6.033 in the news

Act now - 3G is going away in 2022

We’re phasing out our 3G network in February 2022 to make room for an even better one that will improve your experience. Find out how to get ready.

https://www.att.com/support/article/wireless/KM1324171/

If left unaddressed, the stakes could be high in certain cases. Millions of cars, for example, may no longer have the ability to contact first responders after a collision or receive updates such as location or traffic alerts for built-in GPS systems. Some vehicles, including Chevrolet, Buick and Cadillac, have software upgrades for drivers to connect their systems to a 4G network, but other models will reportedly lose this feature for good.

"A few million connected devices in the smart home space still need to be replaced, including my meter for my solar panels," said Roger Entner, analyst and founder of Recon Analytics. "Some companies started reaching out to their customers over the past 2 years informing them that service would soon shut off, but as of 6 months ago, many products still haven’t gotten around to replacing them yet."

https://www.cnn.com/2022/02/15/tech/3g-network-shut-down/index.html
Act now - 3G is going away in 2022

We’re phasing out our 3G network in February 2022 to make room for an even better one that will improve your experience. Find out how to get ready.

https://www.att.com/support/article/wireless/KM1324171/

It's not the first time a network has been phased out nor will it be the last. The 3G shutdown is primarily intended to re-use the spectrum for 4G and 5G, which are newer standards, better technologies and more efficient. The same thing happened with 2G, which AT&T and Verizon shut down around the end of 2017; T-Mobile plans to shut its 2G network in December.

Last month, AT&T and Verizon turned on C-band 5G networks, an important set of higher radio frequencies that will supercharge the internet. The change will allow users to, for example, stream a Netflix movie in 4K resolution or download a movie in seconds. (Verizon said its C-band speeds reach nearly one gigabyte per second, about 10 times as fast as 4G LTE.)
Lecture #8: Routing
distance-vector, link-state, and how they scale
on the Internet, we have to solve all of the “normal” networking problems (addressing, routing, transport) at massive scale, while supporting a diverse group of applications and competing economic interests.
1970s: flexibility and layering
1978: ARPAnet
1980s: growth → change
1980s late: growth → problems
1993: commercialization

hosts.txt
distance-vector routing
TCP, UDP
OSPF, EGP, DNS
(con a link-state routing protocol)
congestion collapse
policy routing
CIDR

application
the things that actually generate traffic

transport
sharing the network, reliability (or not)
examples: TCP, UDP

network
naming, addressing, routing
examples: IP

link
communication between two directly-connected nodes
examples: ethernet, bluetooth, 802.11 (wifi)

today: routing in general
(not specifically on the Internet)

CAIDA’s IPv4 AS Core,
January 2020
(https://www.caida.org/projects/cartography/as-core/2020/)
goal of a routing protocol: allow each switch to know, for every node dst in the network, a **minimum-cost** route to dst
**distributed routing:** nodes build up their own routing tables, rather than having tables given to them by a centralized authority

1. Nodes learn about their neighbors via the HELLO protocol

2. Nodes learn about other reachable nodes via advertisements

3. Nodes determine the minimum-cost routes (of the routes they know about)

All of these steps happen *periodically*, which allows the routing protocol to detect and respond to failures, and adapt to other changes in the network.
link-state routing: disseminate full topology information so that nodes can run a shortest-path algorithm.

A’s advertisement: [(B,7),(D,2),(F,1)]
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm.

A’s advertisement: $[(B,7),(D,2),(F,1)]$
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm.

A’s advertisement: 

\[((B,7),(D,2),(F,1)]

**link state**

what’s in an advertisement

its **link costs** to each of its **neighbors**

who gets a node’s advertisement

effectively, every other node (via flooding)
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding)

Its **link costs** to each of its **neighbors**

**what’s in an advertisement**

- A’s advertisement: [(B, 7), (D, 2), (F, 1)]

Nodes keep track of which advertisements they’ve forwarded so that they don’t re-forward them. They can also be a bit smarter about flooding, and not forward an advertisement back to the node that sent it.
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding).

Nodes **integrate** advertisements by running Dijkstra’s Algorithm.

**link state**

what’s in an advertisement

its **link costs** to each of its **neighbors**

who gets a node’s advertisement
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm.

![Diagram of a network with nodes A, B, C, D, E, and F connected by weighted edges.]

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>C</td>
<td>?</td>
<td>∞</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>?</td>
<td>∞</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

F does not provide A with a better route to D.

**link state**

what’s in an advertisement

its **link costs** to each of its **neighbors**

who gets a node’s advertisement

effectively, every other node (via flooding)
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding)

- its **link costs** to each of its **neighbors**

---

A's routing table

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>C</td>
<td>?</td>
<td>∞</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-F</td>
<td>6</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

= the cost from A to F + the cost from F to E

---

**link state**

what's in an advertisement

who gets a node's advertisement
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm.

- **Link state:**
  - What's in an advertisement
    - Its link costs to each of its neighbors
  - Who gets a node's advertisement
    - Effectively, every other node (via flooding)

---

**A's routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>?</td>
<td>$\infty$</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-F</td>
<td>6</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm.

**link state**
- what's in an advertisement
  - its link costs to each of its neighbors
- who gets a node's advertisement
  - effectively, every other node (via flooding)

A's routing table

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-F</td>
<td>6</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm.

Link state:
- What's in an advertisement
  - its link costs to each of its neighbors
- Who gets a node's advertisement
  - effectively, every other node (via flooding)

A's routing table:

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-F</td>
<td>6</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm.

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A's routing table

we don’t need to “visit” F; we already know the shortest path to it.
link-state routing: disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding)

link state

what’s in an advertisement
its link costs to each of its neighbors

who gets a node’s advertisement

A’s routing table

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>6</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

notice that A’s route doesn’t change, but the cost needs to update (and the actual path of the packets from A to C has changed)
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding)

A's routing table

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>6</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
link-state routing: disseminate full topology information so that nodes can run a shortest-path algorithm.

A's routing table

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>6</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
link-state routing: disseminate full topology information so that nodes can run a shortest-path algorithm.

- **link state**
  - what’s in an advertisement
    - its link costs to each of its neighbors
  - who gets a node’s advertisement
    - effectively, every other node (via flooding)
  - what happens when things fail?
    - flooding makes link-state routing very resilient to failure
  - what limits scale?
    - the overhead of flooding
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology.

A’s first advertisement: \([ (B, 7), (D, 2), (F, 1) ] \)

A could also include \((A, 0)\) here.

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A’s advertisement reflects its routing table, and right now, A only knows about its neighbors.

**link state**

- what’s in an advertisement
  - its link costs to each of its neighbors

**distance vector**

- what’s in an advertisement
  - its current costs to every node it’s aware of

- who gets a node’s advertisement
  - effectively, every other node (via flooding)

- what happens when things fail?
  - flooding makes link-state routing very resilient to failure

- what limits scale?
  - the overhead of flooding
**distance-vector routing:** disseminate information about the current *costs* to each node, rather than the actual topology.

- **Link State**
  - what’s in an advertisement: its *link costs* to each of its *neighbors*.
  - who gets a node’s advertisement: effectively, *every other node* (via flooding).
  - what happens when things fail?: flooding makes *link-state routing* very resilient to failure.

- **Distance Vector**
  - what’s in an advertisement: its *current costs* to *every node* it’s aware of.
  - who gets a node’s advertisement: only its *neighbors*.

---

**A’s first advertisement:** \[ \{(B,7),(D,2),(F,1)\} \]

A’s routing table:

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A’s advertisement reflects its routing table, and right now, A only knows about its neighbors.

The overhead of flooding limits scale.
**distance-vector routing:** disseminate information about the current *costs* to each node, rather than the actual topology.*

**link state**

- what’s in an advertisement
  - its link costs to each of its neighbors
  - its current costs to every node it’s aware of

**distance vector**

- who gets a node’s advertisement
  - effectively, every other node (via flooding)
  - only its neighbors

**what happens when things fail?**

- flooding makes link-state routing very resilient to failure

**what limits scale?**

- the overhead of flooding

---

A’s first advertisement: \([(B,7),(D,2),(F,1)]\)

A’s routing table

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A’s neighbors do **not** forward A’s advertisements; they **do** send advertisements of their own to A.
**distance-vector routing:** disseminate information about the current costs to each node, rather than the actual topology.

A’s first advertisement: [(B,7),(D,2),(F,1)]

A’s routing table:

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A’s neighbors do not forward A’s advertisements; they do send advertisements of their own to A.
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology.

- **Link State**
  - what’s in an advertisement:
    - its link costs to each of its neighbors
  - who gets a node’s advertisement:
    - effectively, every other node (via flooding)
    - only its neighbors

- **Distance Vector**
  - what happens when things fail?
    - flooding makes link-state routing very resilient to failure
  - what limits scale?
    - the overhead of flooding

---

**Graph Representation**

- **A’s routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

- **Advertisements**
  - B’s first adv: [(A,7), (C,3), (D,1)]
  - D’s first adv: [(B,1), (C,5), (E,3), (F,4)]
  - F’s first adv: [(A,1), (D,4), (E,5)]

- **A receives advertisements from B, D, and F**
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology. 

---

**link state**
- what’s in an advertisement
  - its **link costs** to each of its **neighbors**

**distance vector**
- what’s in an advertisement
  - its current costs to every node it’s aware of

---

**who gets a node’s advertisement**
- effectively, every other node (via flooding)
  - only its **neighbors**

---

**what happens when things fail?**
- flooding makes link-state routing very resilient to failure

---

**what limits scale?**
- the **overhead** of flooding

---

**A’s routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-B</td>
<td>7</td>
</tr>
<tr>
<td>C</td>
<td>A-B</td>
<td>10</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

B’s *first* adv: [(A, 7), (C, 3), (D, 1)]
**distance-vector routing:** disseminate information about the current costs to each node, rather than the actual topology.

- **link state**
  - what’s in an advertisement
    - its link costs to each of its neighbors
    - effectively, every other node (via flooding)
    - only its neighbors

- **distance vector**
  - what’s in an advertisement
    - its current costs to every node it’s aware of
  - who gets a node’s advertisement
  - what happens when things fail?
    - flooding makes link-state routing very resilient to failure
  - what limits scale?
    - the overhead of flooding

---

**A’s routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology of its link costs to each of its neighbors effectively, every other node (via flooding) only its neighbors

what's in an advertisement
its link costs to each of its neighbors its current costs to every node it's aware of

who gets a node's advertisement
effectively, every other node (via flooding) only its neighbors

what happens when things fail?
flooding makes link-state routing very resilient to failure

what limits scale?
the overhead of flooding

---

**A's routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

*F's first adv: [(A,1), (D,4), (E,5)]*
**distance-vector routing:** disseminate information about the current costs to each node, rather than the actual topology.

- **link state**
  - what’s in an advertisement: its link costs to each of its neighbors
  - who gets a node’s advertisement: effectively, every other node (via flooding)
  - what happens when things fail? flooding makes link-state routing very resilient to failure

- **distance vector**
  - what’s in an advertisement: its current costs to every node it’s aware of
  - who gets a node’s advertisement: only its neighbors

---

**A’s routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

This is A’s routing table after one round of advertisements; note that it does not have knowledge of the min-cost route to C yet.

---

Katrina LaCurts | lacurts@mit.edu | 6.033 2022
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology.

- **link state**
  - what’s in an advertisement
    - its link costs to each of its neighbors
    - every other node (via flooding)
  - who gets a node’s advertisement
    - only its neighbors

- **distance vector**
  - what’s in an advertisement
    - its current costs to every node it’s aware of
  - who gets a node’s advertisement
    - effectively, every other node (via flooding)

- **what happens when things fail?**
  - flooding makes link-state routing very resilient to failure

- **what limits scale?**
  - the overhead of flooding

---

**A’s routing table**

<table>
<thead>
<tr>
<th>dst</th>
<th>route</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>A-D</td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>A-D</td>
<td>7</td>
</tr>
<tr>
<td>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>A-D</td>
<td>5</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A’s second adv: \[(B,3), (C,7), (D,2), (E,5), (F,1)\]

A will learn about the correct min-cost path to C in the next round of advertisements; try that out for yourself!
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology.

- **link state**: what’s in an advertisement
  - its link costs to each of its neighbors
  - effectively, every other node (via flooding)
- **distance vector**: what’s in an advertisement
  - its current costs to every node it’s aware of
  - only its neighbors
- **what happens when things fail?**: flooding makes link-state routing very resilient to failure
  - failures can be complicated because of timing
- **what limits scale?**: the overhead of flooding
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology.

A sends advertisements at $t=0, 10, 20, \ldots$; B sends advertisements at $t=5, 15, 25, \ldots$

*every link has cost 1*

<table>
<thead>
<tr>
<th>Time</th>
<th>A: Self, 0</th>
<th>A: B-&gt;A, 1</th>
<th>B: A-&gt;B, 1</th>
<th>B: Self, 0</th>
<th>C: A-&gt;B, 2</th>
<th>C: None, inf</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t=9$</td>
<td>B: Self, 0</td>
<td>B: A-&gt;B, 1</td>
<td>A: B-&gt;A, 1</td>
<td>C: B-&gt;A, 3</td>
<td>A: B-&gt;A, 1</td>
<td>C: B-&gt;A, 3</td>
</tr>
<tr>
<td></td>
<td>(2+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

continues until both costs to C are INFINITY

---

**link state**

- **what's in an advertisement**
  - its **link costs** to each of its neighbors

**distance vector**

- its current costs to every node it's aware of

**who gets a node's advertisement**

- effectively, every other node (via flooding)
  - only its neighbors

**what happens when things fail?**

- flooding makes link-state routing very resilient to failure
  - failures can be complicated because of timing

**what limits scale?**

- the overhead of flooding

Katrina LaCurts | lacurts@mit.edu | 6.033 2022
**distance-vector routing**: disseminate information about the current costs to each node, rather than the actual topology

A sends advertisements at t=0, 10, 20, ...; B sends advertisements at t=5, 15, 25, ...
every link has cost 1

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: Self, 0</td>
<td>A: B-&gt;A, 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B: A-&gt;B, 1</td>
<td>B: Self, 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C: A-&gt;B, 2</td>
<td>C: None, inf</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**link state**
- **what's in an advertisement**
  - its link costs to each of its neighbors

**distance vector**
- **its current costs to every node it's aware of**

**who gets a node's advertisement**
- effectively, every other node (via flooding)
- only its neighbors

**what happens when things fail?**
- flooding makes link-state routing very resilient to failure
- failures can be complicated because of timing

**what limits scale?**
- the overhead of flooding

**new strategy ("split horizon")**: don't send advertisements about a route to the node providing the route

split horizon takes care of this particular case

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A: Self, 0</td>
<td>A: B-&gt;A, 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B: A-&gt;B, 1</td>
<td>B: Self, 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C: None, inf</td>
<td>C: None, inf</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[(A,0)\]

\[(B,0),(C,\text{inf})\]
**distance-vector routing:** disseminate information about the current *costs* to each node, rather than the actual topology.

<table>
<thead>
<tr>
<th>D</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>C: D-&gt;B, 2</td>
<td>C: A-&gt;B, 2</td>
<td>C: None, inf</td>
<td>B&lt;-&gt;C fails</td>
</tr>
<tr>
<td>C: None, inf</td>
<td>C: A-&gt;B, 2</td>
<td>C: None, inf</td>
<td>B’s advertisement to A gets lost (so A makes no changes)</td>
</tr>
<tr>
<td>C: D-&gt;A, 3</td>
<td>C: A-&gt;B, 2</td>
<td>C: None, inf</td>
<td>A advertises about C to D (not to B because of split horizon)</td>
</tr>
<tr>
<td>C: D-&gt;A, 3</td>
<td>C: A-&gt;B, 2</td>
<td>C: B-&gt;D, 4</td>
<td>D advertises about C to B</td>
</tr>
<tr>
<td>C: D-&gt;A, 3</td>
<td>C: A-&gt;B, 5</td>
<td>C: B-&gt;D, 4</td>
<td>B advertises about C to A</td>
</tr>
</tbody>
</table>

**continues until all costs to C are INFINITY**

**new strategy ("split horizon"):** don’t send advertisements about a route to the node providing the route.

---

**link state**

- **what’s in an advertisement**
  - its *link costs* to each of its *neighbors*
  - its *current costs* to every node it’s aware of

- **who gets a node’s advertisement**
  - **effectively, every other node** (via flooding)
  - only its *neighbors*

- **what happens when things fail?**
  - flooding makes link-state routing very resilient to failure
  - failures can be complicated because of timing

- **what limits scale?**
  - the overhead of flooding
  - failure handling
neither one of these algorithms will scale to the size of the internet, nor do either of them allow for policy routing.
IP networks can route using either distance-vector routing (RIP) or link-state routing (OSPF).