Data and Computer Communications

Chapter 12 – Routing in Switched Networks

Eighth Edition

by William Stallings

Lecture slides by Lawrie Brown

Routing in Switched Networks

"I tell you," went on Syme with passion, "that every time a train comes in I feel that it has broken past batteries of besiegers, and that man has won a battle against chaos. You say contemptuously that when one has left Sloane Square one must come to Victoria. I say that one might do a thousand things instead, and that whenever I really come there I have the sense of hairbreadth escape. And when I hear the guard shout out the word 'Victoria', it is not an unmeaning word. It is to me the cry of a herald announcing conquest. It is to me indeed 'Victoria'; it is the victory of Adam."

—*The Man Who Was Thursday*, G.K. Chesterton

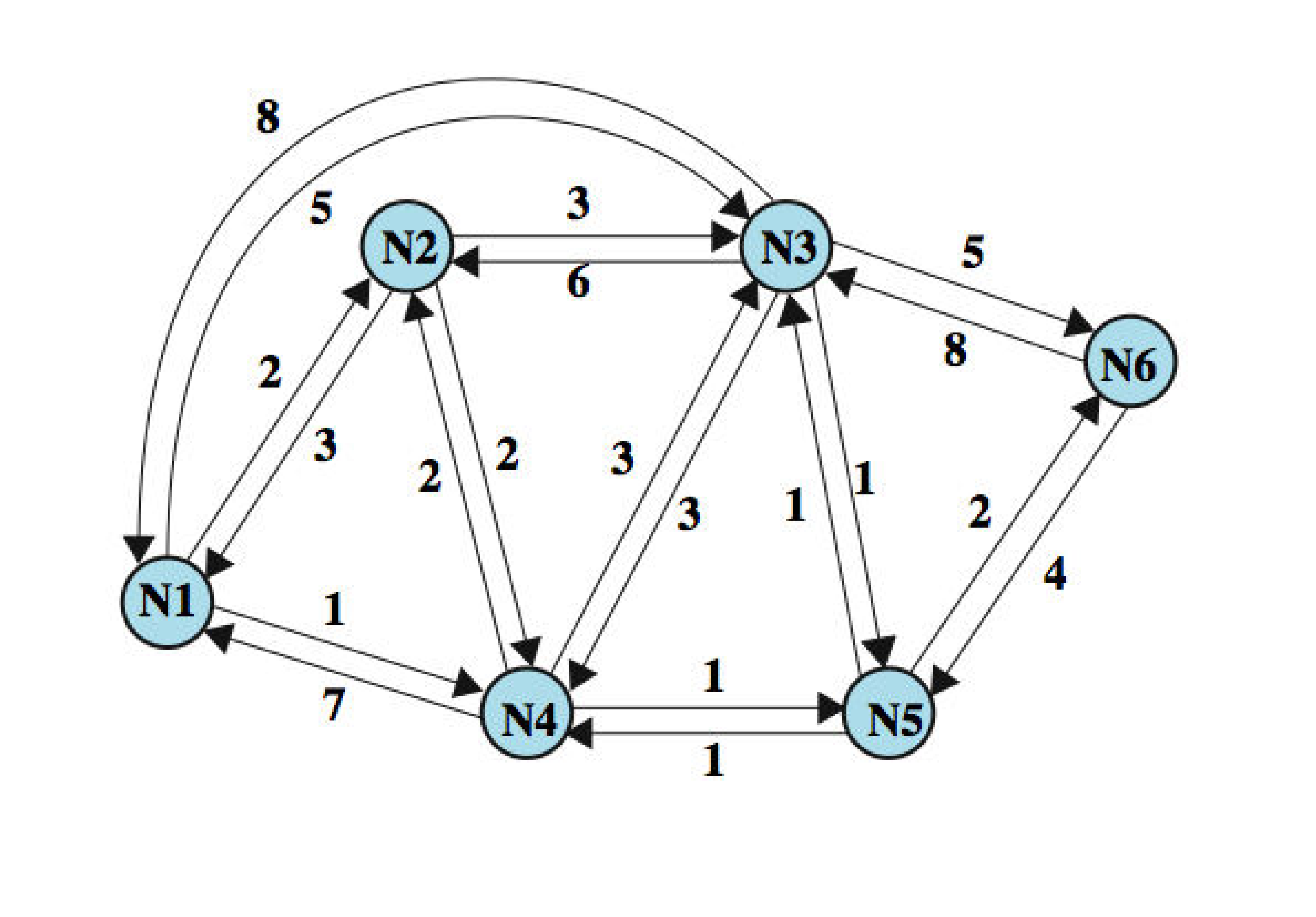
Routing in Packet Switched Network

- key design issue for (packet) switched networks
- select route across network between end nodes
- characteristics required:
  - correctness
  - simplicity
  - robustness
  - stability
  - fairness
  - optimality
  - efficiency

Performance Criteria

- used for selection of route
- simplest is “minimum hop”
- can be generalized as “least cost”

Example Packet Switched Network



Decision Time and Place

- time
  - packet or virtual circuit basis
  - fixed or dynamically changing
- place
  - distributed - made by each node
  - centralized
  - source

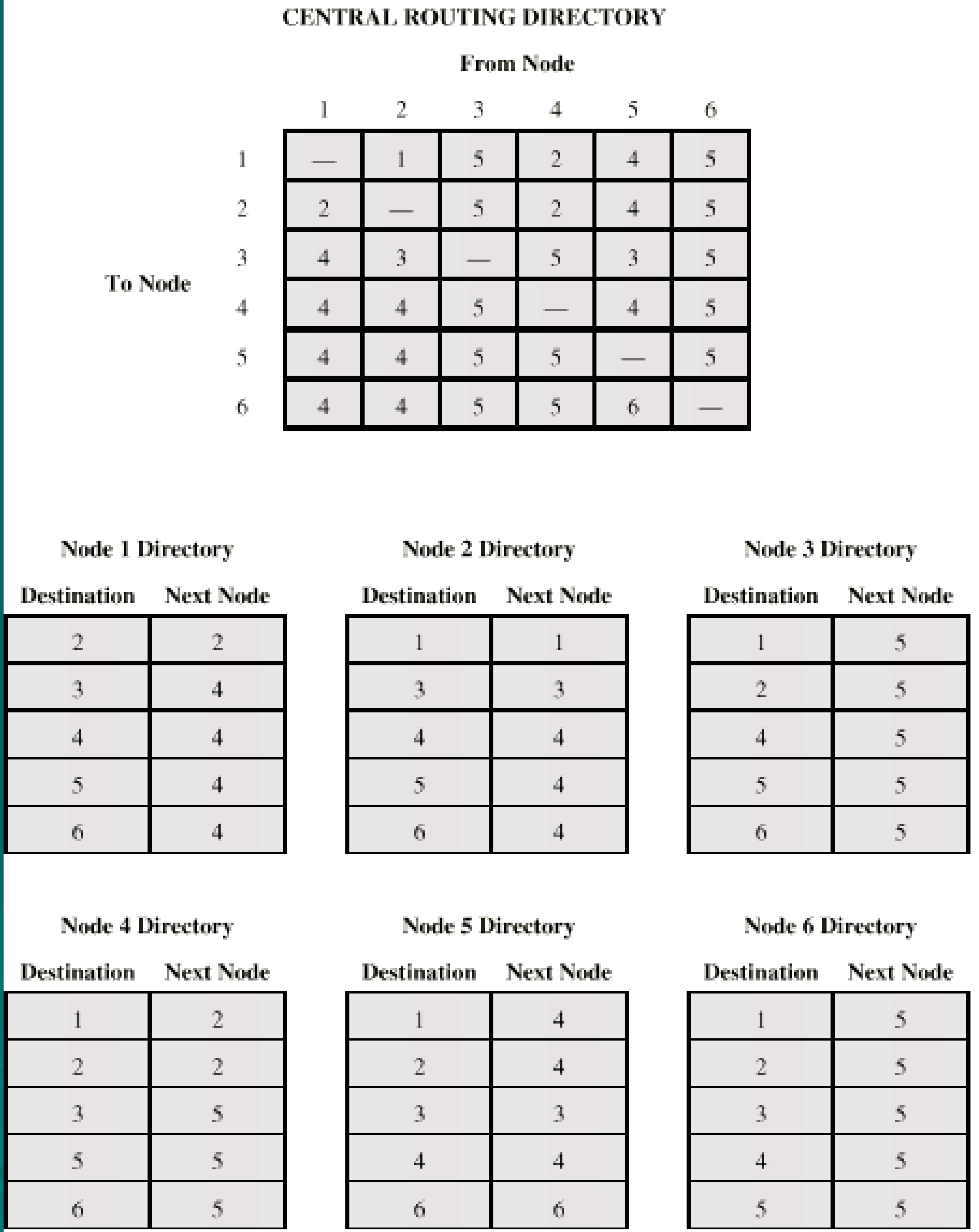
Network Information Source and Update Timing

- routing decisions usually based on knowledge of network (not always)
  - distributed routing
    - using local knowledge, info from adjacent nodes, info from all nodes on a potential route
  - central routing
    - collect info from all nodes
- issue of update timing
  - when is network info held by nodes updated
  - fixed - never updated
  - adaptive - regular updates

Routing Strategies - Fixed Routing

- use a single permanent route for each source to destination pair
- determined using a least cost algorithm
- route is fixed
  - at least until a change in network topology
  - hence cannot respond to traffic changes
- advantage is simplicity
- disadvantage is lack of flexibility

Fixed Routing Tables



CENTRAL ROUTING DIRECTORY

		From Node					
		1	2	3	4	5	6
To Node	1	—	1	5	2	4	5
	2	2	—	5	2	4	5
	3	4	3	—	5	3	5
	4	4	4	5	—	4	5
	5	4	4	5	5	—	5
	6	4	4	5	5	6	—

Node 1 Directory

Destination	Next Node
2	2
3	4
4	4
5	4
6	4

Node 2 Directory

Destination	Next Node
1	1
3	3
4	4
5	4
6	4

Node 3 Directory

Destination	Next Node
1	5
2	5
4	5
5	5
6	5

Node 4 Directory

Destination	Next Node
1	2
2	2
3	5
5	5
6	5

Node 5 Directory

Destination	Next Node
1	4
2	4
3	3
4	4
6	6

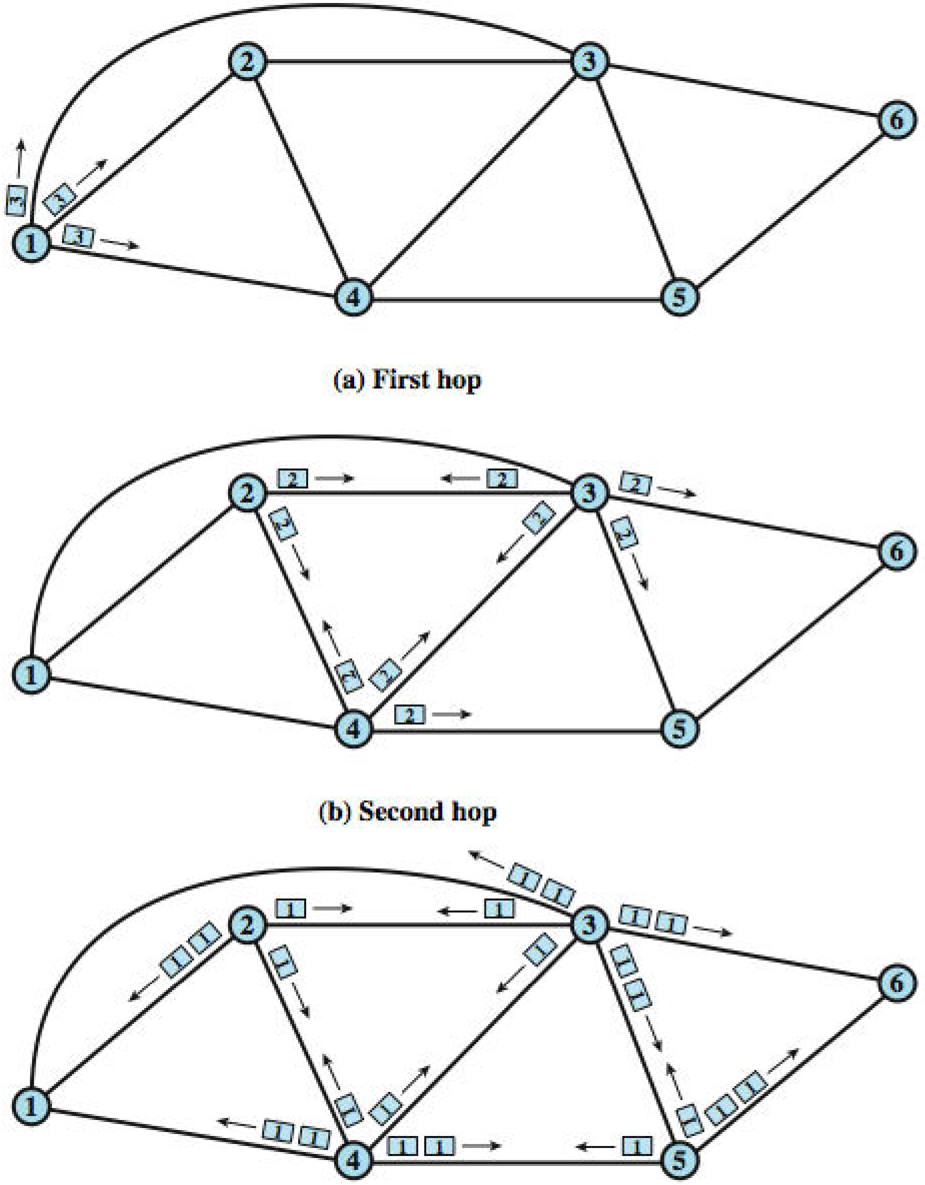
Node 6 Directory

Destination	Next Node
1	5
2	5
3	5
4	5
5	5

Routing Strategies - Flooding

- packet sent by node to every neighbor
- eventually multiple copies arrive at destination
- no network info required
- each packet is uniquely numbered so duplicates can be discarded
- need some way to limit incessant retransmission
  - nodes can remember packets already forwarded to keep network load in bounds
  - or include a hop count in packets

Flooding Example



(a) First hop

(b) Second hop

(c) Third hop

Properties of Flooding

- all possible routes are tried
  - very robust
- at least one packet will have taken minimum hop count route
  - can be used to set up virtual circuit
- all nodes are visited
  - useful to distribute information (eg. routing)
- disadvantage is high traffic load generated

Routing Strategies - Random Routing

- simplicity of flooding with much less load
- node selects one outgoing path for retransmission of incoming packet
- selection can be random or round robin
- a refinement is to select outgoing path based on probability calculation
- no network info needed
- but a random route is typically neither least cost nor minimum hop

Routing Strategies - Adaptive Routing

- used by almost all packet switching networks
- routing decisions change as conditions on the network change due to failure or congestion
- requires info about network
- disadvantages:
  - decisions more complex
  - tradeoff between quality of network info and overhead
  - reacting too quickly can cause oscillation
  - reacting too slowly means info may be irrelevant

Adaptive Routing - Advantages

- improved performance
- aid congestion control
- but since is a complex system, may not realize theoretical benefits
  - cf. outages on many packet-switched nets

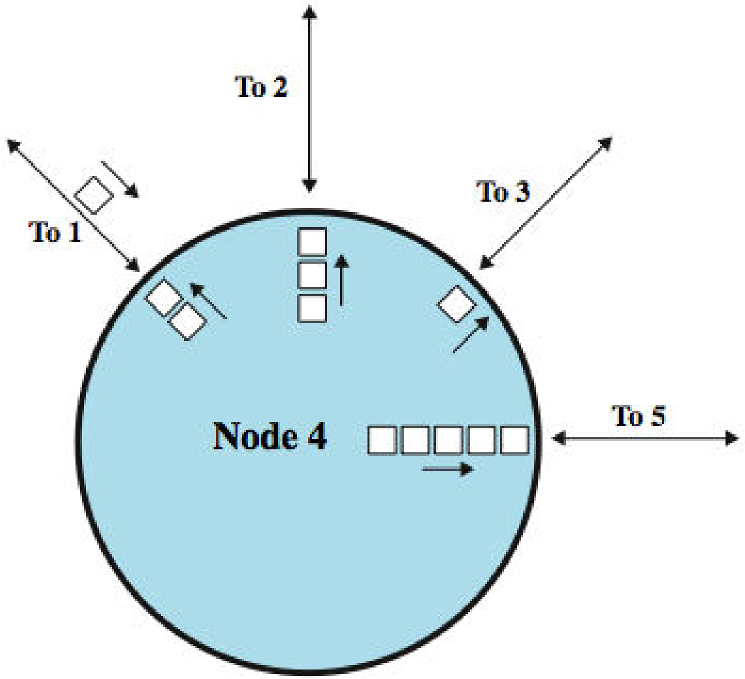
Classification of Adaptive Routing Startegies

- based on information sources
  - local (isolated)
    - route to outgoing link with shortest queue
    - can include bias for each destination
    - Rarely used - does not make use of available info
  - adjacent nodes
    - takes advantage on delay / outage info
    - distributed or centralized
  - all nodes
    - like adjacent

Isolated Adaptive Routing

Node 4's Bias
Table for
Destination 6

Next Node	Bias
1	9
2	6
3	3
5	0



ARPANET Routing Strategies 1st Generation

- 1969
- distributed adaptive using estimated delay
  - queue length used as estimate of delay
- using Bellman-Ford algorithm
- node exchanges delay vector with neighbors
- update routing table based on incoming info
- problems:
  - doesn't consider line speed, just queue length
  - queue length not a good measurement of delay
  - responds slowly to congestion

ARPANET Routing Strategies 2nd Generation

- 1979
- distributed adaptive using measured delay
  - using timestamps of arrival, departure & ACK times
- recomputes average delays every 10secs
- any changes are flooded to all other nodes
- recompute routing using Dijkstra’s algorithm
- good under light and medium loads
- under heavy loads, little correlation between reported delays and those experienced

ARPANET Routing Strategies 3rd Generation

- 1987
- link cost calculations changed
  - to damp routing oscillations
  - and reduce routing overhead
- measure average delay over last 10 secs and transform into link utilization estimate
- normalize this based on current value and previous results
- set link cost as function of average utilization

Least Cost Algorithms

- basis for routing decisions
  - can minimize hop with each link cost 1
  - or have link value inversely proportional to capacity
- defines cost of path between two nodes as sum of costs of links traversed
  - in network of nodes connected by bi-directional links
  - where each link has a cost in each direction
- for each pair of nodes, find path with least cost
  - link costs in different directions may be different
- alternatives: Dijkstra or Bellman-Ford algorithms

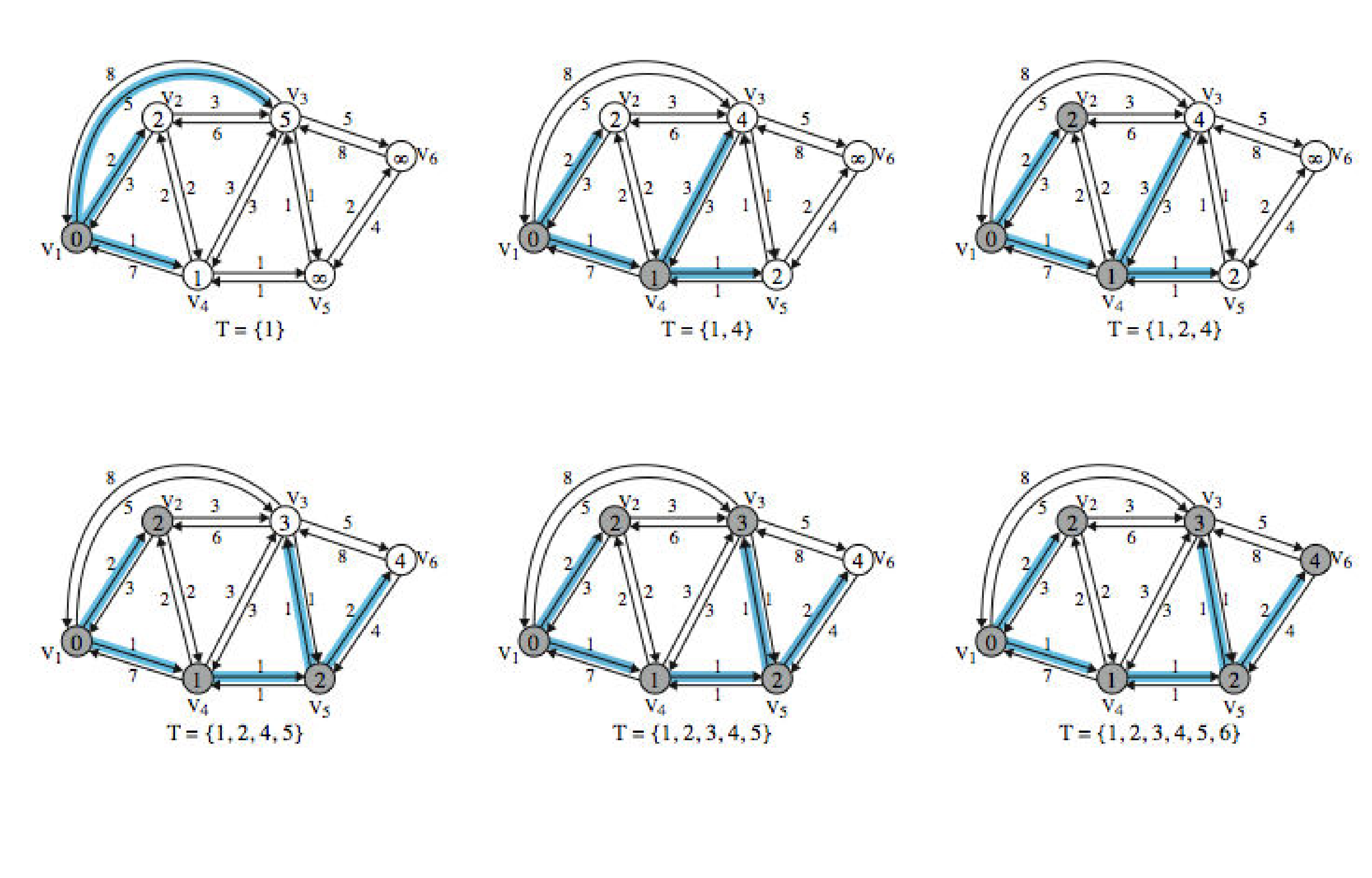
Dijkstra's Algorithm

- finds shortest paths from given source node s to all other nodes
- by developing paths in order of increasing path length
- algorithm runs in stages (next slide)
  - each time adding node with next shortest path
- algorithm terminates when all nodes processed by algorithm (in set T)

Dijkstra’s Algorithm Method

- Step 1 [Initialization]
  - T = {s} Set of nodes so far incorporated
  - L(n) = w(s, n) for n ≠ s
  - initial path costs to neighboring nodes are simply link costs
- Step 2 [Get Next Node]
  - find neighboring node not in T with least-cost path from s
  - incorporate node into T
  - also incorporate the edge that is incident on that node and a node in T that contributes to the path
- Step 3 [Update Least-Cost Paths]
  - L(n) = min[L(n), L(x) + w(x, n)] for all $n \notin T$
  - if latter term is minimum, path from s to n is path from s to x concatenated with edge from x to n

Dijkstra’s Algorithm Example



T = {1}

T = {1, 4}

T = {1, 2, 4}

T = {1, 2, 4, 5}

T = {1, 2, 3, 4, 5}

T = {1, 2, 3, 4, 5, 6}

Dijkstra’s Algorithm Example

Iter	T	L(2)	Path	L(3)	Path	L(4)	Path	L(5)	Path	L(6)	Path
1	{1}	2	1–2	5	1-3	1	1–4	∞	-	∞	-
2	{1,4}	2	1–2	4	1-4-3	1	1–4	2	1-4–5	∞	-
3	{1, 2, 4}	2	1–2	4	1-4-3	1	1–4	2	1-4–5	∞	-
4	{1, 2, 4, 5}	2	1–2	3	1-4-5–3	1	1–4	2	1-4–5	4	1-4-5–6
5	{1, 2, 3, 4, 5}	2	1–2	3	1-4-5–3	1	1–4	2	1-4–5	4	1-4-5–6
6	{1, 2, 3, 4, 5, 6}	2	1-2	3	1-4-5-3	1	1-4	2	1-4–5	4	1-4-5-6

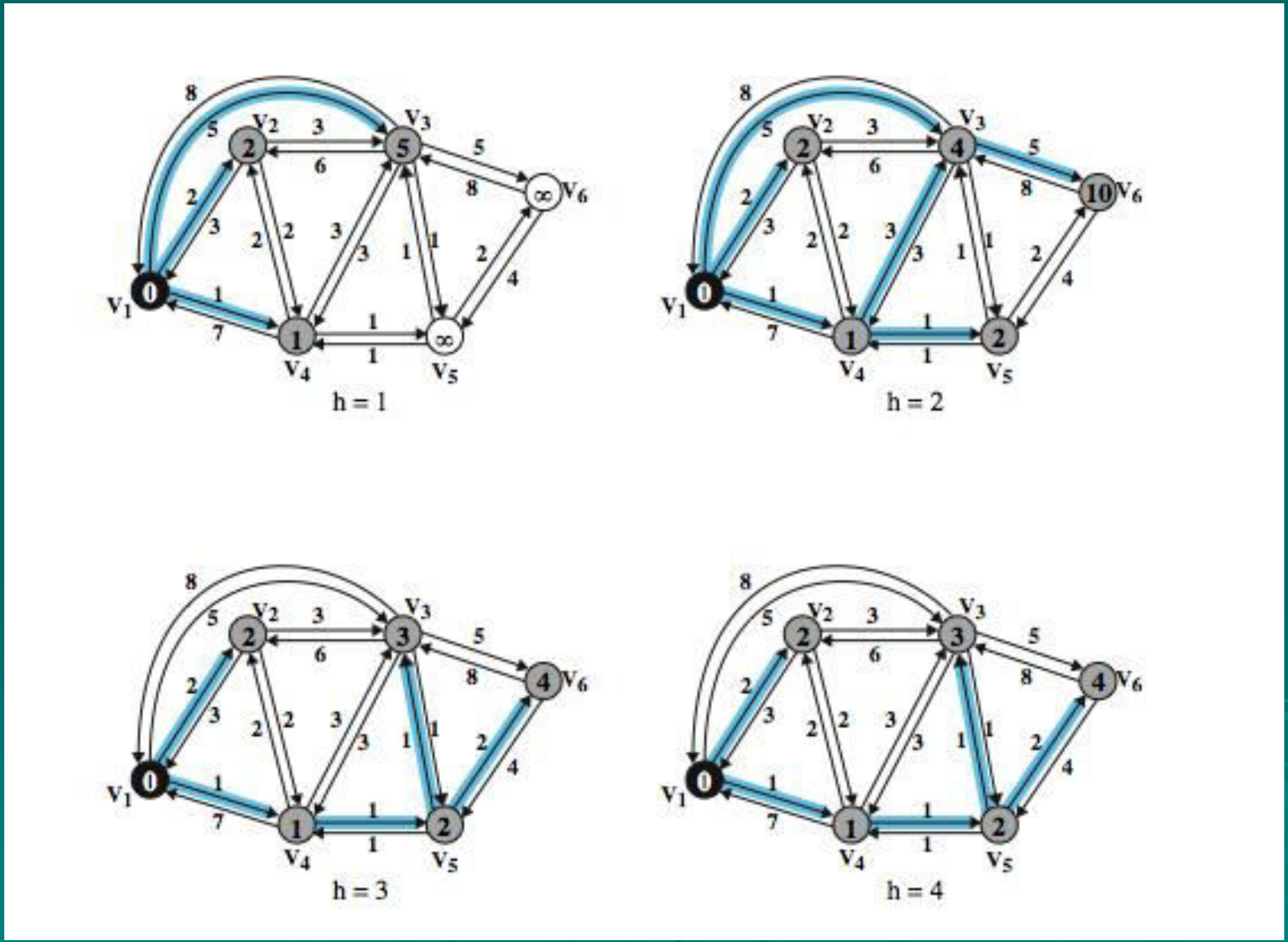
Bellman-Ford Algorithm

- find shortest paths from given node subject to constraint that paths contain at most one link
- find the shortest paths with a constraint of paths of at most two links
- and so on

Bellman-Ford Algorithm

- step 1 [Initialization]
  - $L_0(n) = \infty$, for all $n \neq s$
  - $L_h(s) = 0$, for all h
- step 2 [Update]
  - for each successive $h \geq 0$
    - for each $n \neq s$, compute: $L_{h+1}(n)=\min_j[L_h(j)+w(j,n)]$
  - connect n with predecessor node j that gives min
  - eliminate any connection of n with different predecessor node formed during an earlier iteration
  - path from s to n terminates with link from j to n

Example of Bellman-Ford Algorithm



Results of Bellman-Ford Example

h	$L_h(2)$	Path	$L_h(3)$	Path	$L_h(4)$	Path	$L_h(5)$	Path	$L_h(6)$	Path
0	∞	-	∞	-	∞	-	∞	-	∞	-
1	2	1-2	5	1-3	1	1-4	∞	-	∞	-
2	2	1-2	4	1-4-3	1	1-4	2	1-4-5	10	1-3-6
3	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
4	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6

Comparison

- results from two algorithms agree
- Bellman-Ford
  - calculation for node n needs link cost to neighbouring nodes plus total cost to each neighbour from s
  - each node can maintain set of costs and paths for every other node
  - can exchange information with direct neighbors
  - can update costs and paths based on information from neighbors and knowledge of link costs
- Dijkstra
  - each node needs complete topology
  - must know link costs of all links in network
  - must exchange information with all other nodes

Evaluation

- dependent on
  - processing time of algorithms
  - amount of information required from other nodes
- implementation specific
- both converge under static topology and costs
- both converge to same solution
- if link costs change, algs attempt to catch up
- if link costs depend on traffic, which depends on routes chosen, may have feedback instability

Summary

- routing in packet-switched networks
- routing strategies
  - fixed, flooding, random,adaptive
- ARPAnet examples
- least-cost algorithms
  - Dijkstra, Bellman-Ford