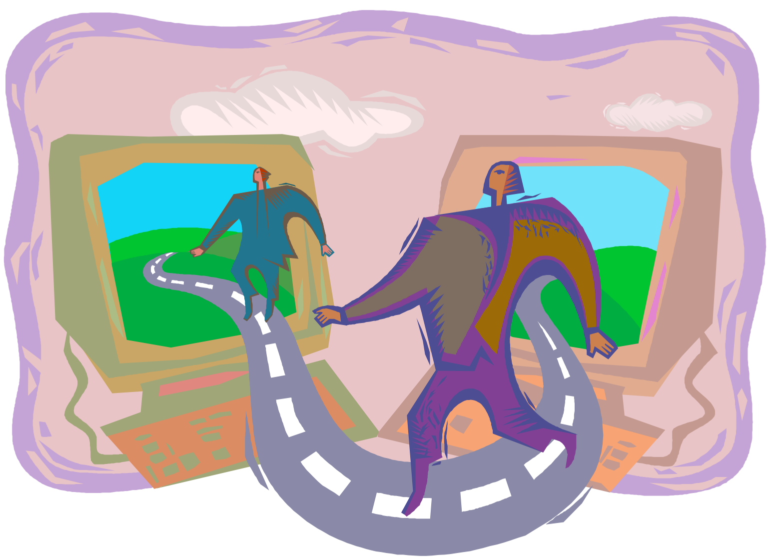
Discussion #13
EE450



-Sample Problems on Routing Protocols

Problem#5: Obtaining a Block of Addresses

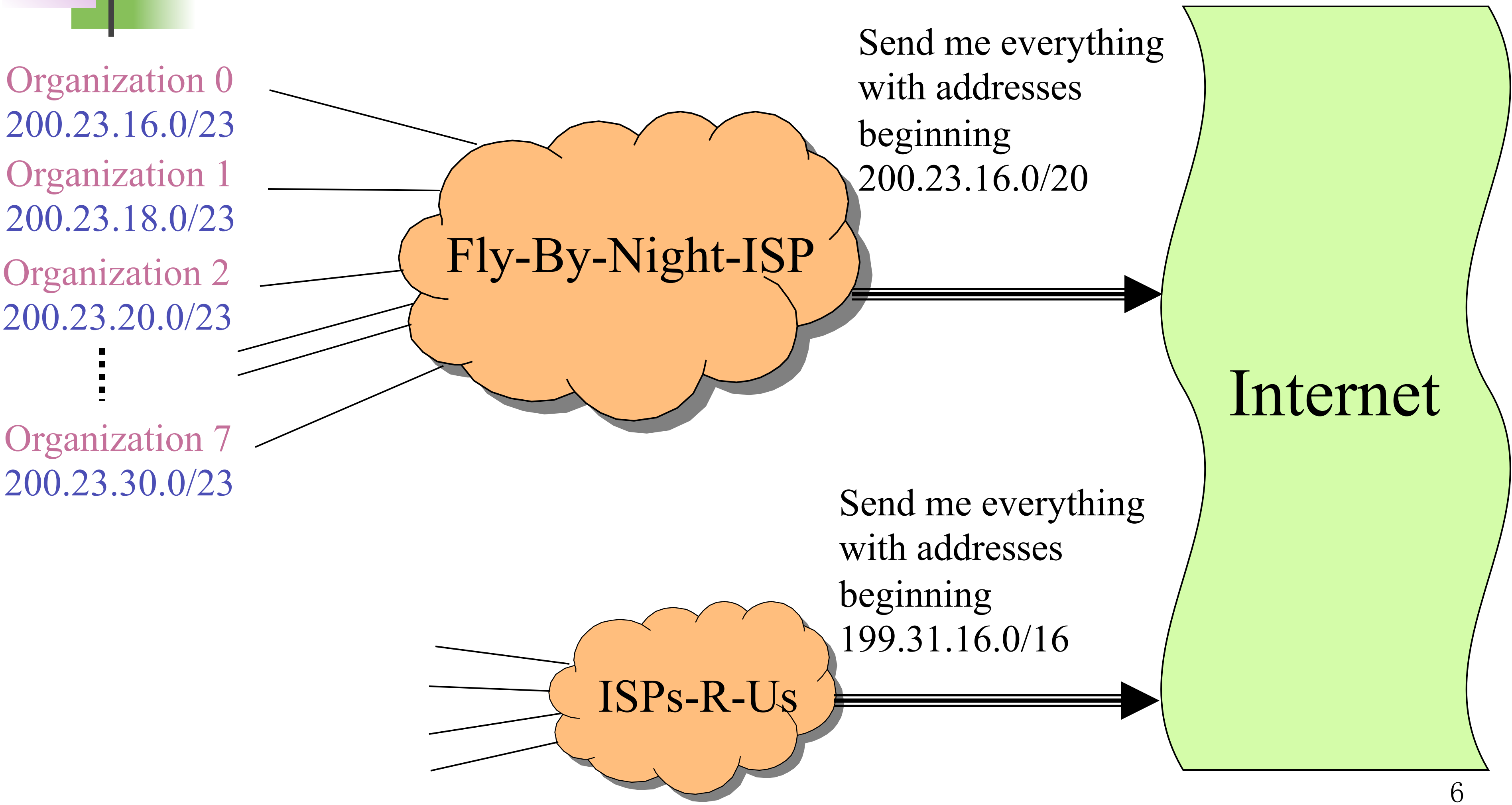
- In order to obtain a block of IP addresses for use within an organization's subnet, a network administrator might first contact its ISP.
- The ISP would provide addresses from a larger block of addresses that had already been allocated to it.
- For example, the ISP may itself have been allocated the address block 200.23.16.0/20.
- It in turn could divide its address block into 8 equal sized contiguous address blocks and give out one of these address blocks to each of the eight organizations that are supported by this ISP.

The eight address blocks

ISP's block	200.23.16.0/20	$\overbrace{11001000\ 00010111\ 0001}^{20}\overbrace{000}^{3}0\ 00000000$

Organization 0	200.23.16.0/23	$\overbrace{11001000\ 00010111\ 0001000}^{23}0\ 00000000$
Organization 1	200.23.18.0/23	11001000 00010111 00010010 00000000
Organization 2	200.23.20.0/23	11001000 00010111 00010100 00000000
Organization 3	200.23.22.0/23	11001000 00010111 00010110 00000000
Organization 4	200.23.24.0/23	11001000 00010111 00011000 00000000
Organization 5	200.23.26.0/23	11001000 00010111 00011010 00000000
Organization 6	200.23.28.0/23	11001000 00010111 00011100 00000000
Organization 7	200.23.30.0/23	11001000 00010111 00011110 00000000

Hierarchical Addressing and Route Aggregation



Address Aggregation And Route Aggregation

- As was shown in the previous Figure, the ISP Fly-By-Night advertises to the outside world that it should be sent any datagrams whose first 20 address bits match 200.23.16.0/20.
- The rest of the world need not know that within the address block 200.23.16.0/20 there are in fact eight other organizations, each with their own subnets.
- This ability to use a single prefix to advertise multiple networks is often referred to as address aggregation (also route aggregation or route summarization).
- This works extremely well when addresses are allocated in blocks to ISPs and then from ISPs to client organizations.

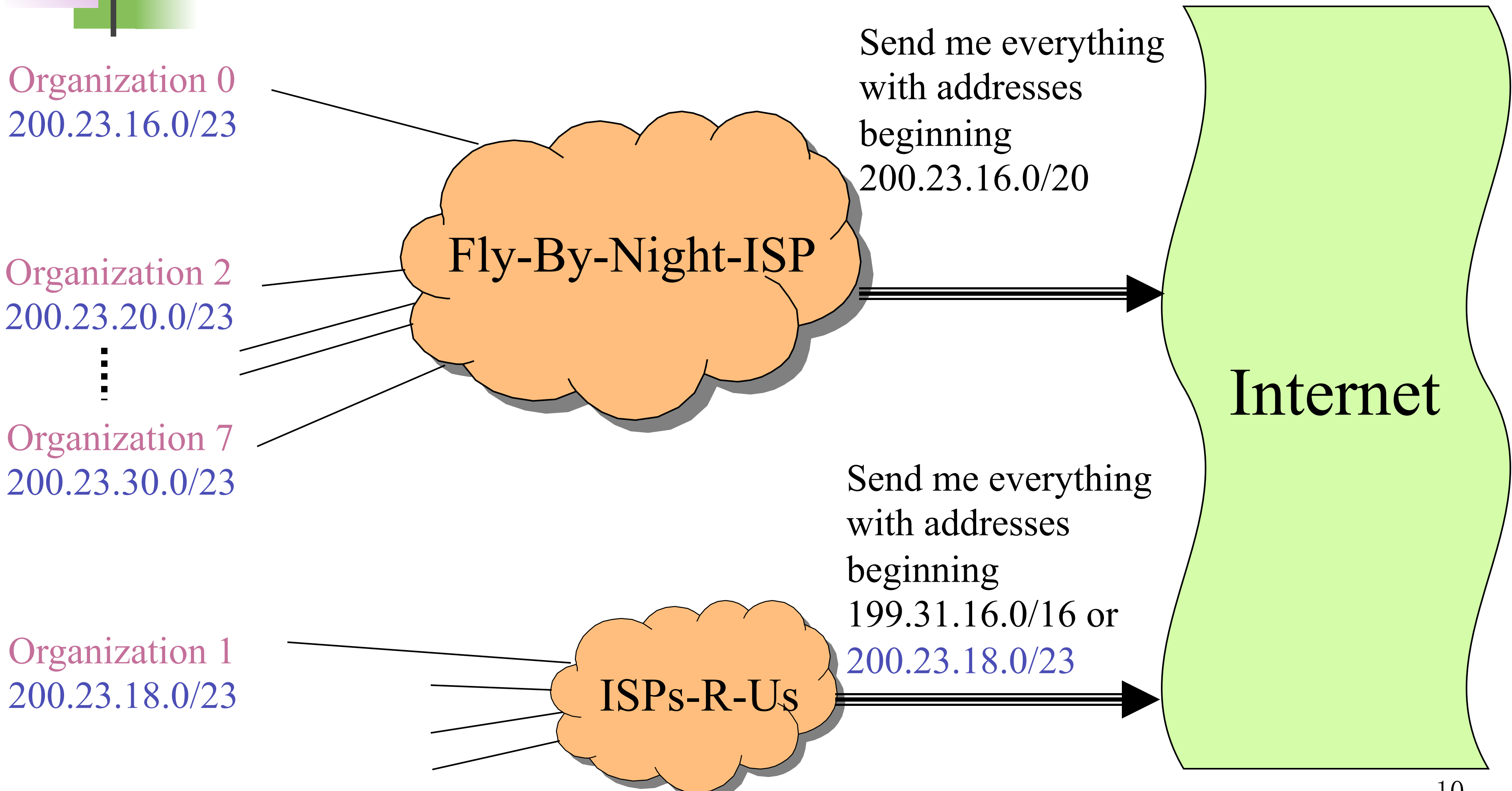
What if the addresses are not allocated in such a hierarchical manner?

- For example , what would happen if ISP Fly-By-Night acquires ISPs-R-Us and then has Organization 1 connect to the Internet through its subsidiary ISPs-R-Us?
- As was shown in the Figure, ISPs-R-Us owns the address block 199.31.0.0/16 but Organization 1's IP addresses are unfortunately outside of this address block.
- What should be done here?

Proposed Solutions

- Organization 1 could renumber all of its routers and hosts to have addresses within the ISPs-R-Us address block.
  - It's a costly solution.
  - Organization 1 might well be reassigned to another subsidiary in the future.
- Organization 1 keeps its IP addresses in 200.23.18.0/23 and ISPs-R-Us advertises the block of addresses for Organization 1 (in addition to its own block of addresses.)
  - When routers in the Internet see the address block 200.23.16.0/20 (from Fly-By-Night) and 200.23.18.0/23 (from ISPs-R-Us), and want to route to an address in the block 200.23.18.0/23, they will use longest prefix matching and route towards ISPs-R-Us as it advertises the longest (most specific) address prefix that matches the destination address.

ISPs-R-Us has a more specific route to Organization 1



Distance-Vector Routing Protocol

- One of the two main classes of routing protocols used in the computer networks (e.g. RIP).
- Uses the *Bellman-Ford algorithm* to calculate paths
- A distance-vector routing protocol requires that
  - A router informs its neighbors of topology changes
    - Periodically
    - Whenever a change is detected in the topology of a network
  - Unlike link-state protocols doesn't require the router to inform all the nodes in a network of topology changes
    - Hence less computational complexity and message overhead

Bellman-Ford Algorithm

- A distributed version of Bellman–Ford algorithm is used in distance-vector routing protocols, (e.g. RIP).
  - Each node (i.e. router) calculates the distances between itself and all other nodes and stores this information as a table.
  - Each node sends its table to all neighboring nodes.
  - When a node receives distance tables from its neighbors, it calculates the shortest routes to all other nodes and updates its own table to reflect any changes.
- Disadvantages of the Bellman–Ford algorithm in this context
  - Scalability
  - Slow convergence
  - Count-to-infinity problem
    - If failure of a link or a node renders a node unreachable from some other nodes, those nodes may indefinitely and gradually increase their cost estimates of the distance to the unreachable node, and routing loops may also be formed.

Count-to-Infinity Problem

- Consider an example topology A—B—C—D—E (hop-count is the metric)
  - A goes down.
  - B does not receive the vector update from A so it concludes that its route of cost 1 to A is no longer available.
  - C doesn't know yet that A is down and tells B that A is 2 hops away form it
  - The wrong info propagates until it reaches infinity.
  - The algorithm then corrects itself using the "Relax property" of Bellman Ford.

- **Partial Solutions**
  - ***Split Horizon:*** prohibits a router from advertising a route back out the interface from which it was learned
  - ***Split Horizon with poison reverse:*** Allows a router to advertise the route back to the router that is used to reach the destination, but marks the advertisement as unreachable.
  - More theoretical than practical, allows more scalable and complex DV protocols

Problem #2: How the *Distance-Vector* Algorithm Builds the Routing Table for Node z

- Consider the network shown below and assume that each node initially knows the costs to each of its neighbors. Consider the distance vector algorithm and show the distance tables entries at node z.

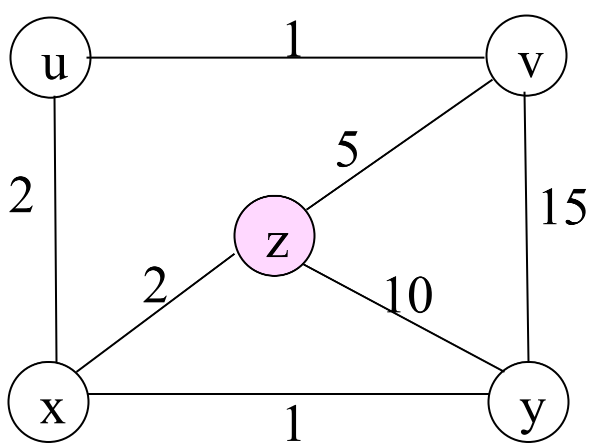
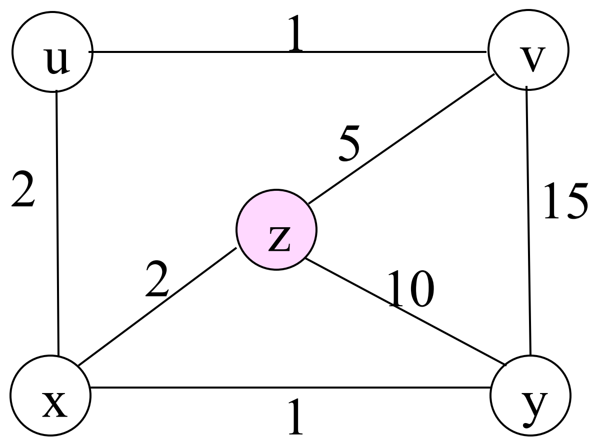


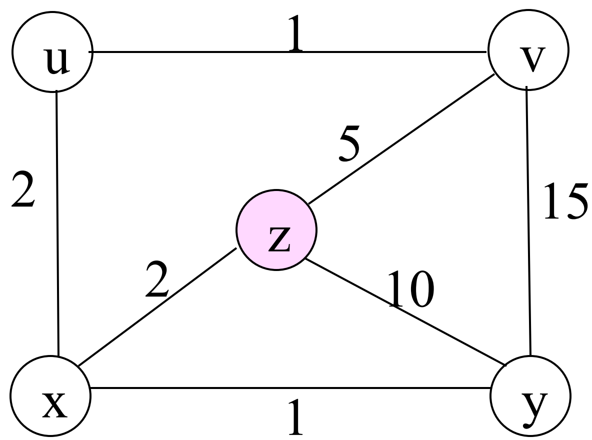
Table at Node Z: Step 1



Cost to

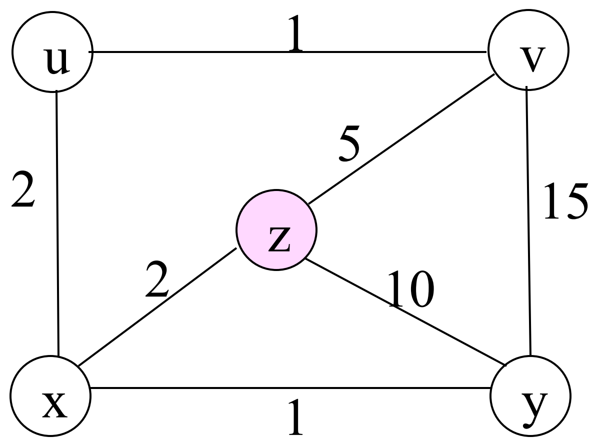
		u	v	x	y	z
	v	∞	∞	∞	∞	∞
From	x	∞	∞	∞	∞	∞
	y	∞	∞	∞	∞	∞
	z	∞	5	2	10	0

Table at Node Z: Step 2



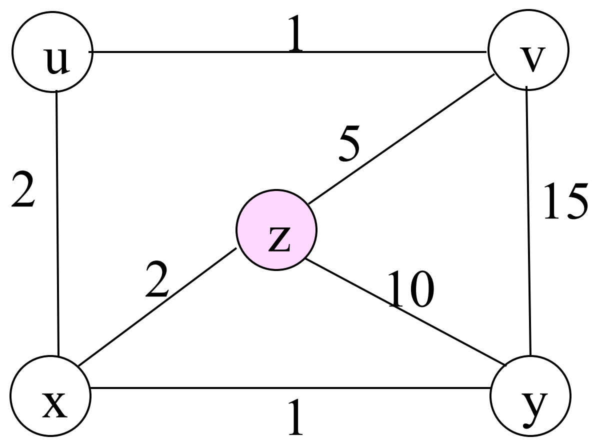
		Cost to				
		u	v	x	y	z
	v	1	0	∞	15	5
From	x	2	∞	0	1	2
	y	∞	15	1	0	10
	z	4	5	2	3	0

Table at Node Z: Step 3



		Cost to				
		u	v	x	y	z
	v	1	0	3	15	5
From	x	2	3	0	1	2
	y	3	15	1	0	3
	z	4	5	2	3	0

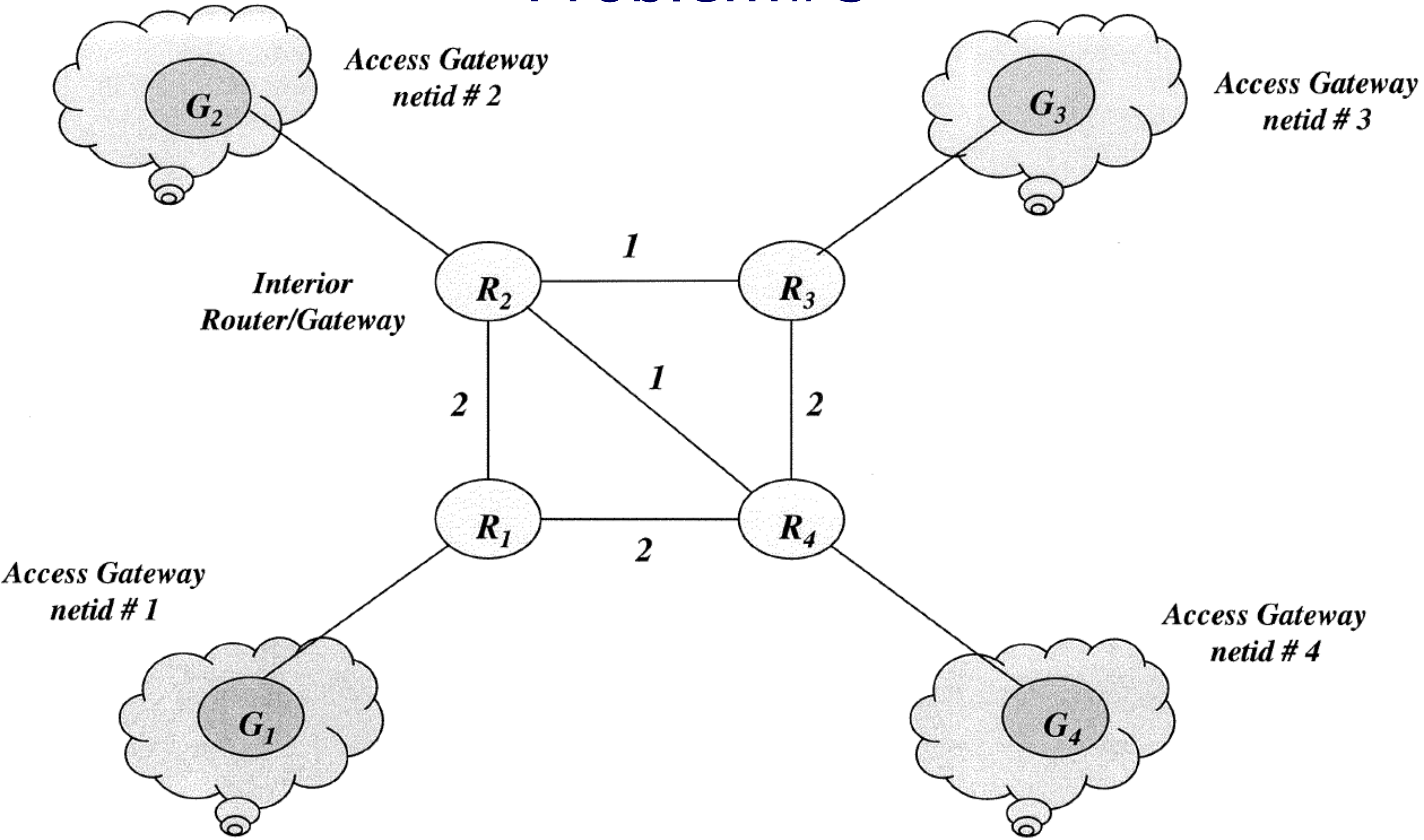
Table at Node Z: Step 4

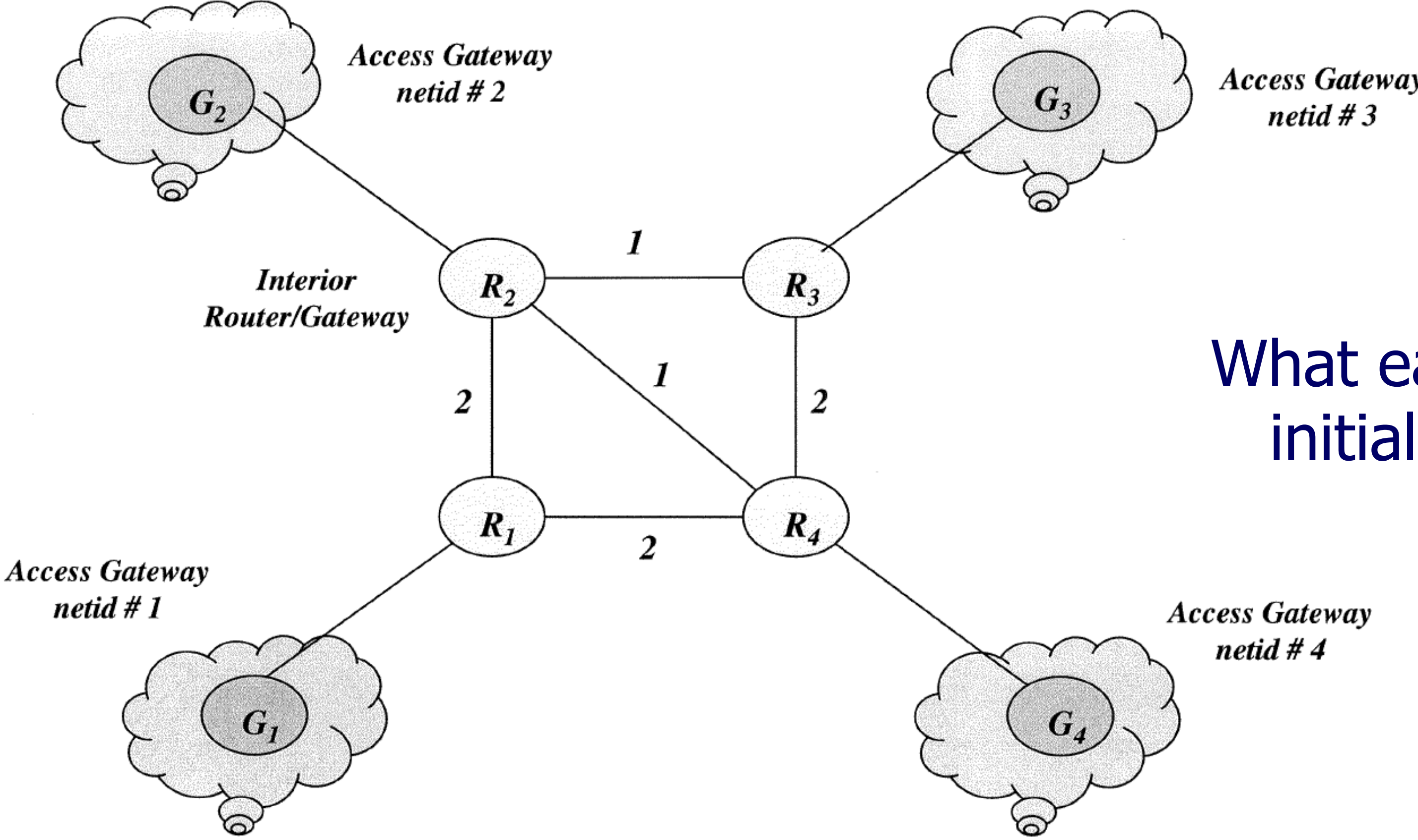


		Cost to				
		u	v	x	y	z
	v	1	0	3	4	5
From	x	2	3	0	1	2
	y	3	4	1	0	3
	z	4	5	2	3	0

Example of a DV Routing

Problem#3





What each router initially knows

Info at	(Next Hop, Distance to)			
	netid1	netid2	netid3	netid4
R1	R1,0	∞	∞	∞
R2	∞	R2,0	∞	∞
R3	∞	∞	R3,0	∞
R4	∞	∞	∞	R4,0

Initial/Intermediate/Final Routing Tables

@ R_1

Netid	*Next Hop, D*
1	R_1, 0

R_2 R_4

Netid	*Next Hop, D*
1	R_1, 0
2	R_2, 2
4	R_4, 2

R_2 R_4

Netid	*Next Hop, D*
1	R_1, 0
2	R_2, 2
3	R_2, 3
4	R_4, 2

@ R_2

Netid	*Next Hop, D*
2	R_2, 0

R_1 R_3 R_4

Netid	*Next Hop, D*
1	R_1, 2
2	R_2, 0
3	R_3, 1
4	R_4, 1

R_1 R_3 R_4

Netid	*Next Hop, D*
1	R_1, 2
2	R_2, 0
3	R_3, 1
4	R_4, 1

@ R_3

Netid	*Next Hop, D*
3	R_3, 0

R_2 R_4

Netid	*Next Hop, D*
2	R_2, 1
3	R_3, 0
4	R_4, 2

R_2 R_4

Netid	*Next Hop, D*
1	R_2, 3
2	R_2, 1
3	R_3, 0
4	R_4, 2

@ R_4

Netid	*Next Hop, D*
4	R_4, 0

R_1 R_2 R_3

Netid	*Next Hop, D*
1	R_1, 2
2	R_2, 1
3	R_3, 2
4	R_4, 0

R_1 R_2 R_3

Netid	*Next Hop, D*
1	R_1, 2
2	R_2, 1
3	R_3, 2
4	R_4, 0