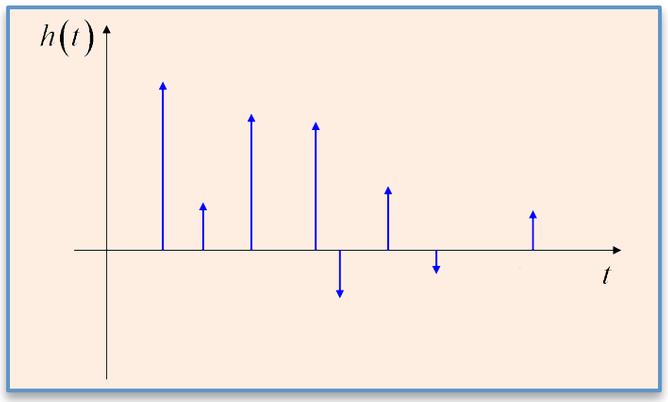
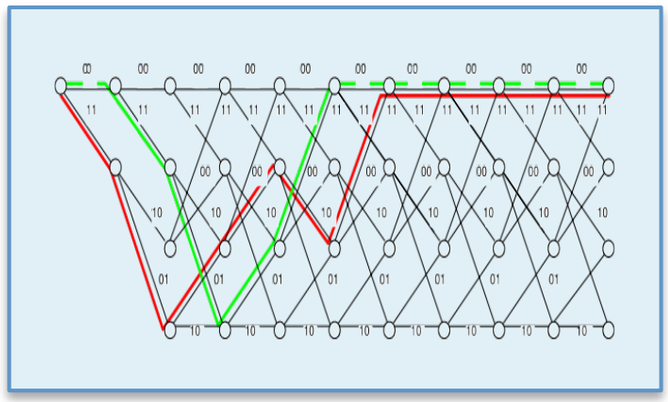
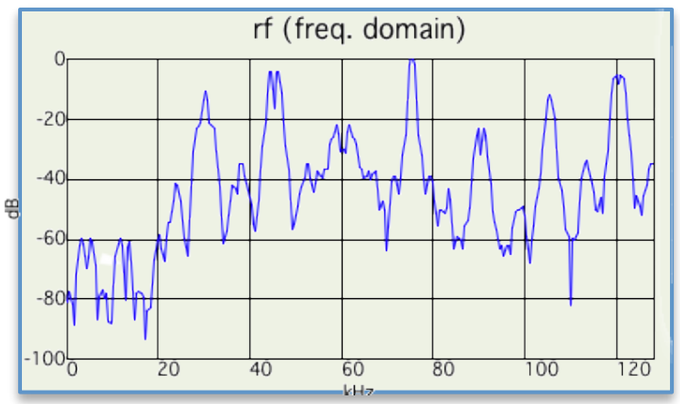
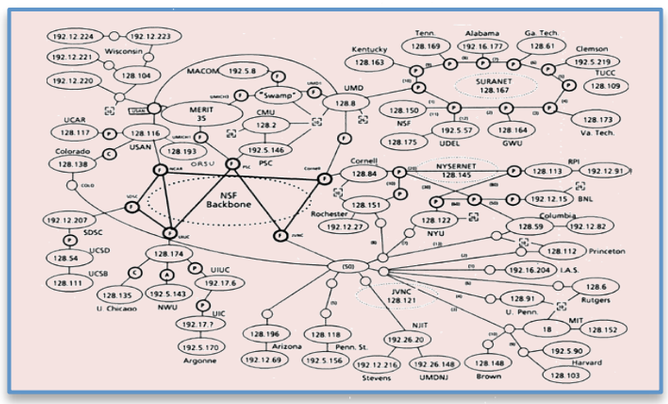








INTRODUCTION TO EECS II

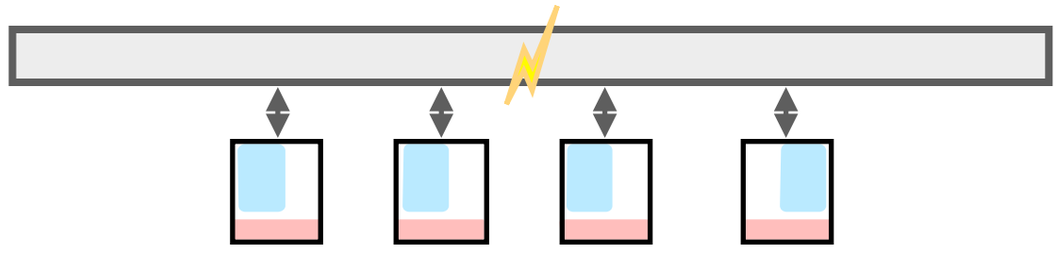
DIGITAL COMMUNICATION SYSTEMS

6.02 Fall 2014
Lecture #20

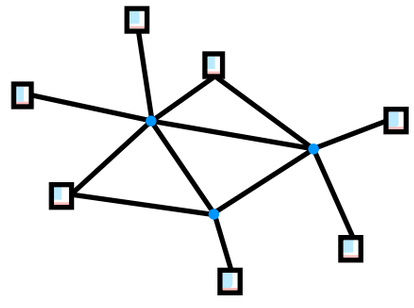
- Distributed routing
- Distance-vector routing protocol
- Link-state routing protocol

Networks so far: how to share

MAC protocols let multiple nodes access the same channel



Circuit- or **packet-switching** lets multiple nodes share a network, even when there are many connections between nodes



But packet-switching causes problems

There are no circuits; nodes need a way to determine routes to other nodes

The network is best-effort; packets can be dropped, reordered, or delayed

Unanswered questions

(about packet-switched networks)

How do nodes **determine routes** to every other node?

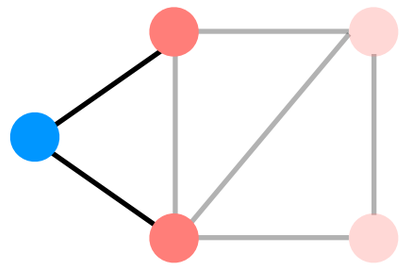
How do nodes **route around link failures**?

How do nodes **communicate reliably**
given that the network is best-effort?

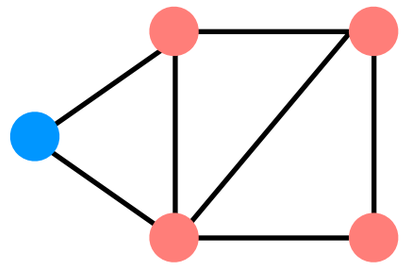
goal of a routing protocol: build a routing table at each node, such that `routing_table[dst]` contains the **minimum-cost route** to `dst`

Distributed Routing

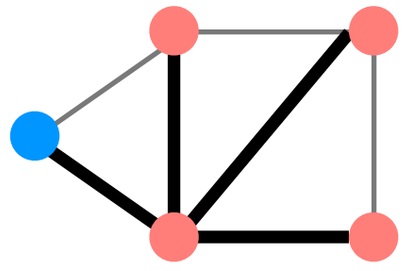
1. Nodes learn about their neighbors via the `HELLO` protocol



2. Nodes learn about other reachable nodes via advertisements



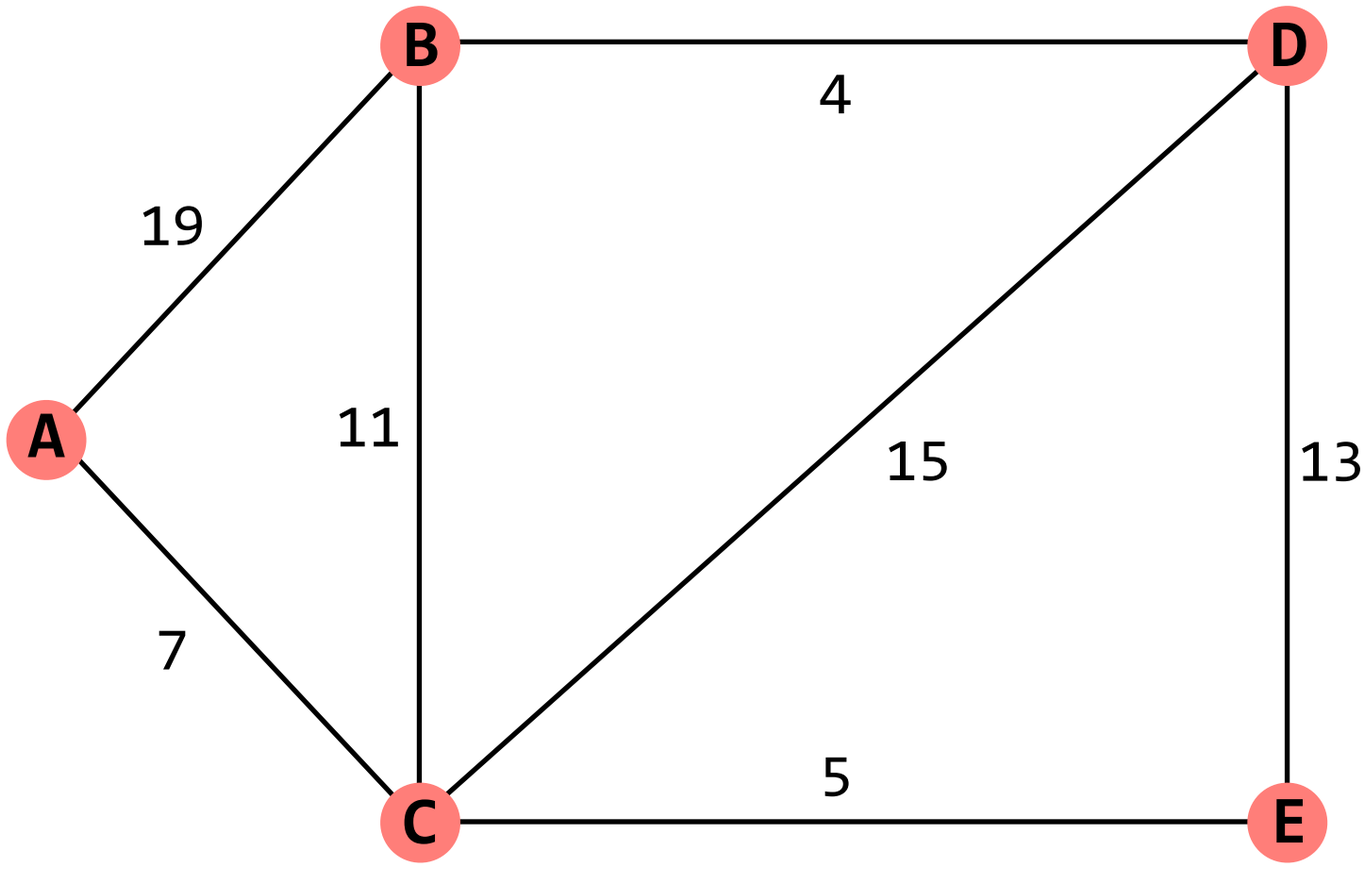
3. Nodes determine the minimum-cost routes (of the routes they know about)



Comparison of Routing Protocols

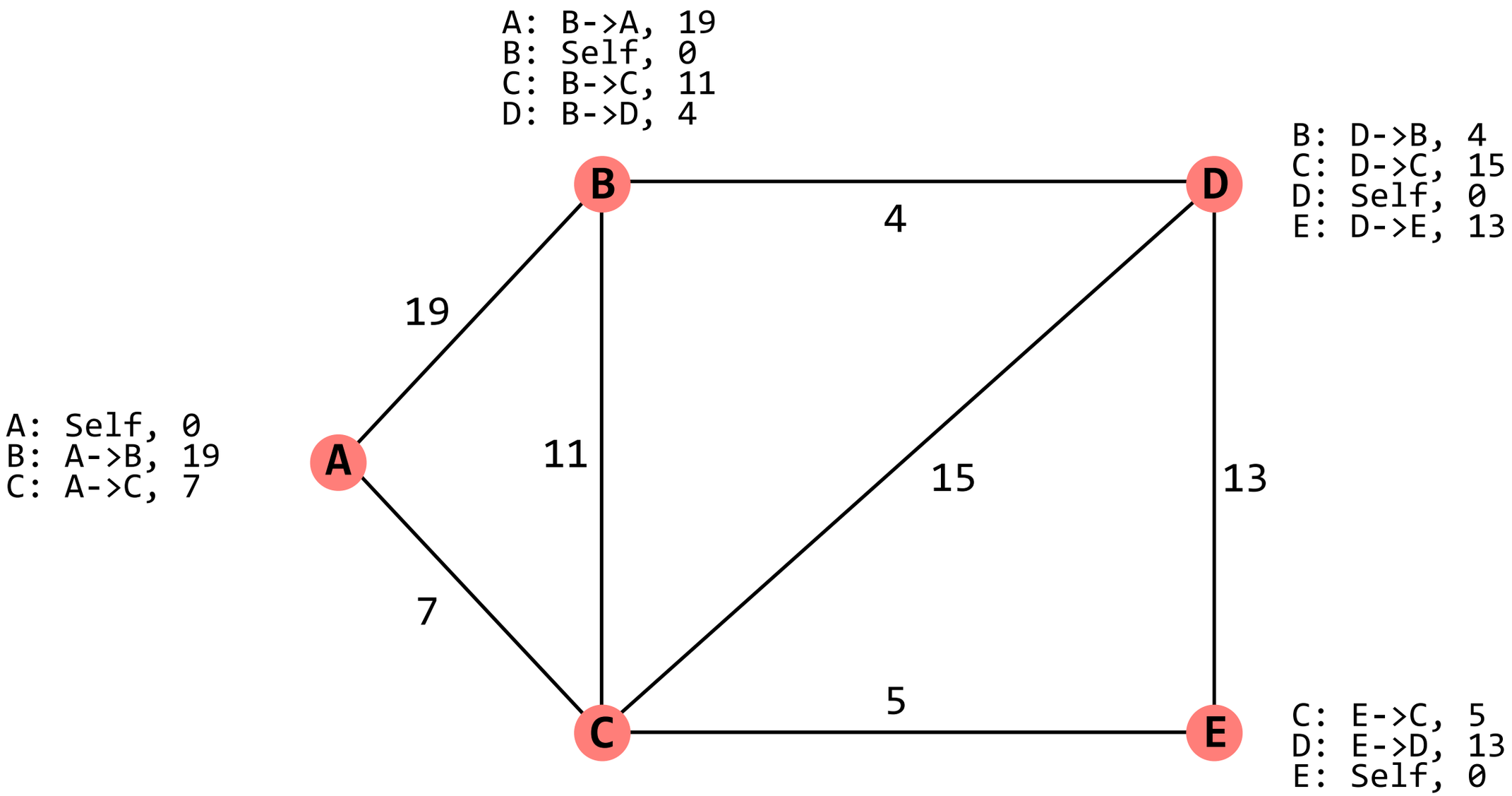
	Distance-vector
Node X's advertisement format	list of all nodes X knows about and the costs to those nodes
Who receives X's advertisement	X's neighbors

Distance-vector Routing



network topology

Distance-vector Routing

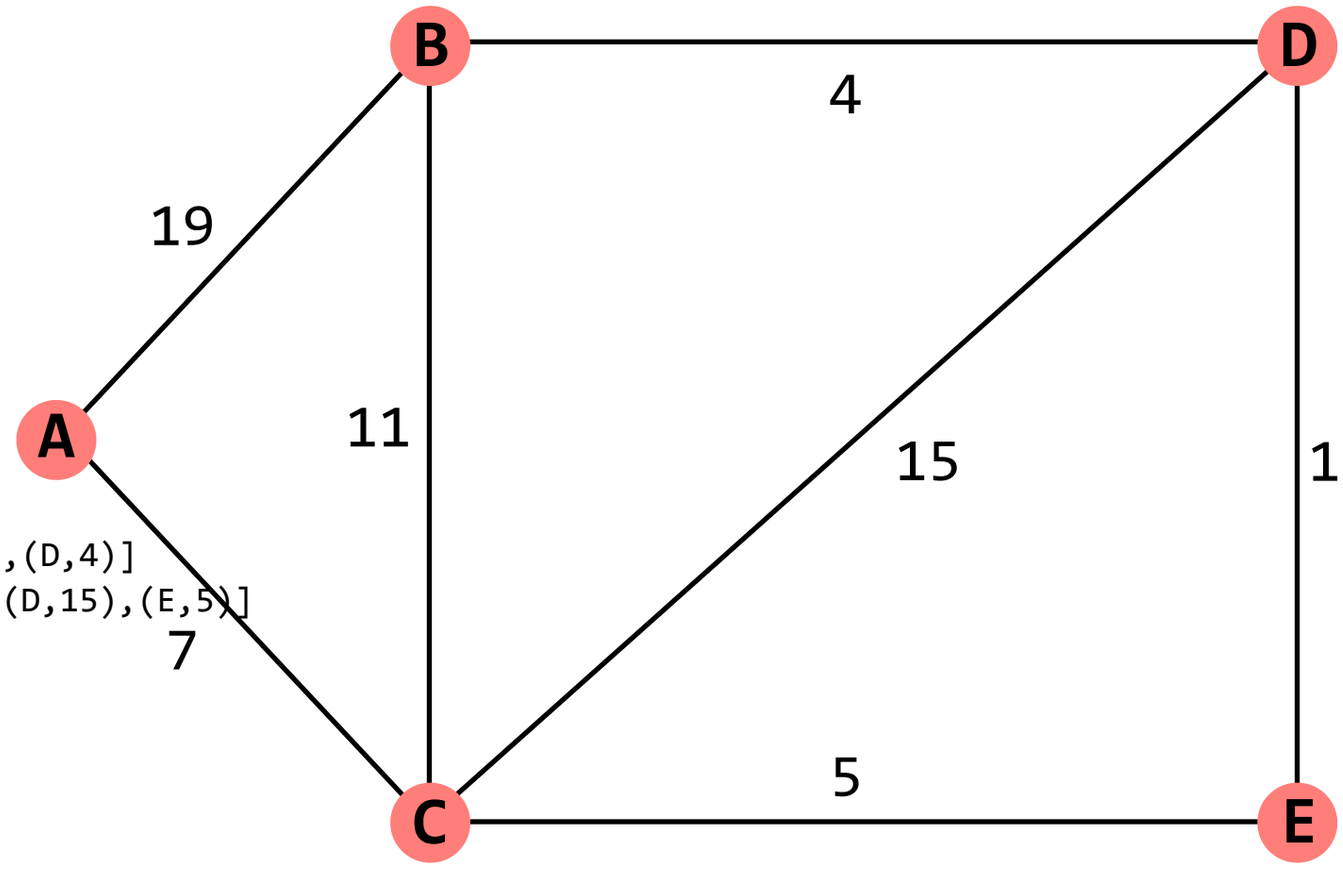


after HELLO

Distance-vector Routing

From A: [(A,0),(B,19),(C,7)]
From C: [(A,7),(B,11),(C,0),(D,15),(E,5)]
From D: [(B,4),(C,15),(D,0),(E,13)]

From B: [(A,19),(B,0),(C,11),(D,4)]
From C: [(A,7),(B,11),(C,0),(D,15),(E,5)]
From E: [(C,5),(D,13),(E,0)]



From B: [(A,19),(B,0),(C,11),(D,4)]
From C: [(A,7),(B,11),(C,0),(D,15),(E,5)]

From A: [(A,0),(B,19),(C,7)]
From B: [(A,19),(B,0),(C,11),(D,4)]
From D: [(B,4),(C,15),(D,0),(E,13)]

From C: [(A,7),(B,11),(C,0),(D,15),(E,5)]
From D: [(B,4),(C,15),(D,0),(E,13)]
From E: [(C,5),(D,13),(E,0)]

received advertisements

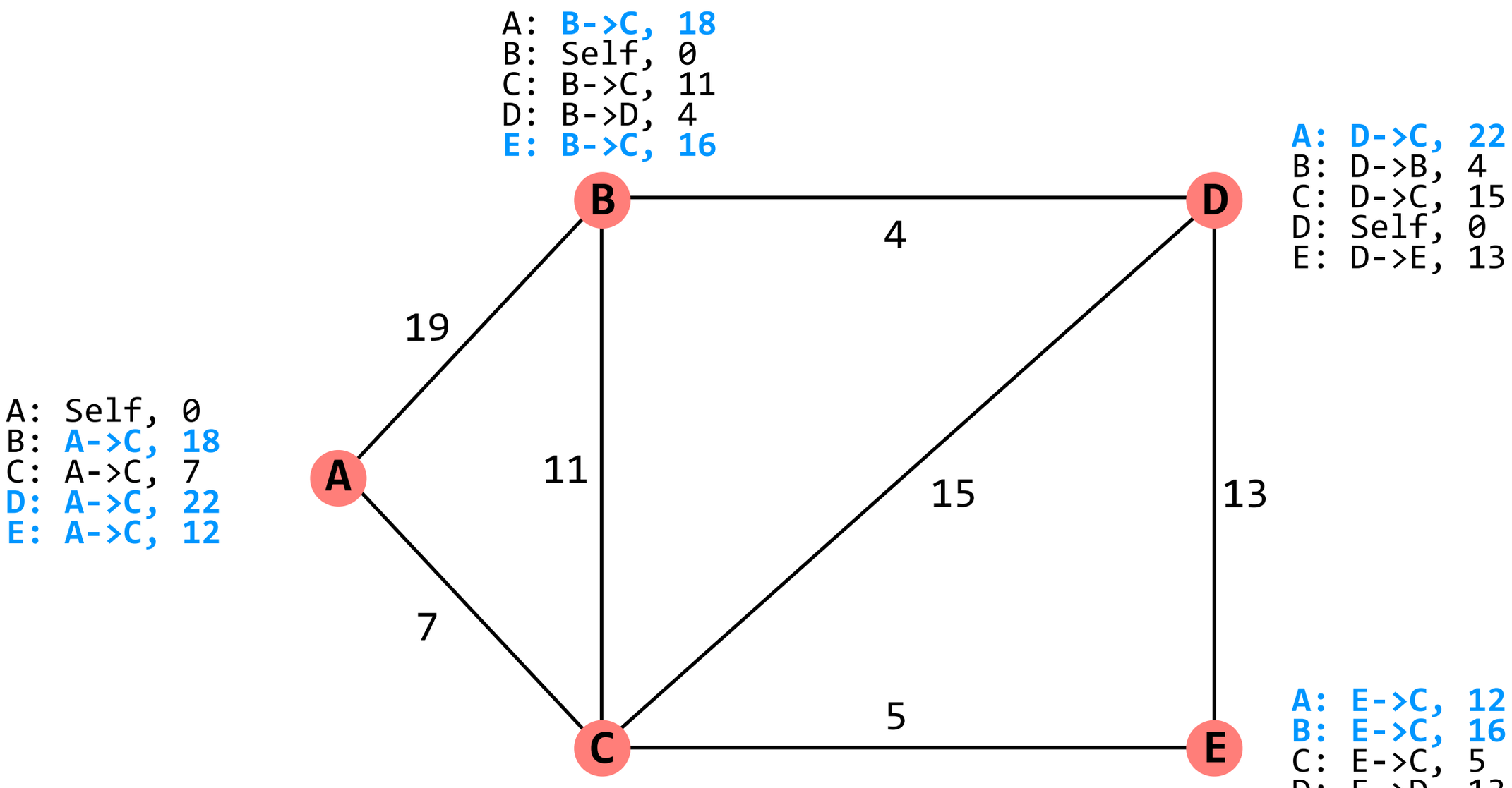
Distance-vector Routing

For every `(dst, cost)` in advertisement

- Calculate cost to `dst` via `Y`
- If we use `Y` to get to `dst` already, update cost
- If not, check if `Y` provides us with a better route to `dst`, and update accordingly

```
to integrate an advertisement from Y
```

Distance-vector Routing

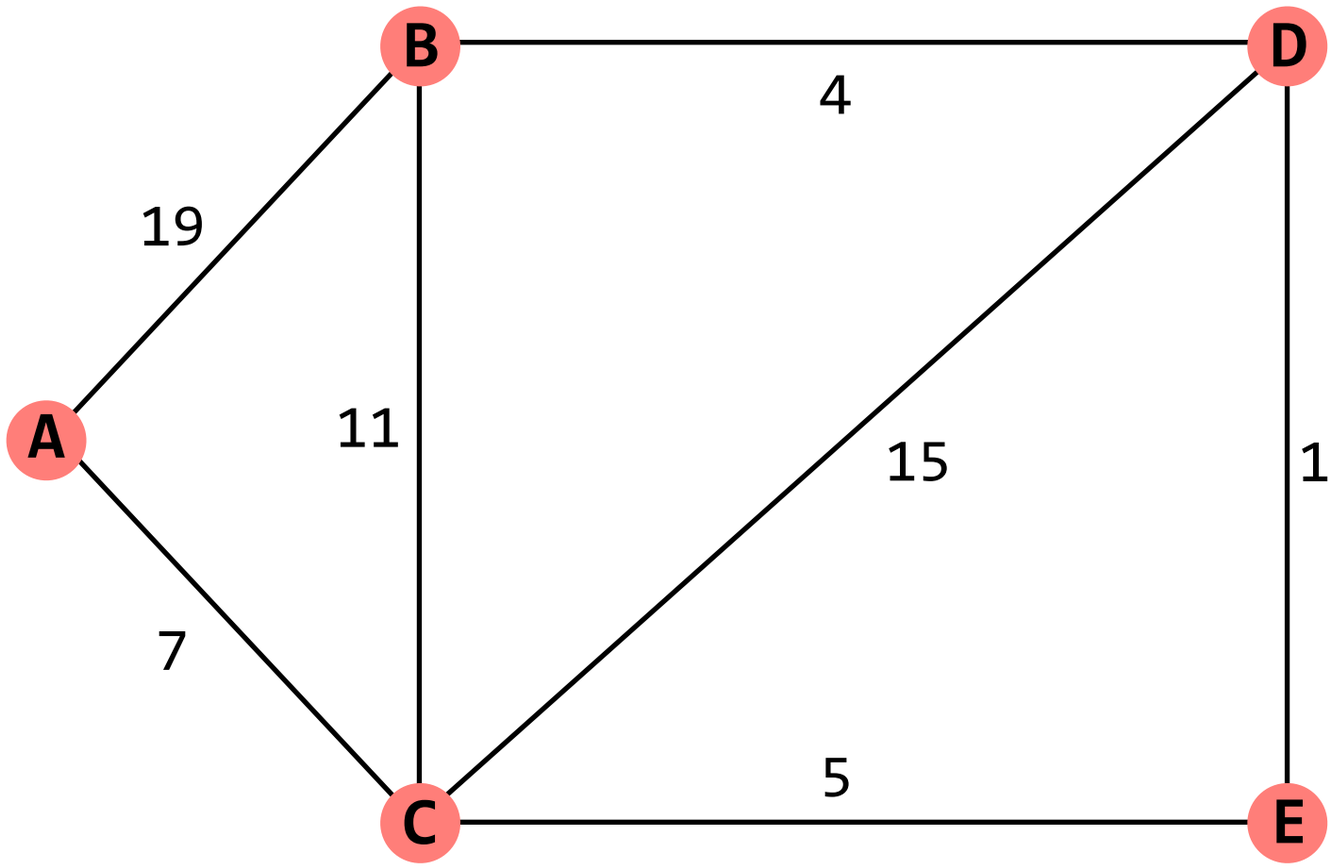


after integration

Comparison of Routing Protocols

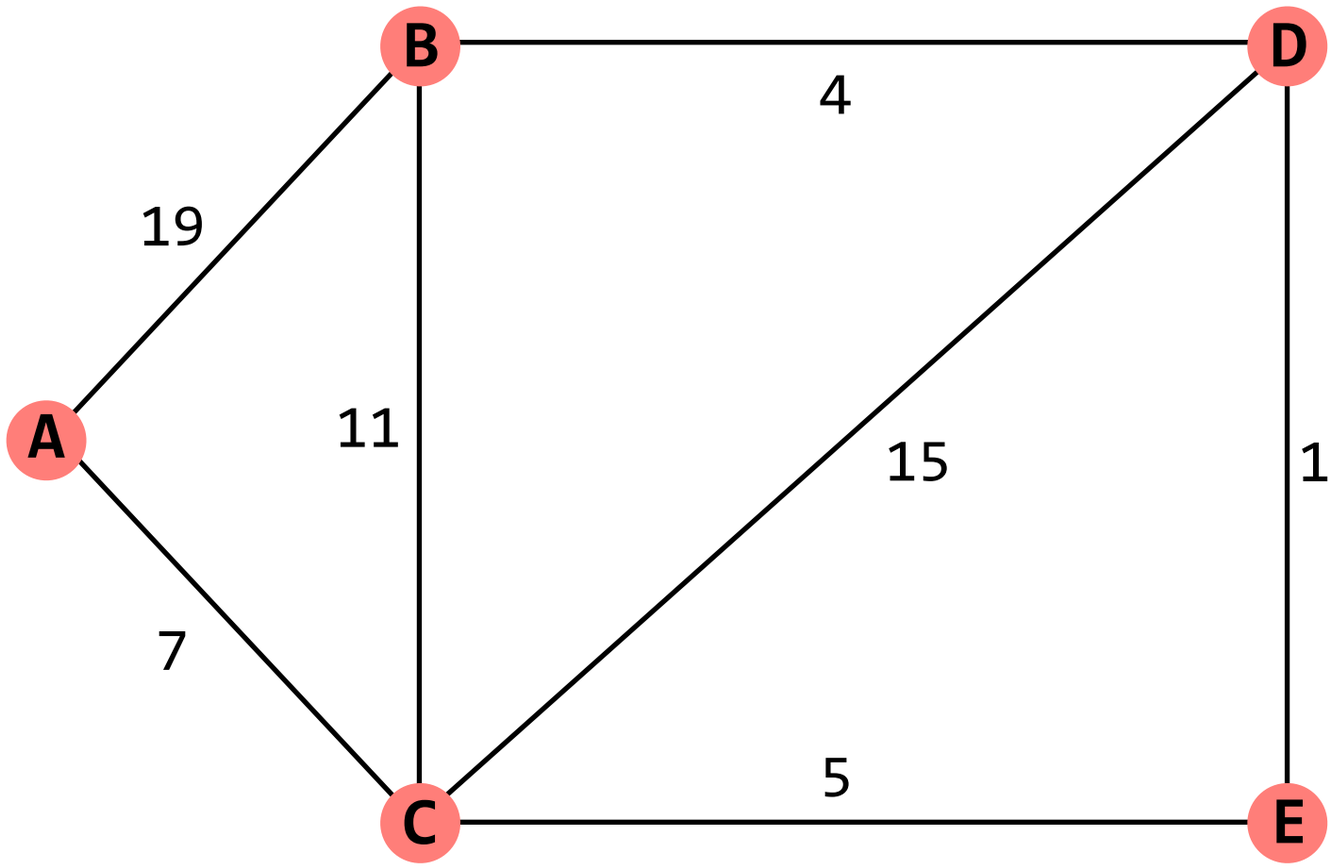
	Distance-vector	**Link-state**
Node X's advertisement format	list of all nodes X knows about and the costs to those nodes	list of all X's neighbors and the link costs to those nodes
Who receives X's advertisement	X's neighbors	all nodes (via flooding)
Integration	Bellman-Ford	Dijkstra's Algorithm

Link-state Routing



network topology

Link-state Routing




```
From A: [(B,19),(C,7)]
From B: [(A,19),(C,11),(D,4)]
From C: [(A,7),(B,11),(D,15),(E,5)]
From D: [(B,4),(C,15),(E,13)]
From E: [(C,5),(D,13)]
```

advertisements

Dijkstra's Algorithm

Until `W` is empty

- Remove `u`, the node in `W` that has the lowest current cost
- Check if **u**'s neighbors provide us with a better route to any node; if so, update

Link-state Routing

results from Dijkstra's algorithm:

			current cost					current route				
Step	**u**	**W**	A	B	C	D	E	A	B	C	D	E
0		{A,B,C,D,E}	0	∞	∞	∞	∞	Self	?	?	?	?
1	A	{B,C,D,E}	0	19	7	∞	∞	Self	A->B	A->C	?	?
2	C	{B,D,E}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C
3	E	{B,D}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C
4	B	{D}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C
5	D	{}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C

final state at node A

Link-state Routing

results from Dijkstra's algorithm:

			current cost					current route				
Step	**u**	**W**	A	B	C	D	E	A	B	C	D	E
0		{A,B,C,D,E}	0	∞	∞	∞	∞	Self	?	?	?	?
1	A	{B,C,D,E}	0	19	7	∞	∞	Self	A->B	A->C	?	?
2	C	{B,D,E}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C
3	E	{B,D}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C
4	B	{D}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C
5	D	{}	0	18	7	22	12	Self	A->C	A->C	A->C	A->C

routing table:

```
A: Self, 0
B: A->C, 18
C: A->C, 7
D: A->C, 22
E: A->C, 12
```

final state at node A

Comparison of Routing Protocols

	Distance-vector	**Link-state**
Node X's advertisement format	list of all nodes X knows about and the costs to those nodes	list of all X's neighbors and the link costs to those nodes
Who receives X's advertisement	X's neighbors	all nodes (via flooding)
	"nodes talk to their neighbors about everyone"	**"nodes talk to everyone about their neighbors"**
Integration	Bellman-Ford	Dijkstra's Algorithm
Convergence time	Proportional to the number of hops in the longest min-cost path	Proportional to flooding time + complexity of Dijkstra's

- **Distributed Routing**
  Each node runs the routing protocol to build its own routing table

- **Distance-vector Routing**
  Nodes send advertisements to their neighbors about their current best routes to all other nodes that they know about, use Bellman-Ford integration

- **Link-state Routing**
  Nodes send advertisements about their neighbors to all nodes (via flooding), use Dijkstra's Algorithm for integration