# Routing Algorithm Classification:

global: All routers have complete topology: "link state Algorithms"

static: routes change slowly over time

Dynamic: routes change more quickly
- periodic updates
- or in response to link cost changes

Decentralized: iterative process, exchange info with neighbors

"Distance Vector Algorithm"

## Link State Algorithm:

### Algorithm:

1. Initialization:
2. $N' = \{u\}$
3. for all nodes v
4. If v adjacent to u
5. then $D(v) = C_{u,v}$
6. else $D(v) = \infty$
7.
8. Loop:
9. find w not in $N'$ such that $D(w)$ is a minimum
10. add w to $N'$
11. update $D(v)$ for all v adjacent w and not in $N'$:
12. $D(v) = \min(D(v), D(w) + C_{w,v})$
15. Until All nodes in $N'$

Algorithm Complexity: $\frac{n(n+1)}{2} \simeq n^2$ ; $O(n^2)$ complexity

More efficient: $O(n \log n)$

## Router Oscillations:

when link costs depend on traffic volume, [described in GPT] path changes back and forth due to traffic every time cost is recomputed.

## Distance Vector Algorithm: (RIP)

Based on Bellman-Ford Equation

$$D_u(x) = \min_{v \in N(u)} \left( c(u,v) + D_v(x) \right)$$

$N(u)$ = Neighbors of u

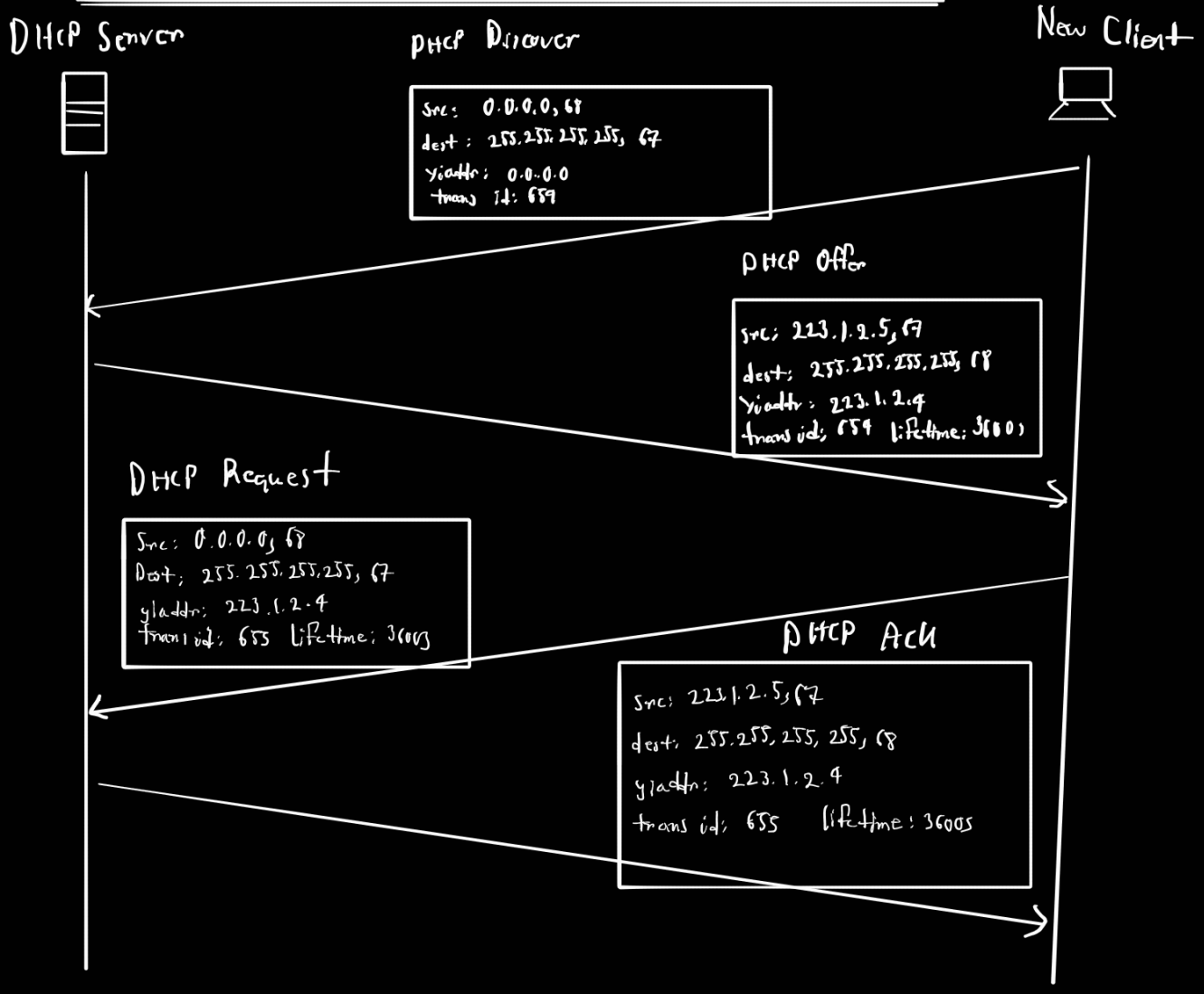
$c(u,v)$ = cost of the link $u \to v$

$D_v(x)$ = v's current estimate to reach x

## Key idea in DV:

1. Periodically, each node sends its own DV estimate to neighbor
2. x updates own DV estimate using B-F equation.
3. Changes to actual least-cost

# DHCP – Dynamic Host Configuration Protocol :




## Process:

New Client:

Src IP = 0.0.0.0

Dest IP = 255.255.255.255

UDP 68 = Client Port

UDP 67 = Server Port

| Ethernet | IP | UDP | DHCP Message |
|---|---|---|---|

## HW: Ch. 4:

**P14:** 128, 119. 40, 129 (Ans)

ISP block = 128, 119, 40, 64/26

/26 → $2^6 = 64 - 2 = 62$ devices

Since 4 subnets with same number of IP,

each subnet needs to have 16 address spaces.

The address prefix will be 128, 119, 40, 64/28

128, 119, 40, 80/28

128, 119, 40, 96/28

128, 119, 40, 112/28

**P15:**

| | | Method | Range |
|---|---|---|---|
| Subnet A = | 250 < 254 → $2^8$ → /24 | 214.97.254.0/24 | 214.97.254.1 – .254 |
| Subnet B = | 120 < 126 → $2^7$ → /25 | 214.97.255.0/25 | 214.97.255.1 – .126 |
| Subnet C = | 120 < 126 → $2^7$ → /25 | 214.97.255.128/25 | 214.97.255.129 – .254 |

D, E, F isn't possible unless /22

Longest Prefix match = match the right subnet prefix

**Forwarding table:**

## SWE Quiz 2:

### Ans. No. 01

| Prefix match | Link interface |
|---|---|
| 11100000 00 | 0 |
| 11100000 01000000 | 1 |
| 11100000 | 2 |
| 11100001 0 | 2 |
| Otherwise | 3 |

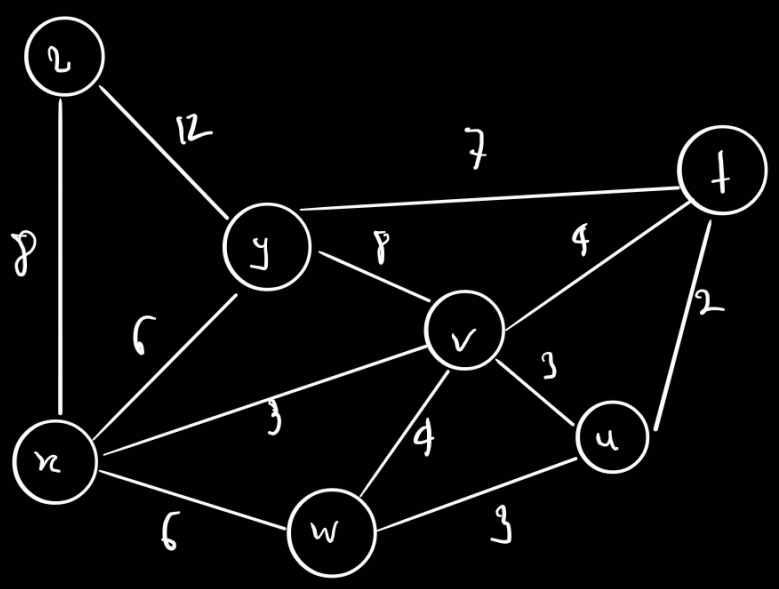
for Interface 2,

11100000 (all 224.*) → 2

11100001 0

## Link State Algorithm:

### SWE Quiz-2:



| Steps | N' | D(t), p(t) | D(u), p(u) | D(v), p(v) | D(w), p(w) | D(y), p(y) | D(z), p(z) |
|---|---|---|---|---|---|---|---|
| 0 | x | ∞, – | ∞, – | 3, x | 6, x | 6, x | 8, x |
| 1 | xv | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 2 | xvu | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 3 | xvuw | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 4 | xvuwy | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 5 | xvuwyt | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 6 | xvuwytz | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |

T.O = $O(n^2)$ → Algorithm Complexity

Message Complexity = $O(n)$ link crossings from source (worst case)

Thus, $O(n^2)$

Route oscillation when link cost depends on traffic

## Distance Vector Algorithm:

D = hop diameter of the shortest-path;

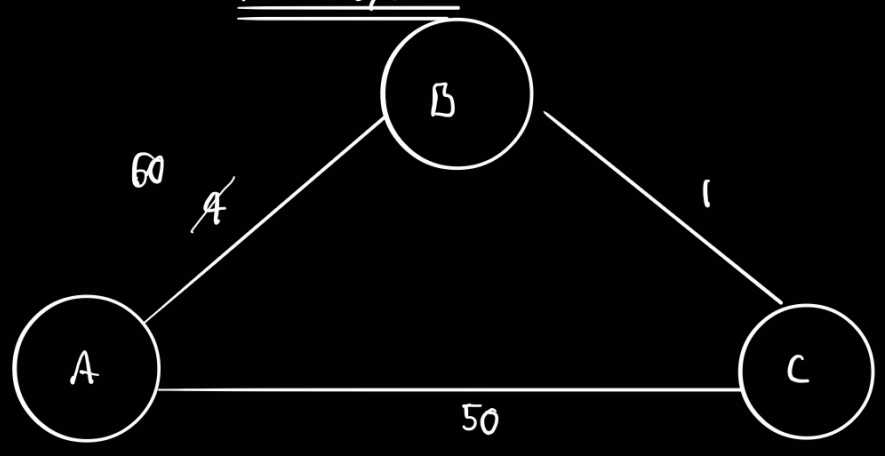
The longest sequence of hop needed to reach the last router from source

|V| = No. of routers

$D \leq |V| - 1$ [worst case]

SWE Final:

Ans. No, 03




(i)

Routing Tables:

B,

| Dest | Distance | Next Hop |
|---|---|---|
| A | 4 | A |
| B | 0 | – |
| C | 1 | C |

C.

| Dest | Distance | Next Hop |
|---|---|---|
| A | 5 | B, A |
| B | 1 | B |
| C | 0 | – |

(ii)

in OSPF, The change is immediately broadcasted to all router, then routers run Dijkstra,

| step | N' | D(A), P(A) | D(B), P(B) |
|---|---|---|---|
| 0 | C | 50, C | 1, C |
| 1 | CB | 5, C | 1, C |
| 2 | CBA | 5, C | 1, C |

Routing Tables.

B, C,

In RIP, DV can lead to 'count-to-infinity' problem when a link cost changes.

Before $t'$, C was advertising $C \to A = C \to B \to A = 5 < 50$ on direct $C \to A$

Immediately after $t'$,

$B \to A = 60$ but from C, $C \to A = 5$

Thus, B chooses $B \to C \to A$ thinking $B \to A$ now B $5+1=6$ via C and updates neighbors.

Update sequence. (to A)

B becomes 6 via C

C becomes 7 via B

B becomes 8 via C

... increases step by step ...

continues until C prefers its direct link to A (cost 50) over 51.

Final RIP matches OSPF table.

(iii)

<u>OSPF:</u> Flooding, immediate recalculation

<u>RIP:</u> May encounter routing loop, eventually chooses the correct path.

K.R. — Ch. 5:

Problems:

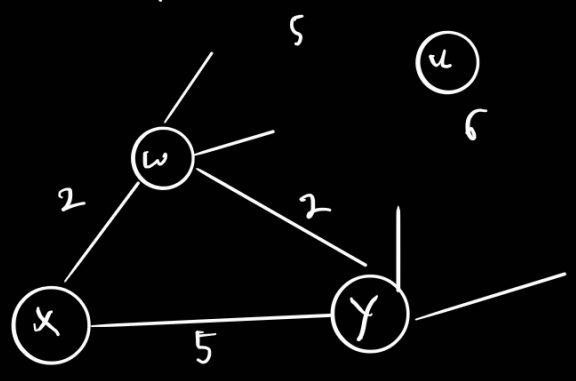
6. Max hop = $|V|-1$

Reason:

1. After $k$ iteration, each node knows best path

2. Shortest path are simple (no cycles)

7.




(a)

To w,

$$D_x(w) = \min\{c(x,w),\ (c(x,y)+c(y,w))\} = \min\{2, 5+2\} = 2$$

To y, $D_x(y) = \min\{c(x,y),\ c(x,w)+c(w,y)\} = \min\{5, 2+2\} = 4$

To u, Next hop, w, cost 7.

(b)

Example Change: if $c(x,w) = 1$ instead of 2

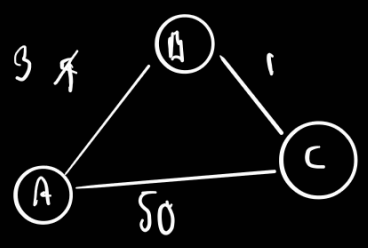
then, $D_x(u) = 1 + D_w(u) = 1+5 = 6$ ; New best cost

Or, if $c(x,y) + c(y,w) < 7$, then $x$ will inform its neighbor.

(c)

if $c(x,w) = 3$, it won't advertise to its neighbor

9.




$\boxed{C \to A = C \to B \to A}$ → Advertises

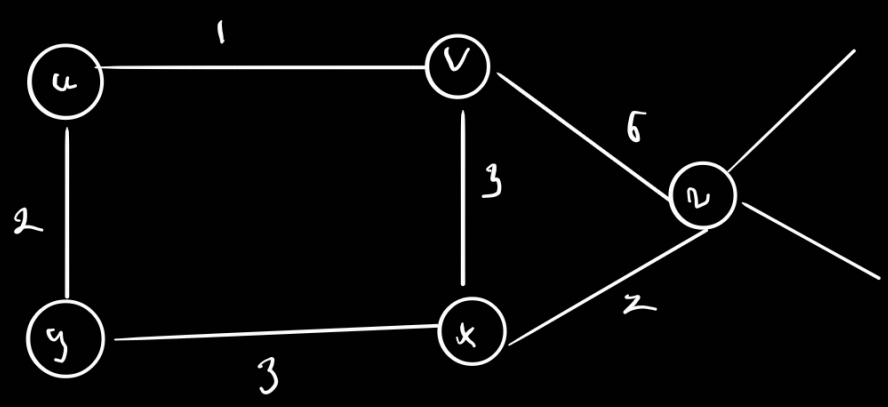
But Now, $B \to A = 3 < 4$ chooses B→A direct

Count-to-infinity problem won't occur.

for new nodes count-to-infinity also will not occur

because its a potential good neighbor.

5.




from the diagram,

| | | |
|---|---|---|
| $c(u,v) = 1$ | $c(y,x) = 3$ | $c(x,z) = 2$ |
| $c(u,y) = 2$ | $c(v,x) = 3$ | $c(v,z) = 6$ |

Distance advertised by neighbors

After convergence,

From x,

$D_x(u) = 4$ ; $x \to v \to u$

$D_x(v) = 3$

$D_x(y) = 3$

$D_x(x) = 0$

$D_x(z) = 2$

From v,

$D_v(u) = 1$

$D_v(x) = 3$

$D_v(y) = 3$

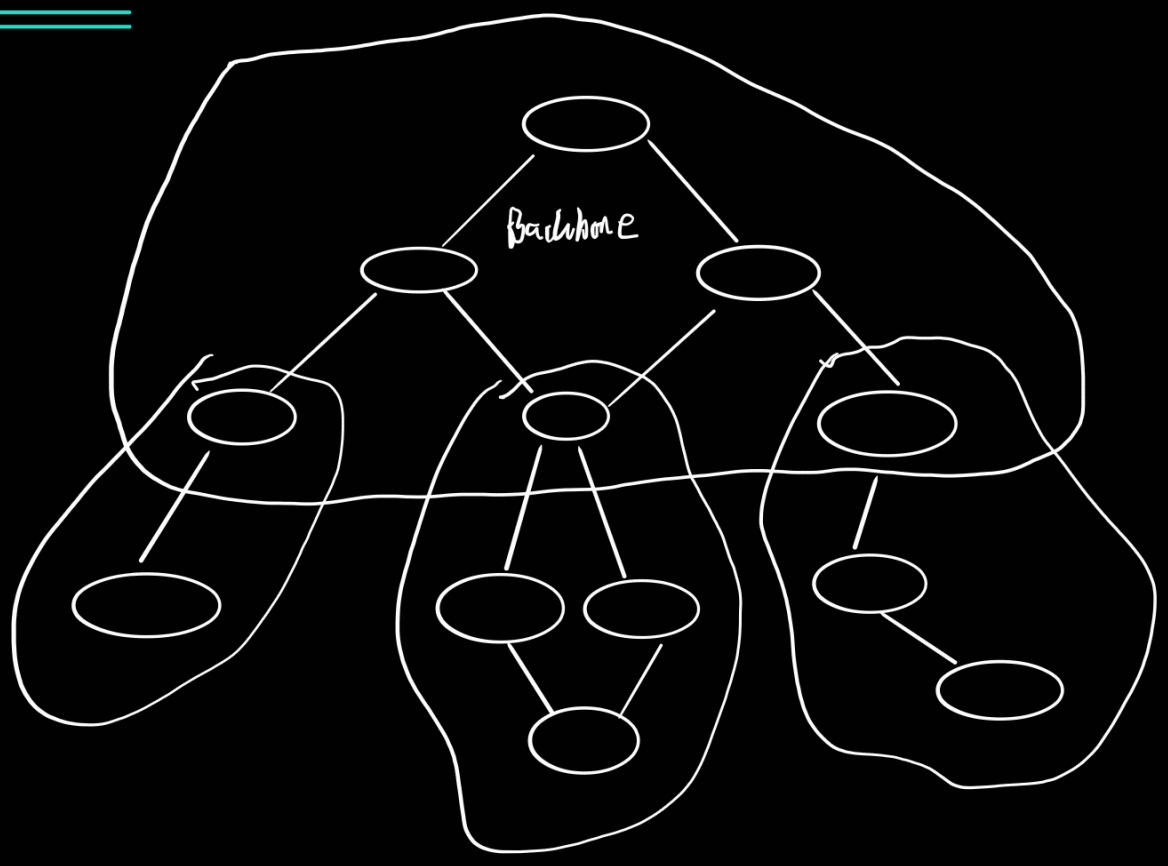
$D_v(v) = 0$

$D_v(z) = 5$ $v \to x \to z$

Distance Table:

| Dest | Distance | Path |
|---|---|---|
| u | 6 | $z \to v \to u$ |
| v | 5 | $z \to x \to v$ |
| y | 5 | $z \to x \to y$ |
| x | 2 | $z \to x$ |
| z | 0 | – |

(Ans)

# Hierarchical OSPF:

1. Local Area
2. Backbone