

# Why Do We Need Routing?

---

Lecture 4, Spring 2026

## Defining the Routing Problem

- **Why Do We Need Routing?**
- Modeling the Network
- What Makes Routing Hard?
- Types of Routing Protocols

## Routing States

- Destination-Based Forwarding
- Least-Cost Routing

## Static Routing

## Today's Goal: Routing

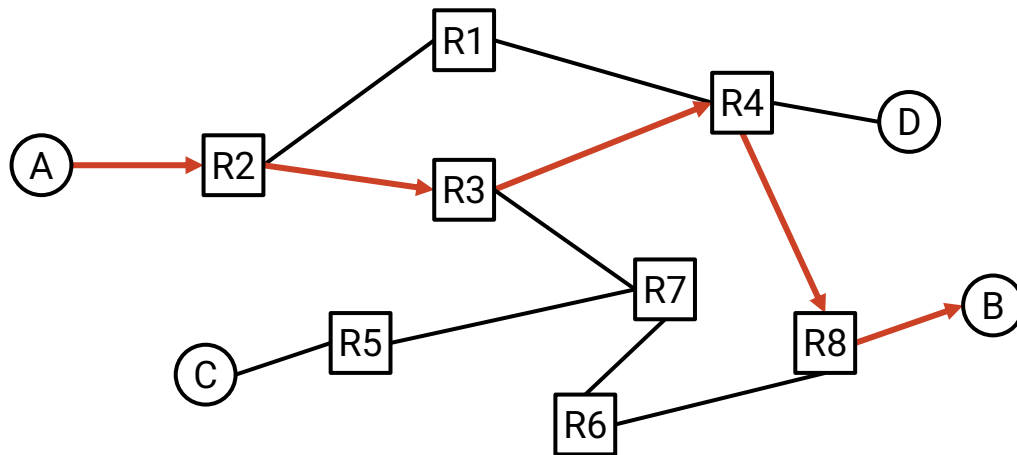
Today's goal is **routing**: Finding paths through the network.

We'll first formally define the problem.

- Why is it needed?
- Why is it a hard problem?
- What types of algorithms exist?

Then, we'll see how to assess a solution.

- What does a solution look like?
- What makes a solution valid?
- What makes a solution good?



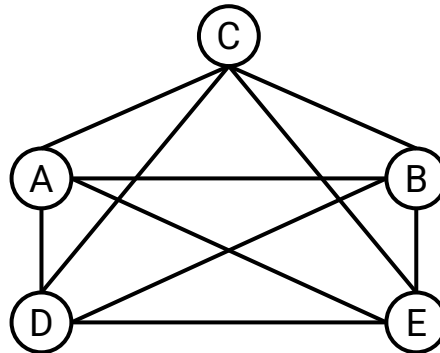
## Why Do We Need Routing? – Full Mesh Topology

---

If 2 machines want to communicate, we can add a link between them.

What if we had 5 machines?

- We could add a link between every pair of machines.
  - The result is called a **full-mesh** topology.
- Problem: This doesn't scale well. Imagine adding a new machine.
- Has benefits in limited settings.
  - Good if we need dedicated high bandwidth between every pair of machines.

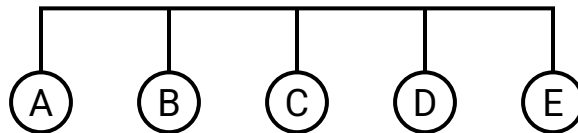


## Why Do We Need Routing? – Single-Link Topology

---

Another approach: Use a single link (wire) to connect all 5 machines.

- Scales better than the full-mesh topology.
- Problem: Less bandwidth available. Everyone has to share the link.



## Why Do We Need Routing? – Introducing Routers

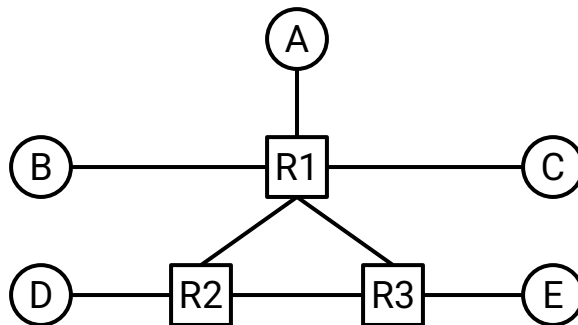
---

To use more sophisticated topologies, we need to introduce a **router**.

- Router: An intermediate machine that can forward data.

If we use routers to connect the machines:

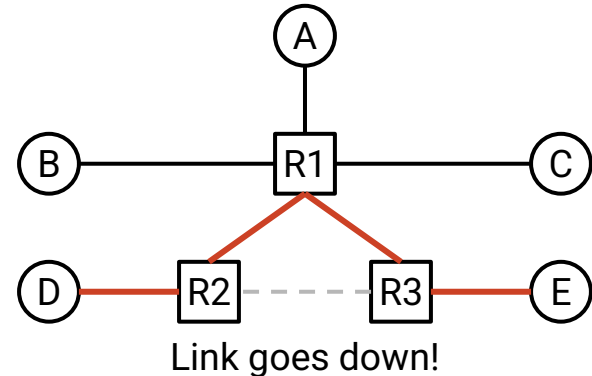
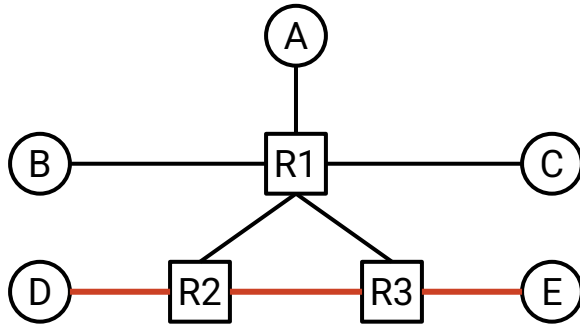
- We use fewer links than the full-mesh topology.
- We have more capacity than just a single link.



## Why Do We Need Routing? – Introducing Routers

Another benefit: If a link goes down, traffic could use another path.

We now need some way to compute paths through the network: Routing protocols!



# Modeling the Network

---

Lecture 4, CS 168, Spring 2026

## Defining the Routing Problem

- Why Do We Need Routing?
- **Modeling the Network**
- What Makes Routing Hard?
- Types of Routing Protocols

## Routing States

- Destination-Based Forwarding
- Least-Cost Routing
- Static Routing

## Modeling the Network – Routers vs. Hosts

---

We'll assume every machine on the network is one of two types.

**End hosts** send and receive packets, to and from other end hosts.

- Example: Your personal computer.
- End hosts don't forward intermediate packets.

**Routers** forward intermediate packets.

- Example: Your home router, heavy-duty router in a datacenter.
- For now, assume routers don't send and receive packets of their own.

Industrial-sized routers.



Home router.



Router symbol in diagrams.





## Modeling the Network – Graph

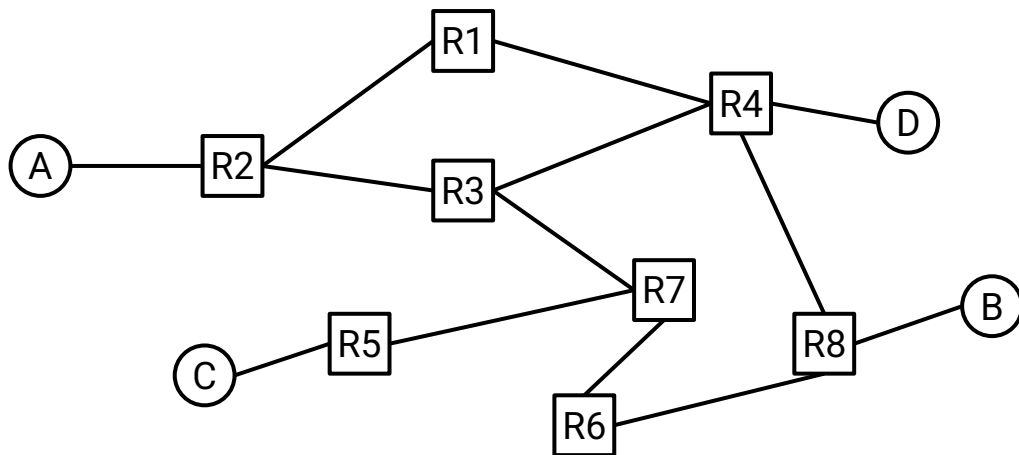
---

We'll draw the network as a graph.

- Each edge represents a link. For now, assume a link connects exactly 2 machines.
- For now, assume each machine is identified by a unique label.
  - We'll think more about addressing later.

(A) = End host

[R1] = Router



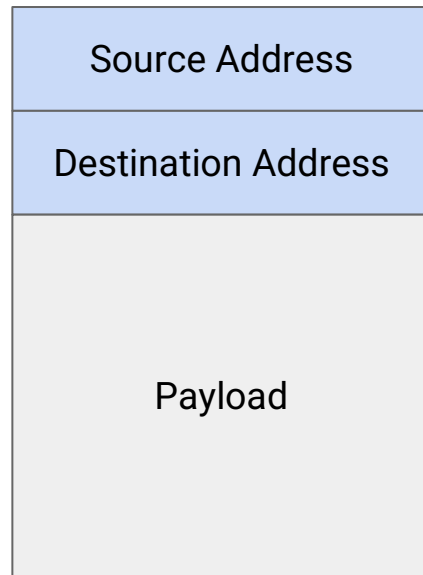
**Packets** are the basic unit of data sent across the network.

Packets have a header with metadata.

- For now, we only care about the source address and destination address fields.

Packets have a payload with the application data.

- Example: Website contents, image, etc.
- Routing isn't concerned about the payload. It's just a blackbox sequence of 1s and 0s to forward.



# What Makes Routing Hard?

---

Lecture 4, CS 168, Spring 2026

## Defining the Routing Problem

- Why Do We Need Routing?
- Modeling the Network
- **What Makes Routing Hard?**
- Types of Routing Protocols

## Routing States

- Destination-Based Forwarding
- Least-Cost Routing

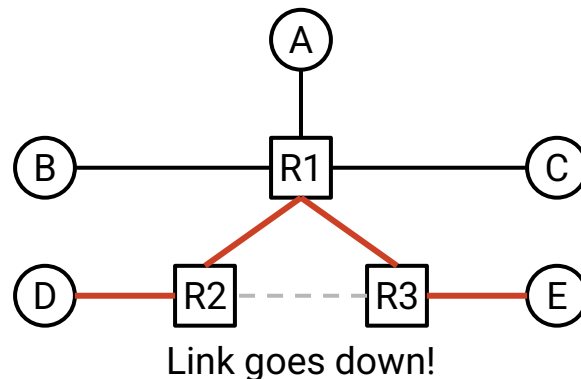
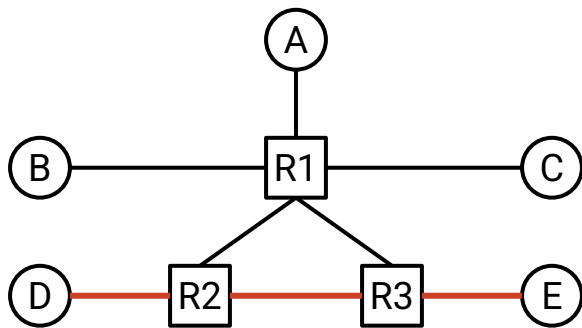
## Static Routing

## What Makes Routing Hard? (1/3) – Changing Topologies

The network graph is constantly changing.

- Hosts join and leave the network.
- Links can fail, and new links can be added.

Our routing protocol needs to be robust to changes.



## What Makes Routing Hard? (2/3) – Distributed Protocols

Routers don't have a global, birds-eye view of the network.

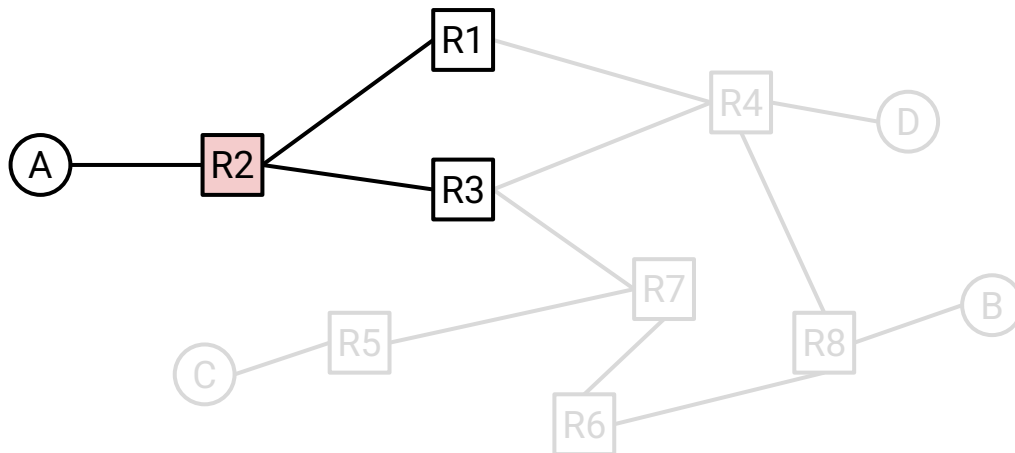
- If a link fails, routers don't automatically know about it.

Routing protocols have to be **distributed**.

- There's no central mastermind computing the answer.
- Each router computes its own part of the answer.
- Routers must coordinate with each other in the protocol.

R2 can't see the whole network.

R2 might only be able to know about its direct neighbors.



At Layer 3, links are best-effort. Packets could get dropped.

In summary, challenges of routing:

1. Topology changes over time.
2. Routers don't have a global view of the network.
3. Links are best-effort.

# Types of Routing Protocols

---

Lecture 4, CS 168, Spring 2026

## Defining the Routing Problem

- Why Do We Need Routing?
- Modeling the Network
- What Makes Routing Hard?
- **Types of Routing Protocols**

## Routing States

- Destination-Based Forwarding
- Least-Cost Routing
- Static Routing

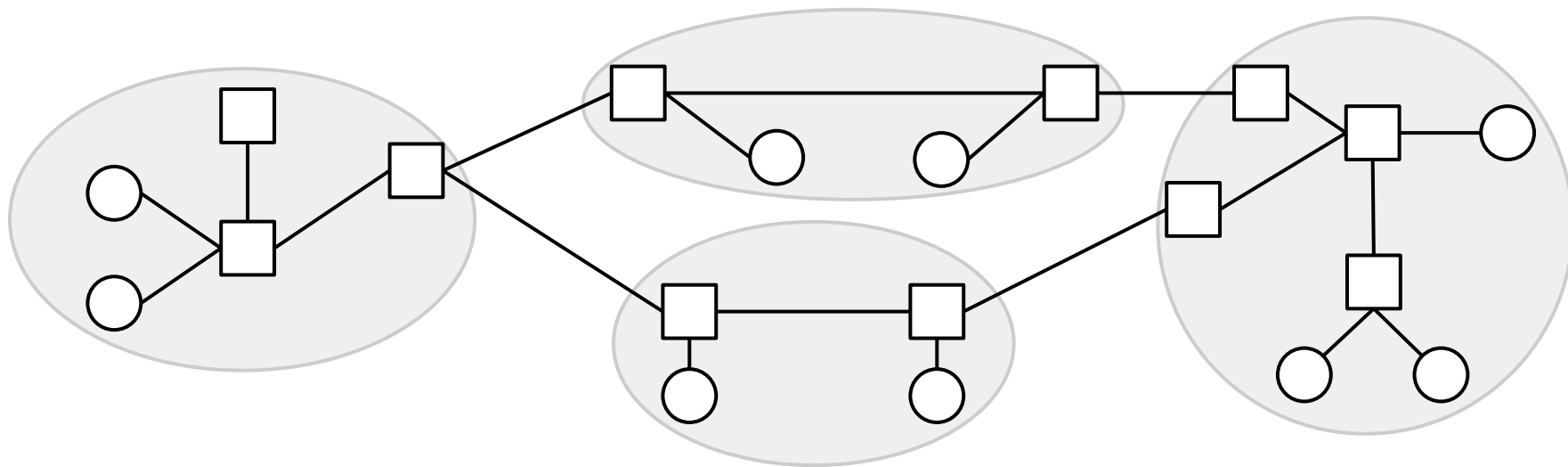
## Routing Protocols – Disclaimer

There are an endless number of possible routing protocols.

- We'll see some alternative approaches later.

The Internet is a network of networks.

- The Internet does not have a single giant routing protocol.





## Intra-Domain vs. Inter-Domain Routing

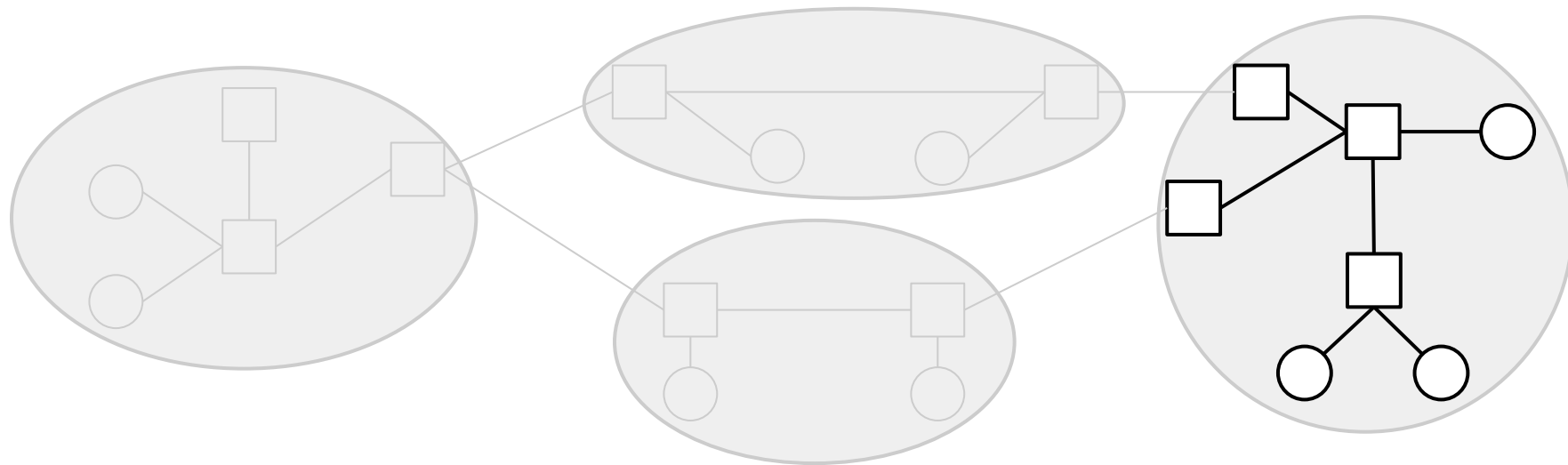
---

**Intra-domain** routing protocols compute routes within a single network.

- Sometimes called **Interior Gateway Protocols (IGPs)**.

Each network can choose their own intra-domain protocol based on their needs.

- Networks differ in size (geographic, number of hosts), capacity (bandwidth, latency), support (money, staff), etc.

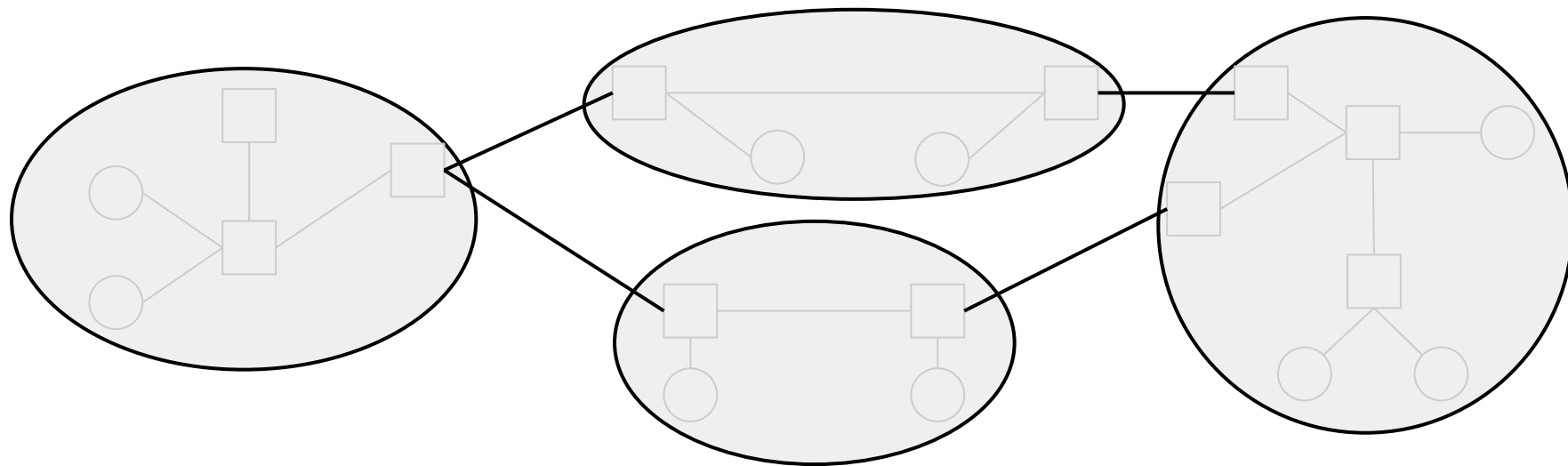


**Inter-domain** routing protocols compute routes between networks..

- Sometimes called **Exterior Gateway Protocols (EGPs)**.

Everybody has to agree on the same protocol.

- The Internet has used BGP since the 1990s.



## Classifying Routing Protocols

---

Classifying routing protocols by *where* they operate:

- Let individual networks choose how to route *inside* their network. (Intra-domain.)
- Have all networks agree on how to route *between* each other. (Inter-domain.)

In practice, the lines between intra-domain and inter-domain routing are blurred.

- Example: BGP is sometimes used inside networks, as well as between networks.

We can also classify routing protocols by *how* they operate:

- Distance-vector protocols.
- Link-state protocols.

# Destination-Based Forwarding

---

Lecture 4, CS 168, Spring 2026

## Defining the Routing Problem

- Why Do We Need Routing?
- Modeling the Network
- What Makes Routing Hard?
- Types of Routing Protocols

## Routing States

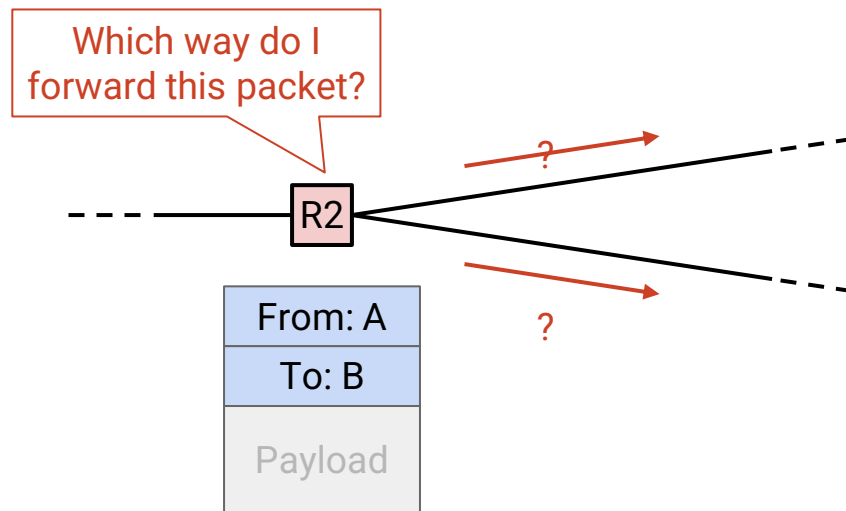
- **Destination-Based Forwarding**
- Least-Cost Routing

## Static Routing

## Routing Decisions

The basic challenge: When a packet arrives at a router, how does the router know where to send it next, such that it will eventually arrive at the desired destination?

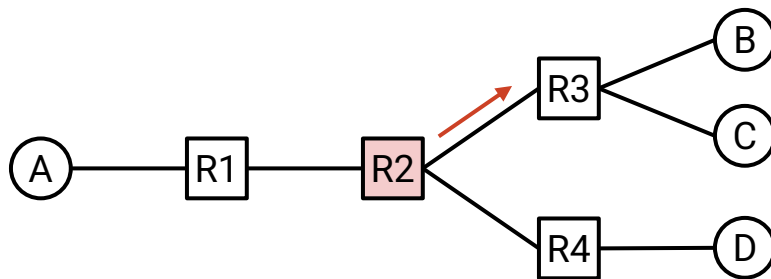
- The **next hop** is where to forward the packet next.



## Routing Decisions

The Internet uses **destination-based forwarding**.

- Each router keeps a table, mapping destinations to next hops.
- The decision only depends on the destination field of the packet.



When a packet arrives, look up the destination...

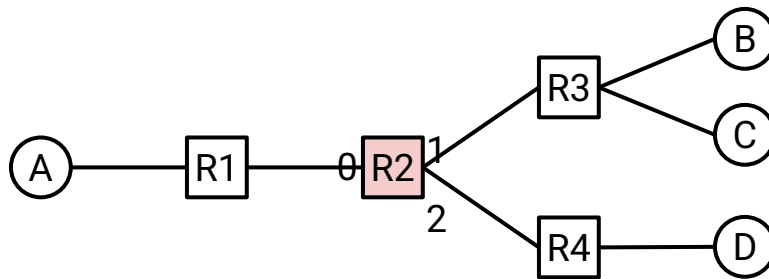
From: A
To: B
Payload

...and send along the corresponding next hop.

R2's Table	
Destination	Next Hop
A	R1
B	R3
C	R3
D	R4

In real life, the table often uses physical ports instead of next hops.

- Conceptual: "Send to next-hop of R3."
- Reality: "Send out of physical port 1."
- We'll use the conceptual picture for simplicity.



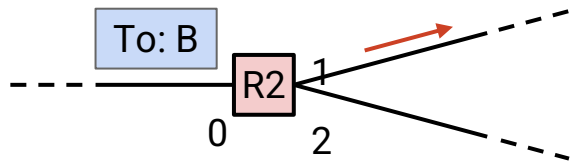
R2's Table (Conceptual)	
Destination	Next Hop
A	R1
B	R3
C	R3
D	R4

R2's Table (Reality)	
Destination	Port
A	0
B	1
C	1
D	2

# Routing vs. Forwarding

## Forwarding:

- Look up packet's destination in table, and send packet to neighbor.
- Inherently *local*. Depends only on arriving packet and local table.
- Occurs every time a packet arrives (nanoseconds).

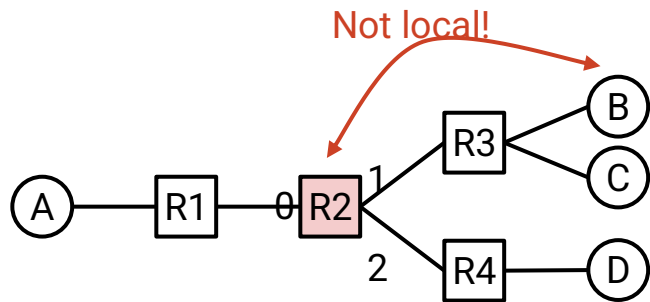


Forwarding is local.

R2's Table	
Destination	Port
A	0
B	1
C	1
D	2

## Routing:

- Communicates with other routers to determine how to populate tables.
- Inherently *global*. Must know about all destinations, not just local ones.
- Occurs every time the network changes (e.g. a link fails).



Routing is global.



## Defining the Routing Problem

- Why Do We Need Routing?
- Modeling the Network
- What Makes Routing Hard?
- Types of Routing Protocols

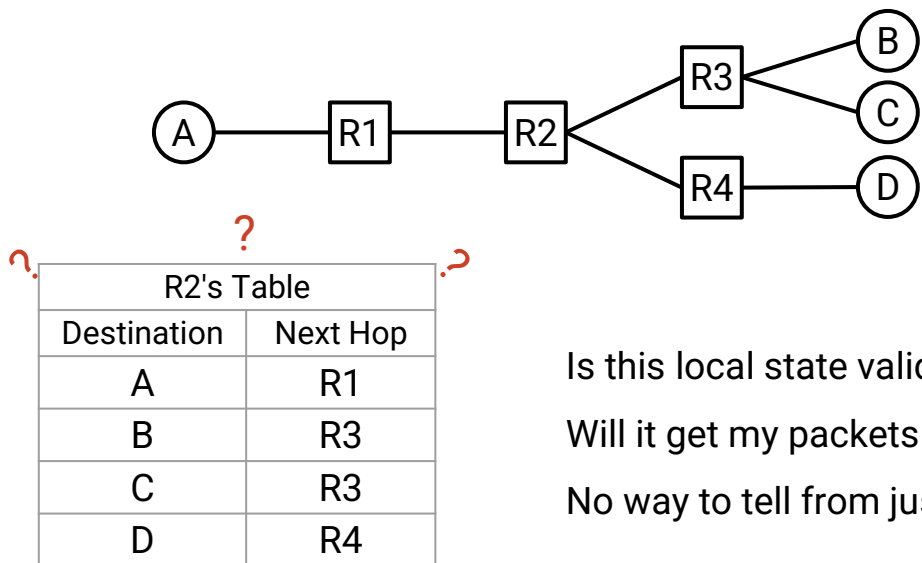
## **Routing States**

- Destination-Based Forwarding
- Least-Cost Routing
- Static Routing

A routing state is **valid** if packets actually reach their destinations.

A **local** routing state is a table in a single router.

- By itself, the state in a single router can't be evaluated for validity.



Is this local state valid?

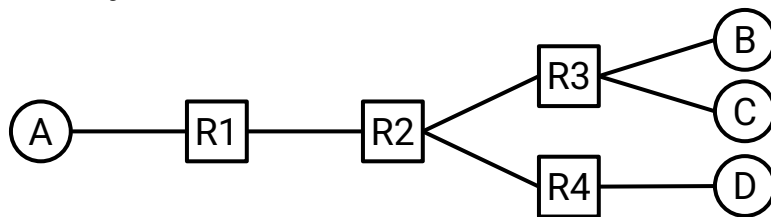
Will it get my packets to their destinations?

No way to tell from just this info!

A **global** routing state is a collection of tables in all routers.

- Global state determines the paths a packet takes.
- Global state is valid if it produces forwarding decisions that deliver packets to their destinations.

Given a global state, how can you tell if it's valid?



R1's Table	
Destination	Next Hop
A	A
B	R2
C	R2
D	R2

R2's Table	
Destination	Next Hop
A	R1
B	R3
C	R3
D	R4

R3's Table	
Destination	Next Hop
A	R2
B	B
C	C
D	R4

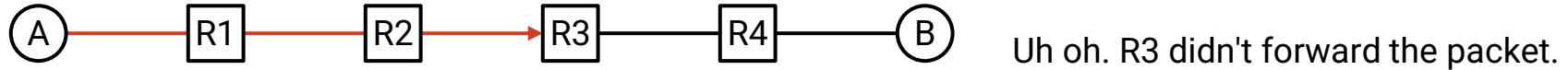
R4's Table	
Destination	Next Hop
A	R2
B	R2
C	R2
D	D

## Conditions

A global routing state is valid *if and only if* there are no dead ends and no loops.

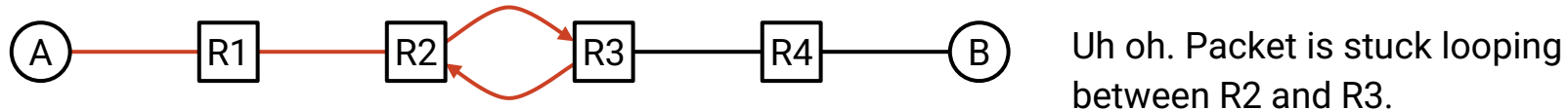
**Dead end:** A packet arrives at a router, but there is no next hop to forward it.

- Packet arriving at destination doesn't count as a dead end.



**Loop:** A packet cycles around the same set of routers.

- If forwarding only depends on destination field, if a packet gets stuck in a loop, it can never escape.



# Least-Cost Routing

---

Lecture 4, CS 168, Spring 2026

## Defining the Routing Problem

- Why Do We Need Routing?
- Modeling the Network
- What Makes Routing Hard?
- Types of Routing Protocols

## Routing States

- Destination-Based Forwarding
- **Least-Cost Routing**
- Static Routing

We now know what a *valid* solution looks like (no loops, no dead ends).

What makes a solution *good*?

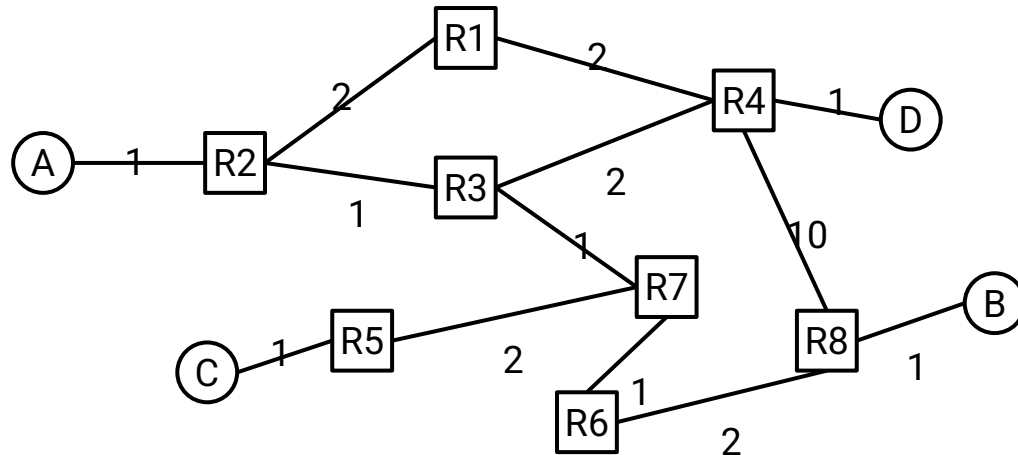
Many different ways to define good, including:

- Price.
- Propagation delay.
- Distance.
- Unreliability.
- Bandwidth constraints.

## Least-Cost Routing – Definition

**Least-cost routing:** Assign costs to every edge, and find paths with lowest cost.

- Cost depends on the metric the operator wants to minimize.
- Costs can be arbitrary. Routing protocols don't care where the costs come from.

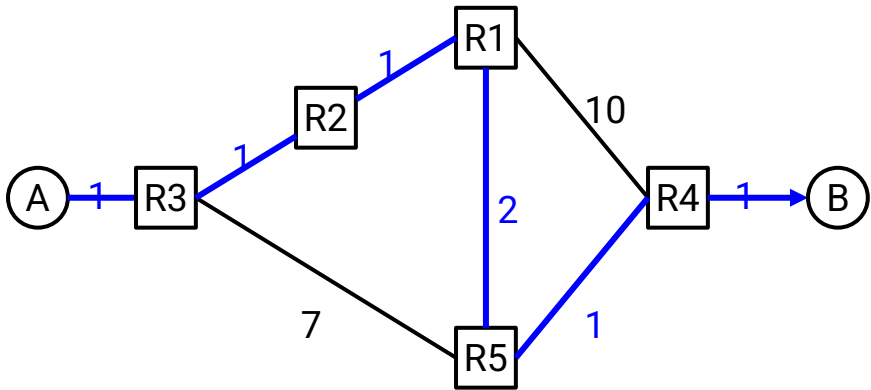


# Least-Cost Routing – Definition

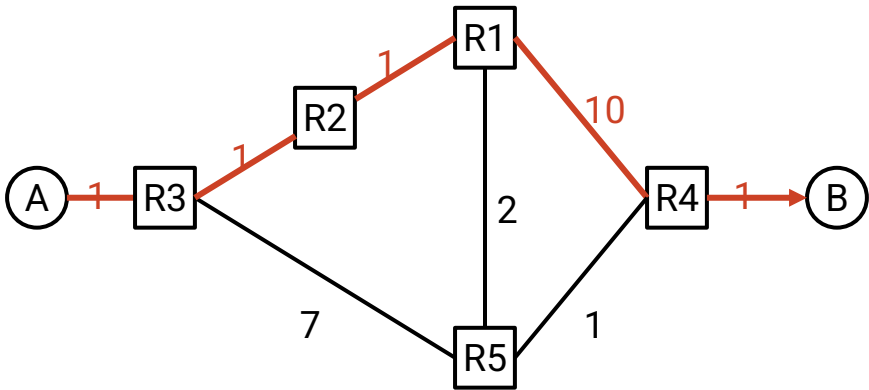
In least-cost routing, routers should forward packets such that they take the lowest-cost path to the destination.

Example: If all costs are 1, the protocol finds paths with the fewest hops.

- Usually, if edges are unlabeled, assume they have cost 1.



A → B Cost = 7



A → B Cost = 14



## Least-Cost Routing – Properties

---

Where do costs come from?

- Costs are local to a router. Each router knows the cost of its own links.
- In practice, generally configured by an operator.
  - Some protocols allow for auto-configuration.

Properties of costs:

- Costs are always positive integers.
  - Consistent with almost any practical metric you'd use.
- Costs are always symmetrical.
  - $A \rightarrow B$  costs the same as  $B \rightarrow A$ .
  - Exceptions possible (e.g. different upload/download bandwidth).
- These two assumptions will simplify our protocols.

*Good* paths (least-cost) also *valid* paths (no loops, no dead ends).

- Routes are still destination-based.
- Routes still form a spanning tree.

# Static Routing

---

Lecture 4, CS 168, Spring 2026

## Defining the Routing Problem

- Why Do We Need Routing?
- Modeling the Network
- What Makes Routing Hard?
- Types of Routing Protocols

## Routing States

- Destination-Based Forwarding
- Least-Cost Routing
- **Static Routing**

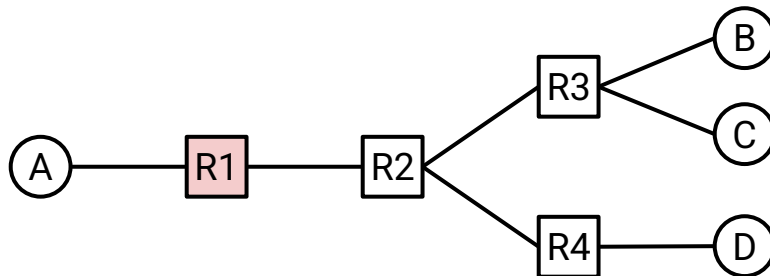
## Special Route Types

Recall: In a routing protocol, routers talk to each other to populate forwarding tables and learn paths to destinations.

Some table entries can be manually hard-coded.

- **Connected/Direct** routes let us forward to things we're connected to directly.
- These routes are created when the operator configures the router.
- No routing protocol needed for these entries.

R1's Table	
Destination	Next Hop
A	Direct
B	R2
C	R2
D	R2



Some table entries can be manually hard-coded.

- **Static** routes are hard-coded entries that we always want to be there.
- Router isn't necessarily directly connected to the destination.
- The route is static because:
  - It never changes.
  - No routing protocol used to discover it.
- Again, often manually created by an operator.

## Summary: Routing Principles

Routing allows us to find **paths** through the network.

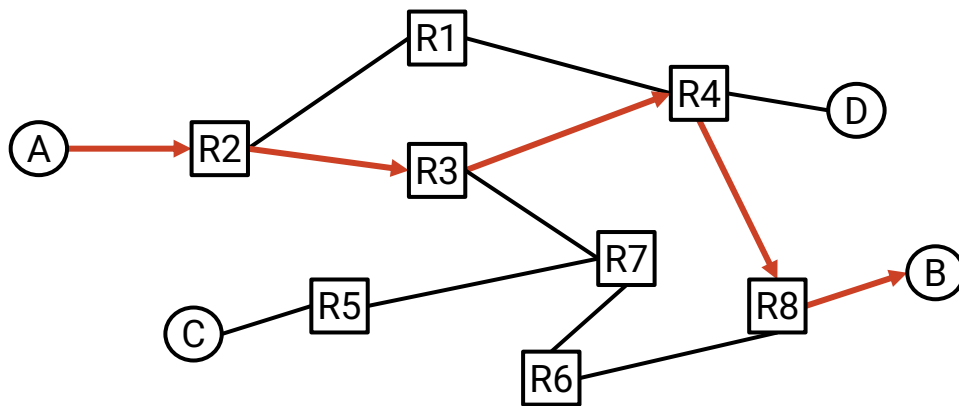
- Have to handle constant change, a non-global view, and best-effort links
- There are both intra-domain (choices!) and inter-domain routing protocols (BGP)

The Internet uses **destination-based forwarding**.

- Each router keeps a table, mapping destinations to next hops.

We need a valid directed delivery tree with **no dead ends and no loops!**

Static or direct  
routes can be  
hard coded!



R3's Table (Conceptual)	
Destination	Next Hop
A	R2
B	R4
C	R7
D	R4