# E6998-02: Internet Routing

## Lecture 7
## Distance-Vector
## RIP and RIPv2

**John Ioannidis**

AT&T Labs – Research

`ji+ir@cs.columbia.edu`

# Announcements

Lectures 1-5, 7 are available.

Lecture 6 has no notes.

Lecture on 9/26 will be at 4:10pm in 1024.

W4180 on 9/26 will be at 2:40pm in 1127.

Homework 2 is due on Thursday at 3am.

    Don't submit it late, I won't accept it.

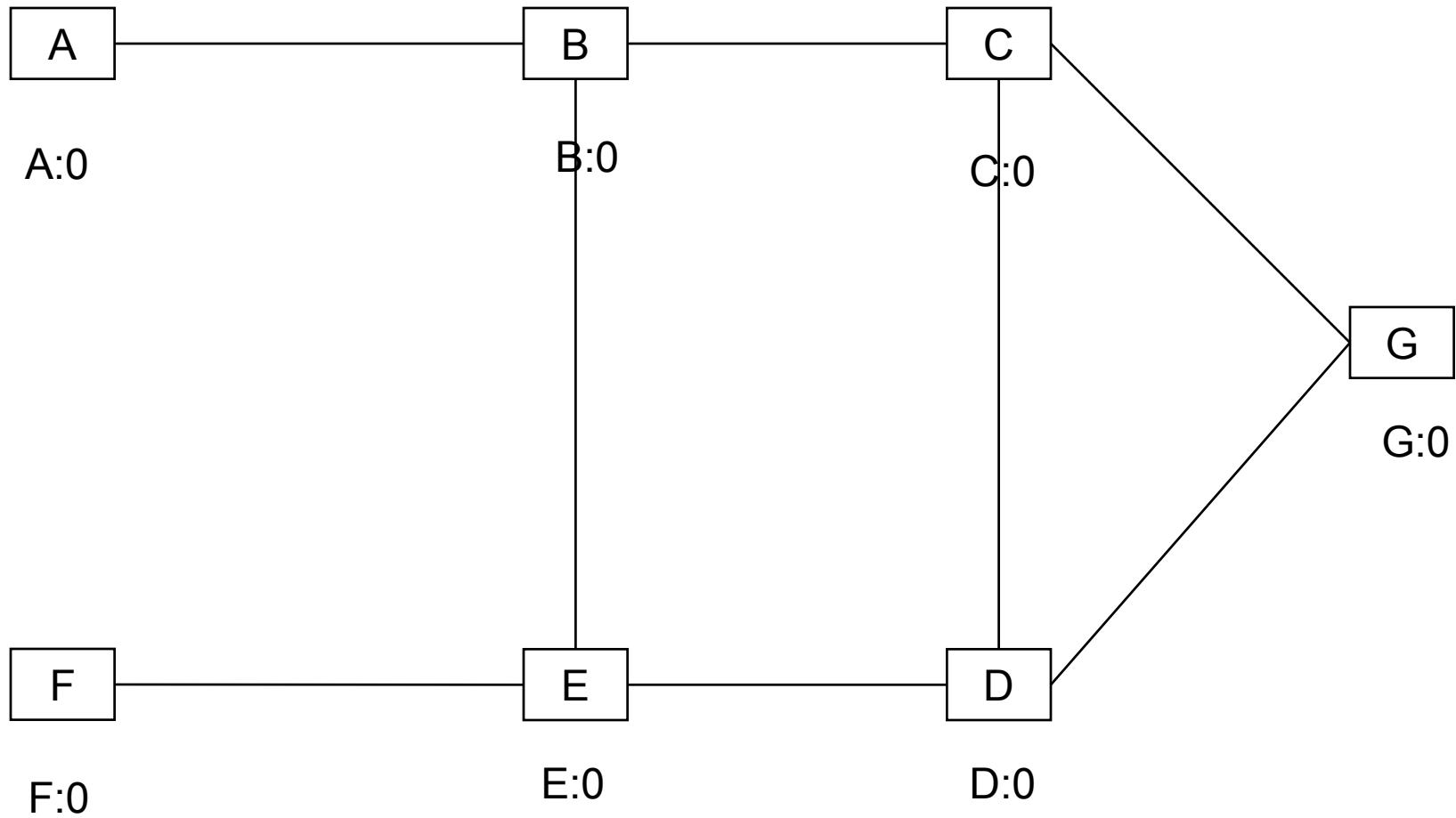    Get a good night's sleep, I need you awake in class!

# Distance-Vector [repeat]

- Variations of Bellman-Ford algorithm.
- Each router starts by knowing:
  - Prefixes of its attached networks ("zero" distance).
  - Its next hop routers (how to find them?)
- Each router advertises only to its neighbors:
  - All prefixes it knows about.
  - Its distance from them.
- Each router learns:
  - All prefixes its neighbors know about.
  - Their distance from them.
- Each router figures out, for each destination prefix:
  - The "distance" (how far away it is).
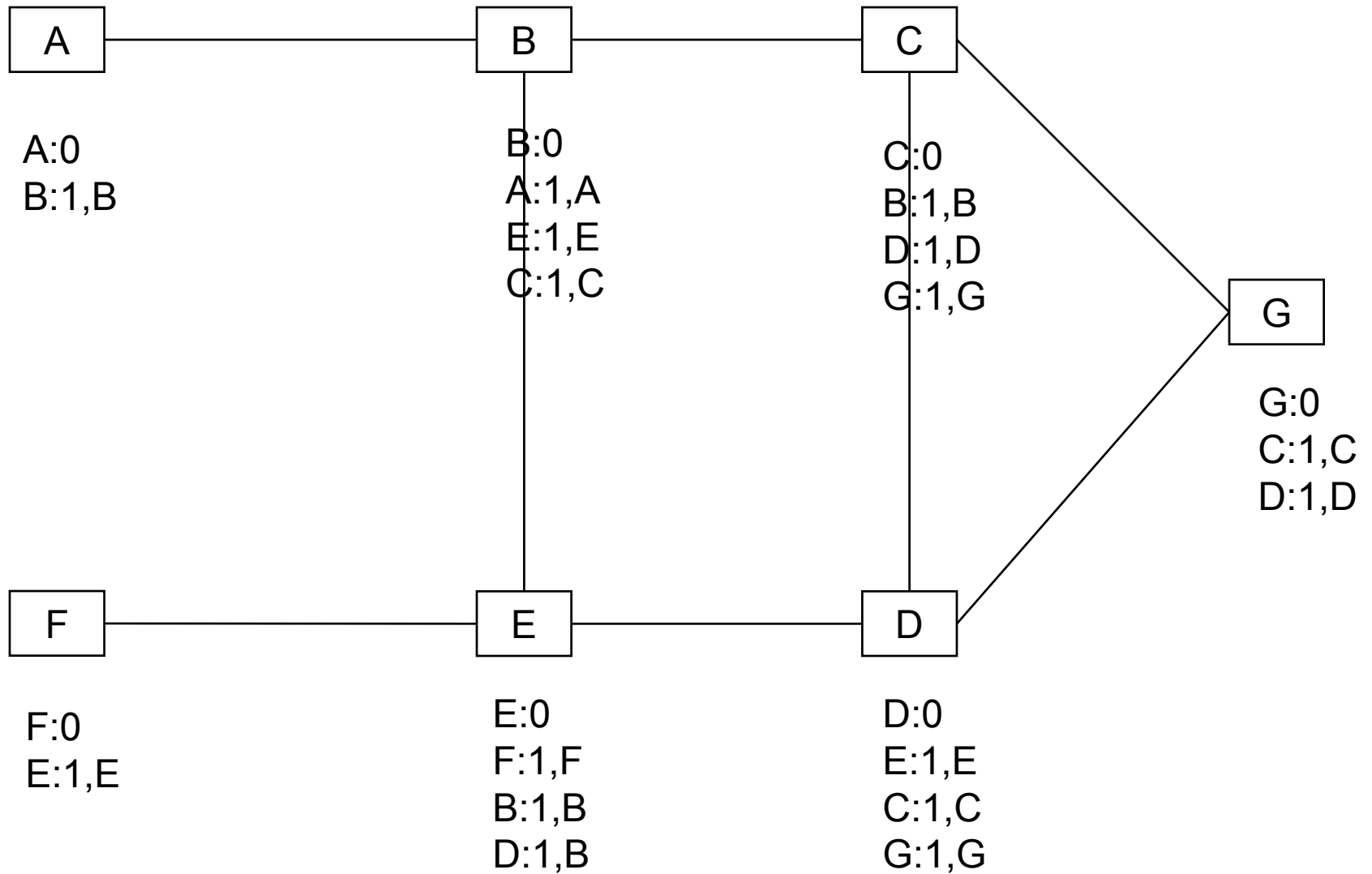  - The "vector" (the next hop router).

# DV Computes the Shortest Path

- DV protocols are variants of the "distributed Bellman-Ford Shortest-Path algorithm".

- "Routing by rumor".

  - Each router believes what its neighbors tell it.

- In steady-state, each router has the "shortest" (smallest metric) path to the destination.

- Convergence time is (on the average) proportional to the diameter of the network.

- Any link change affects the entire network.

# DV

# DV
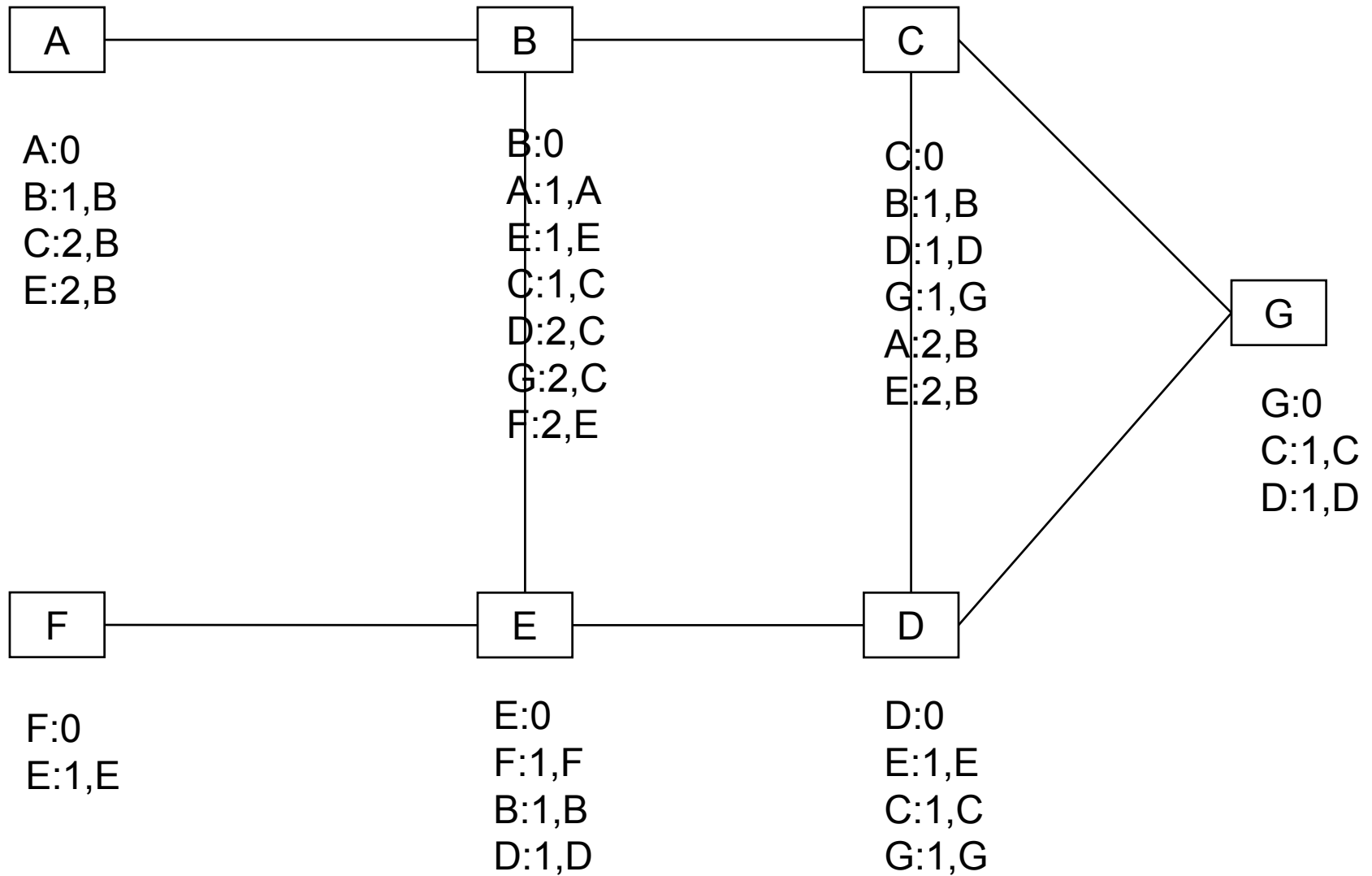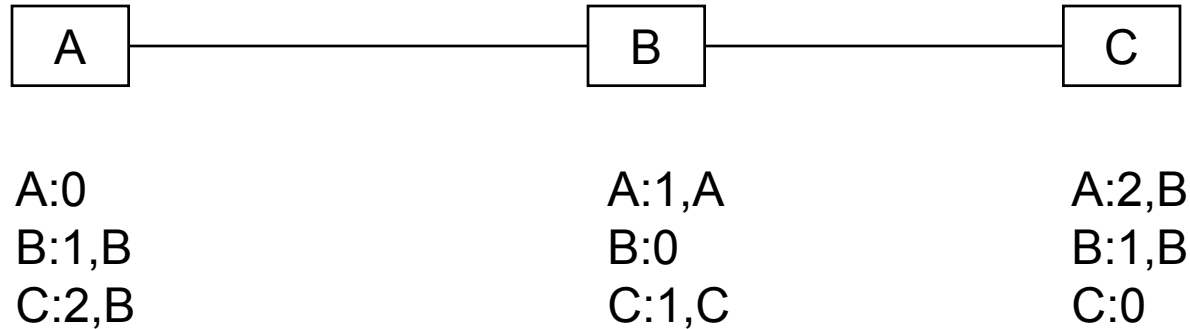


A:0
B:1,B

B:0
A:1,A
E:1,E
C:1,C

C:0
B:1,B
D:1,D
G:1,G

G:0
C:1,C
D:1,D

F:0
E:1,E

E:0
F:1,F
B:1,B
D:1,B

D:0
E:1,E
C:1,C
G:1,G

# DV

A

A:0
B:1,B
C:2,B
E:2,B

B

B:0
A:1,A
E:1,E
C:1,C
D:2,C
G:2,C
F:2,E

C

C:0
B:1,B
D:1,D
G:1,G
A:2,B
E:2,B

G

G:0
C:1,C
D:1,D

F

F:0
E:1,E

E

E:0
F:1,F
B:1,B
D:1,D

D

D:0
E:1,E
C:1,C
G:1,G

# Counting to Infinity

- What happens when a link dies?



| A | | B | | C |
|---|---|---|---|---|
| A:0 | | A:1,A | | A:2,B |
| B:1,B | | B:0 | | B:1,B |
| C:2,B | | C:1,C | | C:0 |

# Counting to Infinity

- What happens when a link dies?



A:0
B:1,B
C:2,B

A:1,A
B:0

A:2,B
B:1,B
C:0

# Counting to Infinity

- What happens when a link dies?

A            B      ✕      C

A:0                 A:1,A              A:2,B
B:1,B              B:0                 B:1,B
C:2,B              C:3,A              C:0

# Counting to Infinity

- What happens when a link dies?



A:0
B:1,B
C:4,B

A:1,A
B:0
C:3,A

A:2,B
B:1,B
C:0

# Counting to Infinity

- What happens when a link dies?



```
A                       B                       C

A:0                     A:1,A                   A:2,B
B:1,B                   B:0                     B:1,B
C:4,B                   C:5,A                   C:0
```
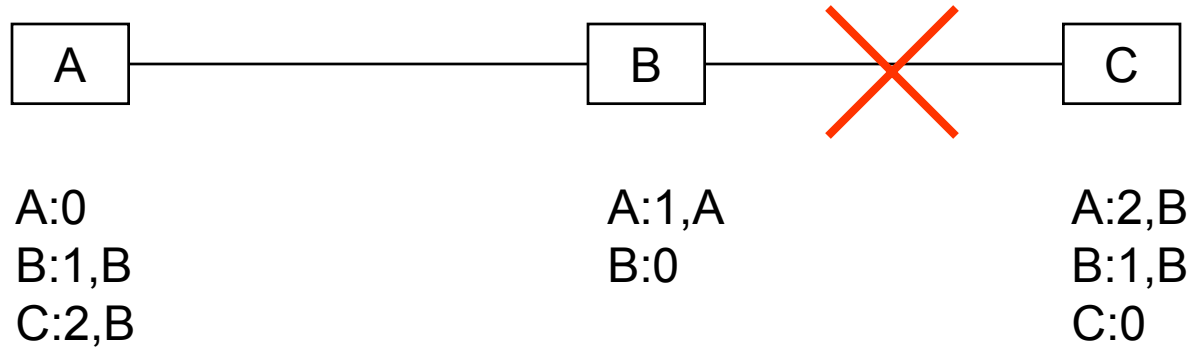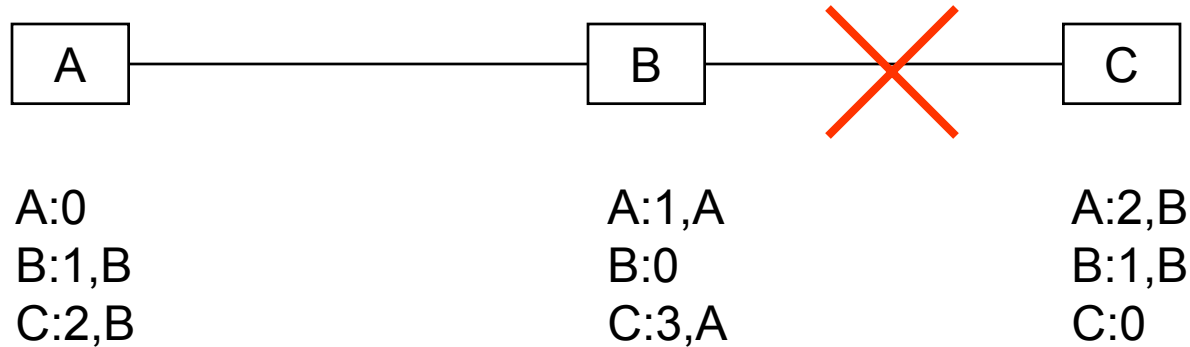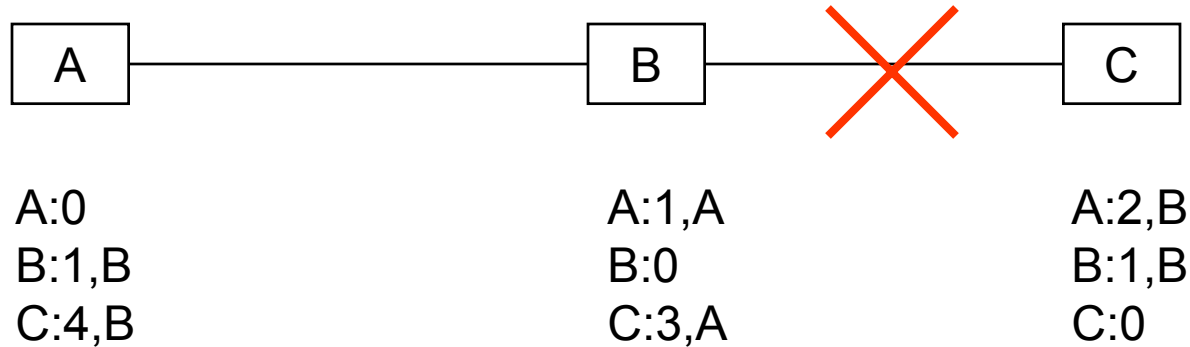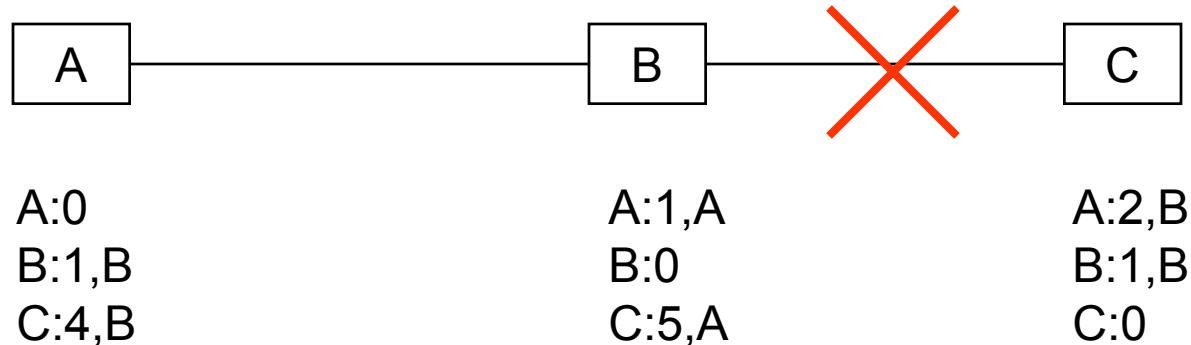
- And so on.

- Counting to infinity can take a long time.

- RIP defines infinity as 16.

- When infinity is reached, the route to C is declared dead.

# Split Horizon

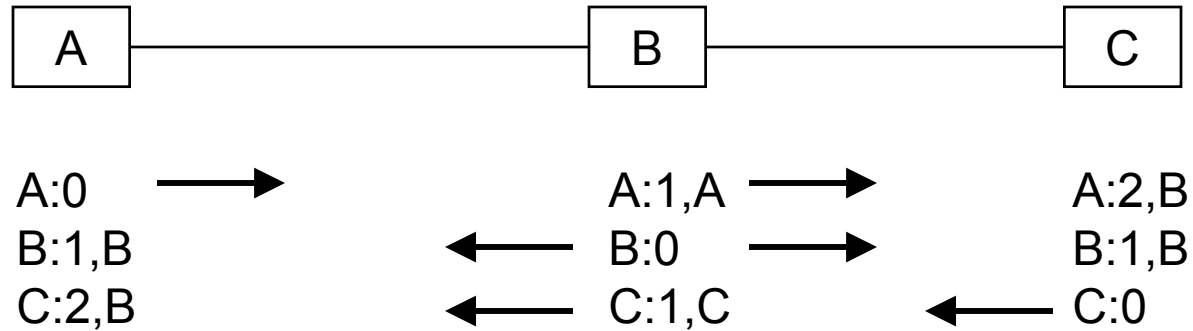- To combat the effects of counting-to-infinity.

- Routers do not announce routes to the link from which they learned them.

# Split Horizon

```
┌─────┐                      ┌─────┐                    ┌─────┐
│  A  │──────────────────────│  B  │────────────────────│  C  │
└─────┘                      └─────┘                    └─────┘
```

A:0      ──────▶             A:1,A ──────▶              A:2,B
B:1,B         ◀──────        B:0   ──────▶              B:1,B
C:2,B         ◀──────        C:1,C         ◀──────      C:0

# Split Horizon

- What happens when a link dies?



A:0

B:1,B

C:2,B

A:1,A

B:0

A:2,B

B:1,B

C:0

# Split Horizon

- What happens when a link dies?

A        B       ✗       C

A:0 →         A:1,A →✗     A:2,B

B:1,B ←       B:0 →✗     B:1,B

                             ✗← C:0
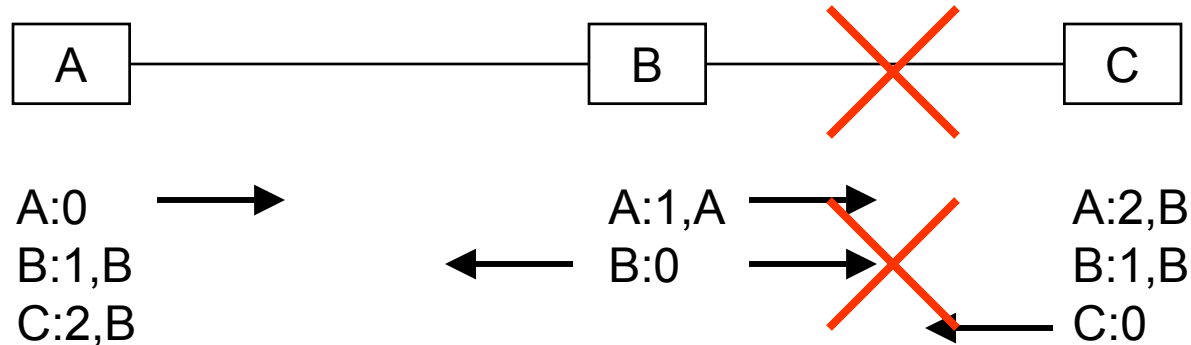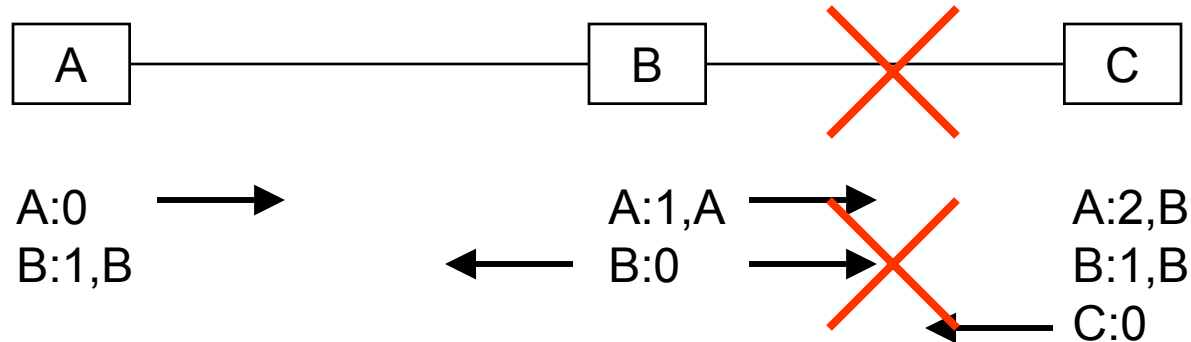
- Route to C expires after it hasn't been refreshed for a while.
- This way we avoid counting to infinity.

# Split Horizon with Poison Reverse
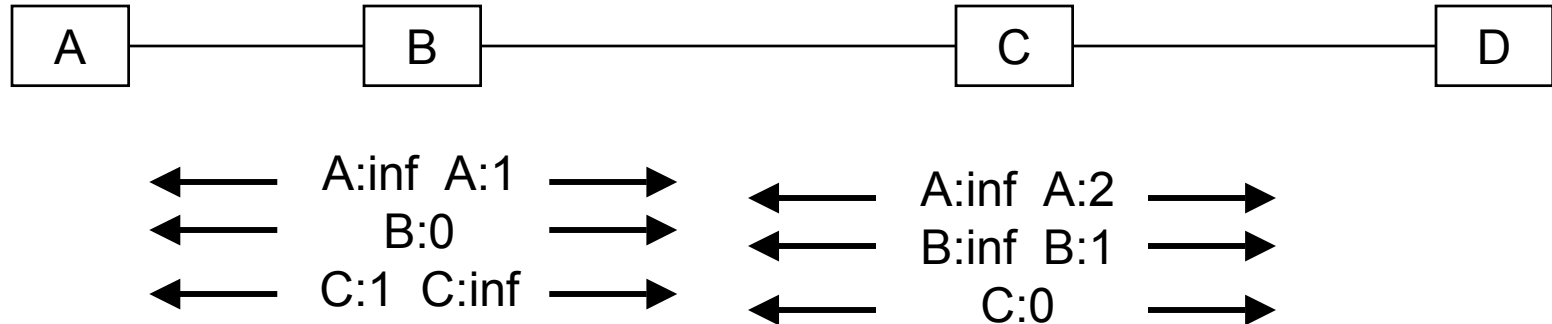
- Instead of suppressing routes, advertise them with infinite metric:

```
┌───┐              ┌───┐                     ┌───┐                  ┌───┐
│ A │──────────────│ B │─────────────────────│ C │──────────────────│ D │
└───┘              └───┘                     └───┘                  └───┘
```

←────── A:inf  A:1 ──────→        ←────── A:inf  A:2 ──────→
←────── B:0 ──────→               ←────── B:inf  B:1 ──────→
←────── C:1  C:inf ──────→        ←────── C:0 ──────→

# Split Horizon with Poison Reverse

- When the A-B link goes down, B believes C's announcement:

```
 ┌───┐      ✗      ┌───┐                    ┌───┐              ┌───┐
 │ A │             │ B │                    │ C │              │ D │
 └───┘             └───┘                    └───┘              └───┘

                    A:inf                ←─────  A:inf  A:2  ─────→
          ←─────    B:0    ─────→         ←─────  B:inf  B:1  ─────→
          ←─────  C:1  C:inf  ─────→      ←─────         C:0  ─────→
```
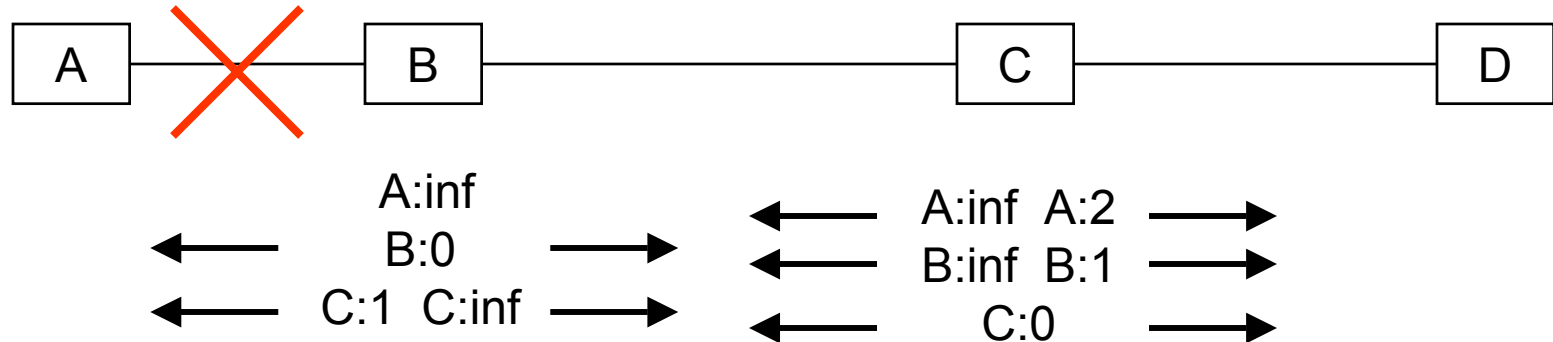
# Split Horizon with Poison Reverse

- Which he propagates (obviously as infinity)

```
┌─────┐       ┌─────┐                    ┌─────┐              ┌─────┐
│  A  │───╳───│  B  │────────────────────│  C  │──────────────│  D  │
└─────┘       └─────┘                    └─────┘              └─────┘

              A:inf    ──────▶      ◀──────   A:inf  A:2   ──────▶
   ◀──────    B:0      ──────▶      ◀──────   B:inf  B:1   ──────▶
   ◀──────    C:1  C:inf  ──────▶   ◀──────          C:0   ──────▶
```
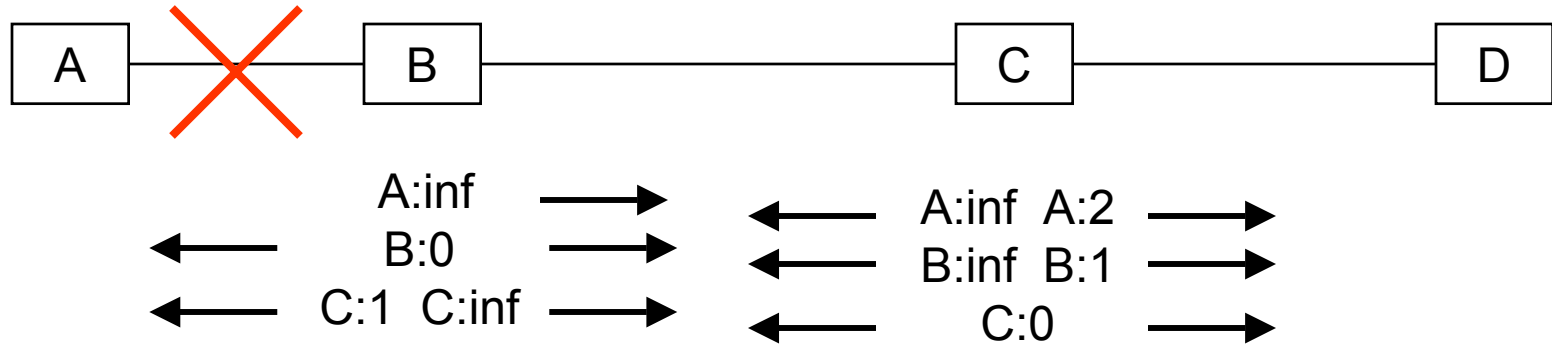
# Split Horizon with Poison Reverse

- Letting C know that A is now unreachable.



```
┌───┐      ╳      ┌───┐                    ┌───┐                    ┌───┐
│ A │             │ B │                    │ C │                    │ D │
└───┘             └───┘                    └───┘                    └───┘

              A:inf  ──→                ←──  A:inf  A:inf  ──→
       ←──    B:0    ──→                ←──  B:inf  B:1    ──→
       ←──    C:1  C:inf ──→            ←──  C:0           ──→
```
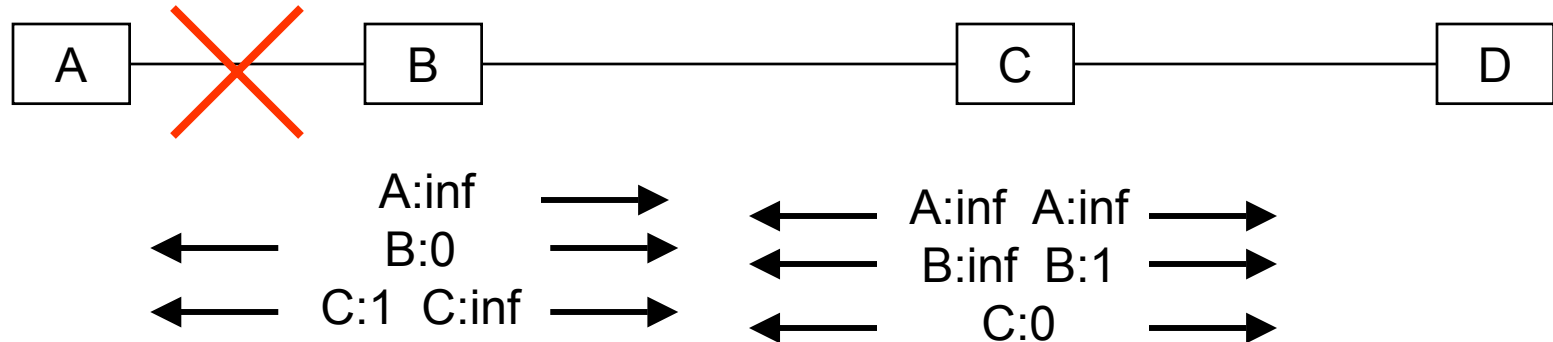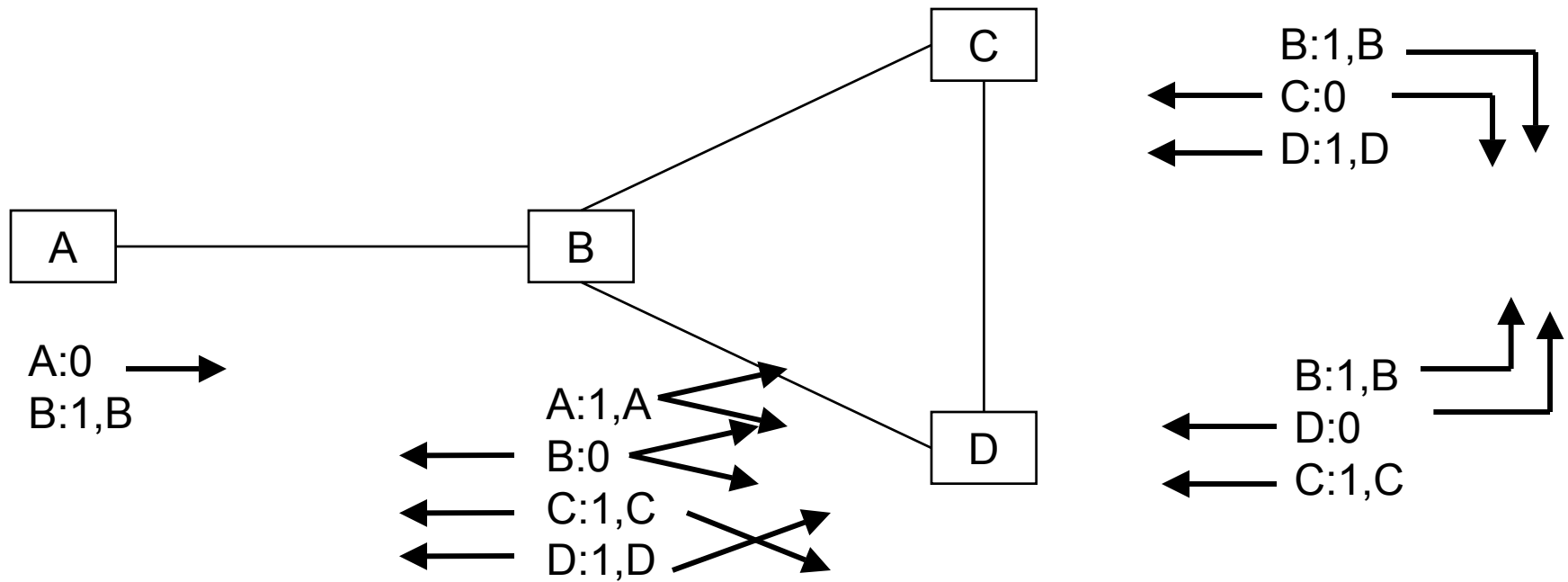
- SHwPR speeds protects against incorrect routes that may appear.

- Bad news is better than no news!

# Split Horizon Does not Always Work
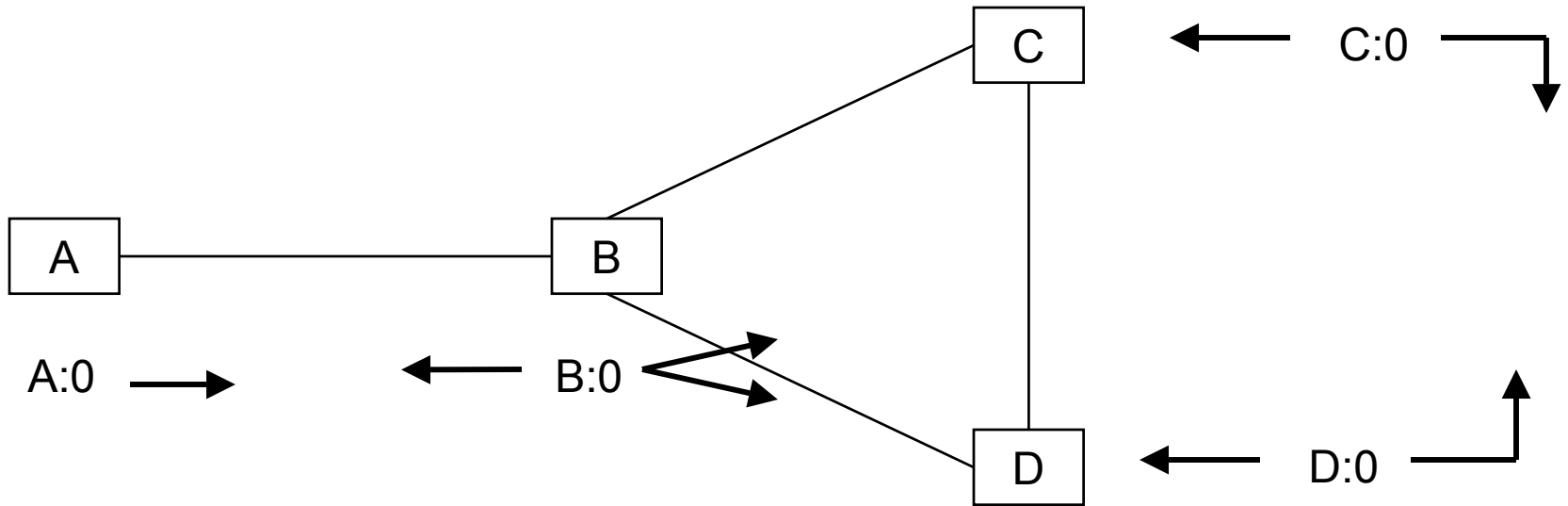


A:0

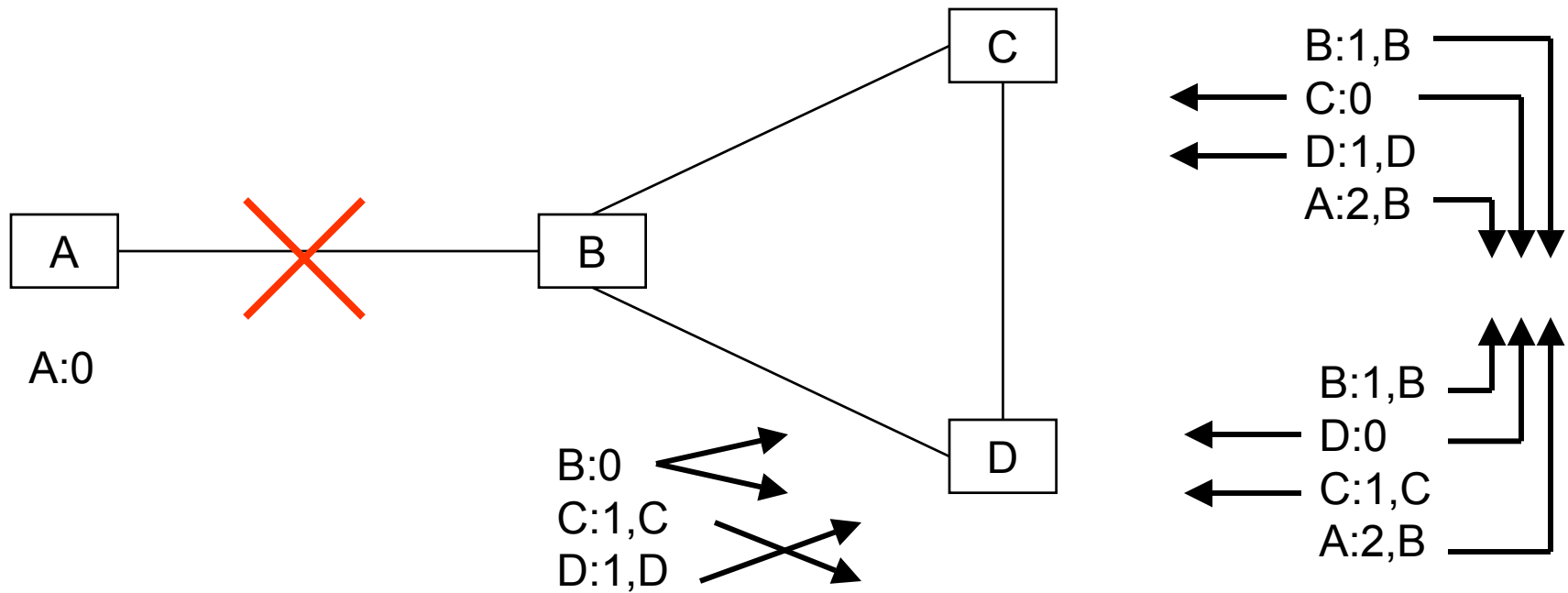B:0

C:0

D:0

# Split Horizon Does not Always Work



C

B:1,B
C:0
D:1,D

A

B

A:0
B:1,B

A:1,A
B:0
C:1,C
D:1,D

D

B:1,B
D:0
C:1,C

# Split Horizon Does not Always Work



C

B:1,B
C:0
D:1,D
A:2,B

A

B

A:0
B:1,B
C:2,B
D:2,B

A:1,A
B:0
C:1,C
D:1,D

D

B:1,B
D:0
C:1,C
A:2,B

# Split Horizon Does not Always Work



A:0

B:0
C:1,C
D:1,D

B:1,B
C:0
D:1,D
A:2,B

B:1,B
D:0
C:1,C
A:2,B

# Split Horizon Does not Always Work

C

B:1,B
C:0
D:1,D
A:3,D

A    B

A:0

B:0
C:1,C
D:1,D

D

B:1,B
D:0
C:1,C
A:3,C

# Split Horizon Does not Always Work

C

B

A

D

A:0

B:1,B
C:0
D:1,D
A:2,B

A:3,C
B:0
C:1,C
D:1,D

B:1,B
D:0
C:1,C
A:2,B
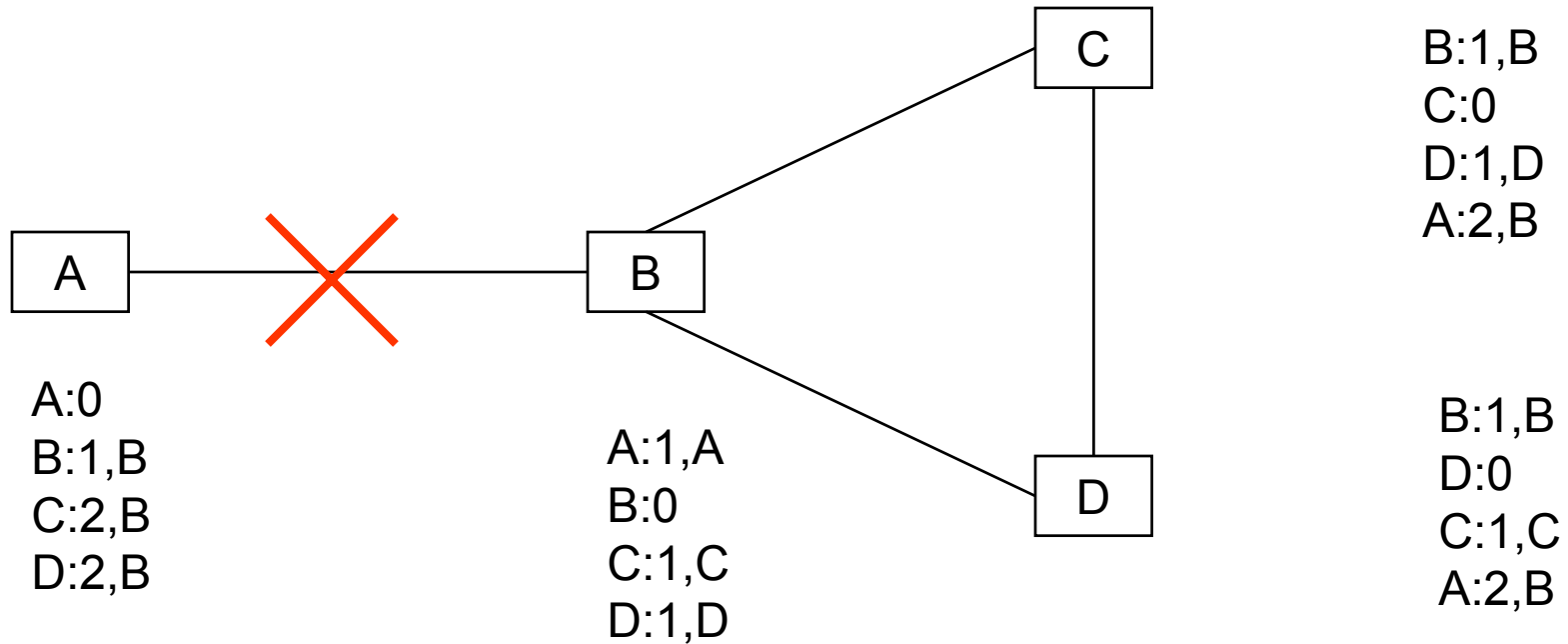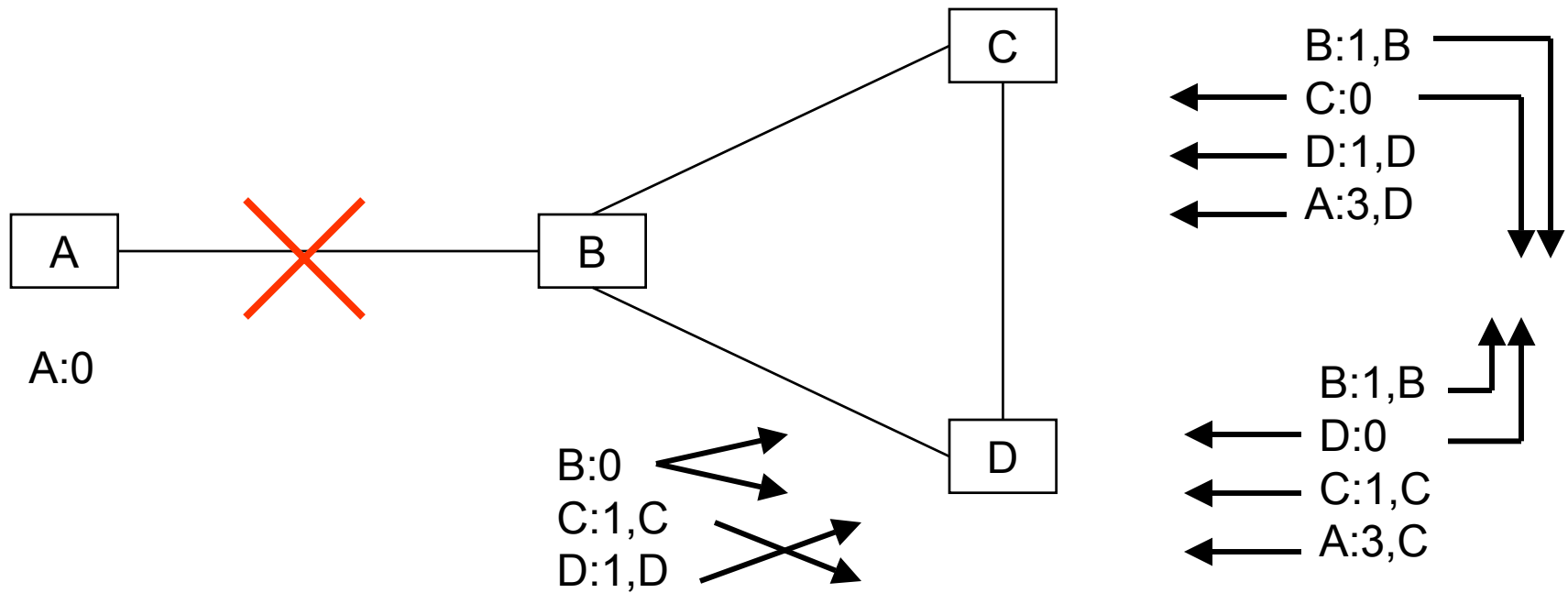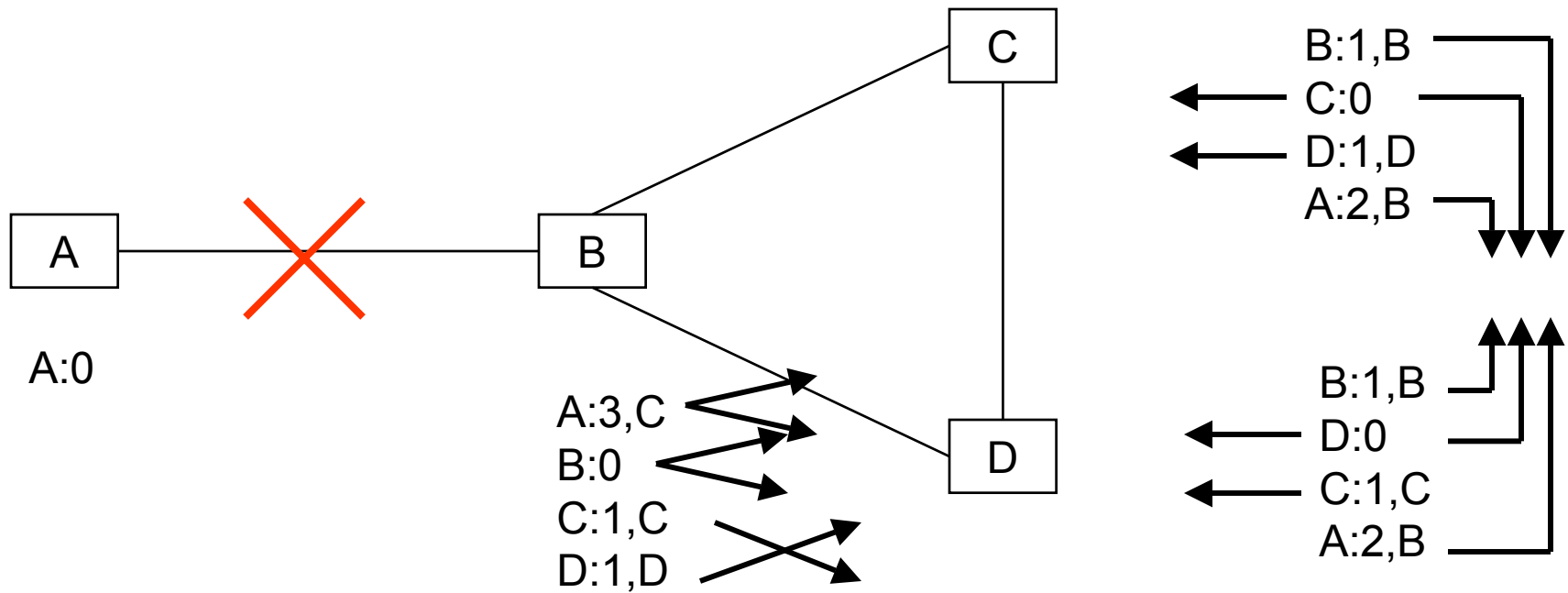
# Triggered Updates

- (aka Flash Updates).

- If a metric changes (up or down), a router immediately propagates that change.

- Decreases convergence time.

- Regular updates still occur.

- Only include prefixes that actually triggered the update.

- It's obvious.

- But RIP didn't do it this way initially.

- Bad information can also propagate very quickly.

# Holddown Timers

- If a route metric increases, no new updates for that route are accepted until the holddown timer expires.

- Too short – ineffective.

- Too long – increases convergence time.

# Asynchronous Updates

- Left unattended, DV routers on a broadcast network synchronize themselves.

- Increased collisions (even with Ethernet CD and exponential backoff).


- Add jitter to each update period (~15%).

# RIP (v1)

- Simplest of the DV protocols currently in use.

- Traces its ancestry to the PUP GWINFO and XNS RIP.

  – Apple RTMP and IPX RIP also trace their ancestry to XNS RIP.

- 4.2BSD (1983) implements RIP in routed(8).

- RFC 1058 written in 1988!


- UDP-based, Port 520, broadcast.

- Classful.

- Request and response messages.

- Metric is hop-count.

- Infinity is 16.

# RIP (v1) Cont'd

- On startup, broadcast request on all interfaces.
- When a request is received, a response is sent.
  - Response contains entire routing table.
- A response is also gratuitously sent every 30s (jittered).
  - Contains full routing table minus entries suppressed by SHwPR.
- Routes are invalidated (set to 16) after 6 update periods.
- Routes are flushed after another 4 update periods.

- Triggered updates do not reset the update timer.
  - TU cannot be re-triggered for 1-5 seconds.

# RIP (v1) Cont'd

- When a response is received:
  - If the route is not in the table, it is inserted.
  - If the route has the same metric, exp timer is reset.
  - If the route has a lower metric than what is in the table, it replaces it.
  - If the route has a higher metric AND update came from same router:
    - Marked as unreachable for holddown period.
    - If adv persists beyond holddown period, route is installed.

# RIP (v1) Message Format

| Command | Version (1) | Reserved (0) |
|---------|-------------|--------------|
| AF (2) | | Reserved (0) |
| IP Address | | |
| Reserved (0) | | |
| Reserved (0) | | |
| Metric | | |

≈≈ *Up to 25 route entries* ≈≈

| AF (2) | Reserved (0) |
|--------|--------------|
| IP Address | |
| Reserved (0) | |
| Reserved (0) | |
| Metric | |

# RIPv2

- Incremental improvement to RIP.

- RFC2453.

- Packets are multicast to 224.0.0.9.

- Additional features:

    – Authentication.

    – Classless.

    – Route tags.

    – Router IP address.

# RIPv2 Message Format

| Command | Version (2) | Reserved (0) | |
|---|---|---|---|
| AF (2) | | Route tag | |
| IP Address | | | |
| Netmask | | | |
| Next Hop | | | |
| Metric | | | |

*Up to 25 route entries*

| AF (2) | Reserved (0) |
|---|---|
| IP Address | |
| Reserved (0) | |
| Reserved (0) | |
| Metric | |

# RIPv2 Authentication

| Command | Version (2) | Reserved (0) | |
|---|---|---|---|
| FFFF | | Auth type | |
| Auth Data (bytes 0-3) | | | |
| Auth Data (bytes 4-7) | | | |
| Auth Data (bytes 8-11) | | | |
| Auth Data (bytes 12-15) | | | |

- Default auth is just a password (useless).

- Cisco uses an MD5 hash that includes a password (somewhat better).

- Up to 24 route entries follow.

# Other DV protocols

- Cisco IGRP and EIGRP.
    - (Enhanced) Interior Gateway Routing Protocol.
    - Proprietary.
    - Flexible (complex!) metric definition.
    - Have areas/autonomous systems.
- Read Doyle and cisco documentation for details.
- BGP is a "path-vector" protocol.
    - We'll examine it in excruciating detail.
- Older, defunct protocols.
    - EGP/Hello.

# Projects

- WAR (Wireless Anonymous Routing).

- ASRAP (AS Routing Authority Protocol).

- IOS config to DML.

- Interface SSF with NAM.

- "Come-From" table feasibility.

- Explore LS protocols for Interdomain routing.

- Packet classifier for BSD routing/tunneling/filtering.

- Routing protocols for overlay networks.

- Route/topology mapping.

- Anything else you can come up with that I will find interesting and appropriate.

# Project Mechanics

- Think about what you would be interested in exploring.
- Start thinking about forming groups (1-4 people; 3 preferred).
  - I can play matchmaker.
- Do your literature search (I can point you to relevant papers).
- Pick a project by 10/8.
  - If not, one will be assigned to you by the court.
- Project proposal (2-4 pages) due 10/22.
- Projects (with a 5-minute presentation) are due 12/10.
- Yes, you can do your project as a joint project with another course (e.g., W4180), but the expected output will be scaled accordingly.
- Any code you write should be open-source under a BSD license.
  - Clear this with your corporate lawyers (esp. CVN people)!