# COMPUTER COMMUNICATION NETWORKS

**M Rajasekar**

Department of Electronics and Communication Engineering

# COMPUTER COMMUNICATION NETWORKS

## Routing algorithms: Dijkstra algorithm

**M Rajasekar**

Department of Electronics and Communication Engineering

## Link state algorithm (Dijkstra's algorithm)

- A central controller broadcasts the link state information
- For each node find the MST considering itself as the root
- Nodes implement the LS algorithm in parallel
  Synchronized
- A forwarding table is constructed based on the MST

## Link state algorithm (Dijkstra's algorithm) (Contd.):

**Initialization:**

```
  N' = {u}
  for all nodes v
    if v is a neighbor of u
      then D(v) = c(u,v)
    else D(v) = ∞
```

**Loop**

```
  find w not in N' such that D(w) is a minimum
  add w to N'
  update D(v) for each neighbor v of w and not in N':
```

$$D(V) = \min_{w} \left(D(V), \{D(W) + C(W,V)\}\right)_{W \in N}$$

$$D_U(V) = \min\{D_U(W) + C(W,V)\}_{W \in N \setminus \{U,V\}}$$

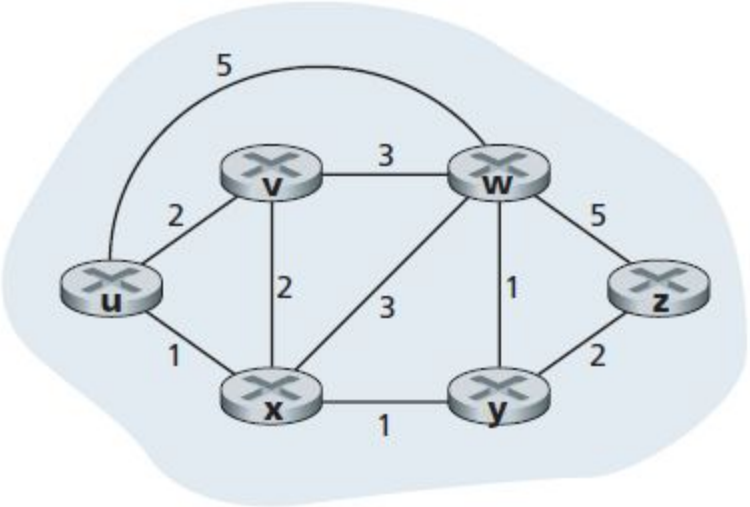
$$D_U(Z) = \min\{ D_U(V) + C(V,Z),\ D_U(X) + C(X,Z), D_U(Y) + C(Y,Z), D_U(W) + C(W,Z)\}$$

```
 /* new cost to v is either old cost to v or known
  least path cost to w plus cost from w to v */
```

**until** N'= N

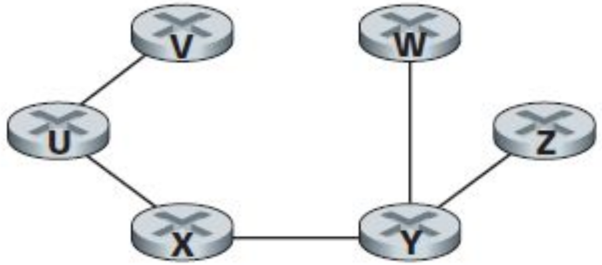
## Link state algorithm (Dijkstra's algorithm) (Contd.):

## Example 1:




| step | N' | D(v),p(v) | D(w),p(w) | D(x),p(x) | D(y),p(y) | D(z),p(z) |
|---|---|---|---|---|---|---|
| 0 | u | 2,u | 5,u | 1,u | ∞ | ∞ |
| 1 | ux | 2,u | 4,x | | 2,x | ∞ |
| 2 | uxy | 2,u | 3,y | | | 4,y |
| 3 | uxyv | | 3,y | | | 4,y |
| 4 | uxyvw | | | | | 4,y |
| 5 | uxyvwz | | | | | |

## Link state algorithm (Dijkstra's algorithm) (Contd.):



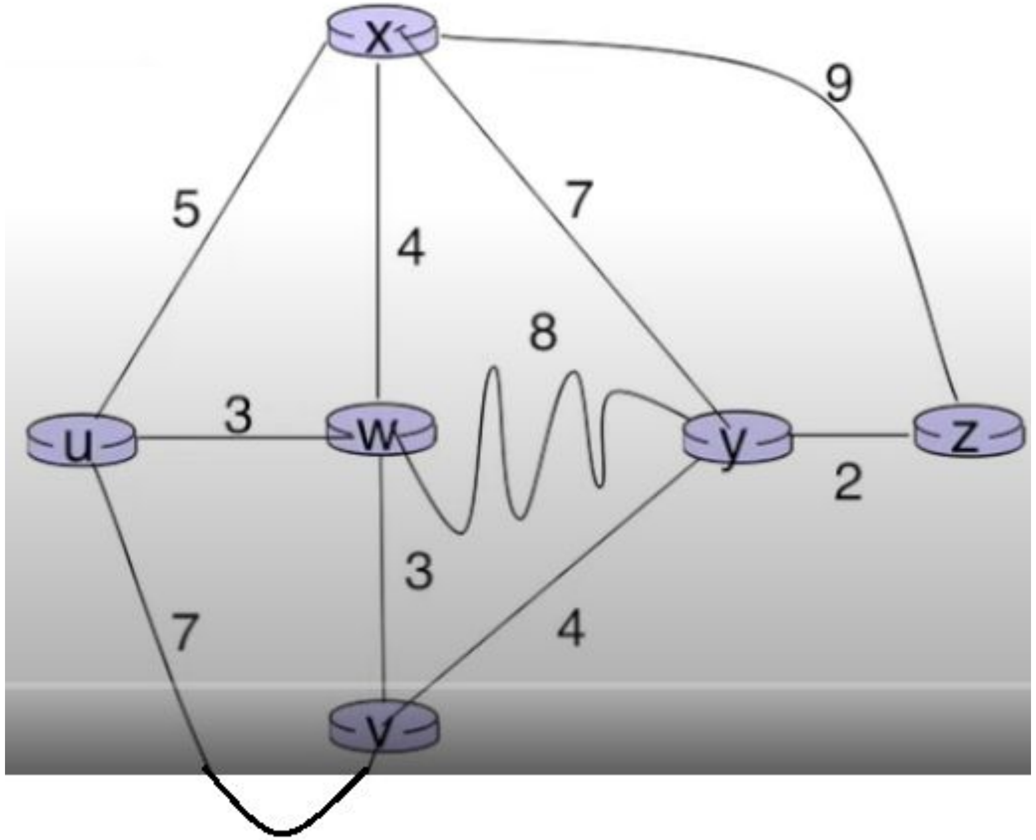

| Destination | Link |
|---|---|
| v | (u, v) |
| w | (u, x) |
| x | (u, x) |
| y | (u, x) |
| z | (u, x) |

**Figure 4.28** ♦ Least cost path and forwarding table for nodule u

| step | N' | D(v),p(v) | D(w),p(w) | D(x),p(x) | D(y),p(y) | D(z),p(z) |
|---|---|---|---|---|---|---|
| 0 | u | 2,u | 5,u | 1,u | ∞ | ∞ |
| 1 | ux | 2,u | 4,x | | 2,x | ∞ |
| 2 | uxy | 2,u | 3,y | | | 4,y |
| 3 | uxyv | | 3,y | | | 4,y |
| 4 | uxyvw | | | | | 4,y |
| 5 | uxyvwz | | | | | |

## Link state algorithm (Dijkstra's algorithm) (Contd.):

## Example 2:



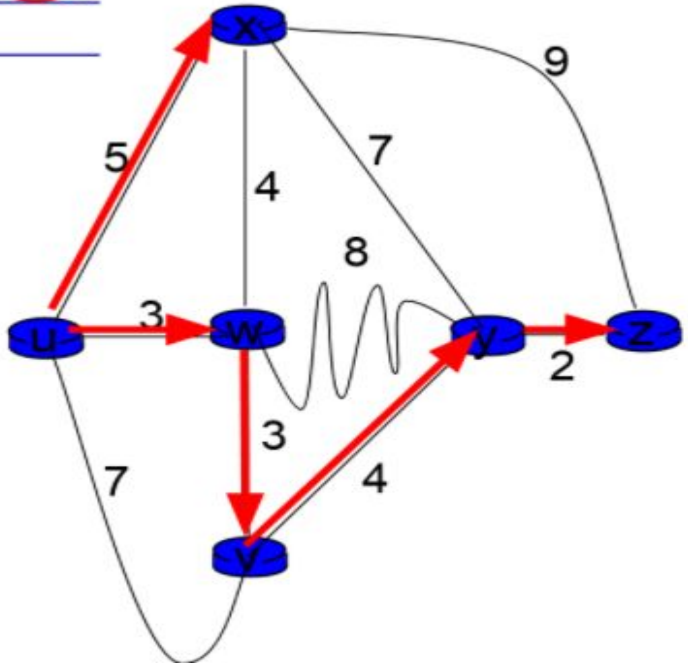
## Link state algorithm (Dijkstra's algorithm) (Contd.):

### Example 2:

| Step | N' | D(v) p(v) | D(w) p(w) | D(x) p(x) | D(y) p(y) | D(z) p(z) |
|---|---|---|---|---|---|---|
| 0 | u | 7,u | 3,u | 5,u | ∞ | ∞ |
| 1 | uw | 6,w | | 5,u | 11,w | ∞ |
| 2 | uwx | 6,w | | | 11,w | 14,x |
| 3 | uwxv | | | | 10,v | 14,x |
| 4 | uwxvy | | | | | 12,y |
| 5 | uwxvyz | | | | | |

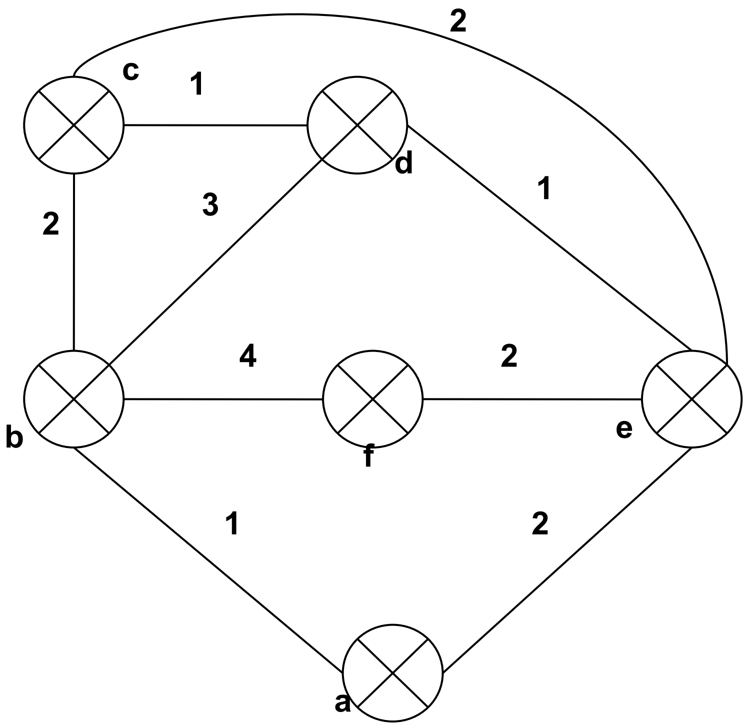
*notes:*

- construct shortest path tree by tracing predecessor nodes
- ties can exist (can be broken arbitrarily)



**Numerical 1:**

Calculate the cost $D(v)$ (where $v \in \{b, c, d, e, f\}$) for each least-cost path from node $a$ to all other nodes in the network using link state algorithm. Add rows and update the $D(v), p(v)$ in the table below for five iterations. Here, $p(v)$ represents the 1st neighbour to $v$ on the least cost path from $a$ to $v$.




| Iteration | | D(b), p(b) | D(c), p(c) | D(d), p(d) | D(e), p(e) | D(f), p(f) |
|---|---|---|---|---|---|---|
| 0 | a | | | | | |

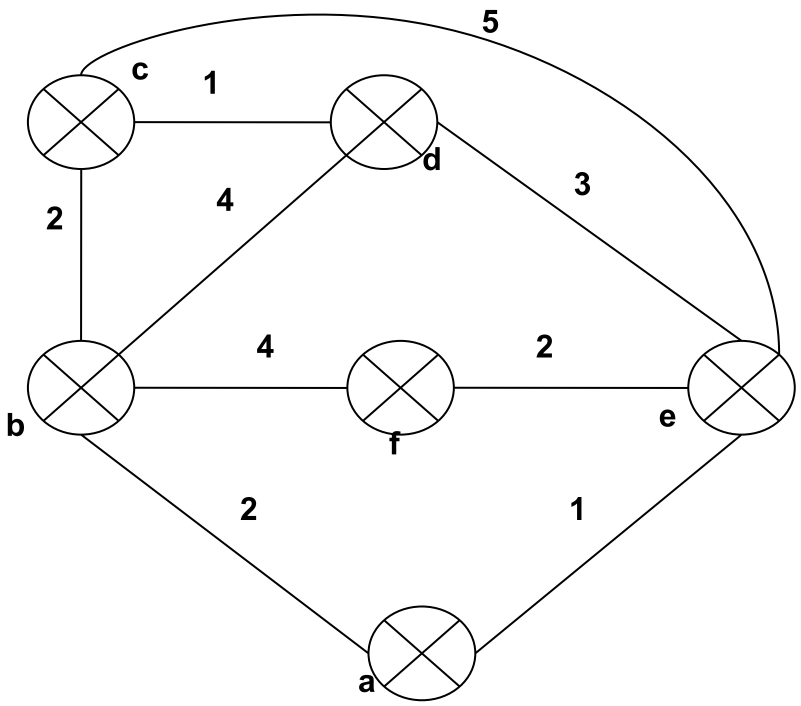
## Routing algorithms: Dijkstra algorithm

| Iteration | | D(b), p(b) | D(c), p(c) | D(d), p(d) | D(e), p(e) | D(f), p(f) |
|---|---|---|---|---|---|---|
| 0 | a | **1,a** | ∞ | ∞ | 2,a | ∞ |
| 1 | a, b | 1,a | 3,b | 4,b | **2,a** | 5,f |
| 2 | a, b, e | 1,a | **3,b** | 3,e | 2,a | 4,e |
| 3 | a, b, e, c<br>or<br>a, b, e, d | 1,a | 3,b | **3,e** | 2,a | 4,e |
| 4 | a, b, e, c, d | 1,a | 3,b | 3,e | 2,a | **4,e** |
| 5 | a, b, e, c, d, f | | | | | |

**Numerical 2:**

Calculate the cost $D(v)$ (where $v \in \{a, b, c, d, e\}$) for each least-cost path from node $f$ to all other nodes in the network using link state algorithm. Add rows and update the $D(v), p(v)$ in the table below for five iterations. Here, $p(v)$ represents the 1st neighbour to $v$ on the least cost path from $f$ to $v$.

| Iteration | | D(a), p(a) | D(b), p(b) | D(c), p(c) | D(d), p(d) | D(e), p(e) |
|---|---|---|---|---|---|---|
| 0 | f | | | | | |



## Routing algorithms: Dijkstra algorithm

| Iteration | | D(a), p(a) | D(b), p(b) | D(c), p(c) | D(d), p(d) | D(e), p(e) |
|---|---|---|---|---|---|---|
| 0 | f | ∞ | 4,f | ∞ | ∞ | **2,f** |
| 1 | f,a | **3,e** | 4,f | 7,e | 5,e | |
| 2 | f,a,e | | **4,f** | 7,e | 5,e | |
| 3 | f,a,e,b | | | 6,b | **5,e** | |
| 4 | f,a,e,b,d | | | **6,b** | | |
| 5 | f,a,e,b,d,c | | | | | |

# THANK YOU

**M Rajasekar**
Department of Electronics and Communication Engineering
**rajasekarmohan@pes.edu**