf{2}_{SRC}$	5_{SRC}	4_{SRC}	∞	∞	∞	∞
1	SRC,A		$\mathbf{4}_A$	$\mathbf{4}_{SRC}$	9_A	∞	$14_A \cancel{14_A} 9_A$	∞
2	SRC,A,B			$\mathbf{4}_{SRC}$	8_B	7_B	$14_A \cancel{14_A} 9_A$	∞
3	SRC,A,B,C				8_B	$\mathbf{7}_B$	$14_A \cancel{14_A} 9_A$	∞
4	SRC,A,B,C,E				$\mathbf{8}_B$		$14_A \cancel{14_A} 9_A$	14_E
5	SRC,A,B,C,E,D						$14_A \cancel{14_A} \mathbf{9}_A$	$\mathbf{13}_D$ 13_D
6	SRC,A,B,C,E,D,DST ~~SRC,A,B,C,E,D,DST~~ SRC,A,B,C,E,D,F						$14_A \cancel{14_A}$	$\mathbf{12}_F$
7	SRC,A,B,C,E,D,DST,F ~~SRC,A,B,C,E,D,DST,F~~ SRC,A,B,C,E,D,F,DST							



The least-cost path from SRC to DST has then a cost of **12** over the path **SRC-A-F-DST** (instead of $SRC - A - B - D - DST$).

Denote

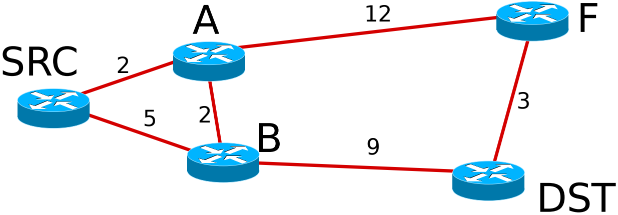
- D^x distance table of node x
  - example: $D^x(y, z) = 4$ route to destination y has cost 4 via next hop z
- F^x forwarding table of node x
  - example: $F^x(y) = (4, z)$ shortest path to destination y via next hop z has cost 4

Mode of operation

- node z computes the shortest path to destination y as $c(z, y) = \min_w\{D^z(y, w)\}$
- node z stores the next hop w that achieves the minimum in its forwarding table $F^z(y) = (c(z, y), w)$
- node z sends its forwarding table to it's neighboring node x
- node x computes $D^x(y, z) = c(x, z) + c(z, y)$

4 Distance Vector Routing

for all neighbors y **do**
    $D^x(y,y) = c(x,y)$
end for
for all destinations z **do**
    send $\min_w\{D^x(z,w)\}$ to all neighbors y
end for
loop
    if $c(x,y)$ for neighbor y changes by d **then**
        update $D^x(z,y) = D^x(z,y) + d$ for all destinations z
    else if updated $c(y,z)$ received from neighbor y **then**
        recompute $D^x(z,y) = c(x,y) + c(y,z)$
    end if
    if a new $\min_w\{D^x(z,w)\}$ exists for any destination z **then**
        send new $\min_w\{D^x(z,w)\}$ to all neighbors y
    end if
end loop



Least-cost path SRC $\rightarrow$ DST ?

4.1 Distance Vector Algorithm Example

initialization: the distance vector of each host x is initialized with the link cost $c(x, y)$ to the direct node neighbors y:
$D^x(y, y) = c(x, y)$

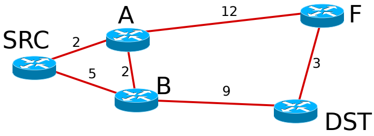
D^S	A	B
A	2	
B		5
F		
D		

D^A	S	B	F
S	2		
B		2	
F			12
D			

D^B	S	A	D
S	5		
A		2	
F			
D			9

D^F	A	D
S		
A	12	
B		
D		3

D^D	B	F
S		
A		
B	9	
F		3



Each node sends an update to all of its neighbors.

4.1 Distance Vector Algorithm Example

T=1

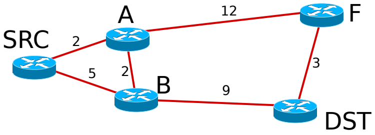
D^S	A	B
A	**2**	7
B	**4**	5
F	**14**	
D		**14**

D^A	S	B	F
S	2	7	
B	7	**2**	
F			**12**
D		**11**	15

D^B	S	A	D
S	5	**4**	
A	7	**2**	
F		14	**12**
D			**9**

D^F	A	D
S	**14**	
A	**12**	
B	14	**12**
D		**3**

D^D	B	F
S	**14**	
A	**11**	15
B	**9**	
F		**3**



Each node sends an update to all of its neighbors when a cheaper path is discovered

4.1 Distance Vector Algorithm Example

T=2

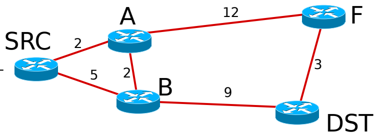
D^S	A	B
A	**2**	7
B	**4**	5
F	**14**	16
D	<u>**13**</u>	14

D^A	S	B	F
S	**2**	6	26
B	6	**2**	24
F	16	14	**12**
D	16	**11**	15

D^B	S	A	D
S	5	**4**	23
A	7	**2**	20
F	19	14	**12**
D	19	13	**9**

D^F	A	D
S	**14**	17
A	**12**	14
B	14	**12**
D	23	**3**

D^D	B	F
S	<u>**13**</u>	17
A	**11**	15
B	**9**	15
F	21	**3**



T=3

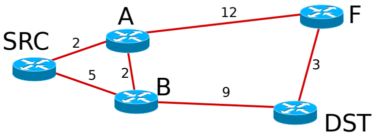
D^S	A	B
A	**2**	7
B	**4**	5
F	**14**	16
D	**13**	14

D^A	S	B	F
S	**2**	6	26
B	6	**2**	24
F	16	14	**12**
D	15	**11**	15

D^B	S	A	D
S	5	**4**	22
A	7	**2**	20
F	19	14	**12**
D	17	13	**9**

D^F	A	D
S	**14**	16
A	**12**	14
B	14	**12**
D	23	**3**

D^D	B	F
S	**13**	17
A	**11**	15
B	**9**	15
F	21	**3**



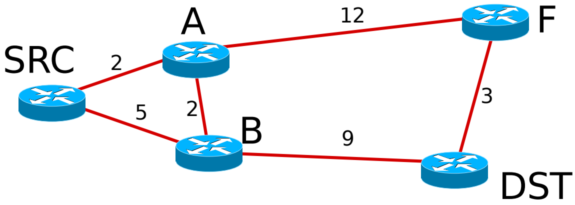
Distance vectors reached stable-state, no more updates sent

Example, Forwarding table for A after step 3:

D^A	S	B	F
S	**2**	6	26
B	6	**2**	24
F	16	14	**12**
D	15	**11**	15

F^A	
S	2,S
B	2,B
F	12,F
D	11,B

4.2 Distance Vector Algorithm Example



Path SRC $\rightarrow$ DST ?

After t=1: SRC $\rightarrow$ B $\rightarrow$ DST

After t=2: SRC $\rightarrow$ A $\rightarrow$ B $\rightarrow$ DST

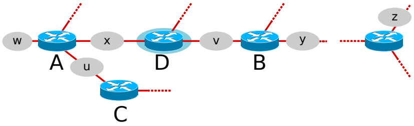
5.1 Difference between Routing Protocols

- **Intra-AS Routing Protocols**
  - Only used within an AS
  - Contain only the information necessary to route packets within the network
  - Protocols: RIP, EIGRP (Distance Vector), OSPF, IS-IS (Link-State)
- **Inter-AS Routing Protocols**
  - Route packets between different ASes
  - Contain routes necessary to reach every other AS on the Internet
  - Prevalent protocol: Border Gateway Protocol (BGP)

Different requirements on the routing procols depending on whether they are used inside (*intra*) or between (*inter*) ASes:

- The ability to specify routing policies for ISPs (inter-AS routing)
- Scalability more important than pure routing speed
- Intra-AS protocols typically optimized for fast routing

5.3 RIP Example



Routing Table of router D:

Destination Subnet	Next Router	Hops to Destination
u	A	2
w	A	2
y	B	2
z	B A	7 1+4=5 5
x	-	1
v	-	1

RIP advertisement from router A:

Destination Subnet	Next Router	Hops to Destination
z	C	4
u	-	1
w	-	1
x	-	1

OSPF areas:

local area subset of the AS
backbone area connects multiple local areas and AS with other ASes

Router types:

internal router within a local area
area border router between backbone and local area
backbone router within backbone area
boundary router between backbone area and other ASes