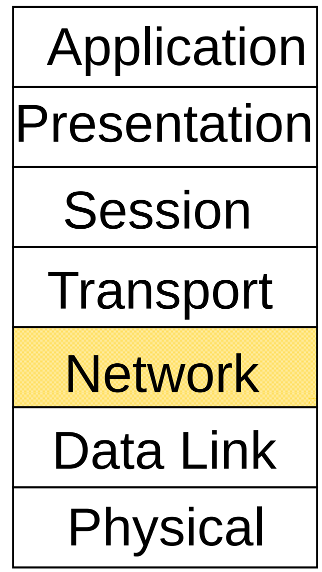
CSE561 – Routing

David Wetherall
djw@cs.washington.edu

Routing

- Focus:
  - How to find and set up paths through networks

- Distance-vector and link-state
- Shortest path routing
- Key properties of schemes
- Multicast

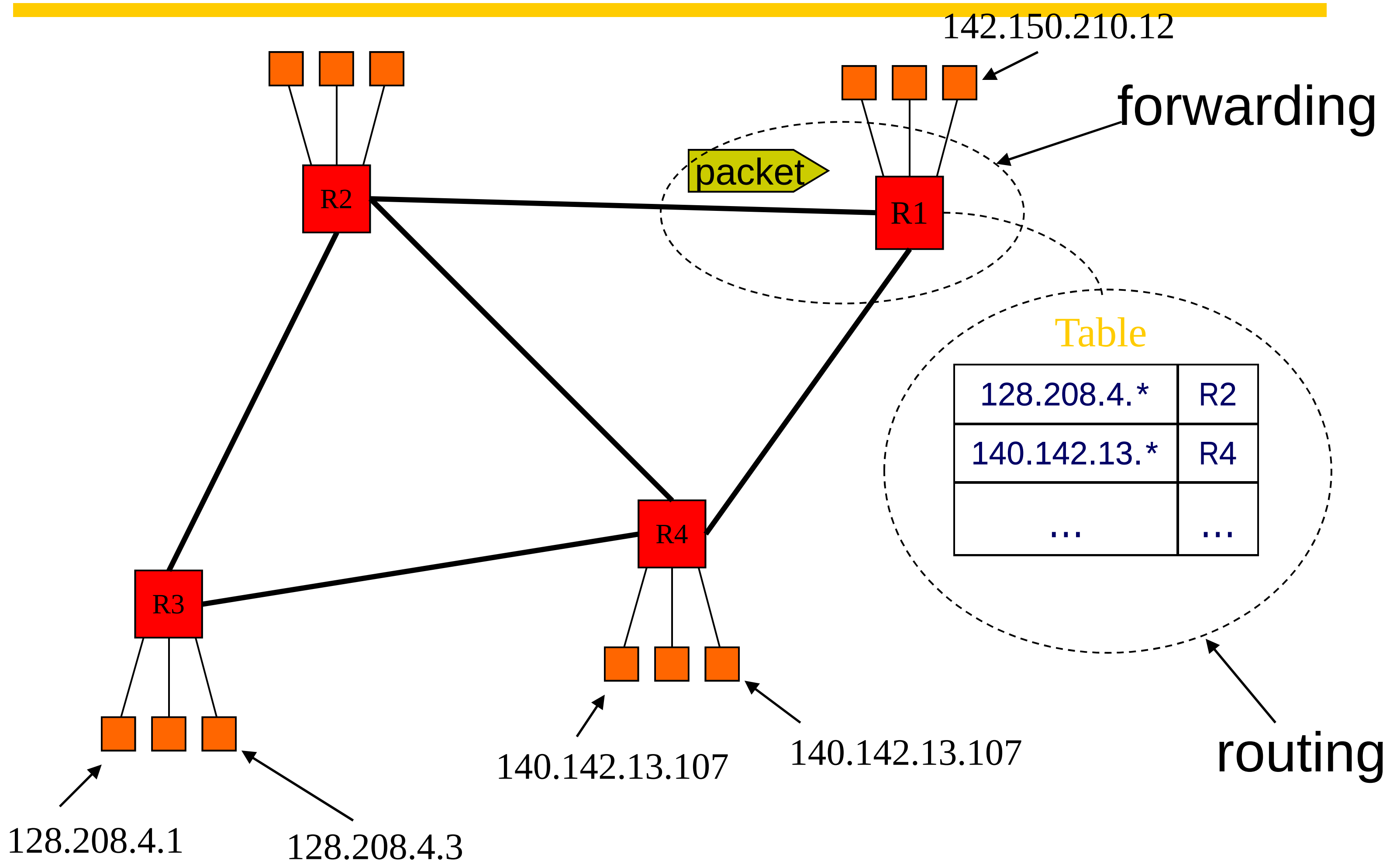


Routing versus Forwarding

- Routing is the process by which all nodes exchange control messages to calculate the *routes* packets will follow
  - Distributed process with *global* goals; emphasis is *correctness*
  - Nodes build a routing table that models the global network

- Forwarding is the process by which a node examines packets and sends them along their *paths* through the network
  - Involves *local* decisions; emphasis is *efficiency*
  - Nodes distill a forwarding table from their routing table (keyed by packet attributes, e.g., address) that gives the *next hop*

Routing versus Forwarding



128.208.4.*	R2
140.142.13.*	R4
…	…

Distance Vector Algorithm

- Each router maintains a vector of costs to all destinations as well as routing table giving next hops
  - Initialize neighbors with known cost, others with infinity
- Periodically send copy of distance vector to neighbors
- On reception of a vector, if your neighbor's path to a destination plus cost to that neighbor cost is better
  - Update the cost and next-hop in your outgoing vectors
- Assuming no changes, will converge to shortest paths

DV problem -- dynamics

- Good news (better routes) propagate quickly
- Bad news (failures) propagate slowly
  - inferred by exploration
- Leads to “count to infinity” loops
  - Many heuristics (split horizon, poison reverse)
  - Takes ordered updates to eliminate (e.g., EGIRP uses diffusing computations) that are complicated and slow convergence
  - No great solutions

- No longer widely used except for resource constrained or legacy networks.

Routing Information Protocol (RIP)

- DV protocol with hop count as metric
  - Infinity value is 16 hops; limits network size
  - Includes split horizon with poison reverse
- Routers send vectors every 30 seconds
  - With triggered updates for link failures
  - Time-out in 180 seconds to detect failures

- RIPv1 specified in RFC1058
  - www.ietf.org/rfc/rfc1058.txt
- RIPv2 (adds authentication etc.) in RFC1388
  - www.ietf.org/rfc/rfc1388.txt

Link State Routing

- Same assumptions/goals, but different idea than DV:
  - Tell all routers the topology and have each compute best paths
  - Two phases:
    1. Topology dissemination (flooding)
    2. Shortest-path calculation (Dijkstra's algorithm)

- Why?
  - In DV, routers hide their computation, making it difficult to decide what to use when there are changes
  - With LS, faster convergence and hopefully better stability
  - It is more complex though …

Open Shortest Path First (OSPF)

- Widely-used Link State protocol today; see also ISIS
- Basic link state algorithms plus many features:
  - Authentication of routing messages
  - Extra hierarchy: partition into routing areas
  - Load balancing: multiple equal cost routes

What is a “best” path anyhow?

- Ideally paths that:
  - Are as direct as possible (low latency)
  - Carry as much traffic as the network will fit (high bandwidth)
  - Carry traffic well for all of the nodes (fairness)

- This is a resource allocation problem with multiple constraints. Depends on topology and who sends how much traffic to who, which changes over time. Yikes!

- We want a simple, distributed solution

Lowest cost (“shortest path”) routes

- Compute paths independently for different node pairs
  - Assign a cost or weight to each link
  - Find lowest total weight path between source/dest
- Typically costs are fixed
  - Does not take hotspots into account
  - Has simple subset optimality properties
- Costs usually set as a function of bandwidth and delay
  - Can tweak (traffic engineering) to match traffic to topology
  - More direct paths help with low latency and high bandwidth, so does a reasonable overall job

Equal-cost multi-path (ECMP)

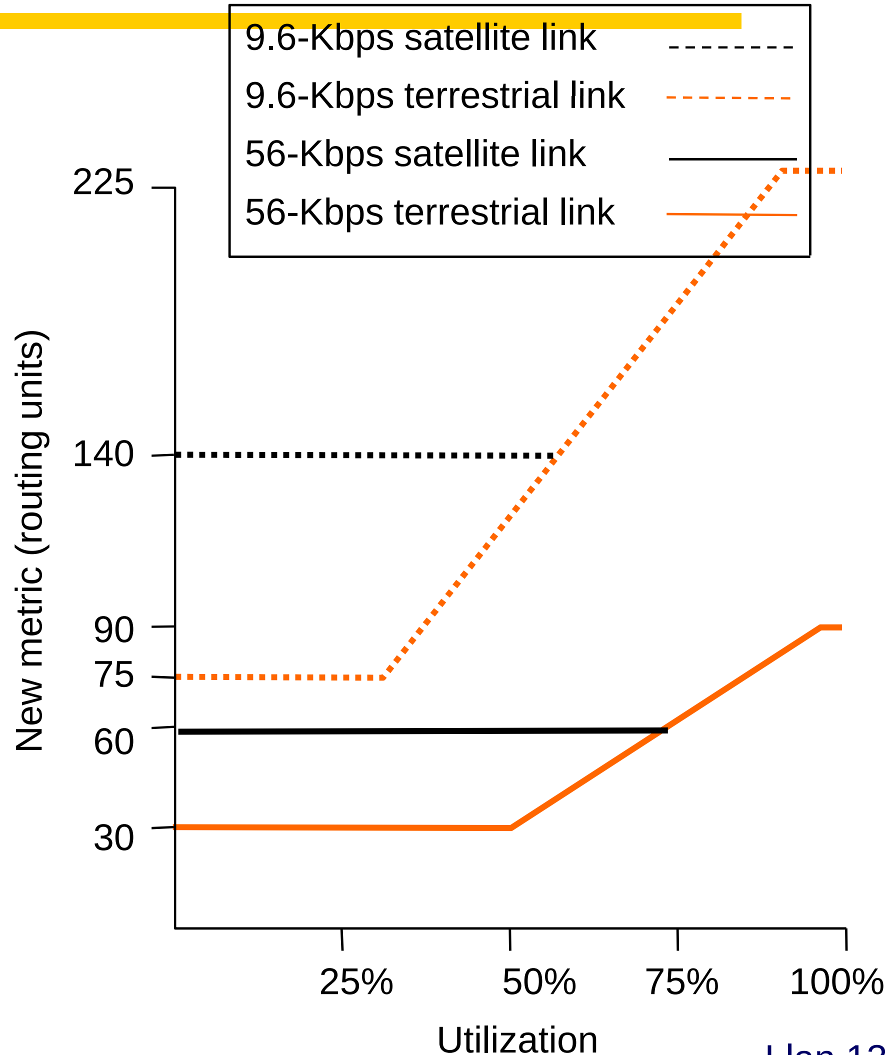
- Generalization for load balancing
  - Allow multiple paths if they have the same lowest cost
  - Remember our fish topology

- Single path lowest cost routing produces a spanning tree
- ECMP produces a directed acyclic graph
  - Still no possibility of loops
  - Simple for nodes: just keep a list of next hops

- Q: How to map traffic to the multiple paths?

What didn't work: Revised ARPANET Cost Metric

- Based on load and link
- Variation limited (3:1) and change damped
- Capacity dominates at low load; we only try to move traffic if high load
- Not stable



Resource allocation timescales today

- From fast (very reactive) to slow (carefully planned)
  - Use of different timescales largely decouples mechanisms

- Congestion control
  - Adapts to packet loss; slows source
- Routing
  - Adapts to failures; finds paths with connectivity
- Traffic engineering
  - Typically manual route adjustments for cost/performance
- Provisioning
  - Build out network to match traffic workload

Desirable properties

- Correctness
- Efficiency
- Fairness

- Rapid convergence
  - To correct routes that are stable after changes, with minimal transient loss
- Scalability
  - Of messages and router state
  - Particularly an issue for large, mobile, or multicast networks

Example

Property	Distance Vector	Link State
Correctness	Yes - Distributed Bellman Ford	Yes - Replicated shortest path
Efficiency	Approx- Least cost paths	Approx - Least cost paths
Fairness	Approx - Least cost paths	Approx - Least cost paths
Convergence	Slow – many exchanges	Fast – prop plus compute
Scalability	Good – O(1) per node/link	Moderate – at least O(edges)

Delivery models

- Unicast
  - single sender to single receiver
- Broadcast
  - Single sender to all receivers
- Multicast
  - Single sender to multiple (but not all) receivers (in a group)
- Anycast
  - Single sender to nearest receiver in a set

Broadcast

- Reverse Path Forwarding (RPF)
  - Simplest broadcast using unicast tables
- Given broadcast from source S. At each router:
  - Look up outgoing interface O to reach S.
  - If packet arrives on O then forward to all other interfaces
- Q: What assumptions does this make?
- Q: How does this compare to flooding?

Anycast

- Simple extension for DV and LS algorithms
- Same destination “appears” at multiple places
  - Each router chooses the next hop with the lowest cost to the destination as before

- Used in the Internet for root nameservers
  - This is BGP routing across ISPs though, not within an ISP

Multicast

- Long history:
  - Multicast is simple on LANs (just broadcast) and useful for service discovery ("Oi! Who is the printer here?")
  - Brilliant idea – let's add it to the Interent
  - But it turned out to be complex, motivated by bandwidth efficiency, and lacking a killer application
  - Finally happening, given simpler schemes and apps like IPTV for an ISP and datacenter distribution
- Requires group membership management
  - To decide who is in the group of receivers
- Key challenges are scalability and cross-ISP deployment
  - Handle dense and sparse cases separately

CBT discussion

- What would an ideal multicast route look like?
- How much state do routers need to keep with a DVMRP or MOSPF protocol?
- How much state do routers need to keep with a CBT protocol?
- What is the penalty for reducing state?
- Where should the core be located?
- Where should the core be located for a video broadcast?