6.1800 Spring 2024

Lecture #9: Routing
distance-vector, link-state, and how they scale
6.1800 in the news

Red Sea cables have been damaged, disrupting internet traffic

By Hanna Ziady, CNN

3 minute read · Updated 9:02 AM EST, Mon March 4, 2024

London (CNN) — Damage to submarine cables in the Red Sea is disrupting telecommunications networks and forcing providers to reroute as much as a quarter of traffic between Asia, Europe and the Middle East, including internet traffic.

HGC estimates that 25% of traffic between Asia and Europe as well the Middle East has been impacted, it said in a statement Monday.

Most large telecoms companies rely on multiple undersea cable systems, allowing them to reroute traffic in the event of an outage to ensure uninterrupted service.
Red Sea cables have been damaged, disrupting internet traffic

By Hanna Ziady, CNN
3 minute read · Updated 9:02 AM EST, Mon March 4, 2024

London (CNN) — Damage to submarine cables in the Red Sea is disrupting telecommunications networks and forcing providers to reroute as much as a quarter of traffic between Asia, Europe and the Middle East, including internet traffic.

HGC estimates that 25% of traffic between Asia and Europe as well the Middle East has been impacted, it said in a statement Monday.

Most large telecoms companies rely on multiple undersea cable systems, allowing them to reroute traffic in the event of an outage to ensure uninterrupted service.
6.1800 in the past

the domain name system (DNS), which maps hostnames (eecs.mit.edu) to IP addresses (18.25.0.23)

a partial view of the DNS hierarchy. each box represents a zone. name servers within a zone keep track of that zone’s mappings

DNS client

e.g., your laptop

query sent to: 198.41.0.4

response:
the **domain name system (DNS)**, which maps hostnames (eecs.mit.edu) to IP addresses (18.25.0.23)

![Diagram of DNS hierarchy](image)

- a partial view of the DNS hierarchy. Each box represents a **zone**. Name servers within a zone keep track of that zone’s mappings.

- How does the DNS client’s query get to 198.41.0.4?

  - Query sent to: 198.41.0.4
  - Response: 198.41.0.4
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

ARPAnet

1978: growth → change

early 80s: growth → change

TCP, UDP

OSPF, EGP, DNS

1980s: growth → problems

congestion collapse

late 80s: growth → problems

policy routing

CIDR

1993: commercialization

hosts.txt
distance-vector routing

network

transport

1970s:

ARPAnet

1978: flexibility and layering

early 80s: growth → change

TCP, UDP

OSPF, EGP, DNS

1980s: growth → problems

congestion collapse

late 80s: growth → problems

policy routing

CIDR

1993: commercialization

application

the things that actually generate traffic

examples: TCP, UDP

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 \( \text{dst} \) in the network, a minimum-cost route to \( \text{dst} \)
**distributed routing:** nodes build up their own routing tables, rather than having tables given to them by a centralized authority
**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
**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
**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)
**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)

what the advertisements contain, and how the nodes use those advertisements to determine the min-cost routes, will change depending on the specific protocol
**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

what the advertisements contain, and how the nodes use those advertisements to determine the min-cost routes, will change depending on the specific protocol
link state
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm.

Its link costs to each of its neighbors.
**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**
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 effectively, every other node (via flooding)

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 effectively, every other node (via flooding)

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 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 advertisement: [(B,7),(D,2),(F,1)]
link-state routing: disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding).

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

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding).

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 extensively, every other node (via flooding)

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

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

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

- Diagram showing a network with nodes A, B, C, D, E, and F, and the link costs between them.
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm effectively, every other node (via flooding).

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

Nodes keep track of which advertisements they’ve forwarded so that they don’t re-forward them.
**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 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.

---

**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-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>
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm 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)

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

<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>

**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.

<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 routing**: disseminate full topology information so that nodes can run a shortest-path algorithm.

A node's link state contains its **link costs** to each of its neighbors, and it effectively advertises its link state to **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-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>
**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-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 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-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>
**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-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>
**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-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>

**question:** what will A’s routing table look like after we’re done visiting all of D’s neighbors?
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm.

- Each node effectively learns the **link costs** to each of its neighbors and what's in an 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-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>
**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>?</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>
**link-state routing:** disseminate full topology information so that nodes can run a shortest-path algorithm.

- Each node effectively learns the cost of its link with each of its neighbors via flooding.
- What's in an advertisement:
  - Each node advertises the cost of its link to each of its neighbors.
- Who gets a node's advertisement:
  - 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>∞</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 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 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>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 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>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>
**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

<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>

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.

In a link state routing protocol, each node maintains a routing table that includes:

- **Destination (dst)**
- **Route**
- **Cost**

<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>

**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)

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>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>
**link-state routing**: disseminate full topology information so that nodes can run a shortest-path algorithm. Link state refers to the link costs to each of its neighbors. What's in an advertisement determines 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>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.

- Each node effectively, every other node (via flooding) broadcasts its link costs to each of its neighbors.
- This information is used to calculate the shortest path to any destination node.

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:

- 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>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 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>

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

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.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

- **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.

- **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?**
**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

effectively, every other node (via flooding)

what happens when things fail?

flooding makes link-state routing very resilient to failure.
link-state routing: disseminate full topology information so that nodes can run a shortest-path algorithm.

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?
**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 *min 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**
- 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 *min costs* to each node, rather than the actual topology.
**distance-vector routing:** disseminate information about the current *min 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)
- **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>D</td>
<td>A-D</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>
**Distance-vector routing:** Disseminate information about the current *min 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
  - What happens when things fail?

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

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

### 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>

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

<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 first advertisement: 

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

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

**distance vector**

- 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 *min costs* to each node, rather than the actual topology

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

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
distance-vector routing: disseminate information about the current \textit{min costs} to each node, rather than the actual topology.

\begin{tabular}{|c|c|c|}
\hline
\textbf{dst} & \textbf{route} & \textbf{cost} \\
\hline
B & A-B & 7 \\
D & A-D & 2 \\
F & A-F & 1 \\
\hline
\end{tabular}

A’s routing table

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

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

link state

- its \textit{link costs} to each of its \textit{neighbors}

- its \textit{current costs} to every node it’s aware of

distance vector

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

who gets a node’s advertisement

- effectively, \textit{every other node} (via flooding)
- only its \textit{neighbors}

what happens when things fail?

- flooding makes link-state routing very resilient to failure

what limits scale?

- the \textit{overhead} of flooding

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min 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 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**

- 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
**distance-vector routing:** disseminate information about the current *min 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 *min 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.

link state | distance vector

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
distance-vector routing: disseminate information about the current min 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 *min 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.

---

**link state**

<table>
<thead>
<tr>
<th>what’s in an advertisement</th>
</tr>
</thead>
<tbody>
<tr>
<td>its <em>link costs</em> to each of</td>
</tr>
<tr>
<td>its <em>neighbors</em></td>
</tr>
</tbody>
</table>

**distance vector**

<table>
<thead>
<tr>
<th>what’s in an advertisement</th>
</tr>
</thead>
<tbody>
<tr>
<td>its <em>current costs</em> to</td>
</tr>
<tr>
<td>every node it’s aware of</td>
</tr>
</tbody>
</table>

who gets a node’s advertisement

<table>
<thead>
<tr>
<th>who gets a node’s advertisement</th>
</tr>
</thead>
<tbody>
<tr>
<td>effectively, <em>every other node</em></td>
</tr>
<tr>
<td>(via flooding)</td>
</tr>
<tr>
<td>only its <em>neighbors</em></td>
</tr>
</tbody>
</table>

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 *min costs* to each node, rather than the actual topology.

---

<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 *min costs* to each node, rather than the actual topology.

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

---

**question**: what are the contents of B’s first advertisement?
**distance-vector routing:** disseminate information about the current *min 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 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>

- **B’s first adv:** [(A, 7), (C, 3), (D, 1)]
- **D’s first adv:** [(A, 2), (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 *min costs* to each node, rather than the actual topology.

<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 routing table**

- **B’s first adv:** [(A,7), (C,3), (D,1)]
- **D’s first adv:** [(A,2), (B,1), (C,5), (E,3), (F,4)]
- **F’s first adv:** [(A,1), (D,4), (E,5)]

**link state**

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

**distance vector**

- its *current costs* to every node it’s aware of

**what’s in an advertisement**

- effectively, *every other node* (via flooding)

**who gets a node’s advertisement**

- only its *neighbors*

**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 *min costs* to each node, rather than the actual topology.

### Link State vs. Distance Vector

<table>
<thead>
<tr>
<th></th>
<th>Link State</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>what’s in an advertisement</strong></td>
<td>its link costs to each of its neighbors</td>
</tr>
<tr>
<td></td>
<td>its current costs to every node it’s aware of</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Distance Vector</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>who gets a node’s advertisement</strong></td>
<td>effectively, every other node (via flooding)</td>
</tr>
<tr>
<td></td>
<td>only its neighbors</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th><strong>what happens when things fail?</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>flooding makes link-state routing very resilient to failure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th><strong>what limits scale?</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>the overhead of flooding</td>
</tr>
</tbody>
</table>

**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>

**B’s first adv:** [(A, 7), (C, 3), (D, 1)]
**distance-vector routing:** disseminate information about the current *min 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 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>

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

<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>

A's routing table

B's first adv: [(A,7), (C,3), (D,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)
  - only its *neighbors*

- 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 *min 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 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 *min 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

---

**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 *min costs* to each node, rather than the actual topology.

<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)]`

D's *first adv*: `[(A,2), (B,1), (C,5), (E,3), (F,4)]`

F's *first adv*: `[(A,1), (D,4), (E,5)]`

**link state**
- 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

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

<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>

**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>

**D's first adv**: \([A,2), (B,1), (C,5), (E,3), (F,4)\]

**link state**

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

**distance vector**

- what's in an advertisement
  - its *current costs* to every node it's aware of
  - effectively, every other node (via flooding)

**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

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

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>

D's first adv: [(A,2), (B,1), (C,5), (E,3), (F,4)]

**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**
- what happens when things fail?
  - flooding makes link-state routing very resilient to failure
- who gets a node's advertisement
  - effectively, every other node (via flooding)
  - only its neighbors

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

<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-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>

A's routing table

D's *first* adv: [(A,2), (B,1), (C,5), (E,3), (F,4)]

**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
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

<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-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>

A's routing table

D’s first adv: [(A,2), (B,1), (C,5), (E,3), (F,4)]

**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
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

### 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>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

D’s *first* adv: [(A,2), (B,1), (C,5), (E,3), (F,4)]

### Link state vs. distance vector

<table>
<thead>
<tr>
<th>Link state</th>
<th>Distance vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>what’s in an advertisement</td>
<td>its <em>link costs</em> to each of its <em>neighbors</em></td>
</tr>
<tr>
<td>who gets a node’s advertisement</td>
<td>effectively, <em>every other node</em> (via flooding) only its <em>neighbors</em></td>
</tr>
<tr>
<td>what happens when things fail?</td>
<td>flooding makes <em>link-state routing</em> very resilient to failure</td>
</tr>
<tr>
<td>what limits scale?</td>
<td>the <em>overhead</em> of flooding</td>
</tr>
</tbody>
</table>

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing:** disseminate information about the current *min costs* to each node, rather than the actual topology.

<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>F</td>
<td>A-F</td>
<td>1</td>
</tr>
</tbody>
</table>

A's routing table

D's first adv: [(A,2), (B,1), (C,5), (E,3), (F,4)]

![Graph showing link costs and node connections]

**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

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing:** disseminate information about the current *min 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*
- **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>

D’s first adv: [(A,2), (B,1), (C,5), (E,3), (F,4)]
**distance-vector routing:** disseminate information about the current *min costs* to each node, rather than the actual topology.

---

### link state
- its *link costs* to each of its neighbors
- effectively, *every other node* (via flooding)

### distance vector
- 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

---

**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>

D’s first adv: 

\[
\text{(A, 2), (B, 1), (C, 5), (E, 3), (F, 4)}
\]
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

<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:

- **B's first adv**: [(A,7), (C,3), (D,1)]
- **D's first adv**: [(A,2), (B,1), (C,5), (E,3), (F,4)]
- **F's first adv**: [(A,1), (D,4), (E,5)]

**link state**
- what's in an advertisement
  - its *link costs* to each of its *neighbors* for 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
**distance-vector routing**: disseminate information about the current *min 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*
  - who gets a node’s advertisement
    - its *current costs* to every node it’s aware of
  - 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 min costs to each node, rather than the actual topology

link state
distance vector

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>

this is A's routing table after one round of advertisements; note that it does not have knowledge of the min-cost path to C yet
distance-vector routing: disseminate information about the current \textit{min costs} to each node, rather than the actual topology

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>

question: what does A's \textit{next} advertisement look like?

link state

- what's in an advertisement
  - its \textit{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
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

**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)]`

---

**link state**

- 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

---

**what limits scale?**

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

<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

*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!

---

**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
**distance-vector routing**: disseminate information about the current *min 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
  - 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 *min 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: B-&gt;C, 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

A: B->A, 1  B: Self, 0  C: B->C, 1
A: B->A, 1  B: Self, 0  C: B->C, 1

---

*link state*

*distance vector*

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

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min 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*

![Diagram]

A: Self, 0  
B: A->B, 1  
C: A->B, 2

A: B->A, 1  
B: Self, 0  
C: B->C, 1

t=9: B<->C fails

---

<table>
<thead>
<tr>
<th>link state</th>
<th>distance vector</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>what’s in an advertisement</strong></td>
<td><strong>its current costs to every node it’s aware of</strong></td>
</tr>
<tr>
<td>its link costs to each of its neighbors</td>
<td>its <strong>link costs</strong> to each of its neighbors</td>
</tr>
</tbody>
</table>

---

**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

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min 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

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: Self, 0</th>
<th>A: B-&gt;A, 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>B: A-&gt;B, 1</td>
<td>B: Self, 0</td>
<td>t=9: B&lt;-C fails</td>
</tr>
<tr>
<td>C: A-&gt;B, 2</td>
<td>C: None, inf</td>
<td></td>
</tr>
</tbody>
</table>

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.

**what's in an advertisement**

- **link state**: 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?**

- **floodings 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 | 6.1800 2024
**distance-vector routing:** disseminate information about the current *min 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>A: Self, 0</th>
<th>B: A-&gt;B, 1</th>
<th>C: A-&gt;B, 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>B: Self, 0</td>
<td></td>
<td>None, inf</td>
</tr>
</tbody>
</table>

$t=9$: $B<->C$ fails

$t=10$: $B$ receives the following advertisement from $A$: 

$[(A, 0), (B, 1), (C, 2)]$

**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
- failures can be complicated because of timing

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

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing:** disseminate information about the current *min 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>B: Self, 0</td>
<td></td>
</tr>
<tr>
<td>B: A-&gt;B, 1</td>
<td>C: None, inf</td>
<td>C: A-&gt;B, 2</td>
<td></td>
</tr>
<tr>
<td>C: A-&gt;B, 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$t=9$: B--C fails

$t=10$: B receives the following advertisement from A:

$$[(A, 0), (B, 1), (C, 2)]$$

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

**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

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min 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: Self, 0</th>
<th>B: A-&gt;B, 1</th>
<th>C: A-&gt;B, 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>t=9: B&lt;-&gt;C fails</td>
<td>A: B-&gt;A, 1</td>
<td>B: Self, 0</td>
<td>C: None, inf</td>
</tr>
<tr>
<td>t=10: B receives the following advertisement from A: [(A,0),(B,1),(C,2)]</td>
<td>A: Self, 0</td>
<td>B: A-&gt;B, 1</td>
<td>C: B-&gt;A, 3 (2+1)</td>
</tr>
<tr>
<td>t=15: A receives the following advertisement from B: [(A,1),(B,0),(C,3)]</td>
<td>A: B-&gt;A, 1</td>
<td>B: Self, 0</td>
<td>C: None, inf</td>
</tr>
</tbody>
</table>

**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
  - failures can be complicated because of timing

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

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min 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></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>

\(t=9\): \(B\leftarrow C\) fails

\(t=10\): \(B\) receives the following advertisement from \(A\):

\([(A,0),(B,1),(C,2)]\)

\(t=15\): \(A\) receives the following advertisement from \(B\):

\([(A,1),(B,0),(C,3)]\)

**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

---

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology

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

---

**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

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing**: disseminate information about the current *min 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

---

**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
  - failures can be complicated because of timing

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

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing**: disseminate information about the current *min 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>Time (t)</th>
<th>Advertisement</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>t=9</td>
<td>B-&gt;C fails</td>
<td>A: Self, 0</td>
<td>A: B-&gt;A, 1</td>
<td>B: Self, 0</td>
</tr>
<tr>
<td>t=10</td>
<td>B receives the following advertisement from A: [(A,0),(B,1),(C,2)]</td>
<td>A: Self, 0</td>
<td>A: B-&gt;A, 1</td>
<td>B: Self, 0</td>
</tr>
<tr>
<td></td>
<td>t=15: A receives the following advertisement from B: [(A,1),(B,0),(C,3)]</td>
<td>A: Self, 0</td>
<td>A: B-&gt;A, 1</td>
<td>B: Self, 0</td>
</tr>
<tr>
<td></td>
<td>t=20: B receives the following advertisement from A: [(A,0),(B,1),(C,4)]</td>
<td>A: Self, 0</td>
<td>A: B-&gt;A, 1</td>
<td>B: Self, 0</td>
</tr>
</tbody>
</table>

continues until both costs to C are INFINITY

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way

**link state**
- what's in an advertisement
- who gets a node's advertisement
- what happens when things fail?

**distance vector**
- its current costs to every node it's aware of
- only its neighbors
- failures can be complicated because of timing

the overhead of flooding

**Overhead**
- the overhead of flooding

**what limits scale?**
**Distance-vector routing**: Disseminate information about the current *min 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.*

- **What's in an advertisement**
  - Distance vector: its *current costs* to every node it's aware of
  - Link state: its link costs to each of its neighbors

*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.

---

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing:** disseminate information about the current *min 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

![Diagram](image)

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

t=9: B<->C fails

**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
- 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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing:** disseminate information about the current *min 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>Node</th>
<th>Advertisement</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Self, 0</td>
<td>A: B-&gt;A, 1</td>
</tr>
<tr>
<td>B</td>
<td>A-&gt;B, 1</td>
<td>B: Self, 0</td>
</tr>
<tr>
<td>C</td>
<td>A-&gt;B, 2</td>
<td>C: None, inf</td>
</tr>
</tbody>
</table>

$t=9$: B->C fails

$t=10$: B receives the following advertisement from A: $[(A,0)]$

**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

---

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min 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.

A: Self, 0  \quad B: B->A, 1
B: A->B, 1  \quad B: Self, 0
C: A->B, 2  \quad C: None, inf

\[ t=9: \ B\leftarrow C \text{ fails} \]

\[ t=10: \ B \text{ receives the following advertisement from } A: \ [(A,0)] \]

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

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.

**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
  - failures can be complicated because of timing

- what limits scale?
  - the overhead of flooding

Katrina LaCurts  |  lacurts@mit.edu  |  6.1800 2024
**distance-vector routing:** disseminate information about the current *min 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

```
A: Self, 0     A: B->A, 1
B: A->B, 1     B: Self, 0
C: A->B, 2     C: None, inf
```

t=9: B<->C fails

t=10: B receives the following advertisement from A:

```
[(A,0)]
```

t=15: A receives the following advertisement from B:

```
[(B,0), (C,inf)]
```

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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
distance-vector routing: disseminate information about the current min 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

A: Self, 0    A: B->A, 1
B: A->B, 1    B: Self, 0
C: A->B, 2    C: None, inf

A: Self, 0    A: B->A, 1
B: A->B, 1    B: Self, 0
C: A->B, 2    C: None, inf

A: Self, 0    A: B->A, 1
B: A->B, 1    B: Self, 0
C: None, inf  C: None, inf

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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way

link state
distance vector

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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing:** disseminate information about the current *min 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

![Diagram of network with nodes A, B, and C]

A: Self, 0  
B: A→B, 1  
C: A→B, 2

B: Self, 0  
C: None, inf

A: Self, 0  
B: A→B, 1  
C: None, inf

B: Self, 0  
C: None, inf

C: None, inf

**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**
- its *current costs* to every node it's aware of
- what happens when things fail?
  - flooding makes link-state routing very resilient to failure
  - failures can be complicated because of timing

**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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing:** disseminate information about the current *min costs* to each node, rather than the actual topology

- Link state
  - what's in an advertisement
    - its *link costs* to each of its *neighbors* (via flooding)
    - only its *neighbors* (via flooding)

- 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
    - 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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

---

**link state**

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

- 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.

---

C: D->B, 2  C: A->B, 2  C: B->C, 1  B<->C fails

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**  
disseminate information about the current *min 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  
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

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way

---

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology

![Diagram of network with nodes A, B, C, D and link costs]

- C: D→B, 2
- C: A→B, 2
- C: None, inf

B→C fails

**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
- 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

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way
**distance-vector routing**: disseminate information about the current *min 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**
  - 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.

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

- **C**: D->B, 2  C: A->B, 2  C: None, inf
- **C**: None, inf  C: A->B, 2  C: None, inf

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

- **B**-><C fails

**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

---

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.
**Distance-Vector Routing**: Disseminate information about the current *minimum 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) for its neighbors.
  - **What happens when things fail?**: Link-state routing is very resilient to failure. Failures can be complicated because of timing.
  - **What limits scale?**: The overhead of flooding.

- **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.
  - **What happens when things fail?**: Flooding makes link-state routing very resilient to failure.
  - **What limits scale?**: The overhead of flooding.

**New Strategy ("Split Horizon")**: Don't send advertisements about a route to the node providing the route.

- **Diagrams**: Nodes C and D are connected to B. Node A is also connected to B. Node C has a link state of C: D->A, 3, C: A->B, 2, C: None, inf. Node B has a link state of B<->C fails, B's advertisement to A gets lost (so A makes no changes). Node A advertises about C to D (not to B because of split horizon).

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min 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**

- 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.

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology

---

---

---

**link state**

**distance vector**

<table>
<thead>
<tr>
<th></th>
<th>what’s in an advertisement</th>
<th></th>
<th>what’s in an advertisement</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>its <em>link costs</em> to each of</td>
<td></td>
<td>its <em>current costs</em> to</td>
</tr>
<tr>
<td></td>
<td>its neighbors</td>
<td></td>
<td>every node it’s aware of</td>
</tr>
<tr>
<td>who gets a node’s advertisement</td>
<td></td>
<td>who gets a node’s advertisement</td>
<td></td>
</tr>
<tr>
<td>effectively, every</td>
<td>only its <em>neighbors</em></td>
<td></td>
<td></td>
</tr>
<tr>
<td>other node (via</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flooding)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>what happens when</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>things fail?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flooding makes link-</td>
<td></td>
<td></td>
<td>failures can be</td>
</tr>
<tr>
<td>state routing very</td>
<td></td>
<td></td>
<td>complicated because of</td>
</tr>
<tr>
<td>resilient to failure</td>
<td></td>
<td></td>
<td>timing</td>
</tr>
<tr>
<td>what limits scale?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>the <em>overhead</em> of</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>flooding</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

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

---

**in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way**
**distance-vector routing**: disseminate information about the current *min costs* to each node, rather than the actual topology.

---

### Link State vs. Distance Vector

**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* (viaflooding).
- **What happens when things fail?**: flooding makes link-state routing very resilient to failure.
- **What limits scale?**: failures can be complicated because of timing.

---

**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.
- **What happens when things fail?**: 
  - D advertises about C to B
  - B advertises about C to A
- **What limits scale?**: the overhead of flooding.

---

**New Strategy (“Split Horizon”)**: don’t send advertisements about a route to the node providing the route.

---

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**distance-vector routing**: disseminate information about the current *min 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

- 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

---

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way

---

Katrina LaCurts | lacurts@mit.edu | 6.1800 2024
**distance-vector routing:** disseminate information about the current *min 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* (via flooding)
- 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**
- its *current costs* to every node it’s aware of

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

in this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way

```
D -> A -> B
C
```

- C: D→B, 2
- C: A→B, 2
- C: None, inf

- C: None, inf
- C: A→B, 2
- C: None, inf

- C: D→A, 3
- C: A→B, 2
- C: None, inf

- C: D→A, 3
- C: A→B, 2
- C: B→D, 4

- C: D→A, 3
- C: A→B, 5
- C: B→D, 4

B<→C fails
B’s advertisement to A gets lost (so A makes no changes)
A advertises about C to D (not to B because of split horizon)
D advertises about C to B
B advertises about C to A
**distance-vector routing:** disseminate information about the current *min costs* to each node, rather than the actual topology.

![Diagram showing network nodes A, B, C, and D with their current costs and link state updates.]

- **C:** D→B, 2  C: A→B, 2  C: None, inf
- **C:** None, inf  C: A→B, 2  C: None, inf
- **C:** D→A, 3  C: A→B, 2  C: None, inf
- **C:** D→A, 3  C: A→B, 2  C: B→D, 4
- **C:** D→A, 3  C: A→B, 5  C: B→D, 4

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
  - only its neighbors
- who gets a node’s advertisement
  - effectively, every other node (via flooding)
  - failures can be complicated because of timing

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

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

---

In this example, nodes will explicitly include their route/cost to themselves in their advertisements; you can make distance-vector work either way.
**link state**

**distance vector**

---

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

<table>
<thead>
<tr>
<th>link state</th>
<th>distance vector</th>
</tr>
</thead>
<tbody>
<tr>
<td>what’s in an advertisement</td>
<td></td>
</tr>
<tr>
<td>its link costs to each of its neighbors</td>
<td>its current costs to every node it’s aware of</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>who gets a node’s advertisement</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>effectively, every other node (via flooding)</td>
<td>only its neighbors</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>what happens when things fail?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>flooding makes link-state routing very resilient to failure</td>
<td>failures can be complicated because of timing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>what limits scale?</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>the overhead of flooding</td>
<td>failure handling</td>
</tr>
</tbody>
</table>
### Network Evolution

<table>
<thead>
<tr>
<th>Year</th>
<th>Key Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>1970s</td>
<td>ARPAnet</td>
</tr>
<tr>
<td></td>
<td>flexibility and layering</td>
</tr>
<tr>
<td>1978</td>
<td>ARPAnet</td>
</tr>
<tr>
<td></td>
<td>1978: flexibility and layering</td>
</tr>
<tr>
<td>1980s</td>
<td>ARPAnet</td>
</tr>
<tr>
<td></td>
<td>early 80s: growth → change</td>
</tr>
<tr>
<td>1990s</td>
<td>ARPAnet</td>
</tr>
<tr>
<td></td>
<td>late 80s: growth → problems</td>
</tr>
<tr>
<td>1993</td>
<td>ARPAnet</td>
</tr>
<tr>
<td></td>
<td>1993: commercialization</td>
</tr>
</tbody>
</table>

### Protocols and Concepts

- **Transport Layer**
  - **TCP**
  - **UDP**
  - OSPF, EGP, DNS (a link-state routing protocol)
  - Congestion collapse
  - Policy routing
  - CIDR

- **Network Layer**
  - Naming, addressing, routing
  - Examples: IP

- **Link Layer**
  - Communication between two directly-connected nodes
  - Examples: Ethernet, Bluetooth, 802.11 (Wi-Fi)

- **Application Layer**
  - The things that actually generate traffic
  - Examples: TCP, UDP

---

CAIDA's IPv4 AS Core, January 2020
(https://www.caida.org/projects/cartography/as-core/2020/)

IP networks can route using either distance-vector routing (RIP) or link-state routing (OSPF).