ELEC3500/6500
Tute 8 – Solutions

8.1. A centralized routing algorithm computes the least-cost path between a source and destination by using **complete, global knowledge about the network**, that is, the algorithm needs to have the complete knowledge of the connectivity between all nodes and all links' costs. The actual calculation can be run at one site or could be replicated in the routing component of each and every router.

A distributed routing algorithm calculates the lease-cost path in an iterative, distributed manner by the routers. With a decentralized algorithm, **no node has the complete information about the costs of all network links**. Each node begins with only the knowledge of the costs of its own directly attached links, and then through an iterative process of calculation and information exchange with its neighboring nodes, a node **gradually** calculates the least-cost path to a destination or a set of destinations.

OSPF protocol is an example of centralized routing algorithm, and BGP is an example of a distributed routing algorithm.

8.2. Each node begins with only the knowledge of the costs of its own directly attached links. Then, through an iterative process of calculation and exchange of information with its neighboring nodes (that is, nodes that are at the other end of links to which it itself is attached), a node gradually calculates the least-cost path to a destination or set of destinations.

8.3.

y-x-u, y-x-v-u, y-x-w-u, y-x-w-v-u,

y-w-u, y-w-v-u, y-w-x-u, y-w-x-v-u, y-w-v-x-u,

y-z-w-u, y-z-w-v-u, y-z-w-x-u, y-z-w-x-v-u, y-z-w-v-x-u

8.4. Using the Djikstra's algorithm:

	u: D(u), p(u)	v: D(v), p(v)	w: D(w), p(w)	y: D(y), p(y)	z: D(z), p(z)
x	**1, x**	2, x	3, x	1, x	∞
u		2, x	3, x	**1, x**	∞
y		**2, x**	2, y		3, y
v			**2, y**		3, y
w					**3, y**

Bold, red = least-cost path to that node. When the two costs are the same, break tie arbitrarily.

Routing table at node x:

Destination	Output link
u	u
v	v
w	y
y	y
z	y

8.5.

Cost to

		u	v	x	y	z
	v	∞	∞	∞	∞	∞
From	x	∞	∞	∞	∞	∞
	z	∞	6	2	∞	0

Cost to

		u	v	x	y	z
	v	1	0	3	∞	6
From	x	∞	3	0	3	2
	z	7	5	2	5	0

Cost to

		u	v	x	y	z
	v	1	0	3	3	5
From	x	4	3	0	3	2
	z	6	5	2	5	0

Cost to

		u	v	x	y	z
	v	1	0	3	3	5
From	x	4	3	0	3	2
	z	6	5	2	5	0

8.6. Each node runs two iterations. After each iteration, each exchanges/sends its tables to its directly connected neighbours. A node then updates its own table based on (i) the link costs from itself to its neighbours, (ii) "Rumours" or the tables sent by the neighbours. Bellman-Ford's equation is used for updating.

Node x table

		Cost to		
		x	y	z
	x	0	3	4
From	y	∞	∞	∞
	z	∞	∞	∞

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Node y table

		Cost to		
		x	y	z
	x	∞	∞	∞
From	y	3	0	6
	z	∞	∞	∞

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

Node z table

		Cost to		
		x	y	z
	x	∞	∞	∞
From	y	∞	∞	∞
	z	4	6	0

		Cost to		
		x	y	z
	x	0	3	4
From	y	3	0	6
	z	4	6	0

8.7.

(a) Message complexity:

- Link-state (LS) requires each node to know the cost of each link in the network. This requires O(|N| |E|) messages to be sent where |N| is the number of nodes and |E| is the number of links in the network. Whenever a link cost changes, the new link cost must be sent to all nodes.
- Distance-vector (DV) requires message exchanges between directly connected neighbours at each iteration. When link costs change, the DV will propagate the results of the changed link cost only if the new link cost results in a changed least-cost path of one of the nodes attached to that link.

(b) Speed of convergence:

- LS needs $O(|N|^2)$ complexity, typically quite fast and predictable.
- DV can converge slowly and can have count-to-infinity problems.

(c) Robustness:

If a router fails or is sabotaged:

- LS: a router can broadcast an incorrect **cost** of one of its attached links (but no others). But an LS node is computing only its own forwarding tables, so route calculations are somewhat separated under LS, providing a degree of robustness in this case.
- DV: a node can advertise incorrect least-cost paths to any or all destinations. An incorrect node calculation can be diffused through the entire network!