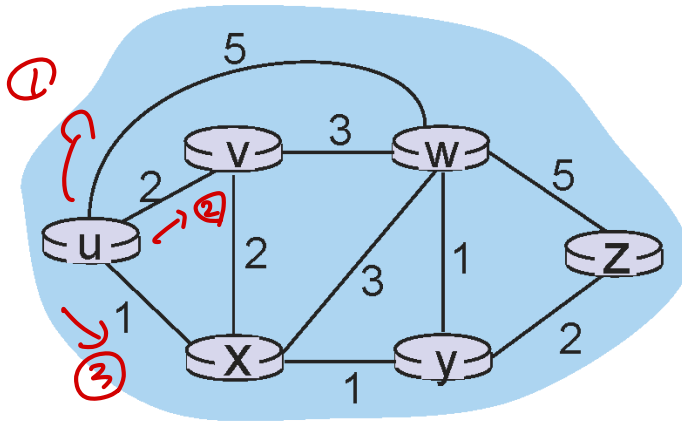


자기 자신이 알고있는 Distance 를 하나의

vector로써 이웃에게 넘겨줌

Bellman-Ford example



clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$\begin{aligned} d_u(z) &= \min \{ c(u,v) + d_v(z), \\ &\quad c(u,x) + d_x(z), \\ &\quad c(u,w) + d_w(z) \} \\ &= \min \{ 2 + 5, \\ &\quad 1 + 3, \\ &\quad 5 + 3 \} = 4 \end{aligned}$$

x

u	w	v	z
---	---	---	---

//

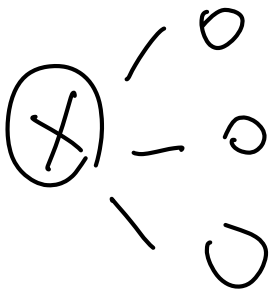
Distance

node achieving minimum is next hop in shortest path, used in forwarding table

반드시 전 알 / 아니면 그래프

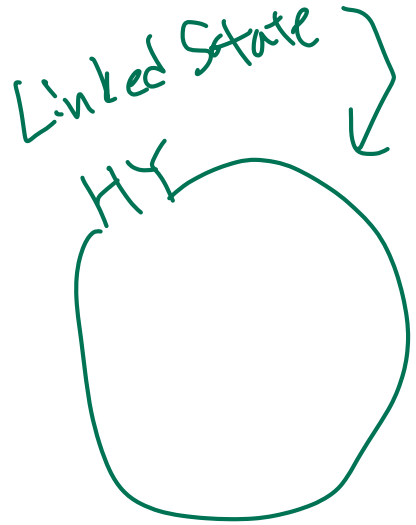
Distance vector algorithm

- ❖ $D_x(y)$ = estimate of least cost from x to y
 - x maintains distance vector $D_x = [D_x(y): y \in N]$
- ❖ node x :
 - knows cost to each neighbor v : $c(x,v)$
 - maintains its neighbors' distance vectors. For each neighbor v , x maintains $D_v = [D_v(y): y \in N]$



$$d_x(y) = \min_v [c(x,v) + d_v(y)]$$

Distance vector algorithm

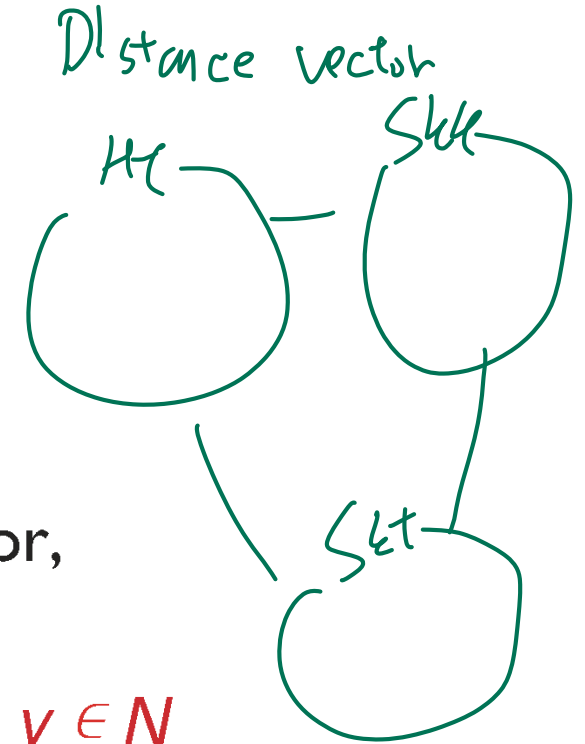


key idea:

- ❖ from time-to-time, each node sends its own distance vector estimate to neighbors
- ❖ when x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow \min_v \{c(x,v) + D_v(y)\} \text{ for each node } y \in N$$

- ❖ under minor, natural conditions, the estimate $D_x(y)$ converge to the actual least cost $d_x(y)$



Distance vector algorithm

iterative, asynchronous:

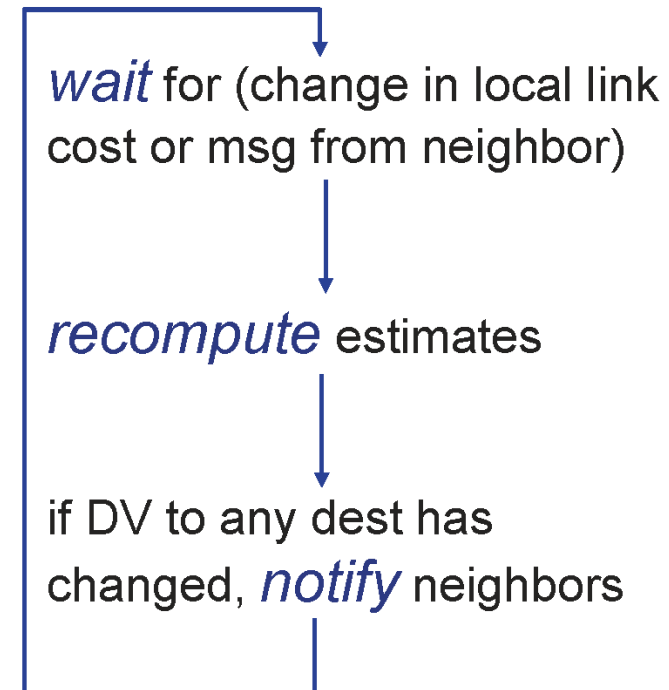
each local iteration
caused by:

- ❖ local link cost change
- ❖ DV update message from neighbor

distributed:

- ❖ each node notifies neighbors *only* when its DV changes
 - neighbors then notify their neighbors if necessary

each node:



$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\} \\ = \min\{2+0, 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\} \\ = \min\{2+1, 7+0\} = 3$$

node x
table

	cost to		
	x	y	z
from x	0	2	7
from y	∞	∞	∞
from z	∞	∞	∞

node y
table

	cost to		
	x	y	z
from x	∞	∞	∞
from y	2	0	1
from z	∞	∞	∞

node z
table

	cost to		
	x	y	z
from x	∞	∞	∞
from y	∞	∞	∞
from z	7	1	0

	cost to		
	x	y	z
from x	0	2	3
from y	2	0	1
from z	7	1	0

	cost to		
	x	y	z
from x	0	2	7
from y	2	0	1
from z	7	1	0

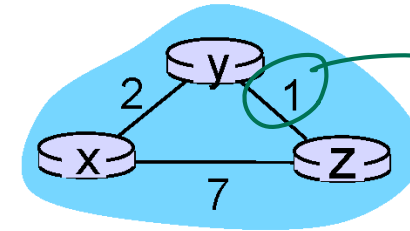
	cost to		
	x	y	z
from x	0	2	7
from y	2	0	1
from z	3	1	0

	cost to		
	x	y	z
from x	0	2	3
from y	2	0	1
from z	3	1	0

	cost to		
	x	y	z
from x	0	2	3
from y	2	0	1
from z	3	1	0

	cost to		
	x	y	z
from x	0	2	3
from y	2	0	1
from z	3	1	0

이것에게 새로운 값이 쓰여
다시 계산해 봐야 함



만약 값이 바뀌면 다시 값을 갱신

count infinity

값이 infinity 일 때 z-z 값을 알려주면 안됨