II. Algorithms of the Internet

1. Routing and Packet Forwarding: Shortest Paths and Autonomous Systems

Tina Schmidt

Ferienakademie im Sarntal 2008
FAU Erlangen-Nürnberg, TU München, Uni Stuttgart

September 2008

Overview

Overview

ARPANET

- Origin of the Internet: ARPANET = **A**dvanced **R**esearch **P**roject **A**gency **Net**work (60s)
  $\Rightarrow$ Barely known in public
- Today: Internet links millions of computers

Overview

Overview over the Layers of the Internet

Application Layer	Peer-to-Peer-networks, e.g. Telnet = Telecommunication Network FTP = File Transfer Protocol HTTP = Hypertext Transfer Protocol SMTP = Simple Mail Transfer Protocol
Transport Layer	TCP = Transmission Control Protocol UDP = User Datagram Protocol
Internet Layer	IP = Internet Protocol ICMP = Internet Control Message Protocol IGMP = Internet Group Management Protocol
Host-to-Network Layer or Link Layer	device drivers (e.g. Ethernet or Token Ring drivers)

Data Encapsulation

Figure: onion-like structure of data encapsulation

Data Encapsulation

Figure: onion-like structure of dataencapsulation

Data Encapsulation

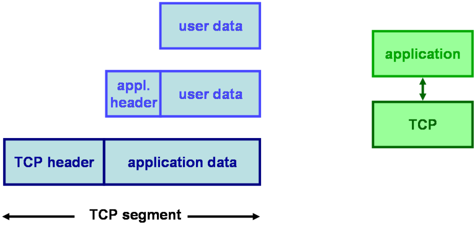


Figure: onion-like structure of dataencapsulation

Data Encapsulation

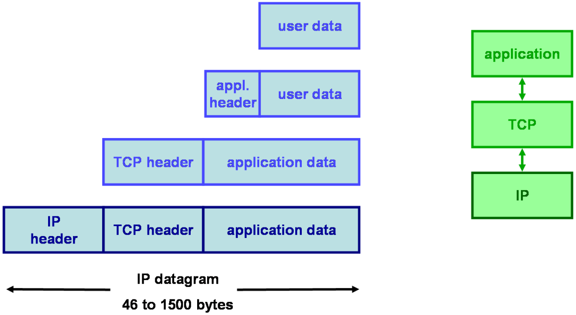


Figure: onion-like structure of dataencapsulation

Data Encapsulation

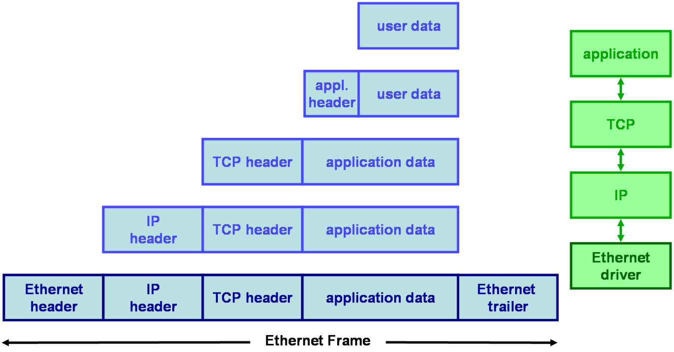


Figure: onion-like structure of dataencapsulation

Data Encapsulation

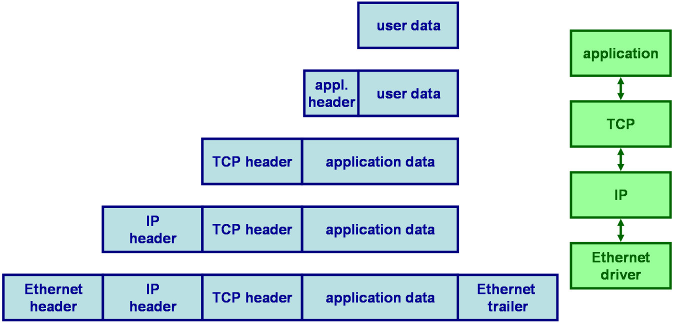


Figure: onion-like structure of dataencapsulation

IPv4

- Two subprotocols: ICMP and IGMP
- Task: transmit datagrams
- Unreliable (simple troubleshooting)
- Connectionless protocol

IPv4 Header

```
 IP Header Format
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Source Address                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Destination Address                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options                    |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure: IPv4 Header

IPv4 Addresses and DNS

- 32-bit-addresses, e.g. 65.114.4.69.
- Until 1993:
  - Host-ID: identification number for a computer (assigned by network administrator)
  - Net-ID: network address (assigned by the Internet Network Information Center)
- Since 1993: CIDR = Classless Inter-Domain Routing
  - Variable splitting of IP address
  - Subnetmask

    IP-Address: `10000100.11100110.10010110.11110011`

    Subnetmask: `11111111.11111111.11111111.00000000`
- DNS = Domain Name System

Overview

Packet Forwarding

- Routing tables to determine the next vertex
- IP refers to the interface of a computer
  - Destination address = interface's IP ⇒ Delivery to transport layer
  - Destination address = other IP of routing table ⇒ Packet is forwarded
  - None of the above cases: default gateway
- TTL entry
  - Forwarding decreases this entry by one
  - If the entry is 0: ICMP-message
  - Prevents infinte packet forwarding

Overview

Routing

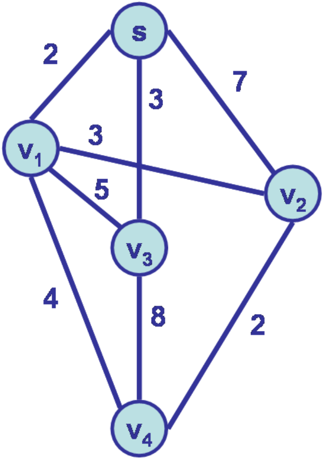
- Static routing: routing tables are created manually
  $\Rightarrow$Only for small networks
- Dynamical routing: routing-tables are created automatically
  protocols: RIP, OSPF
- Constant way

Let $G = (V, E)$ be a weighted graph containing a starting vertex $s \in V$ and weights $w_e \in \mathbb{R}_0^+$ for all edges $e \in E$. Searched are all paths from the starting vertex to all other vertices, which have the least sum of weights.

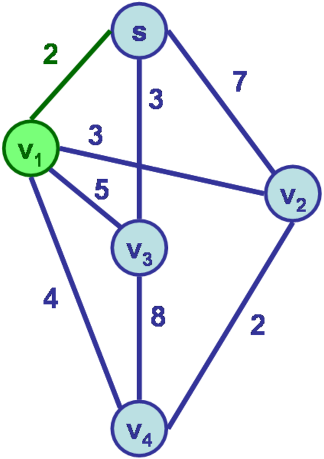
Overview

Dijktra’s Algorithm

Shortest Path Tree

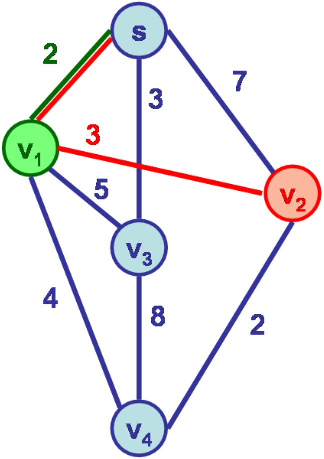


Dijktra's Algorithm
Shortest Path Tree

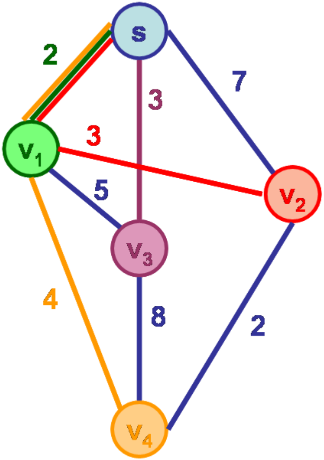


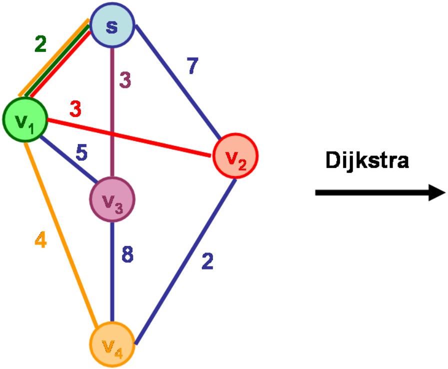
Dijktra's Algorithm

Shortest Path Tree

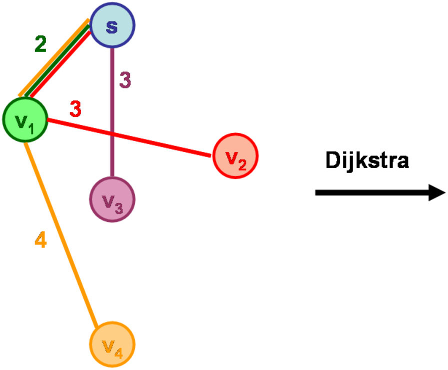


Dijktra's Algorithm
Shortest Path Tree





Vertex	Predecessor
v_1	s
v_2	v_1
v_3	s
v_4	v_1



Vertex	Predecessor
v_1	s
v_2	v_1
v_3	s
v_4	v_1

Dijkstra's Algorithm

- Result: Table of predecessors with edges of shortest-path-tree (Predecessor(u), u)
- Shortest-Path-Tree is built gradually
- Greedy algorithm
- One step: add the vertex with the shortest distance to the starting vertex s
- Similar to breadth-first search

Dijkstra's Algorithm

Input: $G = (V, E)$, $s \in V$, $w : E \rightarrow \mathbb{R}_0^+$

for all $v \in V$ **do**

  Distance[v] $\leftarrow \infty$

  Predecessor[v] $\leftarrow v$

Distance[s] $\leftarrow 0$

$S \leftarrow \emptyset$

$Q \leftarrow V$

while $Q \neq \emptyset$ **do**

  $u \leftarrow$ Element from Q with minimal value Distance[u]

  $S \leftarrow S \cup \{u\}$

  $Q \leftarrow Q \setminus \{u\}$

  for all $v \in V : (u, v) \in E$ **do**

    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**

      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$

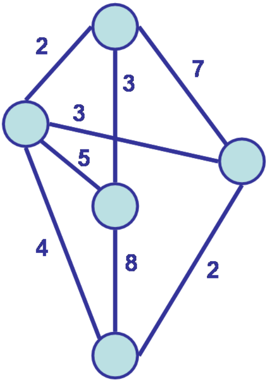
      Predecessor[v] $\leftarrow u$

return $\{(\text{Predecessor}(u), u) | u \in V$ and $u \neq \text{Predecessor}(u)\}$

Dijkstra's Algorithm
Example

Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$

for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm

Example

Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$

for all $v \in V$ **do**

  Distance$[v] \leftarrow \infty$

  Predecessor$[v] \leftarrow v$

Distance$[s] \leftarrow 0$

$S \leftarrow \emptyset$

$Q \leftarrow V$

while $Q \neq \emptyset$ **do**

  $u \leftarrow$ Element from Q with minimal value Distance$[u]$

  $S \leftarrow S \cup \{u\}$

  $Q \leftarrow Q \setminus \{u\}$

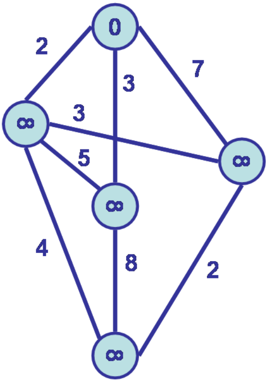
  for all $v \in V : (u, v) \in E$ **do**

    if Distance$[u] + w_{(u,v)} <$ Distance$[v]$ **then**

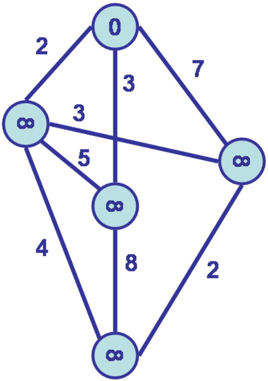
      Distance$[v] \leftarrow$ Distance$[u] + w_{(u,v)}$

      Predecessor$[v] \leftarrow u$

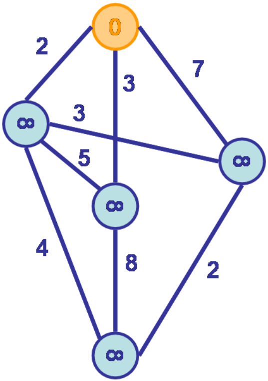
return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



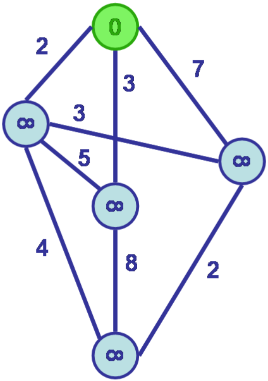
Input: $G = (V, E)$, $s \in V$, $w : E \rightarrow \mathbb{R}_0^+$
for all $v \in V$ **do**
    Distance[v] $\leftarrow \infty$
    Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
    $u \leftarrow$ Element from Q with minimal value Distance[u]
    $S \leftarrow S \cup \{u\}$
    $Q \leftarrow Q \setminus \{u\}$
    for all $v \in V : (u, v) \in E$ **do**
        if Distance[u] + $w_{(u,v)}$ < Distance[v] **then**
            Distance[v] $\leftarrow$ Distance[u] + $w_{(u,v)}$
            Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



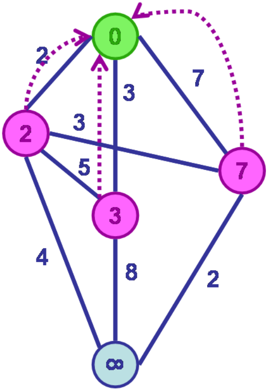
Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance$[v] \leftarrow \infty$
  Predecessor$[v] \leftarrow v$
Distance$[s] \leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance$[u]$
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance$[u] + w_{(u,v)} <$ Distance$[v]$ **then**
      Distance$[v] \leftarrow$ Distance$[u] + w_{(u,v)}$
      Predecessor$[v] \leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V$ and $u \neq \text{Predecessor}(u)\}$



Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance$[v] \leftarrow \infty$
  Predecessor$[v] \leftarrow v$
Distance$[s] \leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance$[u]$
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance$[u] + w_{(u,v)} <$ Distance$[v]$ **then**
      Distance$[v] \leftarrow$ Distance$[u] + w_{(u,v)}$
      Predecessor$[v] \leftarrow u$
return $\{($Predecessor$(u), u) | u \in V$ and $u \neq$ Predecessor$(u)\}$

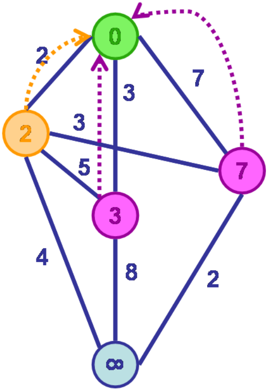


Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm
Example

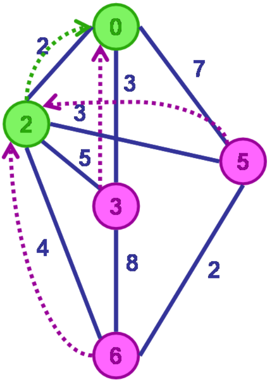
Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V$ and $u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm

Example

Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
    Distance[v] $\leftarrow \infty$
    Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
    $u \leftarrow$ Element from Q with minimal value Distance[u]
    $S \leftarrow S \cup \{u\}$
    $Q \leftarrow Q \setminus \{u\}$
    for all $v \in V : (u, v) \in E$ **do**
        if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
            Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
            Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$

for all $v \in V$ **do**

  Distance[v] $\leftarrow \infty$

  Predecessor[v] $\leftarrow v$

Distance[s] $\leftarrow 0$

$S \leftarrow \emptyset$

$Q \leftarrow V$

while $Q \neq \emptyset$ **do**

  $u \leftarrow$ Element from Q with minimal value Distance[u]

  $S \leftarrow S \cup \{u\}$

  $Q \leftarrow Q \setminus \{u\}$

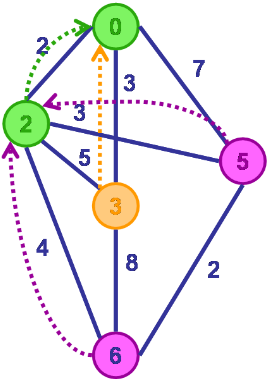
  for all $v \in V : (u, v) \in E$ **do**

    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**

      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$

      Predecessor[v] $\leftarrow u$

return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm

Example

Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$

for all $v \in V$ **do**

  Distance$[v] \leftarrow \infty$

  Predecessor$[v] \leftarrow v$

Distance$[s] \leftarrow 0$

$S \leftarrow \emptyset$

$Q \leftarrow V$

while $Q \neq \emptyset$ **do**

  $u \leftarrow$ Element from Q with minimal value Distance$[u]$

  $S \leftarrow S \cup \{u\}$

  $Q \leftarrow Q \setminus \{u\}$

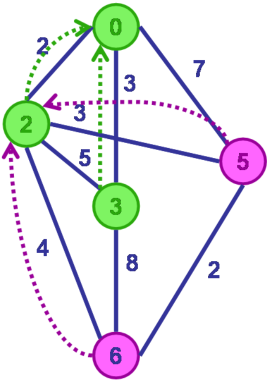
  for all $v \in V : (u, v) \in E$ **do**

    if Distance$[u] + w_{(u,v)} <$ Distance$[v]$ **then**

      Distance$[v] \leftarrow$ Distance$[u] + w_{(u,v)}$

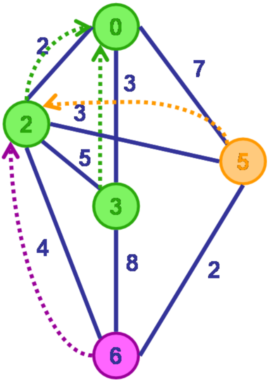
      Predecessor$[v] \leftarrow u$

return $\{(\text{Predecessor}(u), u) | u \in V$ and $u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm
Example

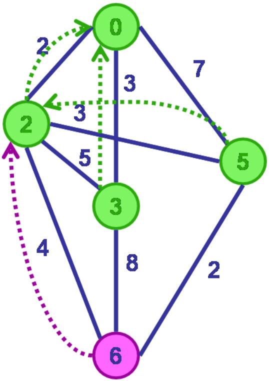
Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V$ and $u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm

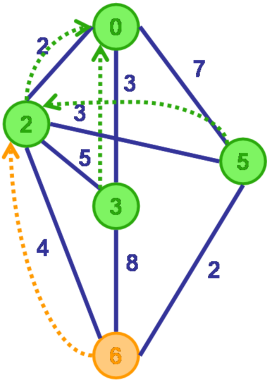
Example

Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{$(Predecessor(u), u)$|u \in V$ and $u \neq$ Predecessor(u)$\}$

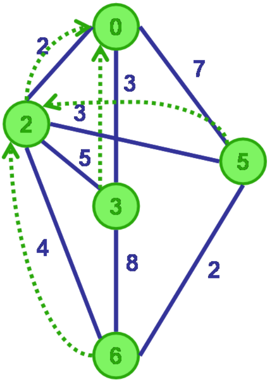


Dijkstra's Algorithm
Example

Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V$ and $u \neq \text{Predecessor}(u)\}$



Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$
for all $v \in V$ **do**
  Distance[v] $\leftarrow \infty$
  Predecessor[v] $\leftarrow v$
Distance[s] $\leftarrow 0$
$S \leftarrow \emptyset$
$Q \leftarrow V$
while $Q \neq \emptyset$ **do**
  $u \leftarrow$ Element from Q with minimal value Distance[u]
  $S \leftarrow S \cup \{u\}$
  $Q \leftarrow Q \setminus \{u\}$
  for all $v \in V : (u, v) \in E$ **do**
    if Distance[u] $+ w_{(u,v)} <$ Distance[v] **then**
      Distance[v] $\leftarrow$ Distance[u] $+ w_{(u,v)}$
      Predecessor[v] $\leftarrow u$
return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



Input: $G = (V, E)$, $s \in V$, $w : E \to \mathbb{R}_0^+$

for all $v \in V$ **do**

  Distance$[v] \leftarrow \infty$

  Predecessor$[v] \leftarrow v$

Distance$[s] \leftarrow 0$

$S \leftarrow \emptyset$

$Q \leftarrow V$

while $Q \neq \emptyset$ **do**

  $u \leftarrow$ Element from Q with minimal value Distance$[u]$

  $S \leftarrow S \cup \{u\}$

  $Q \leftarrow Q \setminus \{u\}$

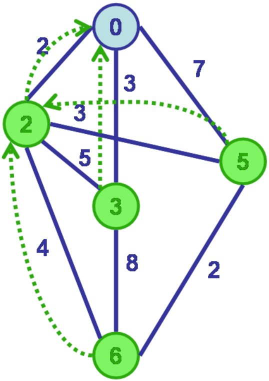
  for all $v \in V : (u, v) \in E$ **do**

    if Distance$[u] + w_{(u,v)} <$ Distance$[v]$ **then**

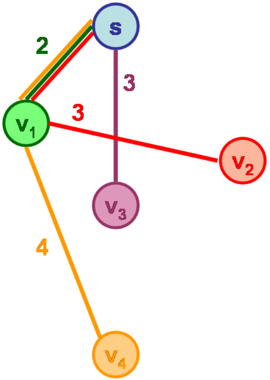
      Distance$[v] \leftarrow$ Distance$[u] + w_{(u,v)}$

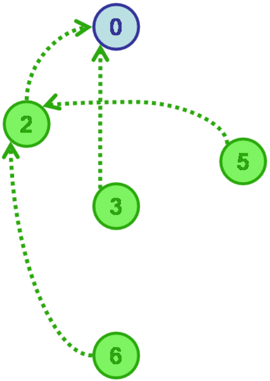
      Predecessor$[v] \leftarrow u$

return $\{(\text{Predecessor}(u), u) | u \in V \text{ and } u \neq \text{Predecessor}(u)\}$



Dijkstra's Algorithm
Example





Overview

Link-State-Routing

- Practical realization of Dijkstra’s Algorithm
- Every vertex needs to know all connections of the network
- Broadcast: spreading connection information through the network
- Low costs for communication in a running state

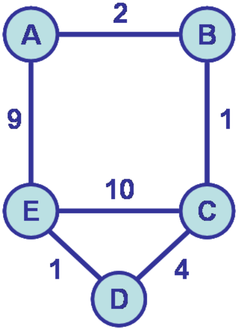
Distance-Vector-Routing

Distance Table for A

from A	over B	E	Routing Table Entry
to B	2	15	B
C	3	19	B
D	7	10	B
E	13	9	E

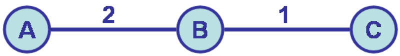
Distance Table for C

from C	over B	D	E	Routing Table Entry
to A	3	14	18	B
B	1	9	16	B
D	6	4	11	D
E	7	5	10	D



Distance-Vector-Routing

Count-to-Infinity-Problem



Distance Table for A

from A	over B	Routing Table Entry
to B	2	B
C	3	B

Distance Table for B

from B	over A	over C	Routing Table Entry
to A	2	4	A
C	5	1	C

Distance Table for A

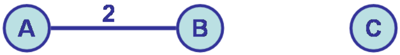
from A	over B	Routing Table Entry
to B	2	B
C	3	B

Distance Table for B

from B	over A	C	Routing Table Entry
to A	2	4	A
C	5	1	C

from A	over B	Routing Table Entry
to B	2	B
C	3	B

from B	over A	C	Routing Table Entry
to A	2	-	A
C	5	-	A



Distance Table for A

from A	over B	Routing Table Entry
to B	2	B
C	3	B

Distance Table for B

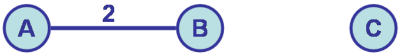
from B	over A	over C	Routing Table Entry
to A	2	-	A
C	5	-	A

from A	over B	Routing Table Entry
to B	2	B
C	7	B

from B	over A	over C	Routing Table Entry
to A	2	-	A
C	5	-	A

Distance-Vector-Routing

Count-to-Infinity-Problem



Distance Table for A

from A	over B	Routing Table Entry
to B	2	B
C	7	B

Distance Table for B

from B	over A	over C	Routing Table Entry
to A	2	-	A
C	5	-	A

from A	over B	Routing Table Entry
to B	2	B
C	7	B

from B	over A	over C	Routing Table Entry
to A	2	-	A
C	9	-	A

Distance-Vector-Routing

Solutions for the Count-to-Infinity-Problem

- Poisoned Reverse
  - Spread information about a lost connection
  - Lost connection: distance is set infinity
- Split Horizon
  - Cannot ask router from which it had learned about the connection
- Problem: both algorithms do not work reliably

Overview

Autonomous Systems

- Impossible to generate routing-tables for all existing routers (too many)
- Implementation of a hierarchy: partition into autonomous systems
- Boundary routers connect the different autonomous systems
- Path for a packet:
  - Intra-AS-Routing
  - Inter-AS-Routing

Overview

- Uses Distance-Vector-Routing algorithm
- New connected computer
    - Only knows distances to neighbors
    - Asks for routing-tables of neighbors
- Routing tables are exchanged regularly
- Advertisement: offer of new routers
- Problems:
    - Routing information does not spread fast
    - Count-to-Infinity Problem

Intra-AS-Routing

Open Shortest Path First = OSPF

- LSD = Link-State-Database: List of all neighbored routers
- Dijkstra's algorithm used to calculate the shortest path
- Larger networks are devided:
  - Local area
  - Backbone
- Today used for IPv4 and IPv6

Intra-AS-Routing

Interior-Gateway-Routing-Protocol = IGRP

- Based on Distance-Vector-Routing Protocol
- RIP: 15 network vertices away ⇒ network is called out of reach
  IGRP: increased to 255
- Better scalability (metric of costs)
- Routing loops are avoided

Overview

Inter-AS-Routing

- BGP = Border Gateway Protcol
- Table with entries about subnets or entries with IP-address-prefixes
- Based on Path-Vector-Protocol

Other Services of IP

- ICMP = Internet Control Message Protocol
- Ping
  - ICMP-Echo-Request-packet
  - ICMP-Echo-Reply-packet
- Traceroute
  - Calculates the route inbetween two computers
  - Measures time for delivery

Conclusion

- The main protocol of the Internet Layer, IP, is a connectionless and unreliable datagram delivery service.
- Dijkstra's Algorithm is used to solve the Shortest Path Problem for Routing.
- Since the Internet contains millions of computers, it is devided into autonomous systems.

- Outlook
  - There was no good solution to the Count-to-Infinity Problem presented.
  - What happens to packets as they pass through other layers?