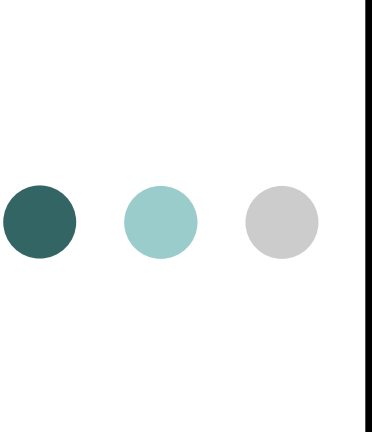




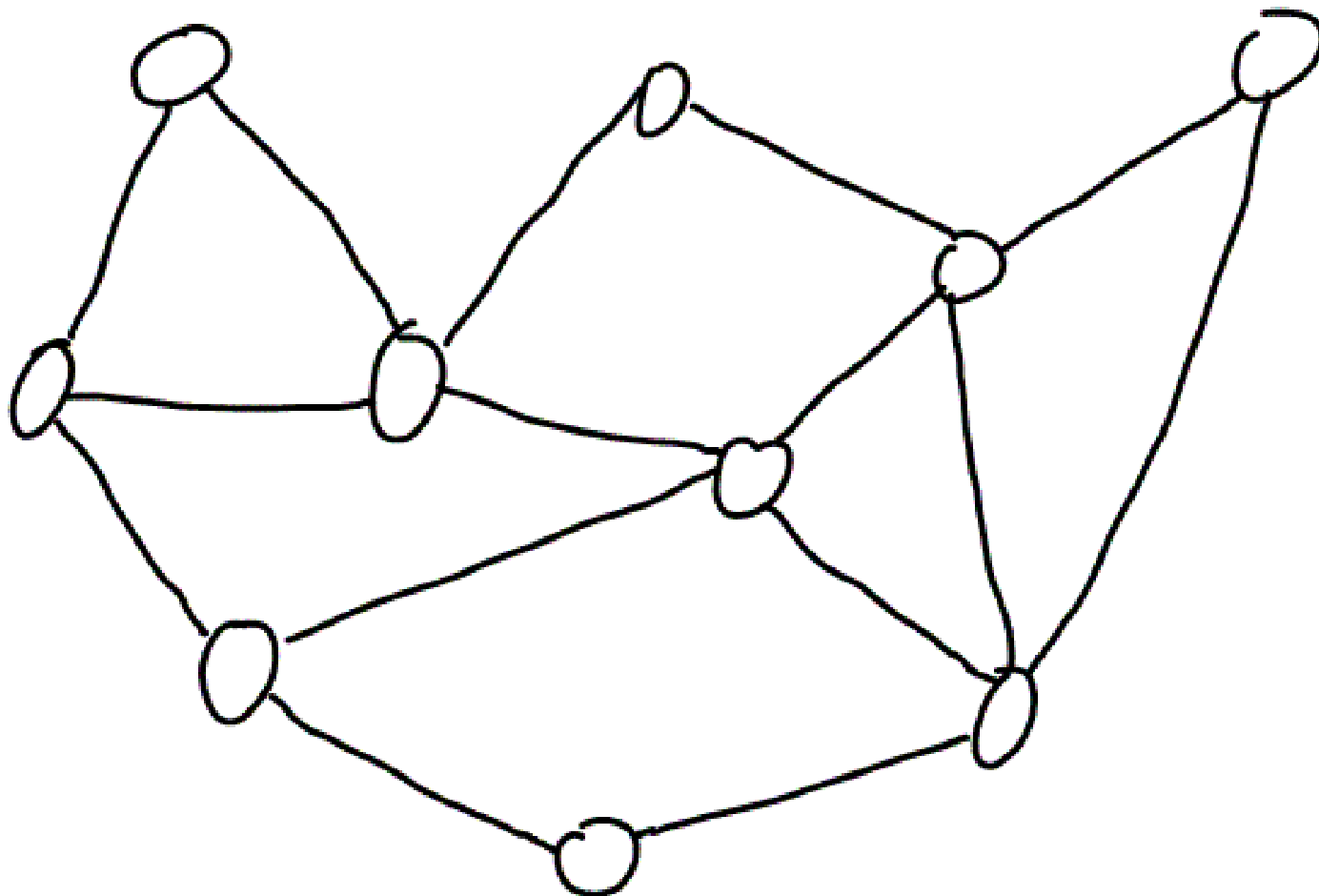
CS419: Computer Networks

Lecture 5, Part 2: Feb 16, 2005

Internet Routing:

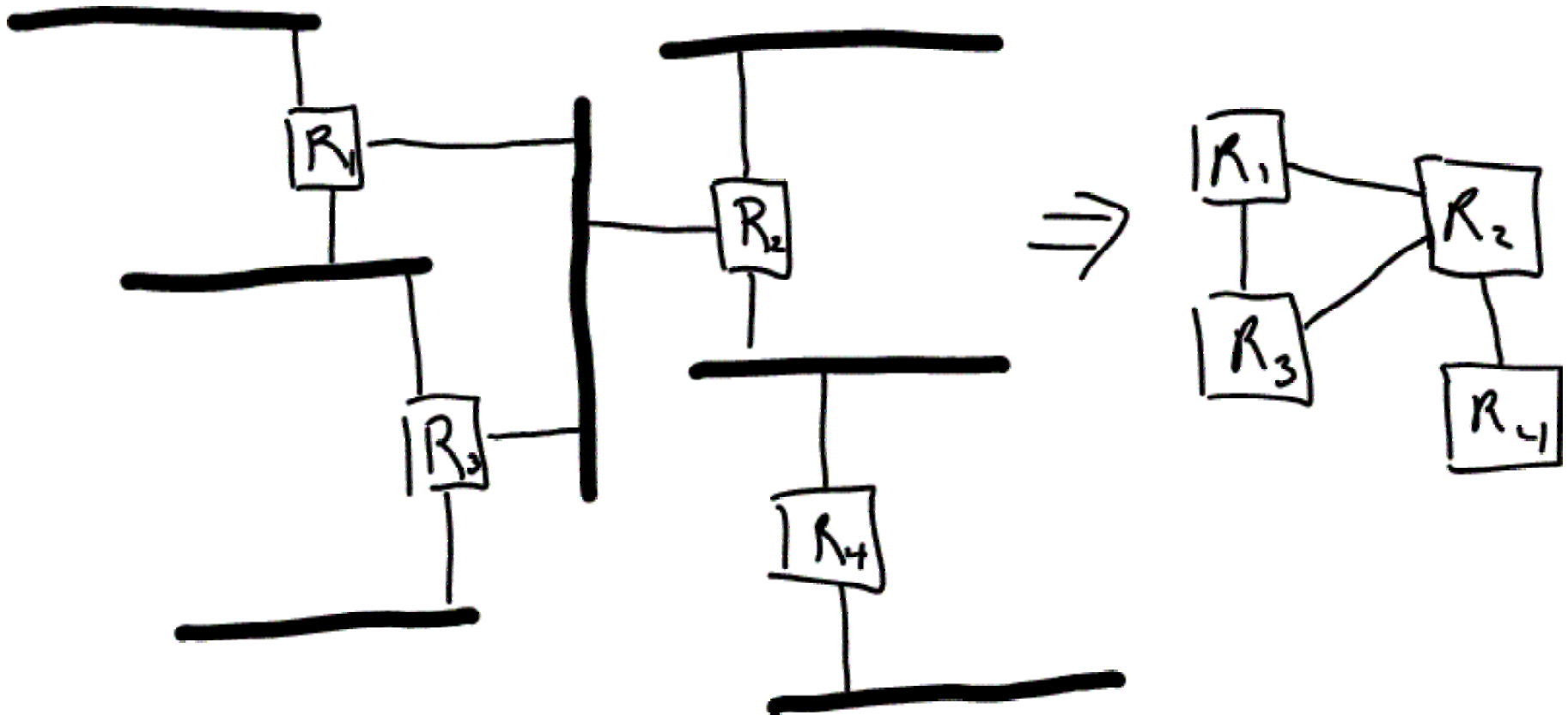
● ● ● | This is a graph

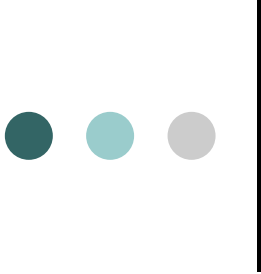
CS419



Networks can be modeled as a graph

CS419



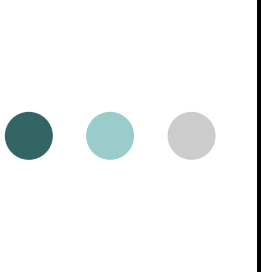


Trees on graphs



CS419

- You can superimpose a *directed tree* on a *graph*
 - No *loops* (or *cycles*)
- This tree is rooted at a *destination*
- Every directed *edge* (link) in this tree represents a “next hop”
- Forming these trees (per destination) is the essence of the routing problem
- By the way, what do you get if you reverse the directions of the tree edges?

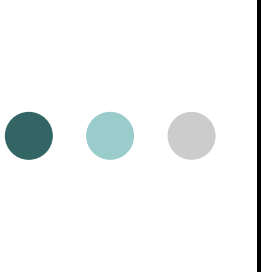


Trees on graphs



CS419

- Lots of trees are possible
- Two of interest:
 - Shortest path spanning tree
 - Sum of weights on links from any node to destination is the smallest
 - Minimum weight spanning tree
 - Sum of weights of all links is the smallest
- In the internet, we calculate shortest paths

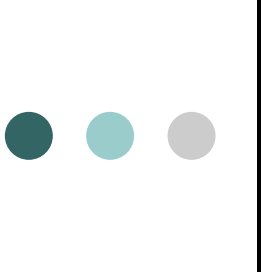


Setting costs on links



CS419

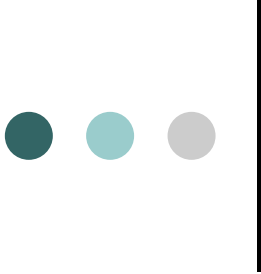
- Each “link” is actually two unidirectional links
 - But we’ll pretend they are bidirectional in the lecture for convenience
- One way: set link cost as the inverse of the BW
 - i.e., take highest BW link, give it a cost of one, weight all other links inverse proportionally
- Does this work?



Setting costs on links

CS419

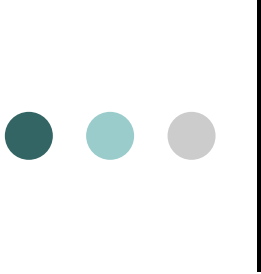
- Ultimate goal is to balance traffic over links
 - No overloaded or underloaded links
- Load on links depends on traffic matrix
 - That is, amount of traffic between every src-dest pair
- In practice, costs are “hand tuned” to get good balance
 - Of course, BW is added or removed as needed
- Link underloaded? Lower its cost a bit . . .



Can we dynamically set costs?

CS419

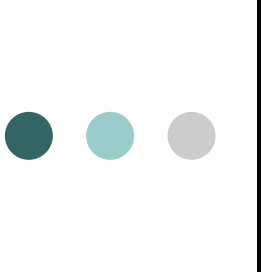
- Measure volume of traffic, increase cost as volume increases, decrease cost as volume decreases
- It turns out that this is hard to get right
 - Route oscillations---must be damped



Can we dynamically set costs?

CS419

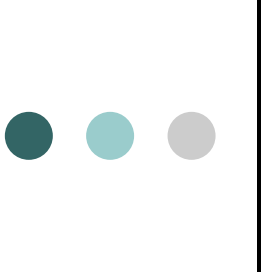
- If network is lightly loaded, traffic-sensitive routing doesn't help
 - All routes are good
- If network is overloaded, traffic-sensitive routing also doesn't help
 - Alternate routes also bad
- What you really want is to throttle sources at times of overload. **This is the big win.**



Multiple link metrics?

CS419

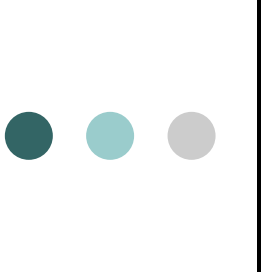
- What if we care about BW and latency and MTU and QoS???
- Many people have studied multi-metric routing
 - It gets complex, and it is hard to figure out what to do with it
- In practice: Over-provision, throttle sources!



Three classes of IP routing algorithm in the Internet today

CS419

- Distance-vector
 - RIP
- Distance-path
 - A variant of distance-vector
 - BGP
- Link-state
 - OSPF, IS-IS



Distance Vector



CS419

- Also known as Bellman-Ford
- Each node's routing table:
 - The distance to each destination via each neighbor
- Building the forwarding table:
 - The next hop to each destination is the neighbor with the shortest distance
- The algorithm:
 - Periodically tell each neighbor the shortest distance to all destinations
 - This is the so-called “distance vector”

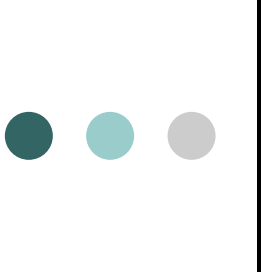
Example RIB, FIB, and routing update message

CS419

<u>RIB</u>	N ₁	N ₂	N ₃
D ₁	5	1	7
D ₂	2	9	3
D ₃	11	2	6
D ₄	3	3	4

<u>FIB</u>	Dest	Next Hop
	D ₁	N ₂
	D ₂	N ₁
	D ₃	N ₂
	D ₄	N ₁

Update Message: (D₁:1)(D₂:2)(D₃:2)(D₄:3)

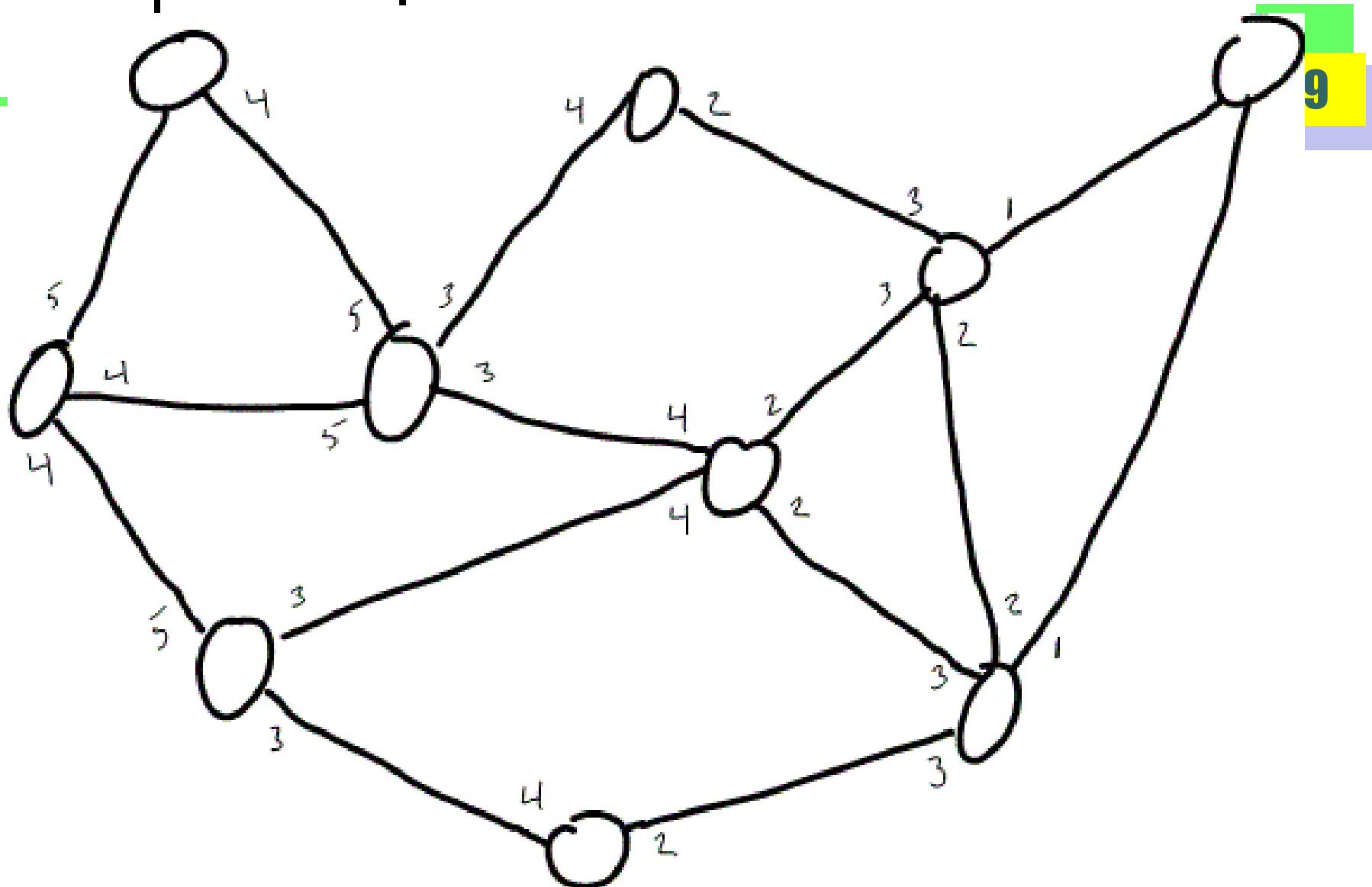


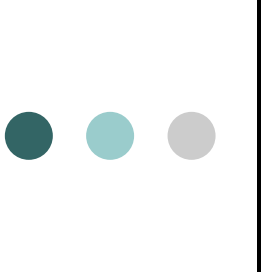
Examples



CS419

- Show establishment of tree
- Show link addition and changes in tree
- Show link deletion and changes in tree
- Show node removal and count-to-infinity



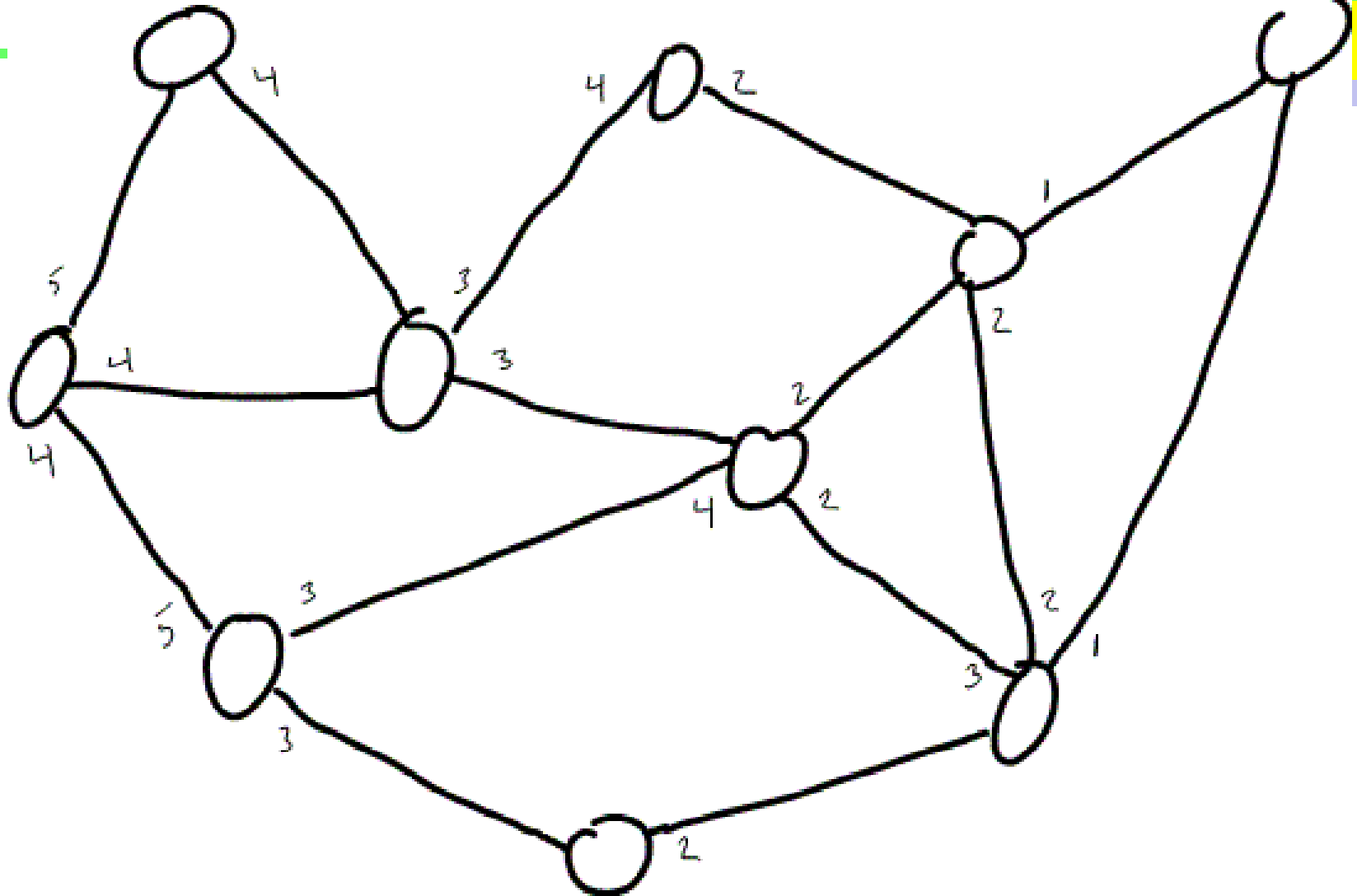


Count-to-infinity fix

CS419

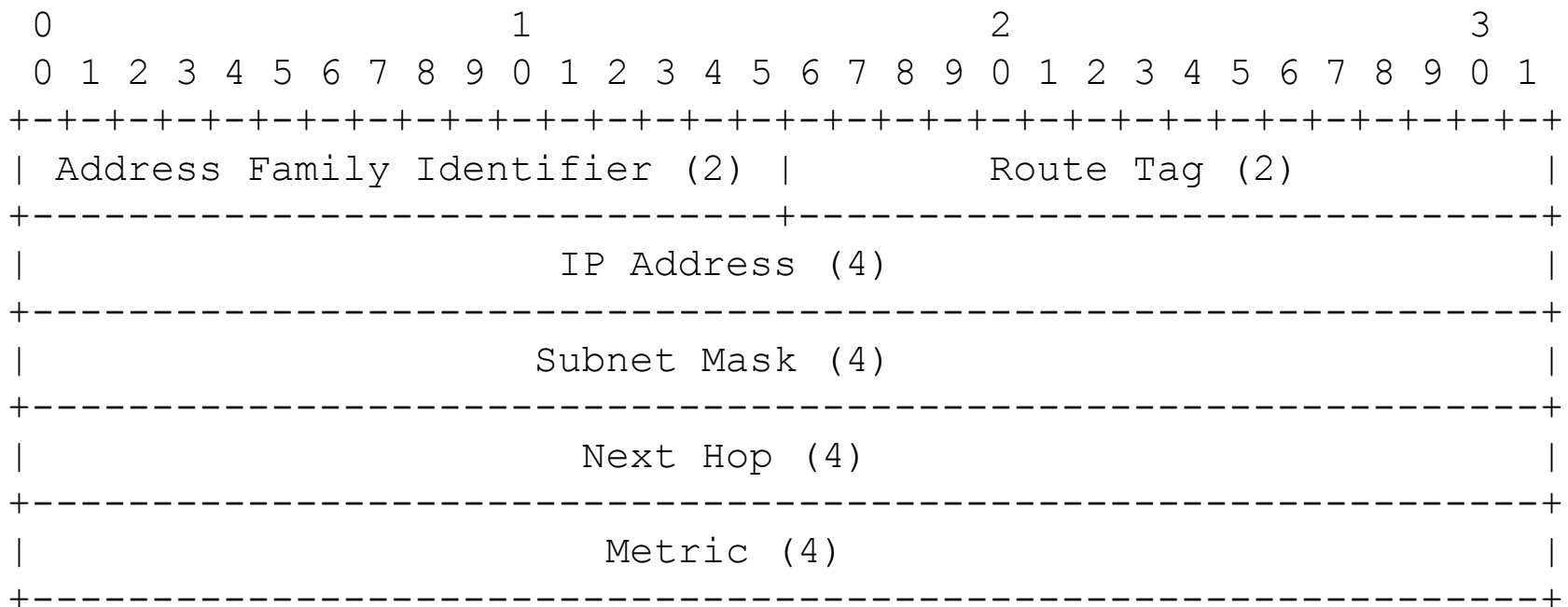
- Split horizon
 - Don't advertise reachability to a neighbor if the destination is reached via that neighbor
- Also triggered updates
 - Instantly report changes (not just periodically)
 - Count-to-infinity fast!
- This fixes “ping-pong” CTI, but doesn't solve the general problem...

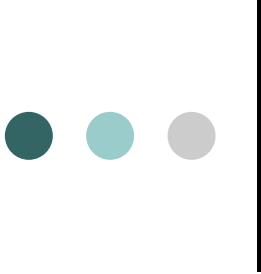
Example with split horizon



RIP-2 header (RFC 2453)

CS419





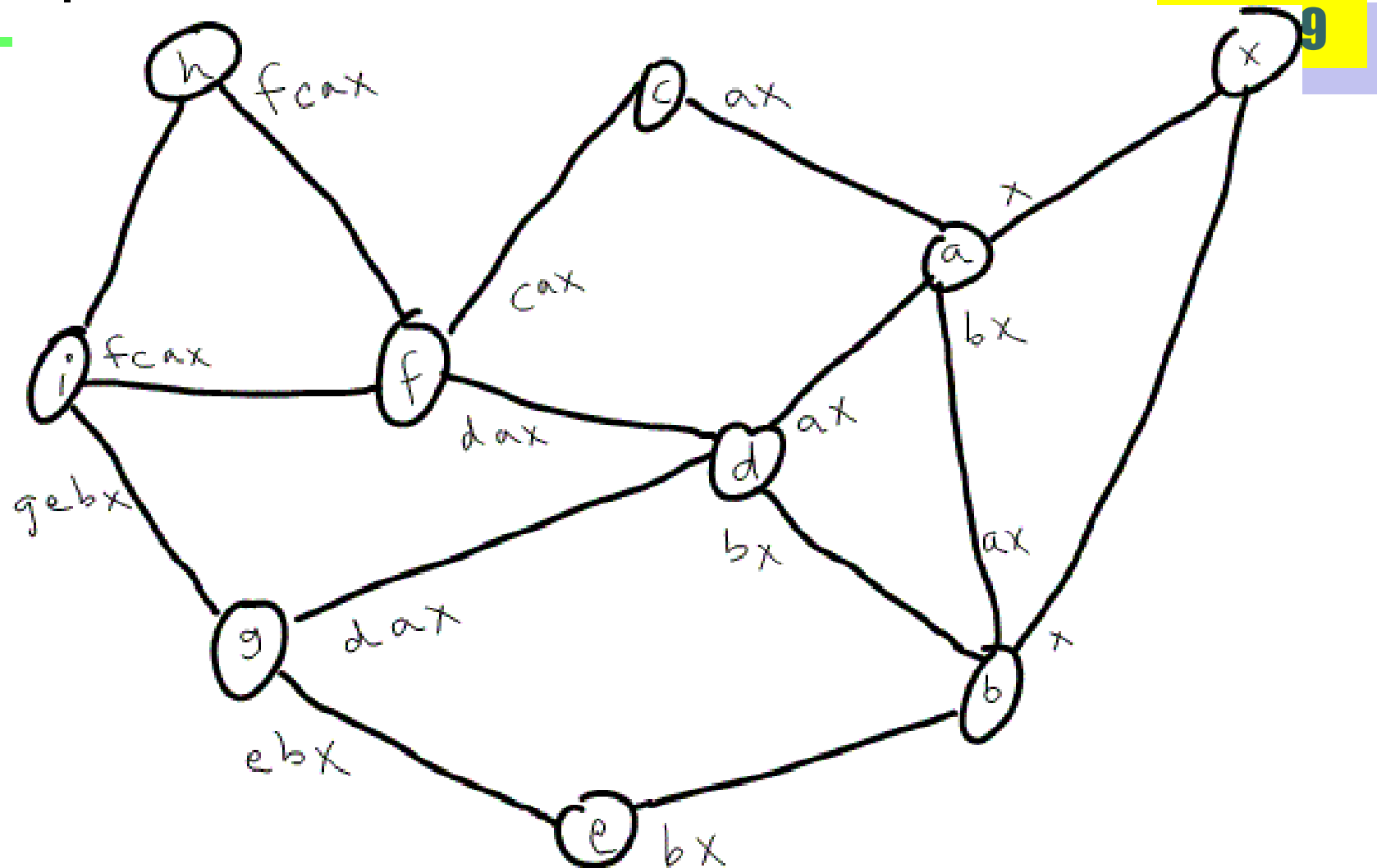
Distance-path

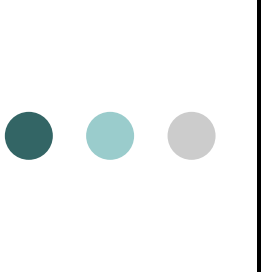


CS419

- The problem with distance-vector is that a node never knows whether a path loops back through itself
- With distance-path algorithm, the entire path to the destination is reported
 - This is not so much overhead, because network diameters are generally small

Distance-path example





Distance-path pros and cons

CS419

- New paths have to be advertised even when the distance doesn't change
 - More traffic overhead...this has caused havoc in BGP
- More policy control (BGP)
 - Path is known, so can make more intelligent selection among alternatives
 - But still “hostage” to policy decisions made before you in the path