

Компјутерски мрежи

Час 19,20

Internet

- Верзија 4 (IP version 4 – IPv4), RFC 791
- Верзија 6 (IP version 6 – IPv6), RFC 2460

IPv6: motivation

- *основна мотивација:* 32-bit адресниот простор наскоро ќе биде цел доделен.
- додатна мотивација:
 - header format помага да се забрза processing/forwarding
 - header changes за имплементација на QoS

IPv6 datagram format:

- Фиксна должина 40 byte header
- Не е дозволена фрагментација

IPv6 datagram format

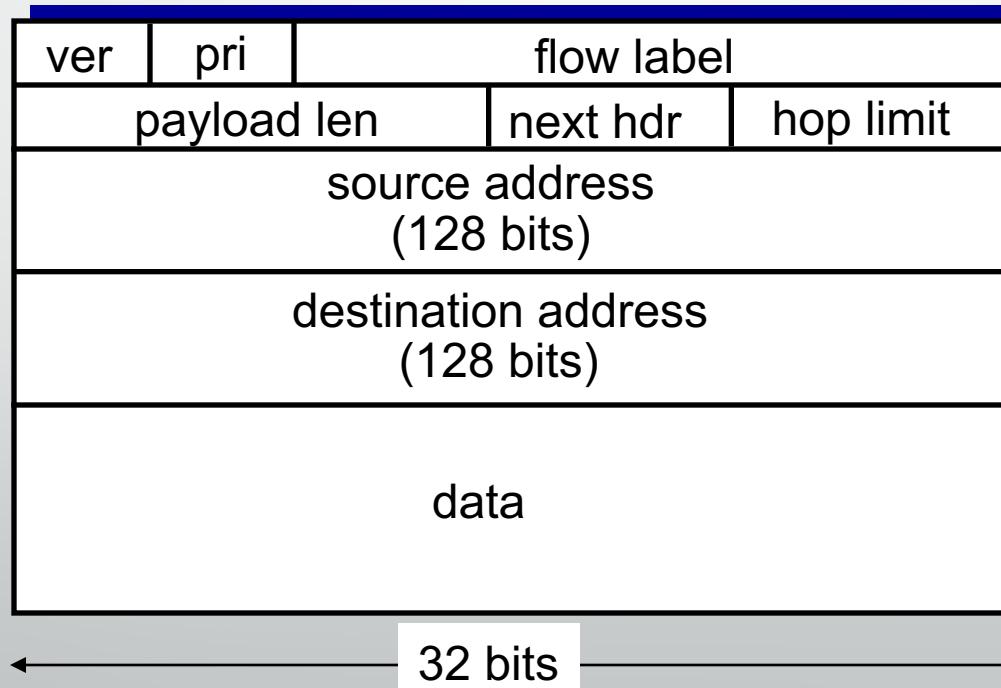
Priority (traffic class): индикација за приоритетите на datagrams (QoS)

flow Label: идентификација на datagrams во ист “flow.”

пр. за VoIP, видео (concept of “flow” not well defined).

next header: идентификација на upper layer protocol (TCP, UDP)

hop limit: = TTL

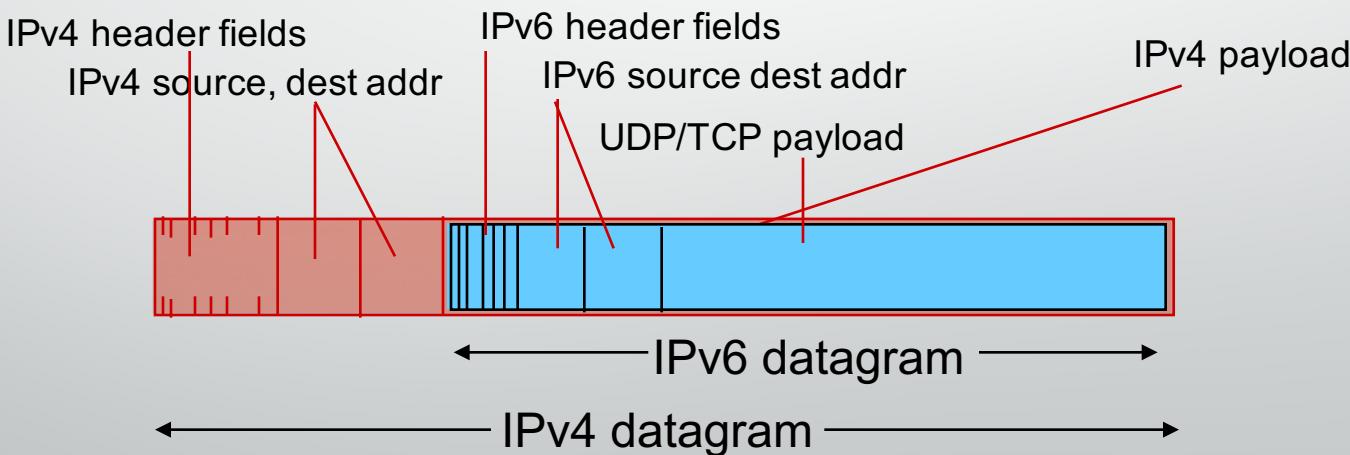


Other changes from IPv4

- *fragmentation*: доколку пакетот е преголем се отфрла и се испраќа назад ICMP error порака па оригиналниот ќе мора да ја смали големината на пакетот
- *checksum*: не постои со цел reduce processing time at each hop
- *options*: дозволени, но надвор од header, indicated со поинтер кон “Next Header” полето
- *ICMPv6*: нова верзија на ICMP
 - additional message types, e.g. “Packet Too Big”
 - multicast group management functions

Transition from IPv4 to IPv6

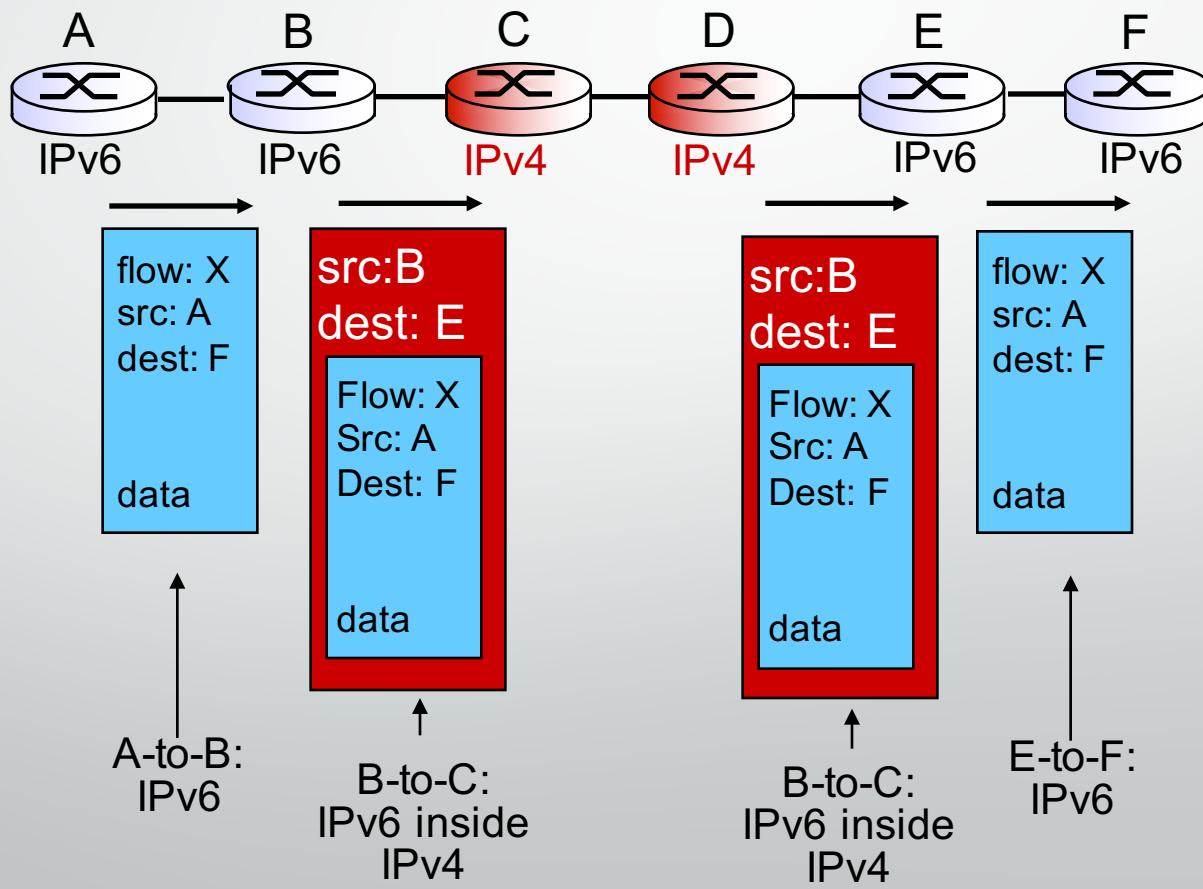
- Сите рутери не можат да се надградат одеднаш
 - Нема време на “транзиција”
 - Како мрежата би работела со mixed IPv4 и IPv6 рутери?
- *tunneling*: IPv6 datagram се пренесува како *payload* во IPv4 datagram низ IPv4 рутери



logical view:



physical view:



Network layer control plane

Што стои зад улогата на control plane

- traditional routing algorithms
- SDN controllers
- Internet Control Message Protocol
- network management

and their instantiation, implementation in the Internet:

- OSPF, BGP, OpenFlow, ODL and ONOS controllers, ICMP, SNMP

Network-layer functions

потсетувње: две функции на network-layer

- *forwarding*: упатување на пакетите од router's input на соодветен router output
- *routing*: определување на патеката која пакетите треба да ја поминат од source до destination

data plane

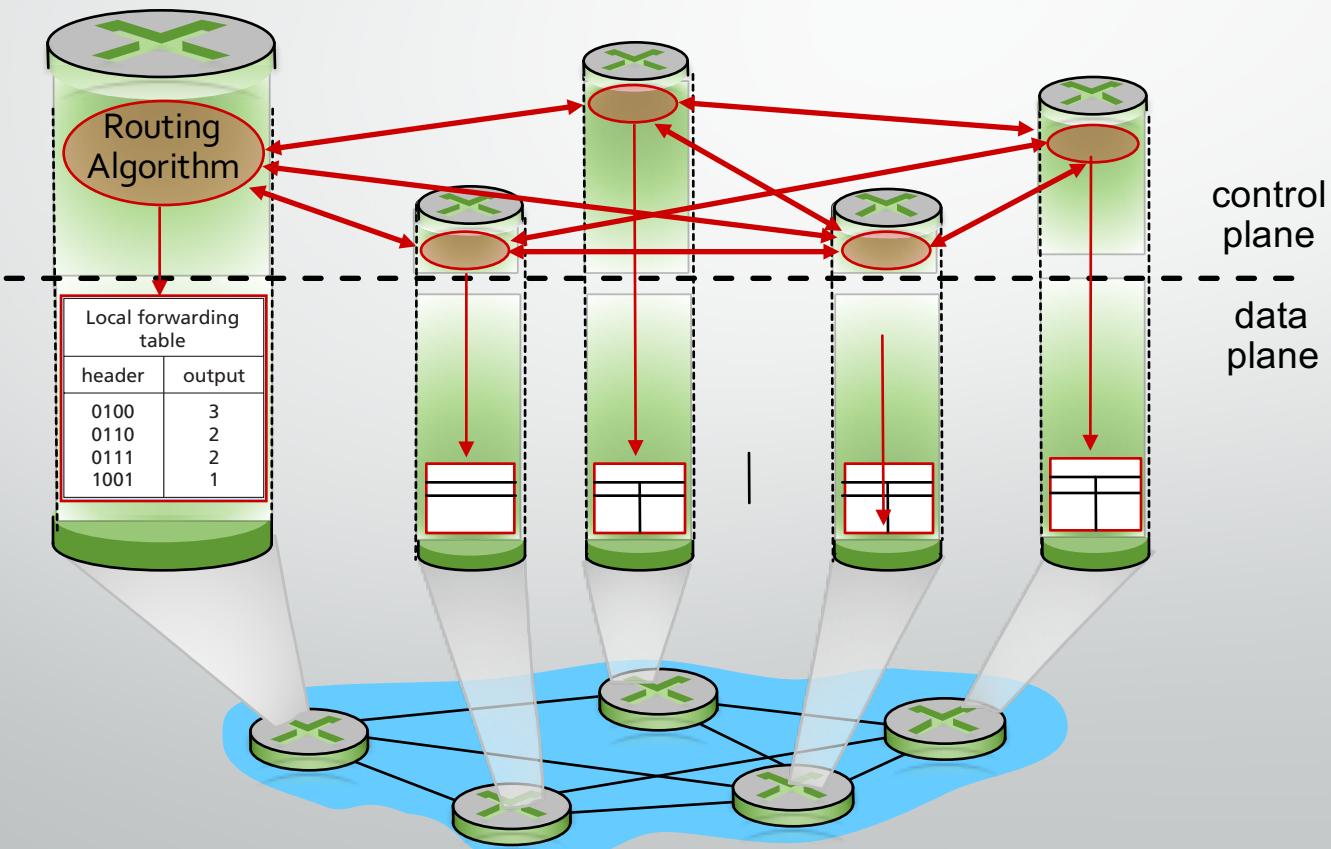
control plane

Два пристапа за структура на network control plane:

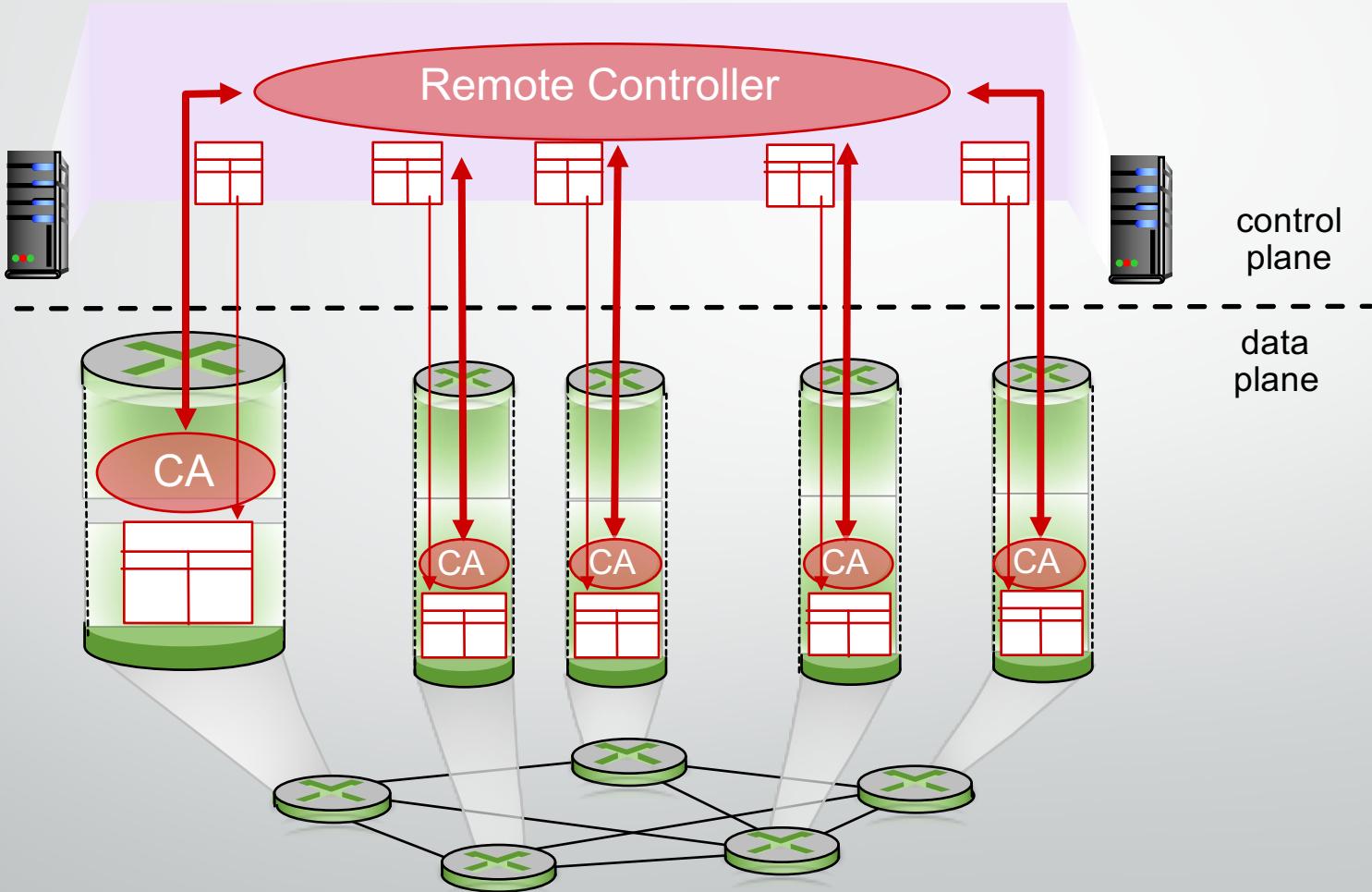
- per-router control (traditional)
- logically centralized control (software defined networking)

Per-router control plane

Поединечните компоненти од алгоритми за упатување се *in each and every router* и комуницираат помеѓу себе во control plane за да ги пресметаат forwarding tables



Logically centralized control plane



Routing protocols

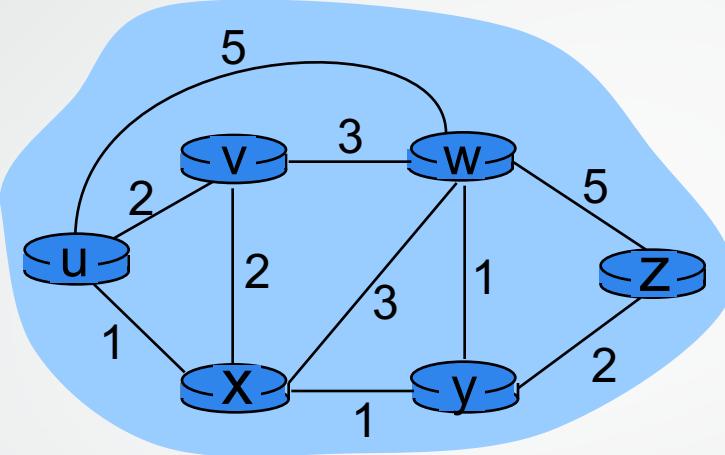
Цел на упатувачкиот протокол: определување на “good” патека (equivalently, routes), од sending hosts до receiving host, преку мрежата од рутери

- патека: секвенцата од пакети ќе преминува преку низа од рутери одод даден иницијален source host до даден final destination host
- “good”: least “cost”, “fastest”, “least congested”

Terminology

- Routing Protocol
 - fills routing table with loop-free routing information
 - RIP, IGRP, EIGRP, OSPF, BGP
- Routed Protocol
 - protocol that defines logical addressing and routing
 - IP, IPX
- Routing Type
 - algorithm and logic of routing protocol

Graph abstraction of the network



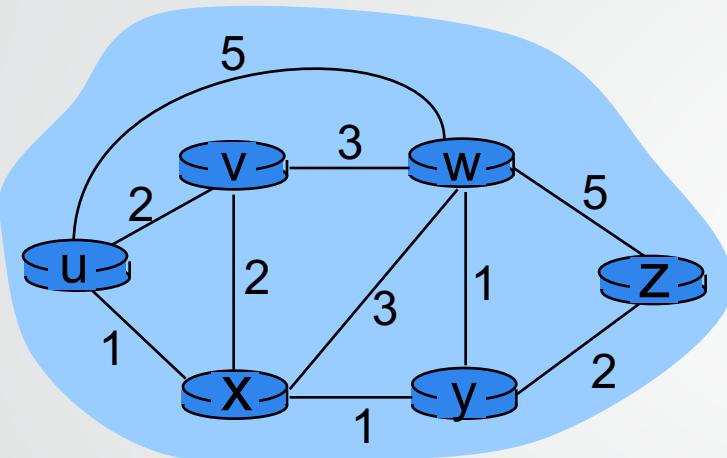
graph: $G = (N, E)$

$N = \text{set of routers} = \{ u, v, w, x, y, z \}$

$E = \text{set of links} = \{ (u,v), (u,x), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) \}$

- дирекционален/недирекционален граф
- Негативни вредности
- Јамки кои се затвараат сами во себе

Graph abstraction: costs



$c(x,x')$ = cost of link (x,x')
e.g., $c(w,z) = 5$

cost може да биде 1, или
Во врска со bandwidth,
Или во врска со загушувањето (доцнење)
Бескрај, ако не постои врска

cost of path $(x_1, x_2, x_3, \dots, x_p) = c(x_1, x_2) + c(x_2, x_3) + \dots + c(x_{p-1}, x_p)$

key question: кој е least-cost патека помеѓу u и z ?

routing algorithm: алоритам кој ја пронаоѓа least cost патека

Класификација на Routing algorithm

Q: глобални или децентрализирани?

global:

- Сите рутери ја имаат комплетната топологија, информација за “цената” на линкот
- Периодично се емитуваат link-state пораки до сите останати nodes (идентитет на јазелот и цена на врските поврзани на тој јазел)
- “link state” algorithms, OSPF (Dijikstra)

decentralized:

- Рутерот ги знае physically-connected соседи, “цената” на линкот до соседите
- iterative process за пресметка, размена на info со соседите
“distance vector” algorithms Bellman-Ford (RIP, BGP)

Класификација на Routing algorithm

Q: статички или динамички?

static:

- Рутите се менуваат споро во текот на времето / најчесто потребна е интервенција на администратор

dynamic:

- Промената на рутите е честа - автоматска
 - periodic update
 - in response to link cost changes

A link-state routing algorithm

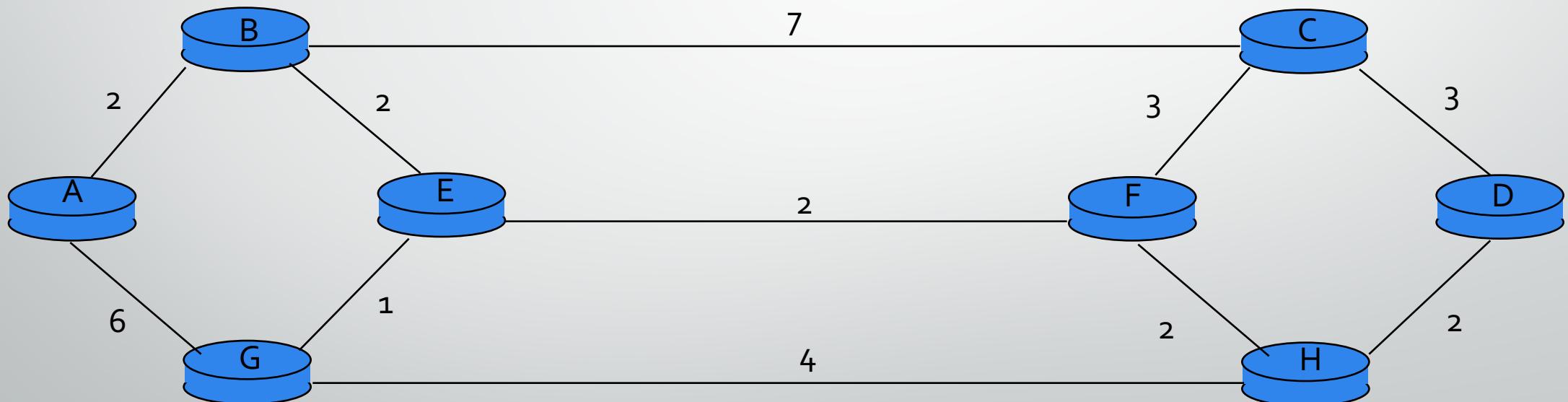
Dijkstra's algorithm

- Треба да е позната топологијата, costs исто треба да е позната до сите nodes
 - Се постигнува преку “link state broadcast”
 - сите nodes имаат иста информација
- Се пресметува least cost патека од еден node (“source”) до сите nodes
 - gives *forwarding table* за тој node
- iterative: после k итерации, ќе биде позната least cost патека до k дестинации

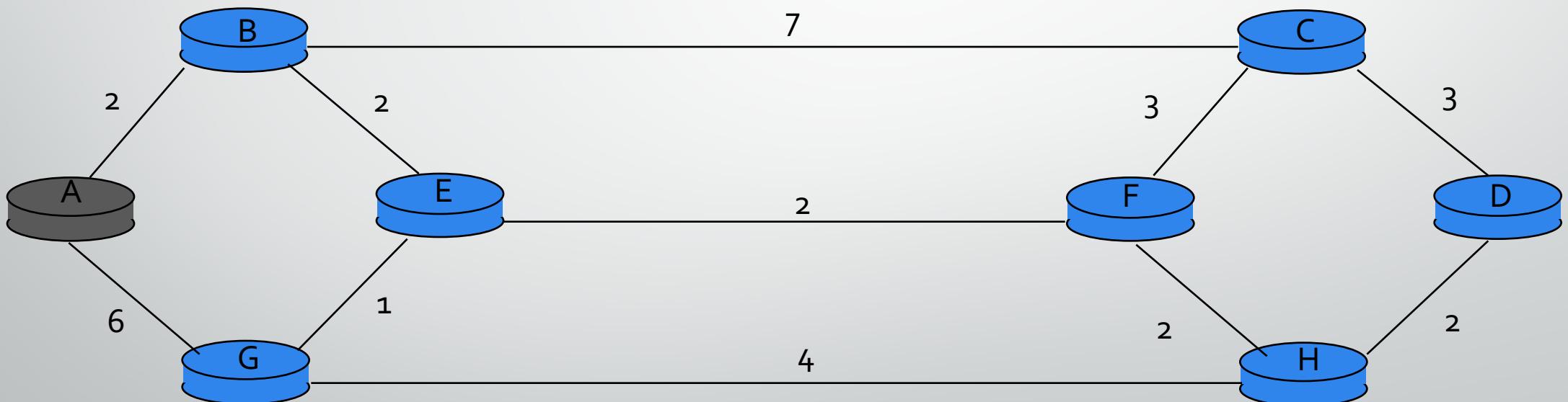
notation:

- $c(x,y)$: link cost from node x to y ; $= \infty$ if not direct neighbors
- $D(v)$: current value of cost of path from source to dest. v
- $p(v)$: predecessor node along path from source to v
- N' : set of nodes whose least cost path definitively known

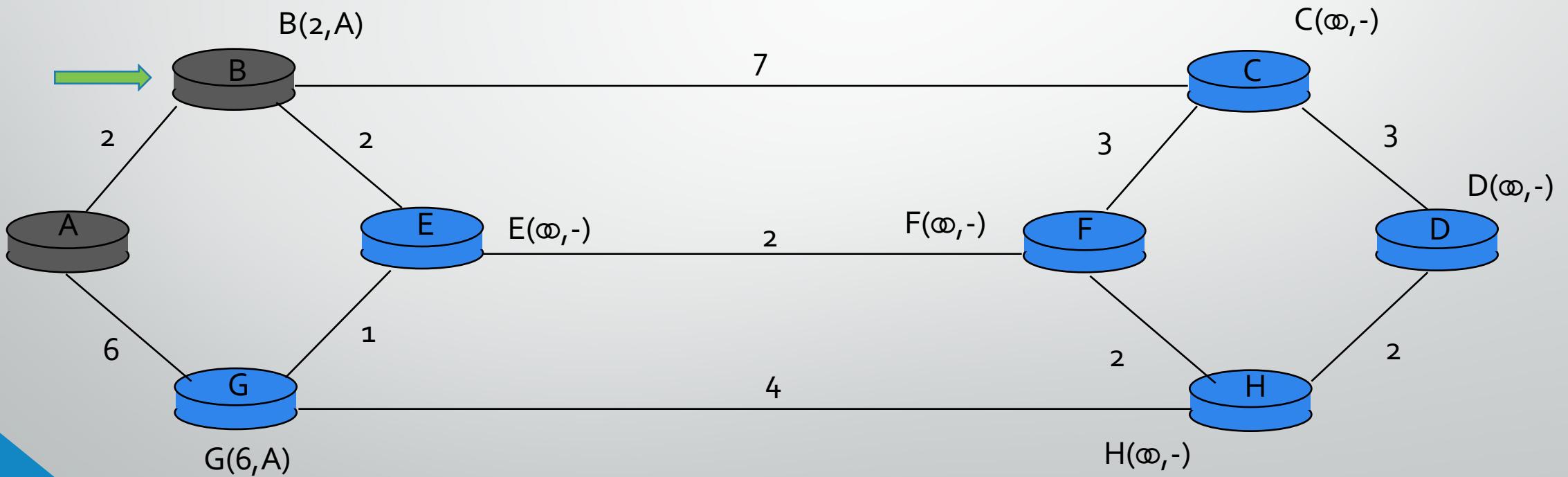
Пример за мрежа на која ќе се примени алгоритмот



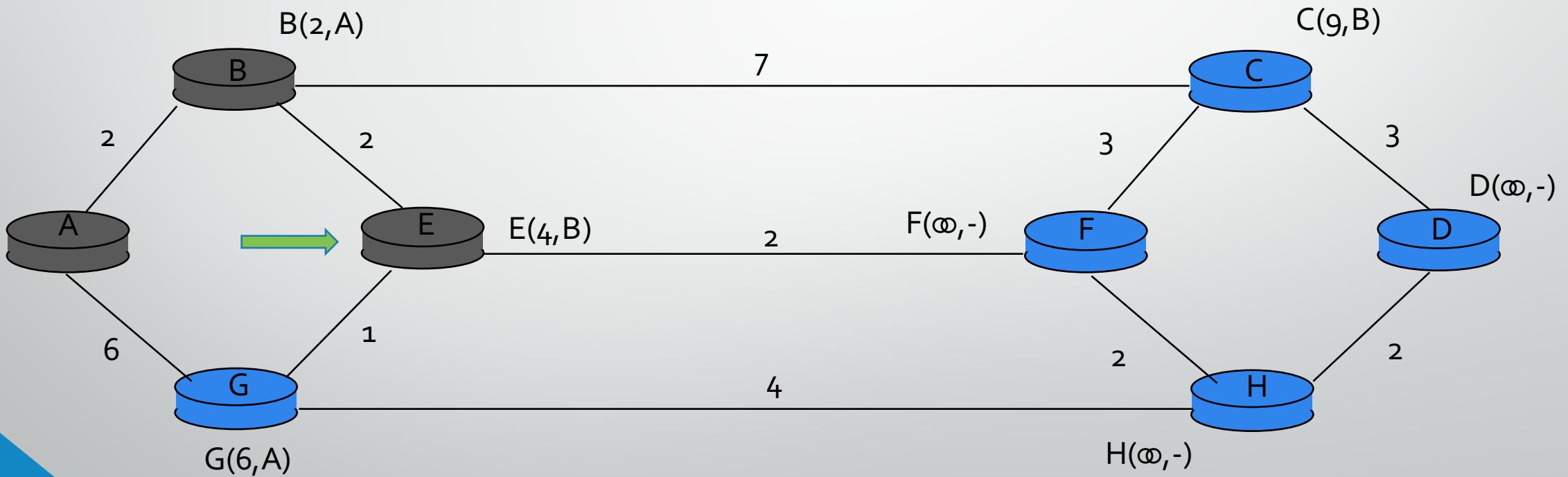
Чекор 1



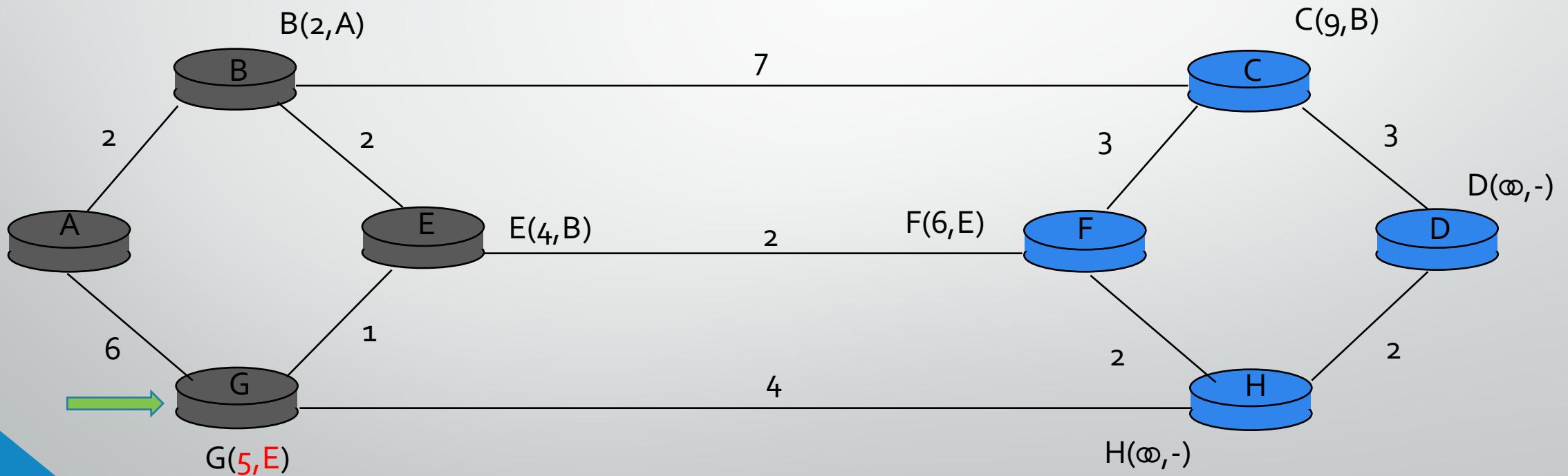
Чекор 2



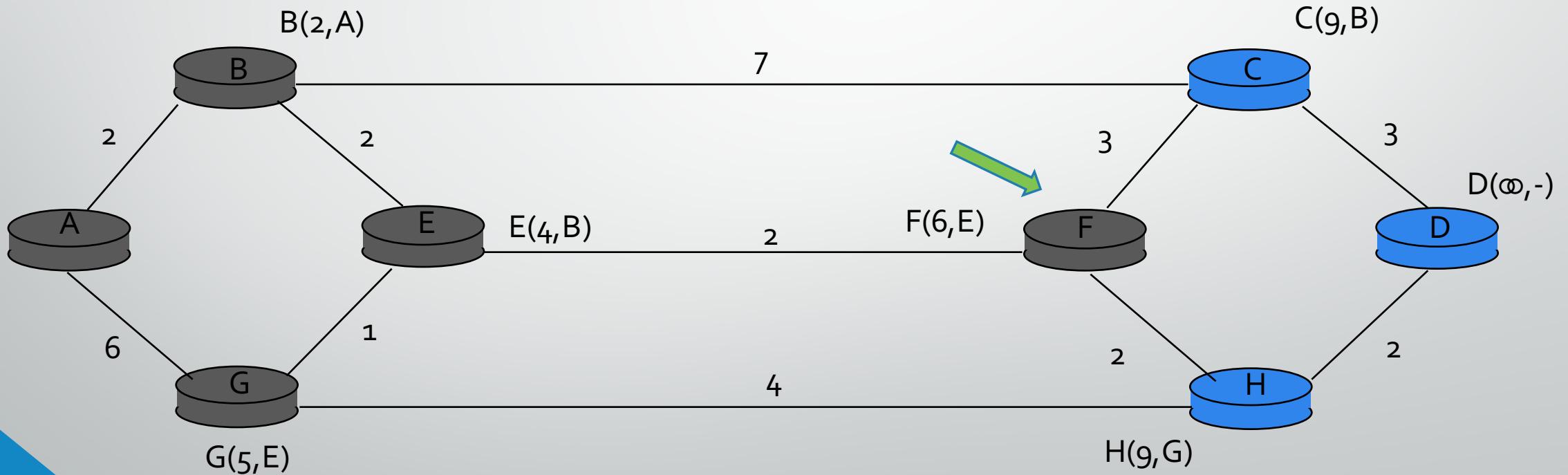
Чекор 3



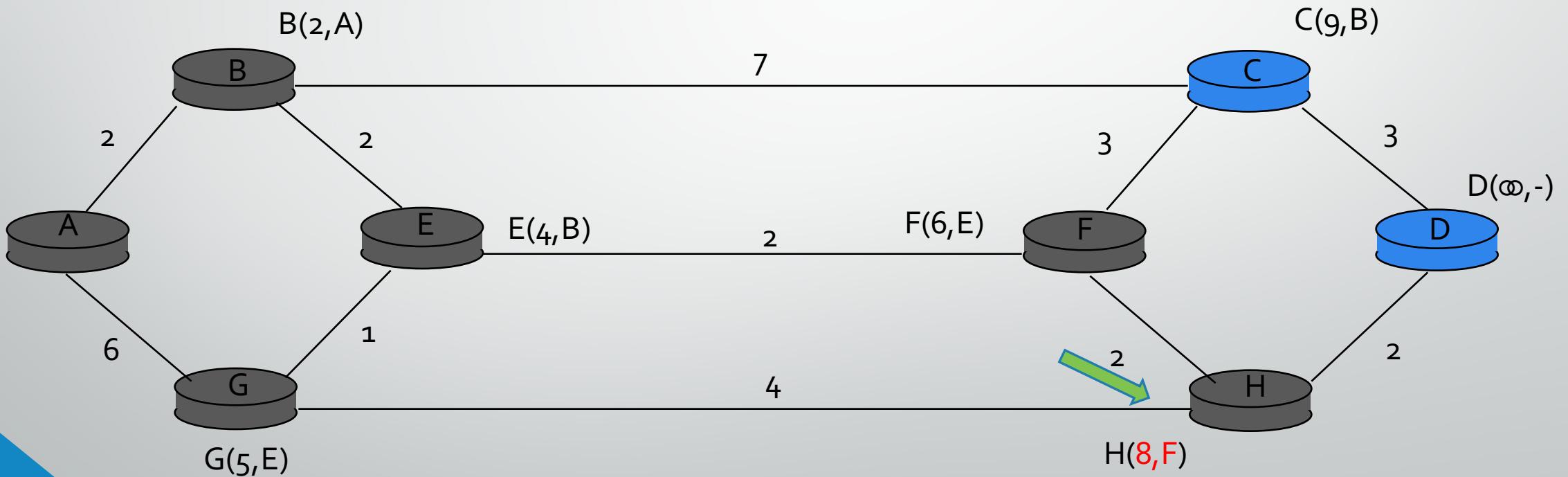
Чекор 4

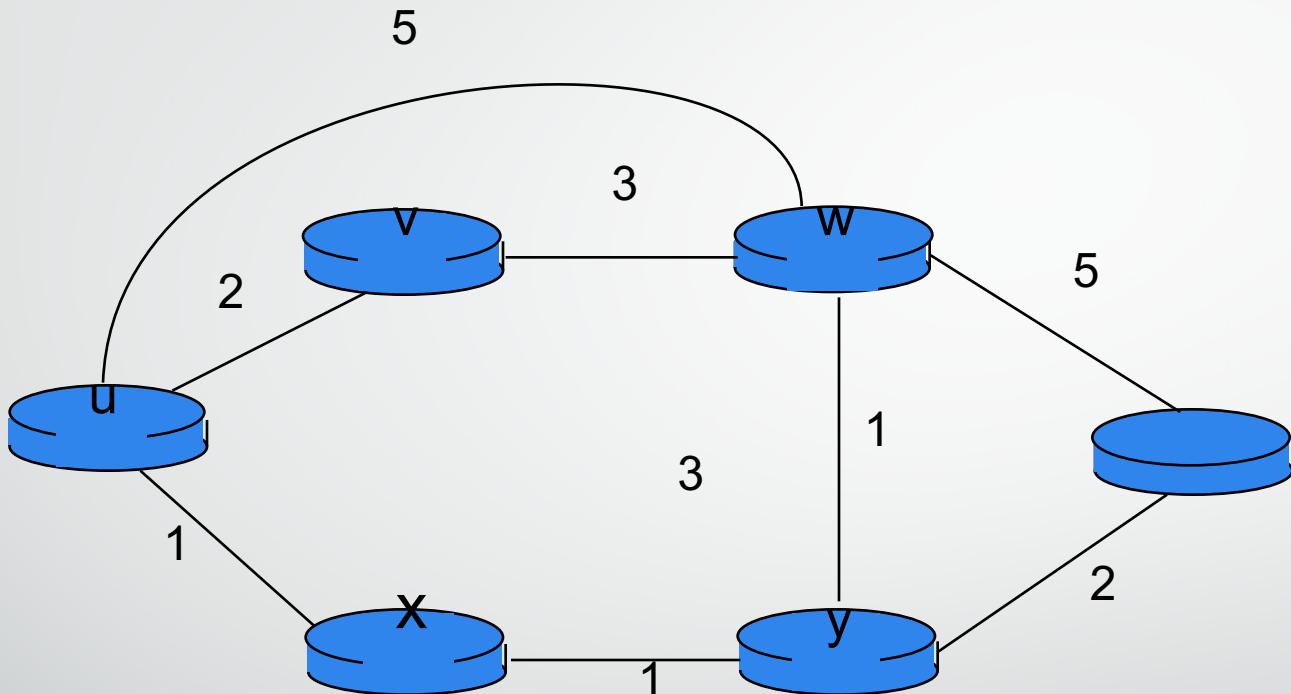


Чекор 5

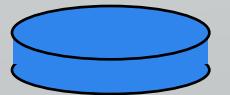


Чекор 6





z



Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

$d_x(y) := \text{cost of least-cost path from } x \text{ to } y$

then

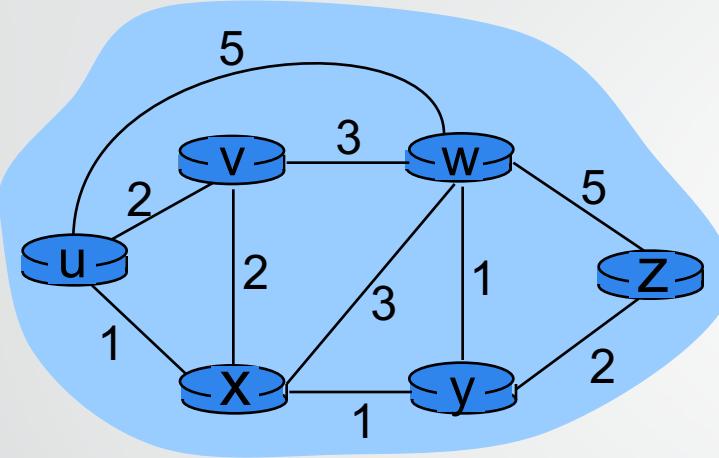
$$d_x(y) = \min \{ c(x,v) + d_v(y) \}$$

- Во равенката \min_v се зема за сите соседи на x
- ако се земе least-cost path од v до y , цената на патеката ќе биде $c(x,v) + d_v(y)$
- Треба да се започне со “патување” до некој сосед v , најмалата cost од x до y е minimum од $c(x,v) + d_v(y)$ земен за сите соседи v

Distance vector algorithm

- $D_x(y)$ = estimate of least cost from x to y
 - x maintains distance vector $\mathbf{D}_x = [D_x(y): y \in N]$
- node x :
 - Ја знае cost до секој neighbor v : $c(x,v)$
 - Ги одржува distance vectori на сите соседи.
За секој сосед v , x одржува инфо
 $\mathbf{D}_v = [D_v(y): y \in N]$

Bellman-Ford example



clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

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

node achieving minimum is next
hop in shortest path, used in forwarding table

Distance vector algorithm

key idea:

- Од време на време, секој node ги испраќа сопствените distance vector до соседите - neighbors
- кога x ќе прими нов DV за некој сосед, ја updates сопствената DV користејќи B-F equation:

$$D_x(y) \leftarrow \min_v \{c(x,v) + D_v(y)\} \text{ for each node } y \in N$$

- ❖ under minor, natural conditions, the estimate $D_x(y)$ converge to the actual least cost $d_x(y)$

Distance vector algorithm

iterative, asynchronous: each local iteration caused by:

- Промена на цената на local link
- DV update message од соседот

distributed:

- each node ги известува соседите само коганеговиот DV changes
 - neighbors then notify their neighbors if necessary

each node:

wait for (change in local link cost or msg from neighbor)

recompute estimates

if DV to any dest has changed, *notify* neighbors

$$\begin{aligned}
 D_x(y) &= \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} \\
 &= \min\{2+0, 7+1\} = 2
 \end{aligned}$$

$$\begin{aligned}
 D_x(z) &= \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} \\
 &= \min\{2+1, 7+0\} = 3
 \end{aligned}$$

**node x
table**

	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

**node y
table**

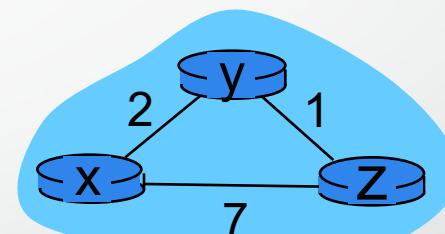
	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

**node z
table**

	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

from

	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0



time

$$\begin{aligned}
 D_x(y) &= \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} \\
 &= \min\{2+0, 7+1\} = 2
 \end{aligned}$$

$$\begin{aligned}
 D_x(z) &= \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} \\
 &= \min\{2+1, 7+0\} = 3
 \end{aligned}$$

node x table

	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

node y table

	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

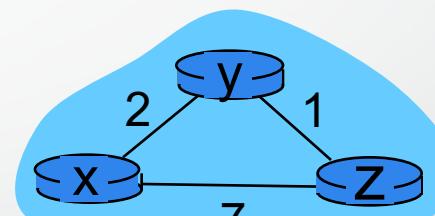
	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0



Консеквенци

- Good news travels fast
- Bad news travels slow

Споредба на LS и DV алгоритмите

message complexity

- **LS:** со n nodes, E вирски, $O(nE)$ msgs sent
- **DV:** размена само помеѓу соседи
 - convergence time varies

speed of convergence

- **LS:** $O(n^2)$ algorithm requires $O(nE)$ msgs
 - Може да се јават осцилации
- **DV:** convergence time varies
 - Може да се појават рутирачки јамки
 - count-to-infinity problem

robustness: што се случува ако рутерот ткаже?

LS:

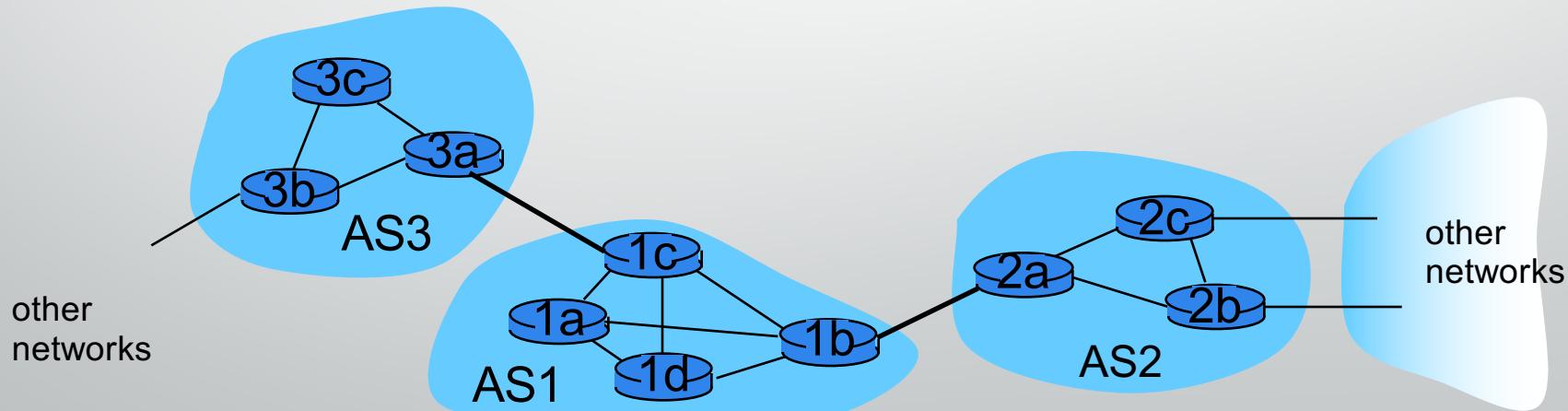
- node ќе испраќа погрешен *link cost*
- секој node ја пресметува само сопствената табела

DV:

- DV може да еmitува погрешен *path cost*
- Секоја node's табела се користи и од останатите
 - Грешката се пропагира низ мрежата

Autonomous System

- Autonomous system е состав од networks под иста administration која користи ист routing protocol или
- Група од routers под заедничка administrative domain кои користат ист routing protocol

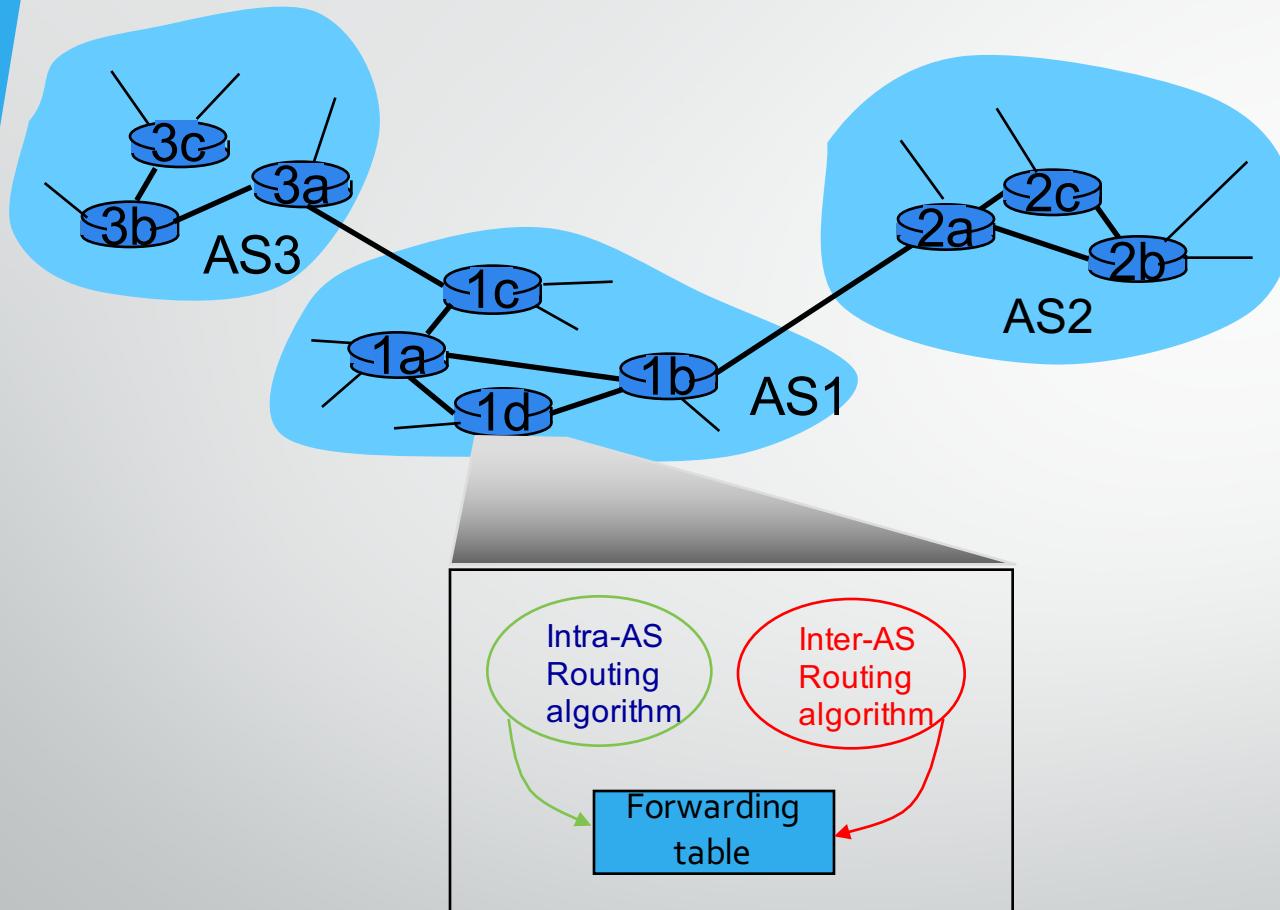


Types of Routing Protocols

- Exterior Gateway Protocols – EGP
 - route помеѓу autonomous systems или routing domains
 - Border Gateway Protocol BGP
- Interior Gateway Protocols – IGP
 - Работи во рамките на autonomous system
 - RIP, IGRP, EIGRP, OSPF

Interior vs. Exterior

Interconnected ASes



- forwarding table се конфигурираат и од intra- и inter-AS routing algorithm
 - intra-AS routing го определува десинациите во рамките на AS
 - inter-AS & intra-AS определува надворешните десинациите

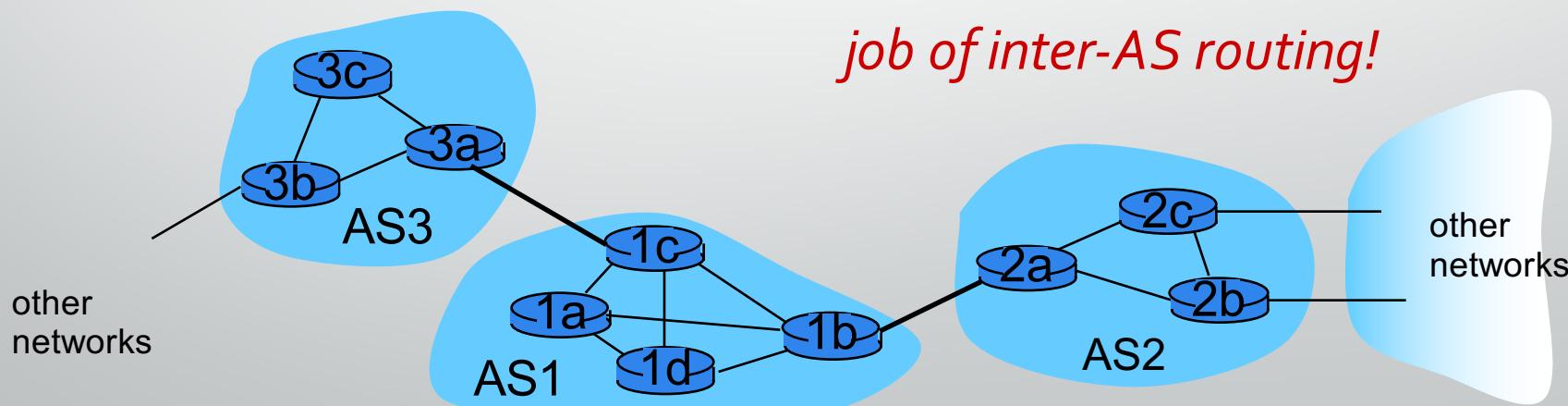
Inter-AS tasks

- suppose router in AS1 receives datagram destined outside of AS1:
 - router should forward packet to gateway router, but which one?

AS1 must:

1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

job of inter-AS routing!



Intra-AS Routing

- Познат како *interior gateway protocols (IGP)*
- најчести intra-AS routing протоколи:
 - RIP: Routing Information Protocol
 - OSPF: Open Shortest Path First (IS-IS protocol essentially same as OSPF)
 - IGRP: Interior Gateway Routing Protocol (Cisco proprietary for decades, until 2016)

IP RIP ver I

RFC 1058

- Routing Information Protocol, најстар протокол (имплементиран во BSD UNIX)
- Distance Vector
- Metric is Hop count, Max 15 hops
- RIP response message секои 30 секунди (routing updates)
- UDP port 520
- Постои и RIP ver II

OSPF (Open Shortest Path First)

RFC 2328

- “open”: јавно достапен
- uses link-state algorithm
 - link state packet dissemination
 - topology map at each node
 - route computation using Dijkstra’s algorithm
- router floods OSPF link-state advertisements до сите рутери во **entire AS**
 - carried in OSPF messages directly over IP (rather than TCP or UDP)
 - link state:for each attached link
 - Можност за автентикација (да се селектира кој може да партиципира)

Internet inter-AS routing: BGP

BGP ver 4 RFC 4271

- **BGP (Border Gateway Protocol):** *the de facto inter-domain routing protocol*
 - “лекак кој го врзува Internet во целина”
- BGP на секој AS му обезбедува начин да:
 - **eBGP:** обезбедува информација за достапноста на subnet од соседни ASes
 - **iBGP:** ја дистрибуира информацијата за reachability до сите AS-internal routers.
 - Определи “good” routes до други мрежи заснован на reachability information и *policy*
- allows subnet to advertise its existence to rest of Internet: “*I am here*”
- Размена на пораки преку TCP port 179