Continued from last lecture ...

- We now know enough to write a (rough) algorithm for processing IP packets:

Algorithm (**IP packet processing**)

Given a packet provided by the lower, link layer:

1. validate header (e.g., use checksum to check for errors),
2. process options in header,
3. check destination address: if the packet is for this host

   3.1 buffer fragments and apply reassembly process, then eventually
   3.2 provide payload to a higher layer (per protocol field)

   otherwise, assuming we want to forward the packet

   3.3 check TTL, and drop if exceeded,
   3.4 look-up next hop in forwarding table,
   3.5 apply fragmentation and header update processes, (e.g., decrement TTL, recompute checksum),
   3.6 transmit packet(s) via lower, link layer

and if/when errors occur, signal them appropriately.

- We now know enough to write a (rough) algorithm for processing IP packets:

Algorithm (IP packet processing)

Given a packet provided by the lower, link layer:

1. validate header (e.g., use checksum to check for errors),
2. process options in header,
3. check destination address: if the packet is for this host

    3.1 buffer fragments and apply reassembly process, then eventually
    3.2 provide payload to a higher layer (per protocol field)

    otherwise, assuming we want to forward the packet

    3.3 check TTL, and drop if exceeded,
    3.4 look-up next hop in forwarding table,
    3.5 apply fragmentation and header update processes, (e.g., decrement TTL, recompute checksum),
    3.6 transmit packet(s) via lower, link layer

and if/when errors occur, signal them appropriately.

- Question: how does the forwarding table get populated with entries?
- Answer: **routing**, which is a *big* topic so we'll focus on uni-cast

  1. **distance vector routing**, and
  2. **link state routing**

  and hence ignore various alternatives.

Concepts (1)

- Recall:
  - routing is the act of deciding a path used when forward packets from a given source to a given destination,
  - forwarding is a *local* process, routing is *global* in the sense it involves the whole (inter-)network,
  - goal is to make best use of connectivity and thus bandwidth: it can be viewed as a form of resource allocation.
- Solution(s):
  1. static (or fixed) routing, i.e., hard-code routing information by hand,
  2. source routing, i.e., let the source pre-determine routing decisions, or
  3. adaptive routing, e.g., i.e., use a distributed **routing algorithm** (or **routing protocol**).

Concepts (2)

- Good news: we can reason about routing by noting

$$\text{network} \equiv \text{graph} \implies \text{graph theory} \subset \text{data structures and algorithms},$$

and that

  - a network graph will be weighted to capture the properties of each connection,
  - we could use directed graphs (e.g., to capture uni-directional connection properties) ...
  - ... but for simplicity we'll consider undirected graphs only

st. our network is modelled by

$$G = (V, E = \langle (u_0, v_0, d_0), (u_1, v_1, d_1), \ldots, (u_{m-1}, v_{m-1}, d_{m-1}) \rangle)$$

where $|V| = n$ and $|E| = m$.

Concepts (3)

- Bad news: any algorithm (ideally) needs to be

correct	$\Rightarrow$	find paths that provide end-to-end connectivity
efficient	$\Rightarrow$	make good use of resources
fair	$\Rightarrow$	will not "stave" nodes of bandwidth
convergent	$\Rightarrow$	initialises/recovers quickly, e.g., after change to topology
scalable	$\Rightarrow$	remains efficient even with large n and/or m

and *must* be **decentralised**: the model is that

1. no (global) controller node exists,
2. nodes typically start with only local knowledge of topology,
3. nodes communicate (concurrently) with neighbours only, and
4. nodes *and* links can fail!

Concepts (4)

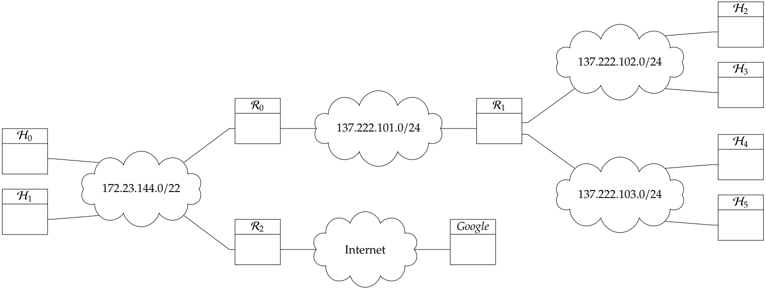
- Recall: we have *already* assumed
  - routers make (global) routing decisions, whereas
  - hosts communicate locally, or forward to nearest router

  so we can route wrt. **routing units** (or regions), *not* per-node destinations.
- The strategy is to address scalablity using hierarchy, so
  1. route *to* region, then
  2. route *in* region

  e.g., by leveraging IP prefixes to coalesce multiple destinations into one region (or block) ...

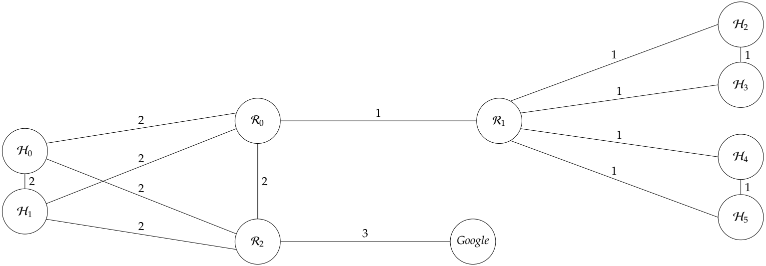
Concepts (5)

- ... so we *significantly* simplify the problem to:



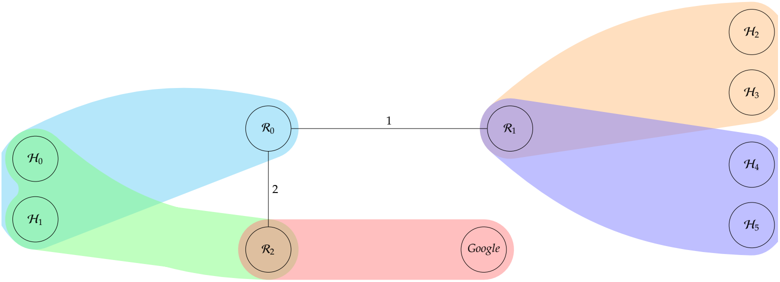
Concepts (5)

- ... so we *significantly* simplify the problem to:



Concepts (5)

- ... so we *significantly* simplify the problem to:



Routing Algorithms (1)

- Idea: in general, we'll have each routing algorithm
  1. maintain some state, i.e., a **routing table**,
  2. periodically transmit, receive and integrate information via (local) communication with neighbouring nodes,
  3. periodically translate the routing table into forwarding table, e.g., via some form of (local) computation

  keeping in mind that

  - periodically means at regular intervals, *plus* when a change is topology occurs, and
  - a cost of ∞ means a link does not exist *or* has failed: either way, avoid it!

- Idea:

$$\textbf{distance vector routing} \simeq \text{distributed Bellman-Ford},$$

  in the sense each node u

  1. maintains a **distance vector**

  $$\langle (v_0, d_0), (v_1, d_1), \ldots, (v_{l-1}, d_{l-1}) \rangle$$

  of next hop and cost tuples, initialised st.

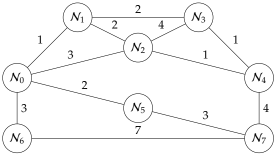
  $$d_i = \begin{cases} 0 & \text{if } v_i = u \\ \infty & \text{otherwise} \end{cases},$$

  2. periodically transmits the distance vector to all neighbours,
  3. periodically updates the distance vector with new information, st.

  $$dist(u, v) = \min_{\forall w,\ (u,w,d) \in E} \left[dist(w, v) + d \right].$$

- Example: **Routing Information Protocol (RIP)** [5].
  - uses hop count as a cost function, assuming $\infty \equiv 16$,
  - transmits distance vectors every $\sim$ 30s with $\sim$ 180s failure time-out.

- Recalling that the graph in question is,



the algorithm proceeds as follows:

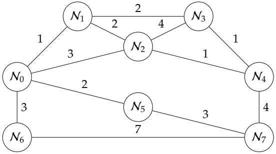
		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	∞	∞	∞	∞	∞	∞	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_1$	*dist* =	∞	0	∞	∞	∞	∞	∞	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_2$	*dist* =	∞	∞	0	∞	∞	∞	∞	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_3$	*dist* =	∞	∞	∞	0	∞	∞	∞	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_4$	*dist* =	∞	∞	∞	∞	0	∞	∞	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_5$	*dist* =	∞	∞	∞	∞	∞	0	∞	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_6$	*dist* =	∞	∞	∞	∞	∞	∞	0	∞
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_7$	*dist* =	∞	∞	∞	∞	∞	∞	∞	0
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$

$\rightsquigarrow$

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	∞	∞	2	3	∞
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\perp$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$
$\mathcal{N}_1$	*dist* =	1	0	2	2	∞	∞	∞	∞
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_2$	*dist* =	3	2	0	4	1	∞	∞	∞
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_3$	*dist* =	∞	2	4	0	1	∞	∞	∞
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_4$	*dist* =	∞	∞	1	1	0	∞	∞	4
	hop =	$\perp$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	∞	∞	∞	∞	0	∞	3
	hop =	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	∞	∞	∞	∞	∞	0	7
	hop =	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	∞	∞	∞	∞	4	3	7	0
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

Routing Algorithms (3) – Distance Vector Routing

- Recalling that the graph in question is,



the algorithm proceeds as follows:

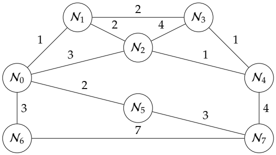
		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	∞	∞	2	3	∞
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\perp$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$
$\mathcal{N}_1$	*dist* =	1	0	2	2	∞	∞	∞	∞
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_2$	*dist* =	3	2	0	4	1	∞	∞	∞
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_3$	*dist* =	∞	2	4	0	1	∞	∞	∞
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\perp$
$\mathcal{N}_4$	*dist* =	∞	∞	1	1	0	∞	∞	4
	hop =	$\perp$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	∞	∞	∞	∞	0	∞	3
	hop =	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	∞	∞	∞	∞	∞	0	7
	hop =	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	∞	∞	∞	∞	4	3	7	0
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

$\leadsto$

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	∞
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	5	6	5
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	∞	∞	5
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	7	11	4
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_7$	$\mathcal{N}_7$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	3	5	∞	7	0	5	3
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	4	6	∞	11	5	0	7
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	5	∞	5	5	4	3	7	0
	hop =	$\mathcal{N}_5$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

Routing Algorithms (3) – Distance Vector Routing

- Recalling that the graph in question is,



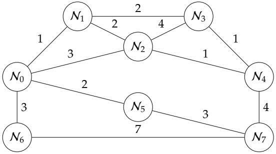
the algorithm proceeds as follows:

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	∞
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	5	6	5
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	∞	∞	5
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	7	11	4
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_7$	$\mathcal{N}_7$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	3	5	∞	7	0	5	3
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	4	6	∞	11	5	0	7
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	5	∞	5	5	4	3	7	0
	hop =	$\mathcal{N}_5$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

$\leadsto$

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	6
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	5	6	5
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	5	6	5
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	6	7	4
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	3	5	5	6	0	5	3
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	4	6	6	7	5	0	7
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	5	6	5	5	4	3	7	0
	hop =	$\mathcal{N}_5$	$\mathcal{N}_5$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

- Recalling that the graph in question is,



the algorithm proceeds as follows:

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	6
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	5	6	5
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	5	6	5
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	6	7	4
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	3	5	5	6	0	5	3
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	4	6	6	7	5	0	7
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	5	6	5	5	4	3	7	0
	hop =	$\mathcal{N}_5$	$\mathcal{N}_5$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

$\rightsquigarrow$

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	6
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	5	6	5
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	5	6	5
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	6	7	4
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_7$
$\mathcal{N}_5$	*dist* =	2	3	5	5	6	0	5	3
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$
$\mathcal{N}_6$	*dist* =	3	4	6	6	7	5	0	7
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$
$\mathcal{N}_7$	*dist* =	5	6	5	5	4	3	7	0
	hop =	$\mathcal{N}_5$	$\mathcal{N}_5$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$

- Idea:

$$\textbf{link state routing} \simeq \text{flooding} + \text{Dijkstra},$$

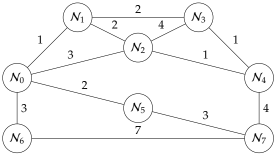
  in the sense each node u

  1. floods network with **link state packets** (i.e., neighbouring nodes plus link costs) yielding global topology, then
  2. use Dijkstra to compute shortest paths.

- Examples:
  - **Open Shortest Path First (OSPF)** [7], and
  - **Intermediate System to Intermediate System (IS-IS)** [8].

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

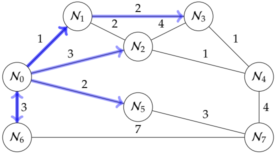


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

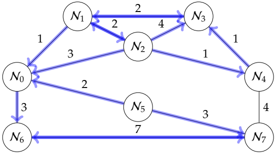
- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

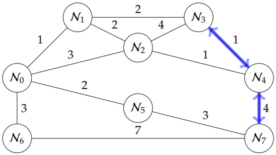
- Recalling that the graph in question is,



the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra.

- Recalling that the graph in question is,

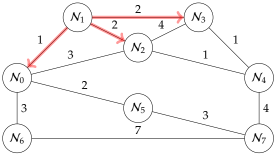


the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

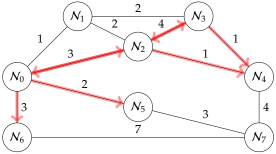


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

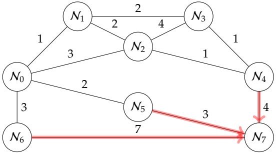
- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

- Recalling that the graph in question is,

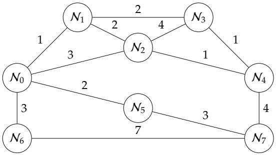


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

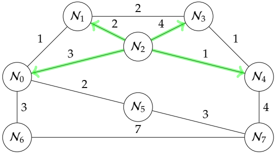


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

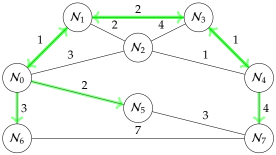


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

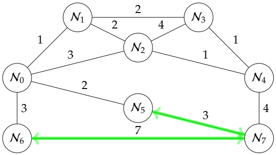


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

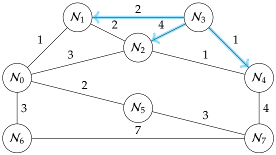
- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

- Recalling that the graph in question is,

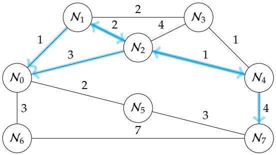


the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

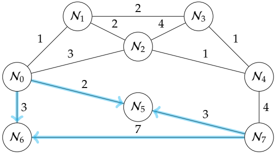
- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

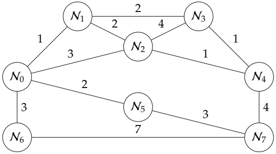
- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

- Recalling that the graph in question is,

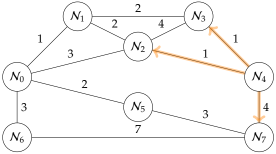


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

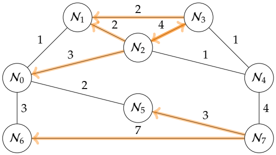


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

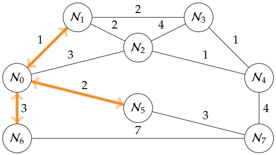


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

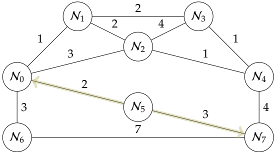


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

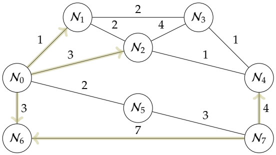


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

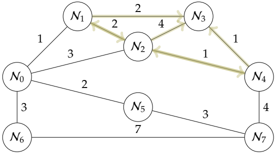


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

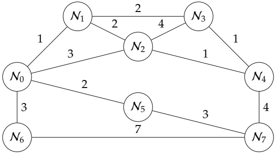


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

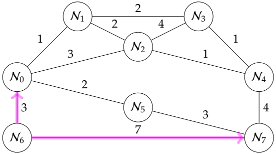


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

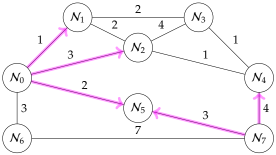


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

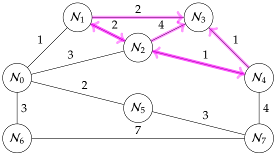


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

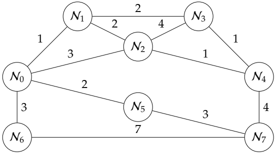


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

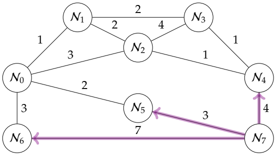


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

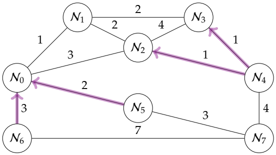
- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

- Recalling that the graph in question is,

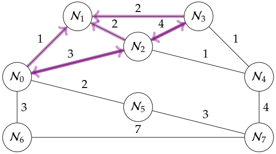


the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

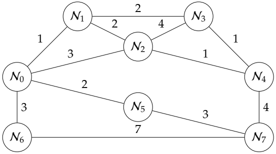


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



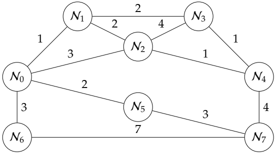
  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

University of BRISTOL

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,

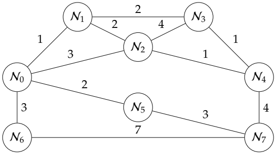


  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra.

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



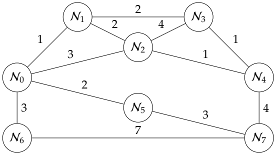
the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		N_0	N_1	N_2	N_3	N_4	N_5	N_6	N_7	
N_0	$dist =$	0	∞	∞	∞	∞	∞	∞	∞	$queue = \langle (N_0, 0), (N_1, \infty), (N_2, \infty), (N_3, \infty), (N_4, \infty), (N_5, \infty), (N_6, \infty), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_1	$dist =$	∞	0	∞	∞	∞	∞	∞	∞	$queue = \langle (N_0, \infty), (N_1, 0), (N_2, \infty), (N_3, \infty), (N_4, \infty), (N_5, \infty), (N_6, \infty), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_2	$dist =$	∞	∞	0	∞	∞	∞	∞	∞	$queue = \langle (N_0, \infty), (N_1, \infty), (N_2, 0), (N_3, \infty), (N_4, \infty), (N_5, \infty), (N_6, \infty), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_3	$dist =$	∞	∞	∞	0	∞	∞	∞	∞	$queue = \langle (N_0, \infty), (N_1, \infty), (N_2, \infty), (N_3, 0), (N_4, \infty), (N_5, \infty), (N_6, \infty), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_4	$dist =$	∞	∞	∞	∞	0	∞	∞	∞	$queue = \langle (N_0, \infty), (N_1, \infty), (N_2, \infty), (N_3, \infty), (N_4, 0), (N_5, \infty), (N_6, \infty), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_5	$dist =$	∞	∞	∞	∞	∞	0	∞	∞	$queue = \langle (N_0, \infty), (N_1, \infty), (N_2, \infty), (N_3, \infty), (N_4, \infty), (N_5, 0), (N_6, \infty), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_6	$dist =$	∞	∞	∞	∞	∞	∞	0	∞	$queue = \langle (N_0, \infty), (N_1, \infty), (N_2, \infty), (N_3, \infty), (N_4, \infty), (N_5, \infty), (N_6, 0), (N_7, \infty) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	
N_7	$dist =$	∞	∞	∞	∞	∞	∞	∞	0	$queue = \langle (N_0, \infty), (N_1, \infty), (N_2, \infty), (N_3, \infty), (N_4, \infty), (N_5, \infty), (N_6, \infty), (N_7, 0) \rangle$
	$hop =$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



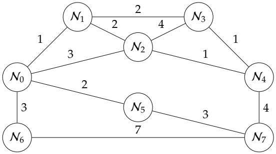
the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	*dist* =	0	1	3	∞	∞	2	3	∞	$queue = \langle (\mathcal{N}_1, 1), (\mathcal{N}_2, 3), (\mathcal{N}_3, \infty), (\mathcal{N}_4, \infty), (\mathcal{N}_5, 2), (\mathcal{N}_6, 3), (\mathcal{N}_7, \infty) \rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\perp$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	
$\mathcal{N}_1$	*dist* =	1	0	2	2	∞	∞	∞	∞	$queue = \langle (\mathcal{N}_0, 1), (\mathcal{N}_2, 2), (\mathcal{N}_3, 2), (\mathcal{N}_4, \infty), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, \infty) \rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\perp$	
$\mathcal{N}_2$	*dist* =	3	2	0	4	1	∞	∞	∞	$queue = \langle (\mathcal{N}_0, 3), (\mathcal{N}_1, 2), (\mathcal{N}_3, 4), (\mathcal{N}_4, 1), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, \infty) \rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\perp$	$\perp$	$\perp$	
$\mathcal{N}_3$	*dist* =	∞	2	4	0	1	∞	∞	∞	$queue = \langle (\mathcal{N}_0, \infty), (\mathcal{N}_1, 2), (\mathcal{N}_2, 4), (\mathcal{N}_4, 1), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, \infty) \rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\perp$	
$\mathcal{N}_4$	*dist* =	∞	∞	1	1	0	∞	∞	4	$queue = \langle (\mathcal{N}_0, \infty), (\mathcal{N}_1, \infty), (\mathcal{N}_2, 1), (\mathcal{N}_3, 1), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, 4) \rangle$
	hop =	$\perp$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_5$	*dist* =	2	∞	∞	∞	∞	0	∞	3	$queue = \langle (\mathcal{N}_0, 2), (\mathcal{N}_1, \infty), (\mathcal{N}_2, \infty), (\mathcal{N}_3, \infty), (\mathcal{N}_4, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, 3) \rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_6$	*dist* =	3	∞	∞	∞	∞	∞	0	7	$queue = \langle (\mathcal{N}_0, 3), (\mathcal{N}_1, \infty), (\mathcal{N}_2, \infty), (\mathcal{N}_3, \infty), (\mathcal{N}_4, \infty), (\mathcal{N}_5, \infty), (\mathcal{N}_7, 7) \rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	*dist* =	∞	∞	∞	∞	4	3	7	0	$queue = \langle (\mathcal{N}_0, \infty), (\mathcal{N}_1, \infty), (\mathcal{N}_2, \infty), (\mathcal{N}_3, \infty), (\mathcal{N}_4, 4), (\mathcal{N}_5, 3), (\mathcal{N}_6, 7) \rangle$
	hop =	$\perp$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



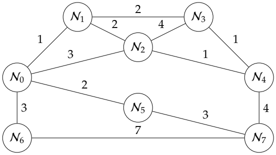
  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	$dist =$	0	1	3	3	∞	2	3	∞	$queue = \langle (\mathcal{N}_2, 3), (\mathcal{N}_3, 3), (\mathcal{N}_4, \infty), (\mathcal{N}_5, 2), (\mathcal{N}_6, 3), (\mathcal{N}_7, \infty) \rangle$
	$hop =$	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	
$\mathcal{N}_1$	$dist =$	1	0	2	2	∞	3	4	∞	$queue = \langle (\mathcal{N}_2, 2), (\mathcal{N}_3, 2), (\mathcal{N}_4, \infty), (\mathcal{N}_5, 3), (\mathcal{N}_6, 4), (\mathcal{N}_7, \infty) \rangle$
	$hop =$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	
$\mathcal{N}_2$	$dist =$	3	2	0	2	1	∞	∞	5	$queue = \langle (\mathcal{N}_0, 3), (\mathcal{N}_1, 2), (\mathcal{N}_3, 2), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, 5) \rangle$
	$hop =$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$	
$\mathcal{N}_3$	$dist =$	∞	2	2	0	1	∞	∞	5	$queue = \langle (\mathcal{N}_0, \infty), (\mathcal{N}_1, 2), (\mathcal{N}_2, 2), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, 5) \rangle$
	$hop =$	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$	
$\mathcal{N}_4$	$dist =$	4	3	1	1	0	∞	∞	4	$queue = \langle (\mathcal{N}_0, 4), (\mathcal{N}_1, 3), (\mathcal{N}_3, 1), (\mathcal{N}_5, \infty), (\mathcal{N}_6, \infty), (\mathcal{N}_7, 4) \rangle$
	$hop =$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_5$	$dist =$	2	3	5	∞	∞	0	5	3	$queue = \langle (\mathcal{N}_1, 3), (\mathcal{N}_2, 5), (\mathcal{N}_3, \infty), (\mathcal{N}_4, \infty), (\mathcal{N}_6, 5), (\mathcal{N}_7, 3) \rangle$
	$hop =$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	$dist =$	3	4	6	∞	∞	5	0	7	$queue = \langle (\mathcal{N}_1, 4), (\mathcal{N}_2, 6), (\mathcal{N}_3, \infty), (\mathcal{N}_4, \infty), (\mathcal{N}_5, 5), (\mathcal{N}_7, 7) \rangle$
	$hop =$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	$\perp$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	$dist =$	5	∞	∞	∞	4	3	7	0	$queue = \langle (\mathcal{N}_0, 5), (\mathcal{N}_1, \infty), (\mathcal{N}_2, \infty), (\mathcal{N}_3, \infty), (\mathcal{N}_4, 4), (\mathcal{N}_6, 7) \rangle$
	$hop =$	$\mathcal{N}_5$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



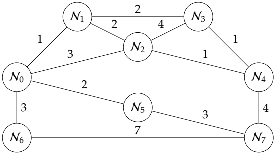
the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	*dist* =	0	1	3	3	∞	2	3	5	$queue = \langle(\mathcal{N}_2,3),(\mathcal{N}_3,3),(\mathcal{N}_4,\infty),(\mathcal{N}_6,3),(\mathcal{N}_7,5)\rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$	
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	∞	$queue = \langle(\mathcal{N}_3,2),(\mathcal{N}_4,3),(\mathcal{N}_5,3),(\mathcal{N}_6,4),(\mathcal{N}_7,\infty)\rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	∞	∞	5	$queue = \langle(\mathcal{N}_0,3),(\mathcal{N}_3,2),(\mathcal{N}_5,\infty),(\mathcal{N}_6,\infty),(\mathcal{N}_7,5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$	
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	∞	∞	5	$queue = \langle(\mathcal{N}_0,3),(\mathcal{N}_2,2),(\mathcal{N}_5,\infty),(\mathcal{N}_6,\infty),(\mathcal{N}_7,5)\rangle$
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$	
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	∞	∞	4	$queue = \langle(\mathcal{N}_0,4),(\mathcal{N}_1,3),(\mathcal{N}_5,\infty),(\mathcal{N}_6,\infty),(\mathcal{N}_7,4)\rangle$
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_5$	*dist* =	2	3	5	5	∞	0	5	3	$queue = \langle(\mathcal{N}_2,5),(\mathcal{N}_3,5),(\mathcal{N}_4,\infty),(\mathcal{N}_6,5),(\mathcal{N}_7,3)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	*dist* =	3	4	6	6	∞	5	0	7	$queue = \langle(\mathcal{N}_2,6),(\mathcal{N}_3,6),(\mathcal{N}_4,\infty),(\mathcal{N}_5,5),(\mathcal{N}_7,7)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	*dist* =	5	∞	5	5	4	3	7	0	$queue = \langle(\mathcal{N}_0,5),(\mathcal{N}_1,\infty),(\mathcal{N}_2,5),(\mathcal{N}_3,5),(\mathcal{N}_6,7)\rangle$
	hop =	$\mathcal{N}_5$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



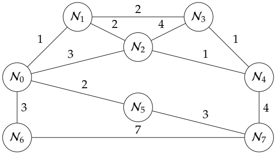
the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5	$queue = \langle(\mathcal{N}_3,3),(\mathcal{N}_4,4),(\mathcal{N}_6,3),(\mathcal{N}_7,5)\rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$	
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	∞	$queue = \langle(\mathcal{N}_4,3),(\mathcal{N}_5,3),(\mathcal{N}_6,4),(\mathcal{N}_7,\infty)\rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\perp$	
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	∞	∞	5	$queue = \langle(\mathcal{N}_0,3),(\mathcal{N}_5,\infty),(\mathcal{N}_6,\infty),(\mathcal{N}_7,5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$	
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	∞	∞	5	$queue = \langle(\mathcal{N}_0,3),(\mathcal{N}_5,\infty),(\mathcal{N}_6,\infty),(\mathcal{N}_7,5)\rangle$
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\perp$	$\perp$	$\mathcal{N}_4$	
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	∞	∞	4	$queue = \langle(\mathcal{N}_0,4),(\mathcal{N}_5,\infty),(\mathcal{N}_6,\infty),(\mathcal{N}_7,4)\rangle$
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\perp$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_5$	*dist* =	2	3	5	5	7	0	5	3	$queue = \langle(\mathcal{N}_2,5),(\mathcal{N}_3,5),(\mathcal{N}_4,7),(\mathcal{N}_6,5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_7$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	*dist* =	3	4	6	6	∞	5	0	7	$queue = \langle(\mathcal{N}_2,6),(\mathcal{N}_3,6),(\mathcal{N}_4,\infty),(\mathcal{N}_7,7)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	*dist* =	5	6	5	5	4	3	7	0	$queue = \langle(\mathcal{N}_1,6),(\mathcal{N}_2,5),(\mathcal{N}_3,5),(\mathcal{N}_6,7)\rangle$
	hop =	$\mathcal{N}_5$	$\mathcal{N}_0$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



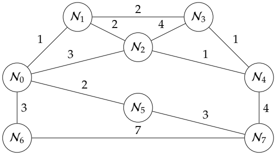
  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	$dist$ =	0	1	3	3	4	2	3	5	$queue = \langle(\mathcal{N}_4,4),(\mathcal{N}_6,3),(\mathcal{N}_7,5)\rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$	
$\mathcal{N}_1$	$dist$ =	1	0	2	2	3	3	4	7	$queue = \langle(\mathcal{N}_5,3),(\mathcal{N}_6,4),(\mathcal{N}_7,7)\rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_2$	$dist$ =	3	2	0	2	1	5	6	5	$queue = \langle(\mathcal{N}_5,5),(\mathcal{N}_6,6),(\mathcal{N}_7,5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_3$	$dist$ =	3	2	2	0	1	5	6	5	$queue = \langle(\mathcal{N}_5,5),(\mathcal{N}_6,6),(\mathcal{N}_7,5)\rangle$
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_4$	$dist$ =	4	3	1	1	0	6	7	4	$queue = \langle(\mathcal{N}_5,6),(\mathcal{N}_6,7),(\mathcal{N}_7,4)\rangle$
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_5$	$dist$ =	2	3	5	5	6	0	5	3	$queue = \langle(\mathcal{N}_3,5),(\mathcal{N}_4,6),(\mathcal{N}_6,5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	$dist$ =	3	4	6	6	7	5	0	7	$queue = \langle(\mathcal{N}_3,6),(\mathcal{N}_4,7),(\mathcal{N}_7,7)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	$dist$ =	5	6	5	5	4	3	7	0	$queue = \langle(\mathcal{N}_1,6),(\mathcal{N}_3,5),(\mathcal{N}_6,7)\rangle$
	hop =	$\mathcal{N}_5$	$\mathcal{N}_0$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



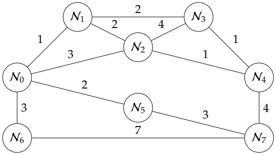
the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	*dist* =	0	1	3	3	4	2	3	5	*queue* = $\langle(\mathcal{N}_4, 4), (\mathcal{N}_7, 5)\rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$	
$\mathcal{N}_1$	*dist* =	1	0	2	2	3	3	4	6	*queue* = $\langle(\mathcal{N}_6, 4), (\mathcal{N}_7, 6)\rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_5$	
$\mathcal{N}_2$	*dist* =	3	2	0	2	1	5	6	5	*queue* = $\langle(\mathcal{N}_6, 6), (\mathcal{N}_7, 5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_3$	*dist* =	3	2	2	0	1	5	6	5	*queue* = $\langle(\mathcal{N}_6, 6), (\mathcal{N}_7, 5)\rangle$
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_4$	*dist* =	4	3	1	1	0	6	7	4	*queue* = $\langle(\mathcal{N}_5, 6), (\mathcal{N}_6, 7)\rangle$
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_5$	*dist* =	2	3	5	5	6	0	5	3	*queue* = $\langle(\mathcal{N}_4, 6), (\mathcal{N}_6, 5)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	*dist* =	3	4	6	6	7	5	0	7	*queue* = $\langle(\mathcal{N}_4, 7), (\mathcal{N}_7, 7)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	*dist* =	5	6	5	5	4	3	7	0	*queue* = $\langle(\mathcal{N}_1, 6), (\mathcal{N}_6, 7)\rangle$
	hop =	$\mathcal{N}_5$	$\mathcal{N}_0$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



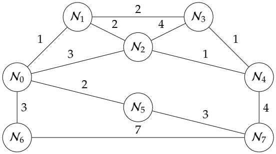
the algorithm

1. uses flooding to communicate the topology to all nodes,
2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	$dist$ =	0	1	3	3	4	2	3	5	$queue = \langle(\mathcal{N}_7, 5)\rangle$
	hop =	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$	
$\mathcal{N}_1$	$dist$ =	1	0	2	2	3	3	4	6	$queue = \langle(\mathcal{N}_7, 6)\rangle$
	hop =	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_5$	
$\mathcal{N}_2$	$dist$ =	3	2	0	2	1	5	6	5	$queue = \langle(\mathcal{N}_6, 6)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_3$	$dist$ =	3	2	2	0	1	5	6	5	$queue = \langle(\mathcal{N}_6, 6)\rangle$
	hop =	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_4$	$dist$ =	4	3	1	1	0	6	7	4	$queue = \langle(\mathcal{N}_6, 7)\rangle$
	hop =	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_5$	$dist$ =	2	3	5	5	6	0	5	3	$queue = \langle(\mathcal{N}_4, 6)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	$dist$ =	3	4	6	6	7	5	0	7	$queue = \langle(\mathcal{N}_7, 7)\rangle$
	hop =	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	$dist$ =	5	6	5	5	4	3	7	0	$queue = \langle(\mathcal{N}_6, 7)\rangle$
	hop =	$\mathcal{N}_5$	$\mathcal{N}_0$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Routing Algorithms (2) – Link State Routing

- Recalling that the graph in question is,



  the algorithm

  1. uses flooding to communicate the topology to all nodes,
  2. has each node (in parallel) compute shortest paths using Dijkstra, i.e.,

		$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_7$	
$\mathcal{N}_0$	$dist =$	0	1	3	3	4	2	3	5	$queue = \emptyset$
	$hop =$	$\perp$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\mathcal{N}_5$	
$\mathcal{N}_1$	$dist =$	1	0	2	2	3	3	4	6	$queue = \emptyset$
	$hop =$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_5$	
$\mathcal{N}_2$	$dist =$	3	2	0	2	1	5	6	5	$queue = \emptyset$
	$hop =$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_3$	$dist =$	3	2	2	0	1	5	6	5	$queue = \emptyset$
	$hop =$	$\mathcal{N}_1$	$\mathcal{N}_1$	$\mathcal{N}_4$	$\perp$	$\mathcal{N}_4$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_4$	
$\mathcal{N}_4$	$dist =$	4	3	1	1	0	6	7	4	$queue = \emptyset$
	$hop =$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_2$	$\mathcal{N}_3$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_5$	$dist =$	2	3	5	5	6	0	5	3	$queue = \emptyset$
	$hop =$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\perp$	$\mathcal{N}_0$	$\mathcal{N}_7$	
$\mathcal{N}_6$	$dist =$	3	4	6	6	7	5	0	7	$queue = \emptyset$
	$hop =$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_0$	$\mathcal{N}_1$	$\mathcal{N}_2$	$\mathcal{N}_0$	$\perp$	$\mathcal{N}_7$	
$\mathcal{N}_7$	$dist =$	5	6	5	5	4	3	7	0	$queue = \emptyset$
	$hop =$	$\mathcal{N}_5$	$\mathcal{N}_0$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_4$	$\mathcal{N}_5$	$\mathcal{N}_6$	$\perp$	

Conclusions

- To summarise the two approaches covered, we can say

Metric	Distance Vector	Link State
correct	distributed Bellman-Ford	replicated Dijkstra
efficient	approximately	approximately
fair	approximately	approximately
convergent	slow: many exchanges	fast: flood then recompute
scalable	excellent	reasonable

i.e., selection is basically a trade-off between convergence and scalablity ...

- ... *plus* we need to cater for various corner cases:
  - distance vector routing can fail if
    - if network is partitioned (cf. graph cut, meaning "count to infinity" problem),

  while
  - link state routing can fail if
    - flooding sequence numbers can overflow or be corrupted,
    - nodes can fail and reset flooding sequence number,
    - if network is partitioned and then re-joined.

Conclusions

- Take away points:
  - Using graph theory to study routing is advantageous in terms of designing and reasoning about solutions ...
  - ... even so, translating theory into practice is *still* a significant challenge wrt. function and efficiency.
  - Routing protocols are instances of more general **consensus protocols** whereby (distributed) parties need to agree on a shared state.
- Additional topics: a (non-exhaustive) list could include at least
  - the **Border Gateway Protocol (BGP)** [6], which is important for large(r) scale inter-networking, and
  - techniques such as **multi-path routing** [10] which allows packets to make use of *multiple* routes to a given destination.

References

[1] Wikipedia: Distance vector routing protocol.
http://en.wikipedia.org/wiki/Distance-vector_routing_protocol.

[2] Wikipedia: Link-state routing protocol.
http://en.wikipedia.org/wiki/Link-state_routing_protocol.

[3] Wikipedia: Routing.
http://en.wikipedia.org/wiki/Routing.

[4] Wikipedia: Routing protocol.
http://en.wikipedia.org/wiki/Routing_protocol.

[5] C. Hedrick.
Routing Information Protocol.
Internet Engineering Task Force (IETF) Request for Comments (RFC) 1058, 1988.
http://tools.ietf.org/html/rfc1058.

[6] K. Loughheed and Y. Rekhter.
A Border Gateway Protocol (BGP).
Internet Engineering Task Force (IETF) Request for Comments (RFC) 1105, 1989.
http://tools.ietf.org/html/rfc1105.

[7] J. Moy.
OSPF specification.
Internet Engineering Task Force (IETF) Request for Comments (RFC) 1131, 1989.
http://tools.ietf.org/html/rfc1131.

References

[8] D. Oran.
OSI IS-IS Intra-domain Routing Protocol.
Internet Engineering Task Force (IETF) Request for Comments (RFC) 1142, 1990.
`http://tools.ietf.org/html/rfc1142`.

[9] W. Stallings.
Chapter 13: Routing in switched data networks.
In *Data and Computer Communications*. Pearson, 9th edition, 2010.

[10] D. Thaler and C. Hopps.
Multipath issues in unicast and multicast next-hop selection.
Internet Engineering Task Force (IETF) Request for Comments (RFC) 2991, 2000.
`http://tools.ietf.org/html/rfc2991`.