Network Layer

Graphs

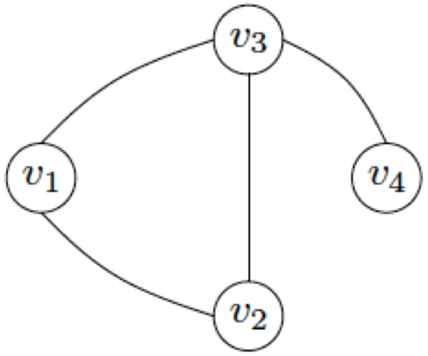
Definition 2.1 (Graph). *A graph G is a pair (V, E), where V is a set of* **nodes** *and $E \subseteq V \times V$ is a set of* **edges** *between the nodes. The number of nodes is denoted by n and the number of edges by m.*

- **Directed graph:** each edge has a direction.
- **Undirected graph:** all the edges have no direction.
- **Weighted graph:** each edge has a weight w(e).
  - The weight of the graph G is $w(G) = \sum w(e)$.

In the internet computers, smartphones, routers etc. are nodes and the wired and wireless connections are edges.

Graphs Representation

- **Adjacency matrix:**
  - *n* x *n* matrix with 1 in location (i,j) iff nodes i and j are connected and 0 otherwise.
  - If the graph is weighted, the 1s are replaced with the weights.
- **Adjacency list:**
  - Every element corresponds to an edge of the graph identified by its endpoints.
  - Better representation for sparse graphs.



	v_1	v_2	v_3	v_4
v_1	0	1	1	0
v_2	1	0	1	0
v_3	1	1	0	1
v_4	0	0	1	0

Figure 2.2: A graph $G = (V, E)$ with node set $V = \{v_1, v_2, v_3, v_4\}$ and edge set $E = \{\{v_1, v_2\}, \{v_1, v_3\}, \{v_2, v_3\}, \{v_3, v_4\}\}$, and the adjacency matrix of G.

Paths and trees

Definition 2.4 (Path). *Let $G = (V, E)$ be a graph. A* ***path*** *between nodes v_1 and v_k is a sequence of nodes $(v_1, v_2, \ldots, v_k)$, where $\{v_i, v_{i+1}\} \in E$ for all $1 \leq i < k$. The path has $k - 1$* ***hops****.*

- **Connected graph:** exists a path between any two nodes.
- **Cycle:** a sequence of connected nodes such that the first and last node of the sequence are the same and no other node appears twice.
- **Tree:** a connected graph that contains no cycles. Has *n-1* edges.
- **Spanning tree:** a tree that connects all nodes in a graph.
- **Rooted tree:** tree with a special root node *r.* Every other node *v* has a parent, i.e., the node adjacent to *v* and closer to *r*.

Spanning trees

Subgraph:

Definition 2.8 (Subgraph). *Let $G = (V, E)$ be a graph. A* ***subgraph*** $G' = (V', E')$ *of G is a graph such that $V' \subseteq V$ and $E' \subseteq E$.*

Spanning tree:

Definition 2.9 (Spanning tree). *Given a graph $G = (V, E)$, a* ***spanning tree*** $T = (V, E')$ *is a subgraph of G that is a tree.*

Minimum spanning tree:

Definition 2.10 (MST). *Given a weighted graph $G = (V, E, \omega)$, a minimum spanning tree (MST) T is a spanning tree that minimizes the total weight $\omega(T)$.*

Minimum spanning tree algorithm

Algorithm 2.11 MST Algorithm

Given a weighted graph $G = (V, E, \omega)$
Let $S = \{u\}$ be a set of visited nodes, initialized with any node $u \in V$
Let T be a tree just consisting of the single node $u \in S$, no edges
while $S \neq V$ **do**
    Find minimum weight edge $e = \{v, w\}$ with $v \in S$ and $w \in V \setminus S$
    Add node w to S
    Add edge e to T
end while

- Greedily adds edges with the lowest weight at each iteration.
- Outputs a mínimum spanning tree.
- The time complexity is O(m log n).

Shortest path

- **Shortest path** between two nodes: path with the minimum total weight.

- **Distance** between two nodes *d(u,v):* total weight of the mínimum path.

- **Shortest path tree (SPT)**: a spanning tree *T*, rooted at r, of graph *G*, where the distance from any node to *r* in *T* equals the distance *d(r,v)* in *G*.

Shortest path algorithm

Algorithm 2.16 SPT Algorithm

Given a weighted graph $G = (V, E, \omega)$ and a node $r \in V$
Set a parent node p_v = null for every node $v \in V$
Set $d_r = 0$ and $d_v = \infty$ for every node $r \neq v \in V$
Let $S = \{r\}$ be the set of visited nodes
while $S \neq V$ **do**
    Find edge $e = \{v, w\}$ with $v \in S$ and $w \in V \setminus S$ with minimum $d_v + \omega(e)$
    Set $p_w = v$
    Set $d_w = d_v + \omega(e)$
    $S = S \cup \{w\}$
end while

- Greedily adds the node with the minimum distance to the root at each iteration.
- The time complexity is O(m log n)

Addressing

Every node in a graph has an **address**. In the internet, an IP address.

- **IPv4:** 32-bit address written in 4 chunks of 8 bits separated by dots.
- **IPv6:** 128-bit address written in 8 chunks of 16 bits separated by colons. Each chunk is written as 4 hexadecimal digits.

Prefix: A prefix of *k* bits corresponds to the first *k* bits of the address.

An address block or **subnet** is a set of addresses that share the same prefix.

Addressing: IPv6

IPv6 is conceived to enlarge the address space given the fast increase of devices connected to the internet.

IPv6 address notation can be compressed:

- Leave out leading zeros in every chunk.
- Consecutive section of zeros replaced with doublé colon (only once).

IPv4 addresses are included in the IPv6 domain. Usually written in hexadecimal as: ::ffff + IPv4

- e.g: IPv4-> 8.8.4.4 | IPv6 -> ::ffff:8:8:4:4

Packets

Definition 2.22 (Packet). *Every network packet contains a* **header** *and a* **payload**. *The payload of a packet corresponds to the actual data of the packet. The header contains information for delivering the payload.*

- Size in IPv4 limited to 65,535 bits -> many packets needed
- **Header**: source and destination addresses and oder options.
  - IPv4: 160 to 480 bits
  - IPv6: 320 bits
- **Time-To-Live (TTL):** number of hops a packet is allowed to travel before it is dropped.

Routing

A routing protocol decides along which path a packet travels from its source to its destination.

- **Routing table of node *v*:** maps every destination address to a neighbour of *v*.
- **Forwarding:** process of an intermidiate node receiving a packet and sending it to the next node.
- **Hierarchical addressing:** match a destination address to longest prefix in the routing table.
- **Default route:** where to forward packets when no specific entry is available in the routing table.

Routing table example

Routing table of v_1	
Destination	Next node
v_1	deliver
v_2	v_2
v_3	v_3
v_4	v_3

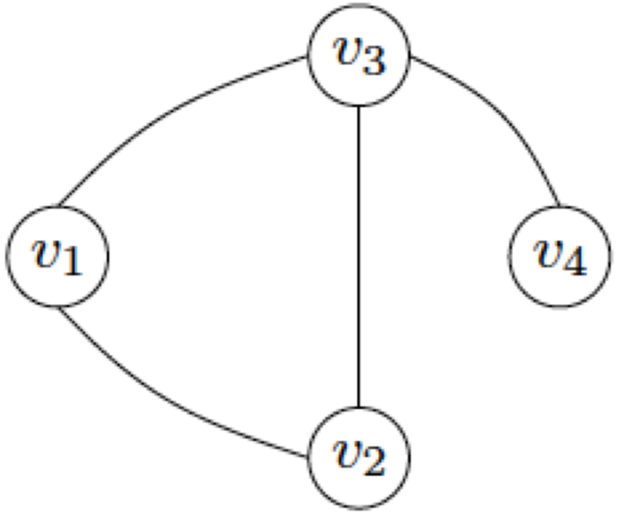


Table 2.24: A simplified routing table.

Link state (LS) routing

Algorithm 2.25 Link-State (LS) Routing Algorithm.

1: Given a weighted graph $G = (V, E, \omega)$
2: Learn $\omega(e)$ for every edge $e \in E$
3: Compute shortest paths to all nodes, e.g., by using Algorithm 2.16

LS routing allows:

- Nodes can discover changes in the network and update their routing tables.
- Other advanced features like multiple path routing.

Drawback: The nodes need to know the whole network. LS routing is feasible only at small scale within Autonomous Systems (AS)

- **Autonomous system:** a collection of nodes owned by a company.

Distance Vector (DV)

Algorithm 2.27 Distance-Vector (DV) Routing Algorithm.

Given a weighted graph $G = (V, E, \omega)$ and a node $u \in V$
Initialize a distance estimate $D(u \to v) = \omega(\{u, v\})$ for all neighbors $N(u)$ and $D(u \to w) = \infty$ for all other nodes
Send distance vector $\mathcal{D}(u) = \{D(u \to v) \mid v \in N(u)\}$ to all neighbors $N(u)$
while true **do**
    Upon receiving a distance vector $\mathcal{D}(v)$ from a neighbor v, update the distance estimate to all destinations accordingly
    if $D(u \to w)$ changed for any w **then**
        Send the updated distance vector $\mathcal{D}(u)$ to all neighbors
    end if
end while

Keeps and updates the distance between all nodes

- **Dsitributed:** nodes do not need the knowledge of the whole network.
- *Count to infinity problem.*

Intra-domain vs inter-domain

- **Intra-domain routing:** routing within an autonomous system.
  - E.g: RIP routing protocol

- **Inter-domain routing:** routing between different autonomous systems -> Border Gateway Protocol (**BGP**):
  - BGP solves count-to-infinity problem.
  - Allows **outbound policies:** a node can decide which traffic to attract
  - Allows **inbound policies:** a node can decide thorugh which neighbour to route.

BGP algorithm

Algorithm 2.29 Border Gateway Protocol (BGP)

1: Basically, BGP is a DV Routing Protocol, see Algorithm 2.27
2: BGP nodes send out annoucements about every 30 seconds
3: BGP nodes send reachability information: every node announces which address blocks (prefixes) it can reach
4: Instead of just distance, nodes announce the whole AS path to each prefix
5: The network is not weighted, an edge between two AS nodes costs 1.

Tunnels

Definition 2.30 (Tunnel). *The payload of a packet is a complete packet, with header and payload. In other words, we have two headers.*

- A tunnel embeds a packet inside another packet. Use cases:
  - Virtual Private Network (VPN).
  - Cross Firewalls.
  - Translate between IPv4 and IPv6.
  - Virtual circuit routing.