CSCD 330
Network Programming
Spring 2017

Lecture 16
Network Layer
Routing Protocols

Reading: Chapter 4

Network Layer

- 4. 1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6

Skip

- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF

Today

Routing Introduction

- Routing packets occurs
  - Between source router and destination router
- What are some problems in routing, algorithm should address?
  - Minimal cost – how to determine that
  - Changing Conditions
    - How to handle ....
    - Congestion, routes disappear and others appear over time
  - Priority of some routes over others, not related to cost

Routing Introduction

- Routing most important function of network layer and …
  - One of the most important functions of whole network
- Routing protocols determine how efficiently traffic moves through network
  - Bad routing protocols hinder networks no matter how ample network resources
- Look at general algorithms and then some specific protocols…

Global Routing

- As we already said ….
- Each corporate entity is responsible for routing within their IP space
- So, they aggregate routers into regions,

Autonomous systems (AS)

- Routers in same AS run same routing protocol

Intra-AS routing protocol

- Routers in different AS's can run different protocols

Inter-AS routing protocol differs from Intra-AS

Interdomain and Intradomain Routing

Intradomain Routing

- Ignores Internet outside AS
- Protocols for Intradomain routing,
- Routing within an AS

Interior Gateway Protocols or **IGP's**.

Routing protocols

- RIP (simple, old)
- OSPF (better)

Interdomain Routing

- Routing between AS's
- Assumes Internet is collection of interconnected AS's
- Normally, dedicated router(s) in each AS handle interdomain traffic

Exterior Gateway Protocols or EGP's.

Routing protocols:

- EGP (older)
- BGP (more recent)

Routing Algorithms

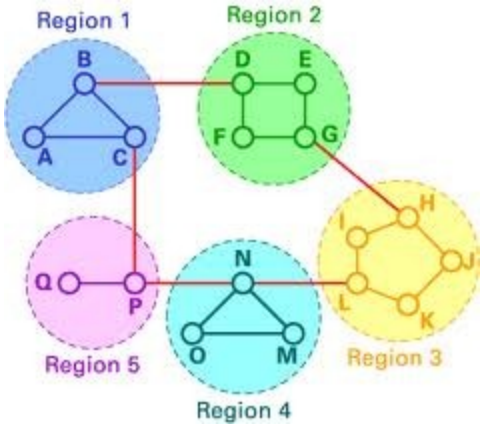
Purpose of Routing Protocols

- Routing protocols are used to exchange routing information between routers.
- A routing protocol is: ***"Set of processes, algorithms, and messages used to exchange routing information and populate the routing table with the best paths"***

- The purpose of dynamic routing protocols includes:
- Discovery of remote networks
- Maintaining up-to-date routing information
- Choosing the best path to destination networks
- Ability to find new best path if current path is no longer available

Classifying Routing Algorithms

- So, … how do you classify routing algorithms?
- Are they pretty much the same or are some better than others?
- Turns out … some are better
- Classifying them lets us describe their characteristics



Routing Algorithm Classification

Two basic types of routing algorithms

1. Decentralized Routing Algorithm, Example: Distance vector
   No single node has complete knowledge,
   Knows its own neighbors and
   ONLY exchanges routing info with neighbors
   Gradually exchanges information and updates routes to arrive at least cost path

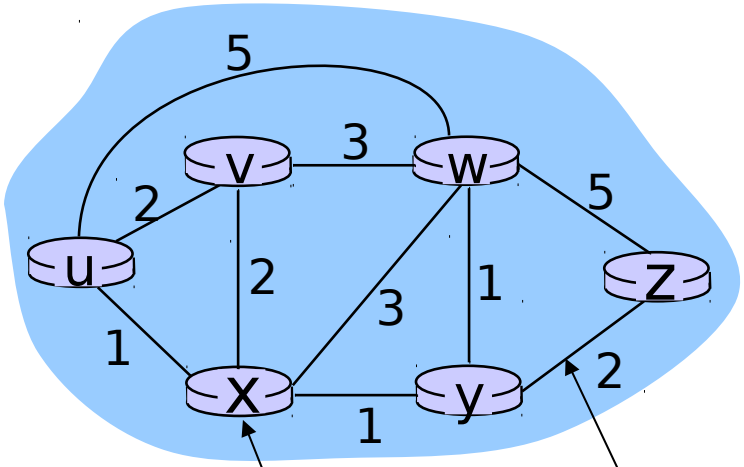
2. Global Routing Algorithm, Example: Link state
   Send everyone your distance to your neighbors
   Once router has all information from updates,
   Router computes shortest path,
   Assume everyone eventually has global knowledge

Link State vs. Distance Vector

- What are the differences? Which is better?
- **Link-state algorithms** flood routing information to all nodes in Internetwork (AS)
  - Each router, sends only portion of routing table that describes state of its own links
- **Distance- vector algorithms** each router sends all or some of its routing table, but only to its neighbors
- **Summary,**
  - Link- state algorithms send small updates everywhere
  - Distance- vector algorithms send larger updates only to neighboring routers

Graph abstraction

One way to view network is a graph

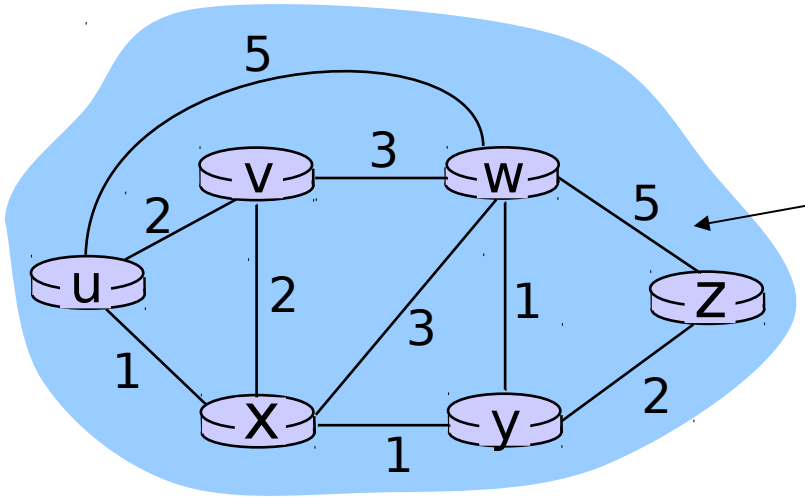


Graph: G = (N,E)

N = nodes = routers = { u, v, w, x, y, z }

E = edges = links { (u,v), (u,x), (v,x),
(v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

Graph Abstraction: Costs



- c(x,x') = cost of link (x,x')

  - e.g., c(w,z) = 5

- Cost could be number of hops between nodes, or related to bandwidth, or related to congestion

Cost of path $(x_1, x_2, x_3, \ldots, x_p) = c(x_1,x_2) + c(x_2,x_3) + \ldots + c(x_{p-1},x_p)$

Question: What's the least-cost path between u and z ?

Link-State Routing Algorithm

Notation

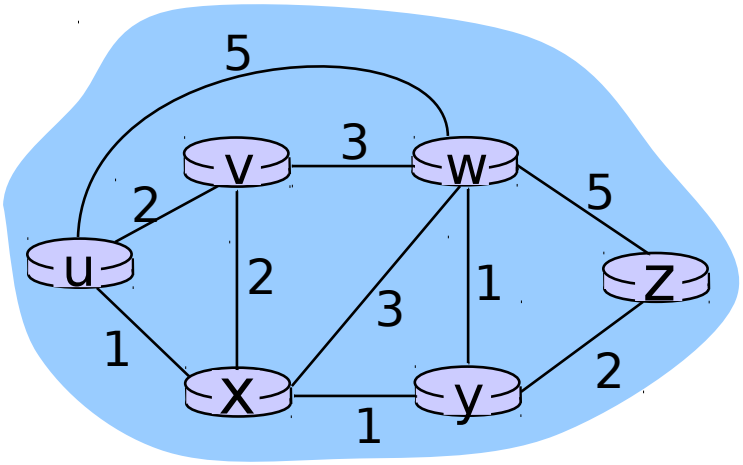
- c(x,y): link cost from node x to y;

  Is ∞, infinity to begin with, if not direct neighbors

- D(v): current value of cost of path from source to destination v
- p(v): predecessor node along path from source to v
- N': set of nodes whose least cost path definitively known, want to keep adding to N' until all nodes are added

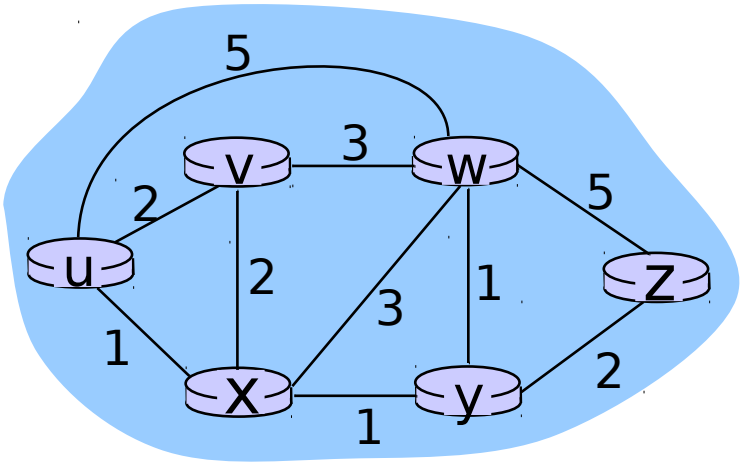
Dijsktra's Algorithm – u to z

```
1  Initialization
2    N' = {u}   //Start node
3    for all nodes v
4      if v adjacent to u
5          then D(v) = c(u,v)
6      else D(v) = ∞
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 to w and not in N' :
12       D(v) = min( D(v), D(w) + c(w,v) )
13    /* new cost to v is either old cost to v or known
14     shortest path cost to w plus cost from w to v */
15  until all nodes in N'
```



Dijkstra's algorithm: example

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					



Loop

find w not in N' such that D(w) is a minimum

add w to N'

update D(v) for all v adjacent to w and not in N'

D(v) = min(D(v), D(w) + c(w,v))

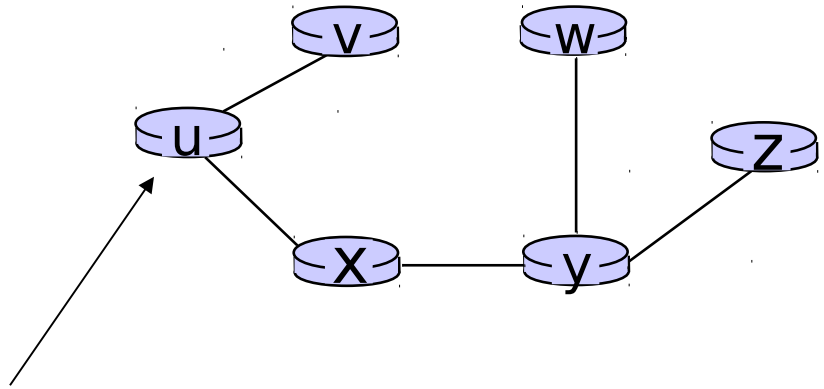
until all nodes in N'

End of Link State

- At end of link state algorithm, you have
  - Each node,
    - Its predecessor along least-cost path
    - From source node
      - Then, have its predecessor and so on ...
      - Can construct entire path from source to all its destinations
  - Forwarding table in a node,
    - Constructed by storing for each destination, next hop node on least cost path

Dijkstra's Algorithm

Resulting shortest-path tree from u



Resulting forwarding table in u

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

In Practice Link State Algorithm

- In practice, network topology known by each node (router)
- Broadcast link state packets to all other nodes in network
- Next ... look at Distance Vector Algorithm

Distance Vector Algorithm

- Want to go from source **x** to destination **y**
  - Let $d_x(y)$ be least-cost
    - From node x to node y
- Least costs are related by the Bellman-Ford equation

$$D_x(y) = \min_v\{c(x,v) + D_v(y)\}$$

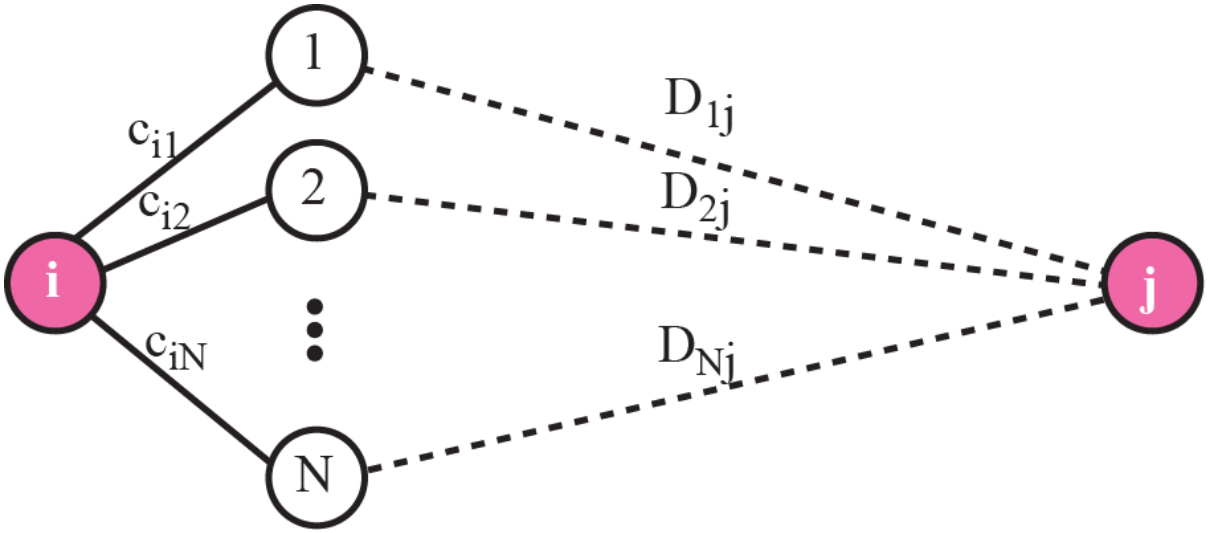
$\min_v$ in equation is taken over all of x's neighbors

- So, after traveling from x to v,
  - Take least cost path from v to y, path cost will be

$c(x,v) + d_v(y)$ then take minimum of this for all neighbors, v

Bellman-Ford algorithm

$$D_{ij} = \text{minimum } \{(c_{i1} + D_{1j}), (c_{i2} + D_{2j}), \ldots (c_{iN} + D_{Nj})$$



Legend

D_{ij} Shortest distance between i and j
c_{ij} Cost between i and j
N Number of nodes

Distance Vector Algorithm

Basic idea

- Each node periodically sends its own Distance Vector (DV) estimate to its neighbors
  - When a node x receives new DV estimate from neighbor
  - It updates its own DV using Bellman-Ford equation

Distance Vector Algorithm

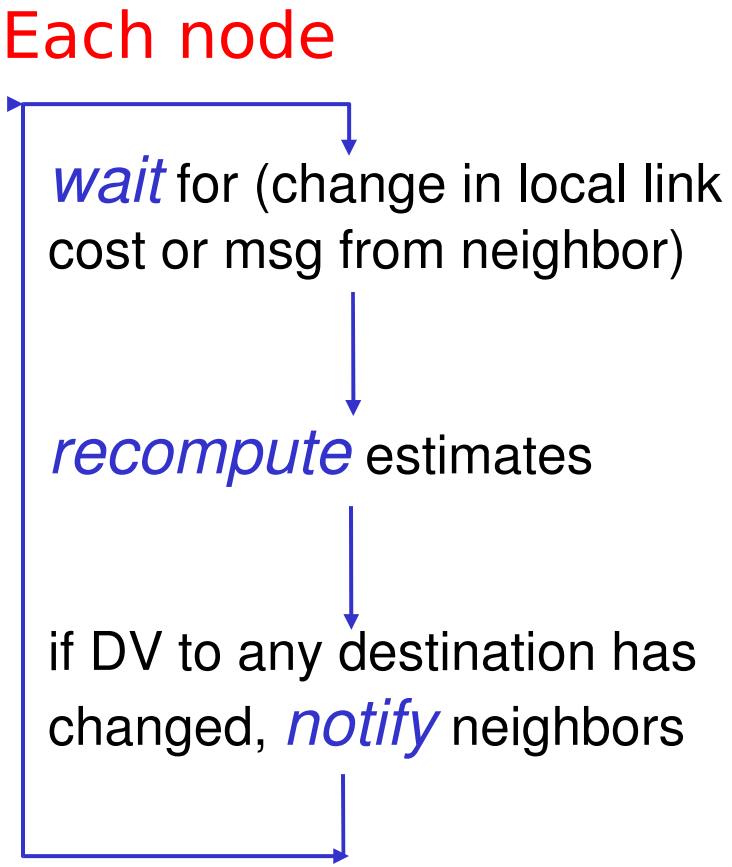
Iterative, Asynchronous

Each local iteration caused by:

- Local link cost change
- DV update message from neighbor

Distributed

- Each node notifies neighbors when its DV changes
  - Neighbors then notify their neighbors if necessary



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

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

node x table

cost to

from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

cost to

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

cost to

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

node y table

cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

cost to

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

cost to

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

node z table

cost to

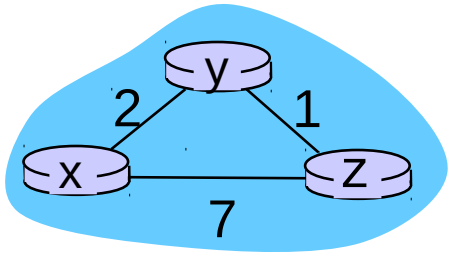
from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

cost to

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

cost to

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

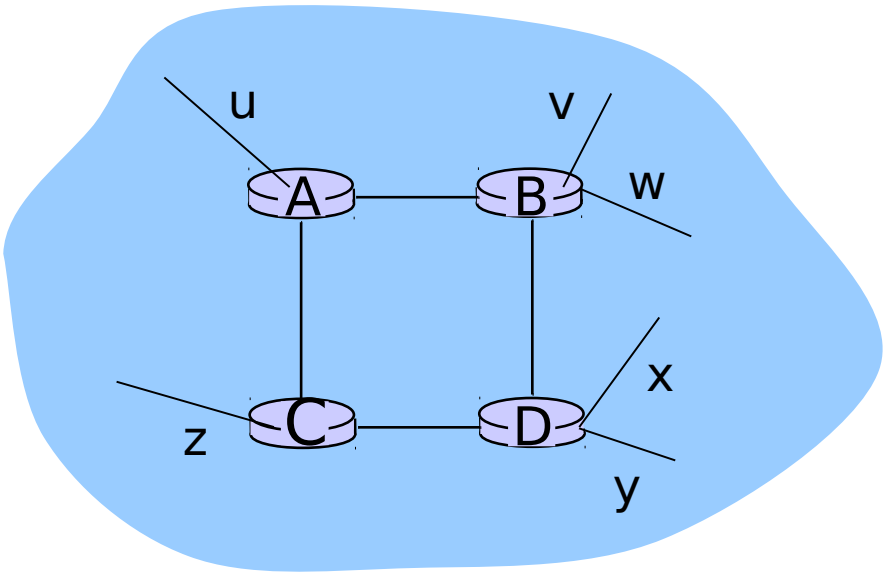


time

Routing Information Protocol
As an Example of Distance Vector

RIP (Routing Information Protocol)

- Distance Vector Algorithm
- Included in BSD-UNIX Distribution in 1982
  - Distributed as Unix daemon program **routed**
- Distance metric: # of hops (max = 15 hops)



From router A to subsets

destination	hops
u	1
v	2
w	2
x	3
y	3
z	2

RIP

- Still used today
  - Distance-Vector protocol
    - Routing tables exchanged between neighbours
      - Updates every 30 to 60 seconds or
      - Whenever routing table changes
  - After certain amount of time, network has **converged**, information does not change
  - Routers just provide stay-alive information
- Uses UDP for route updates
  - Port 520

RIP - Routing Information Protocol

- Straightforward implementation of Distance Vector Routing
  - Maximum hop count 15, with “16” equal to ∞
  - Routes timeout (set to 16) after 3 minutes if not updated, assumed “dead”
  - RIP does not allow router to know exact topology of an Internetwork
    - Since only talking with neighbors

RIP Link Failure and Recovery

A Routing Failure Happens

1. New advertisements sent to neighbors
2. Neighbors in turn send out new advertisements (if tables changed)
3. Link failure propagates to entire network

- Works well if no Loops in Network
  - Problems with potential loops in network
  - See next slides ...

RIP Count-to-Infinity Problem

- Classic distance vector convergence problem is known as count-to-infinity problem
  1. Asynchronous announcement scheme
  2. Passing routes back to originating router

- RIP adds routes to routing table, keeps only best route
    - Update lower cost route with higher cost route **only if** announced by same source as current lower cost route

RIP Count-to-Infinity Problem

- Count-to-infinity
  - One reason why maximum hop count of RIP networks set 15 (16 for unreachable)
    - Higher maximum hop count makes convergence time longer when count-to-infinity occurs
    - Route from Router 1 to Network 3 is through Router 2 and route from Router 2 to Network 3 is through Router 1
    - A **routing loop** exists between Router 1 and Router 2 for Network 3
- How do we fix this?

RIP Count-to-Infinity Problem

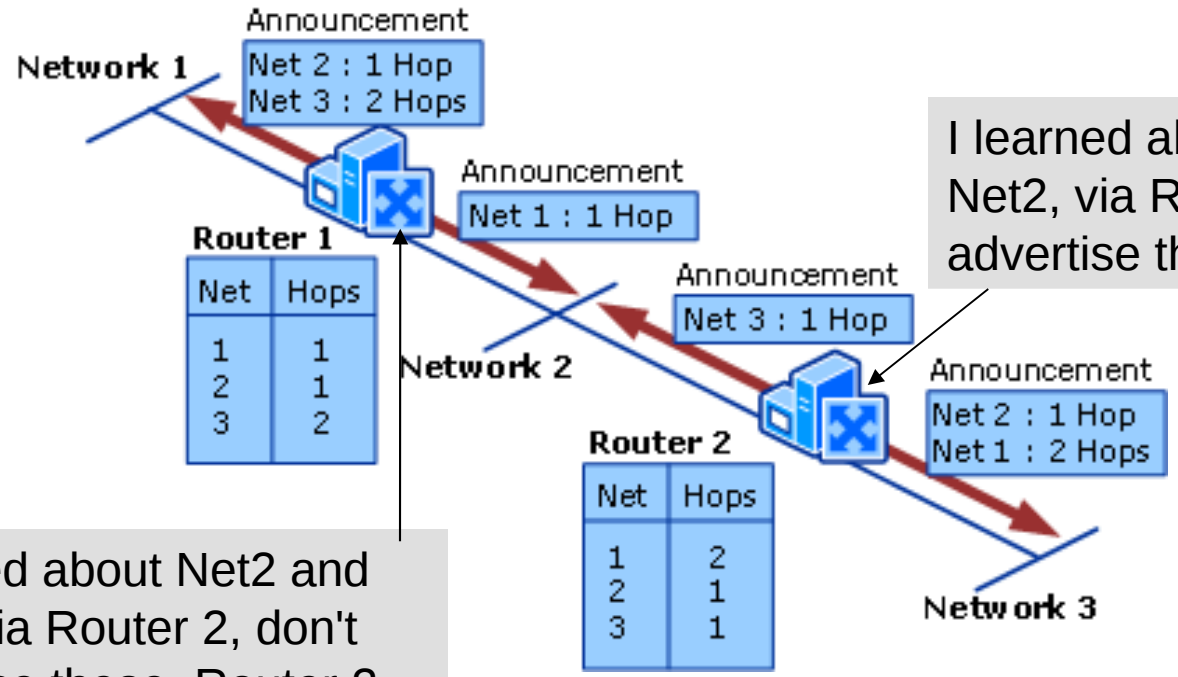
- Solution …
- Split Horizon Rule "do not send information about a route back in the direction it came from"
  - Split horizon reduces convergence time
  - Does not allow routers to advertise networks in direction from which those networks were learned

  - **Only** information sent in RIP announcements
    - Networks beyond neighboring router in opposite direction.
  - Networks learned from neighboring router are not included

  - Split Horizon
    - Eliminates count-to-infinity and routing loops during convergence

Split Horizon Rule

Don't announce routes to routers we got them from



I learned about Net1 and Net2, via Router 1, don't advertise these, Router 1

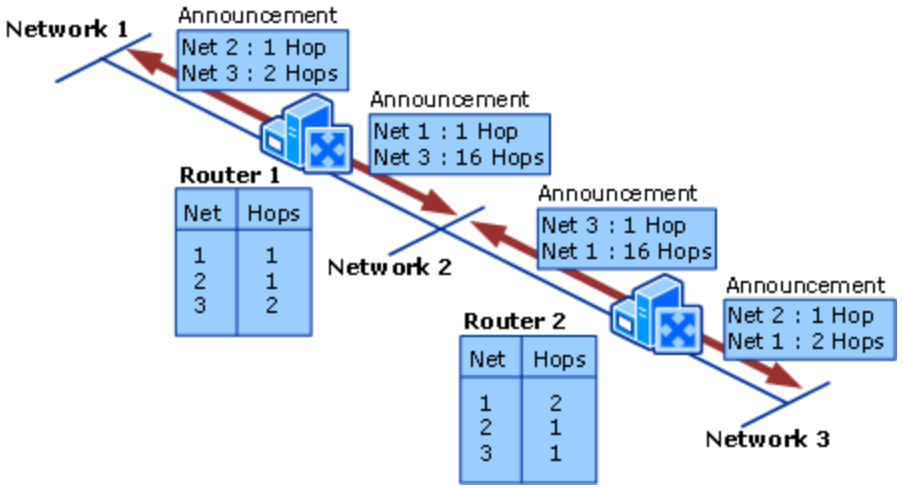
I learned about Net2 and Net3, via Router 2, don't advertise these, Router 2

Split Horizon with Poison Reverse

- Split horizon with poison reverse
  - Routing updates that indicate a network or a subnet is unreachable,
  - Simple split horizon omits routes learned from one neighbor in updates sent to that neighbor,
  - Split horizon with poison reverse includes such routes in updates

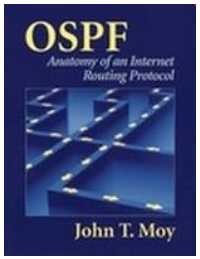
    but announces them

    with a hop count of 16



Open Shortest Path First

OSPF (Open Shortest Path First)



- Open - publicly available
- Uses Link State algorithm
  - Route computed with Dijkstra's algorithm

- Advertisements sent to entire AS (via flooding)
  - Carried in OSPF messages directly over IP (rather than TCP or UDP)

- Supposed to be successor to RIP ...

OSPF

- OSPF
- The OSPF routing protocol is the most important link state routing protocol on the Internet
- The complexity of OSPF is significant

- History:
  - 1989: RFC 1131 OSPF Version 1
  - 1991: RFC1247 OSPF Version 2
  - 1994: RFC 1583 OSPF Version 2 (revised)
  - 1997: RFC 2178 OSPF Version 2 (revised)
  - 1998: RFC 2328 OSPF Version 2 (current version)

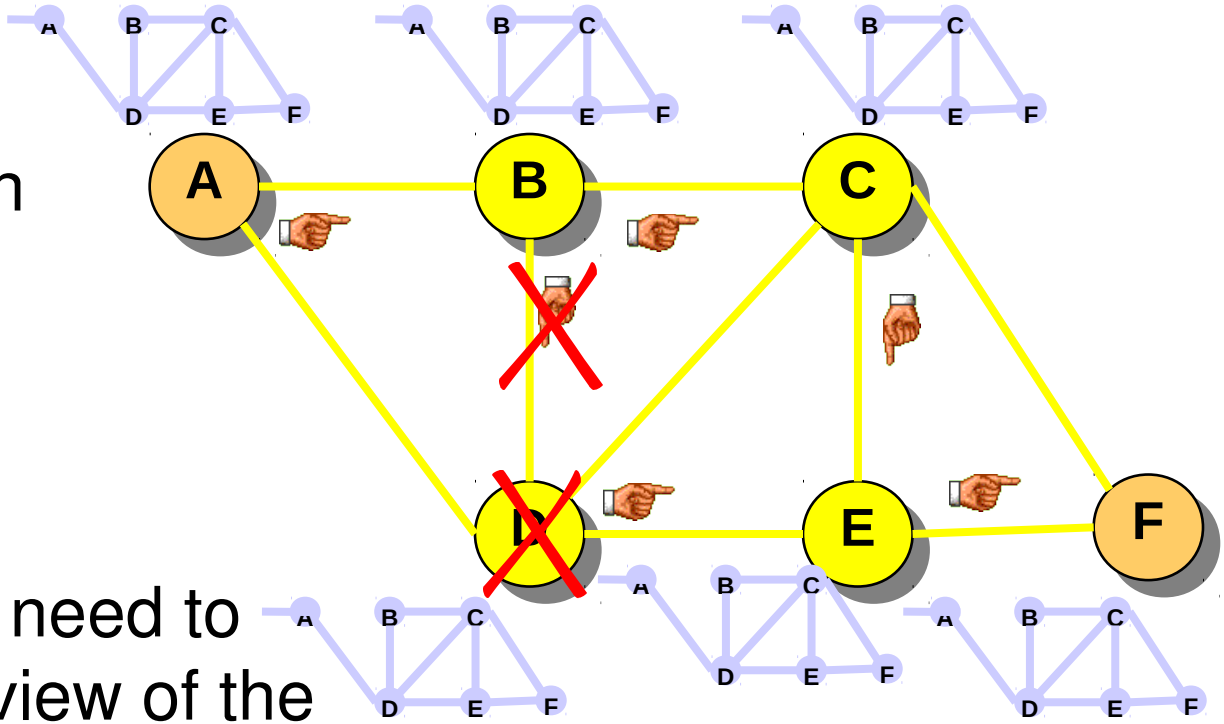
OSPF vs. RIP

- Contrast to RIP,
  - Where a RIP router
    - **Tells all its neighbors about the world,**
  - OSPF routers
    - **Tells the world about the neighbors**

- RIP sends entire routing table every 30 seconds
- OSPF sends its link state information every 30 minutes
- OSPF routers also send each other small update messages
  - Whenever they detect a change in network
  - Such as .... a failure or a new link

Distance Vector vs. Link State Routing

- In link state routing, each node has a complete map of the topology
- If a node fails, each node can calculate the new route
- Difficulty All nodes need to have a consistent view of the network



Link State Routing: Properties

- Each node requires complete topology information
- Link state information must be flooded to all nodes
- Guaranteed to converge!!!

Link State Routing: The Basics

1. Each router establishes relationship ("adjacency") with its neighbors
2. Each router generates Link State Advertisements (LSA's) *and* distributes them to all routers

   LSA = (link id, state of the link, cost, neighbors of the link)

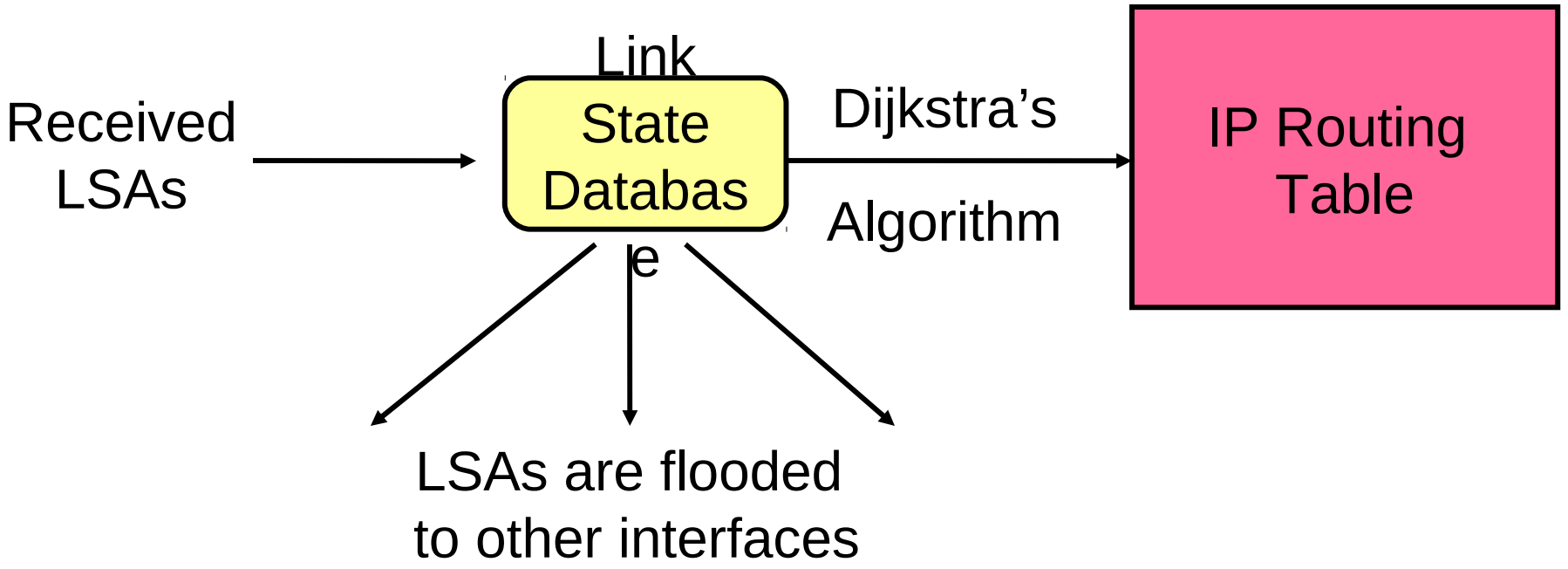
3. Each router maintains database of all received LSAs

   topological database or link state database

   describes network as graph with weighted edges
4. Each router uses link state database to run shortest path algorithm, Dijikstra's algorithm,

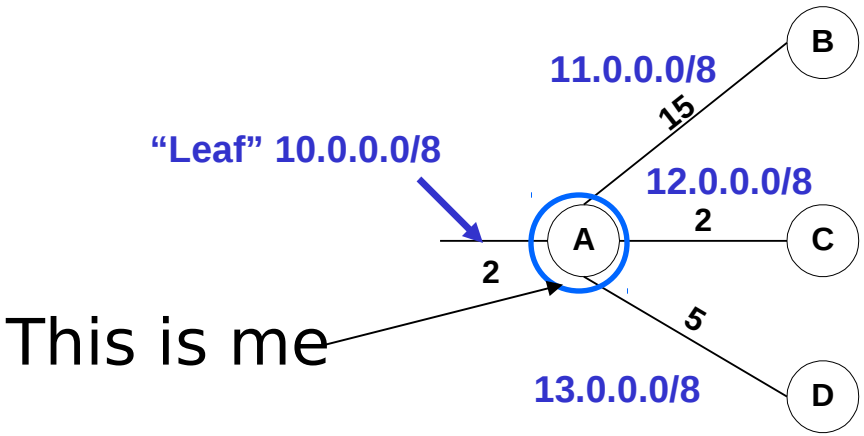
   Result is shortest path to each network
5. Results create the routing table

Operation of Link State Routing protocol



Link State Example
Build Complete Network Graph

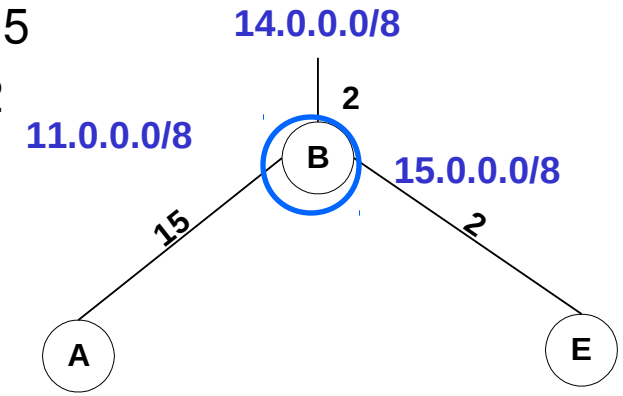
- You are **RouterA** and you have exchanged "Hellos" with:
  - RouterB on your network 11.0.0.0/8 with a cost of 15,
  - RouterC on your network 12.0.0.0/8 with a cost of 2
  - RouterD on your network 13.0.0.0/8 with a cost of 5
  - Have a "leaf" network 10.0.0.0/8 with a cost of 2
- This is your link-state information, which you will flood to all other routers.
- All other routers will also flood their link state information
- Comment- OSPF: Only within an AS area



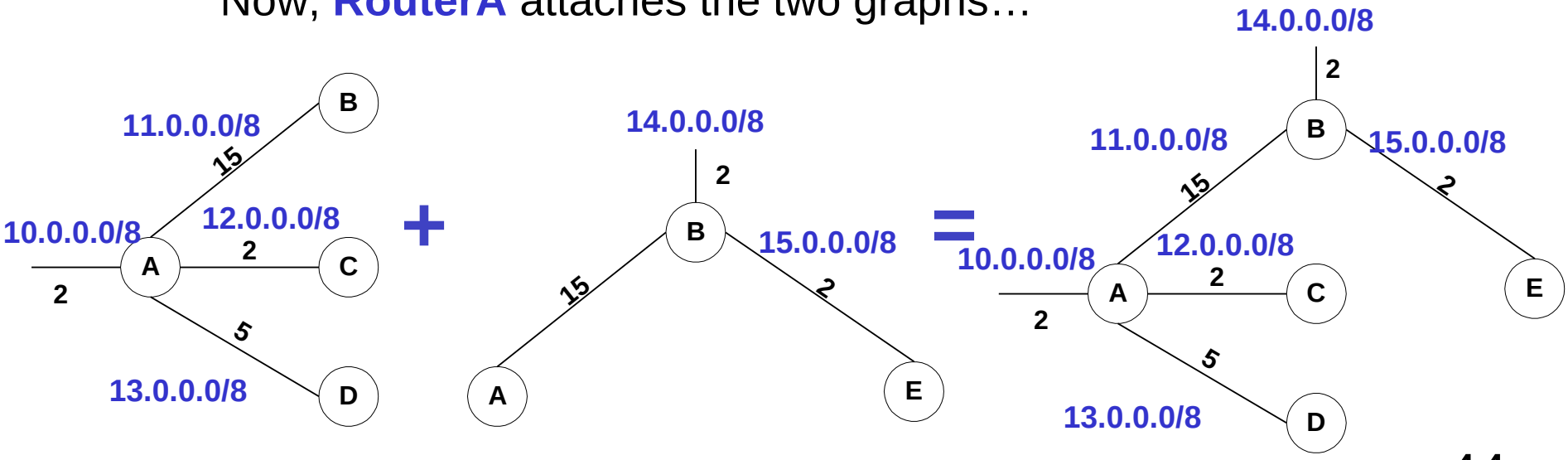
Link State information from Router B

We get following link-state information from **Router B**:

- Connected to RouterA on network 11.0.0.0/8, cost of 15
- Connected to RouterE on network 15.0.0.0/8, cost of 2
- Have a “leaf” network 14.0.0.0/8, cost of 2



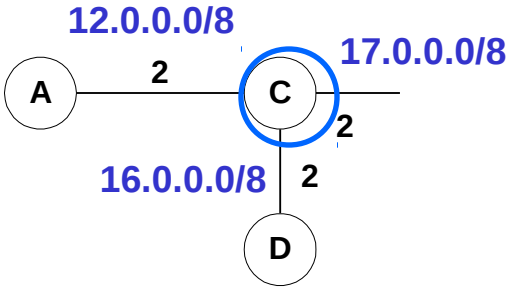
Now, **RouterA** attaches the two graphs…

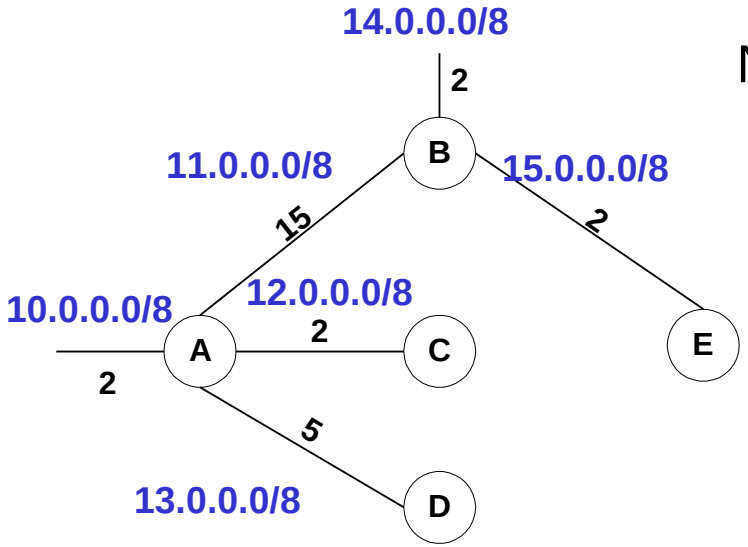


Link State information from Router C

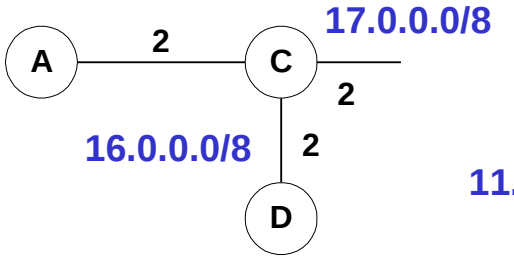
We get the following link-state information from **Router C**

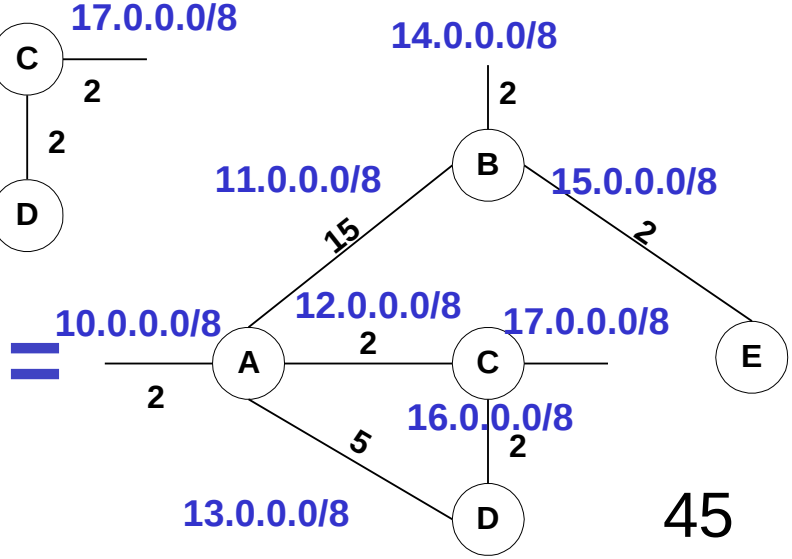
- Connected to RouterA on network 12.0.0.0/8, cost of 2
- Connected to RouterD on network 16.0.0.0/8, cost of 2
- Have a “leaf” network 17.0.0.0/8, cost of 2





Now, **Router A** attaches the two graphs…

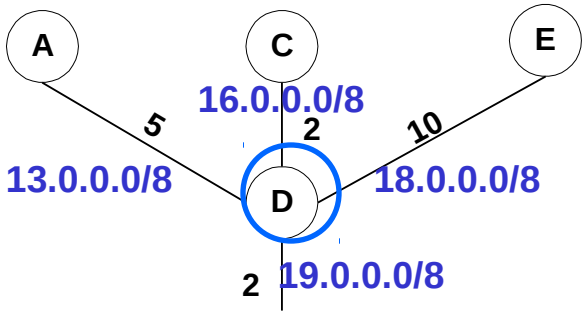




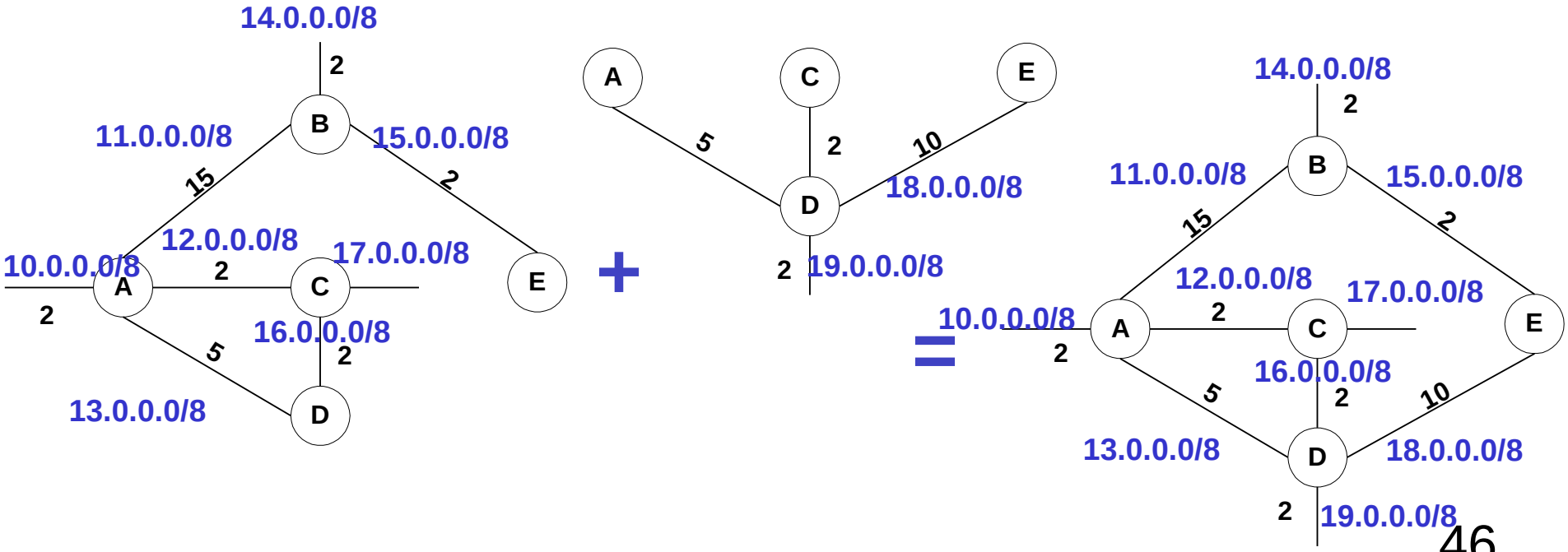
Link State information from Router D

We get the following link-state information from **Router D**:

- Connected to RouterA on network 13.0.0.0/8, cost of 5
- Connected to RouterC on network 16.0.0.0/8, cost of 2
- Connected to RouterE on network 18.0.0.0/8, cost of 2
- Have a “leaf” network 19.0.0.0/8, cost of 2



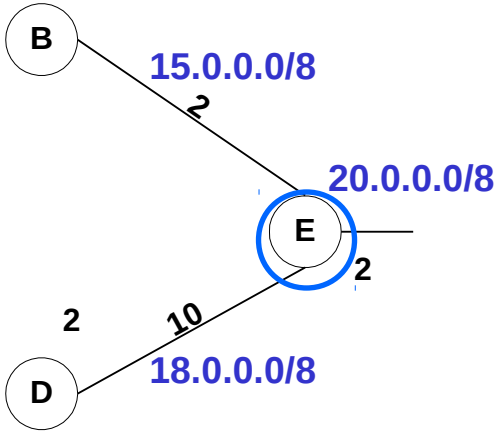
Now, **Router A** attaches the two graphs…



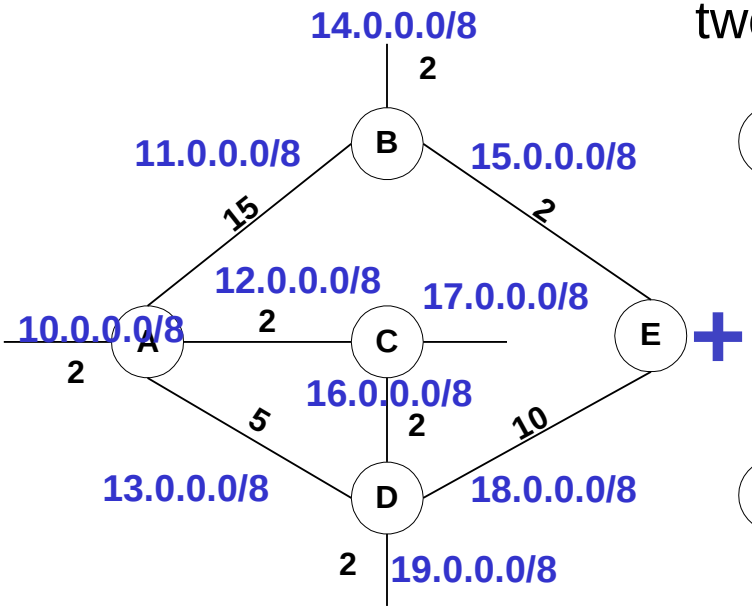
Link State information from Router E

We get the following link-state information from **RouterE**:

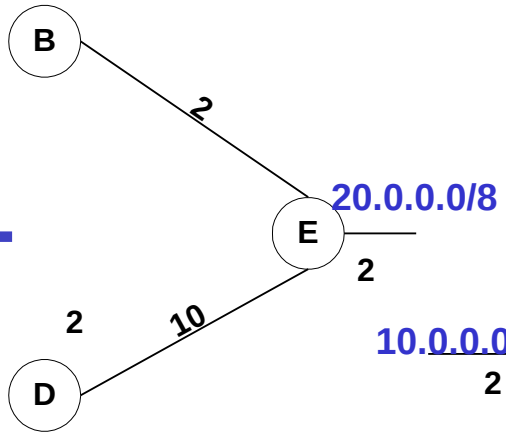
- Connected to RouterB on network 15.0.0.0/8, cost of 2
- Connected to RouterD on network 18.0.0.0/8, cost of 10
- Have a “leaf” network 20.0.0.0/8, cost of 2

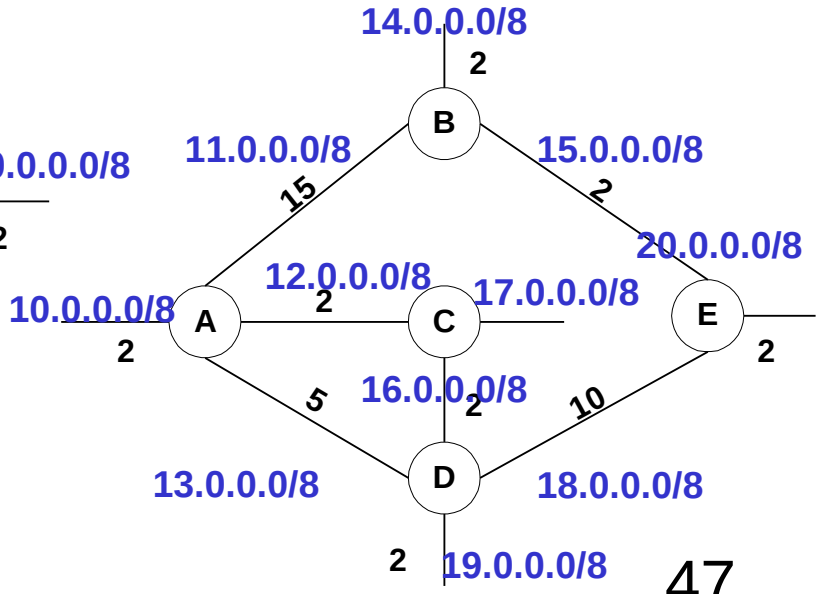


Now, **RouterA** attaches the two graphs…



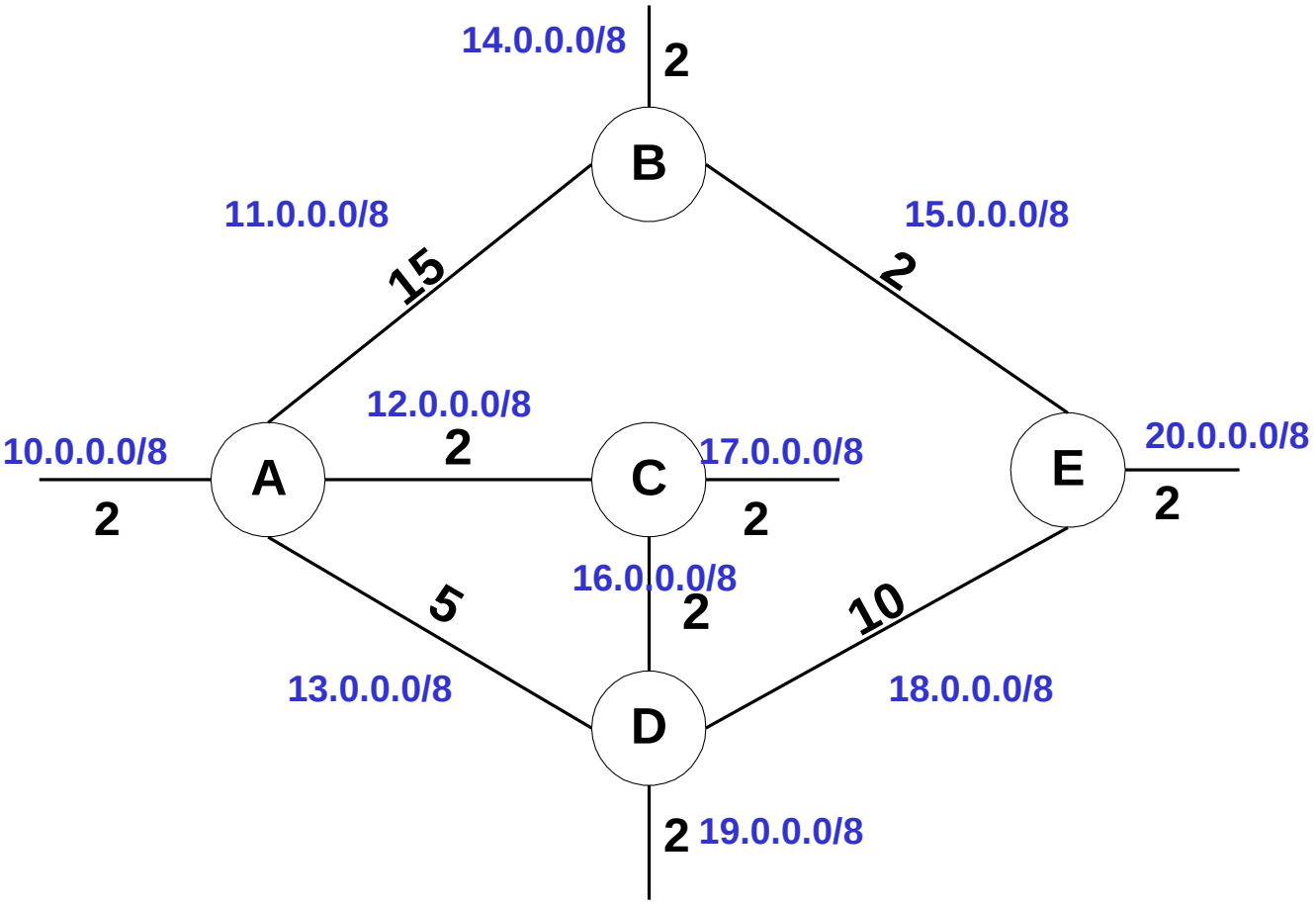
+





Topology

- Using topological information listed, Router A has a complete topology of network.
- What happens next?
- Next step is for link-state algorithm to find best path to each node and leaf network.



Simplified Link State Example

RouterA's Topological Data Base (Link State Database)

RouterB:

- **Connected to RouterA on network 11.0.0.0/8, cost of 15**
- **Connected to RouterE on network 15.0.0.0/8, cost of 2**
- **Has a "leaf" network 14.0.0.0/8, cost of 15**

RouterC:

- **Connected to RouterA on network 12.0.0.0/8, cost of 2**
- **Connected to RouterD on network 16.0.0.0/8, cost of 2**
- **Has a "leaf" network 17.0.0.0/8, cost of 2**

RouterD:

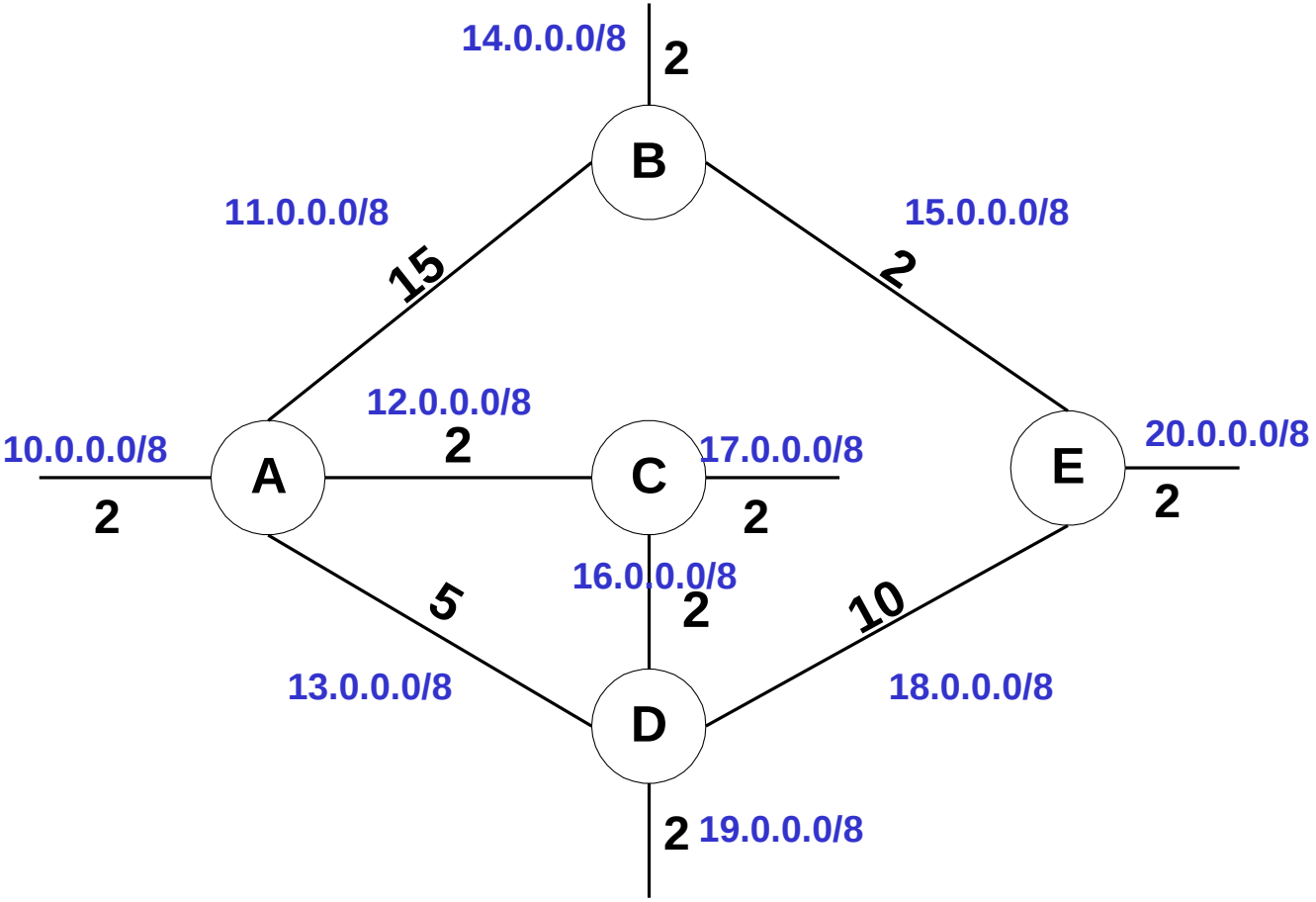
- **Connected to RouterA on network 13.0.0.0/8, cost of 5**
- **Connected to RouterC on network 16.0.0.0/8, cost of 2**
- **Connected to RouterE on network 18.0.0.0/8, cost of 2**
- **Has a "leaf" network 19.0.0.0/8, cost of 2**

RouterE:

- **Connected to RouterB on network 15.0.0.0/8, cost of 2**
- **Connected to RouterD on network 18.0.0.0/8, cost of 10**
- **Has a "leaf" network 20.0.0.0/8, cost of 2**

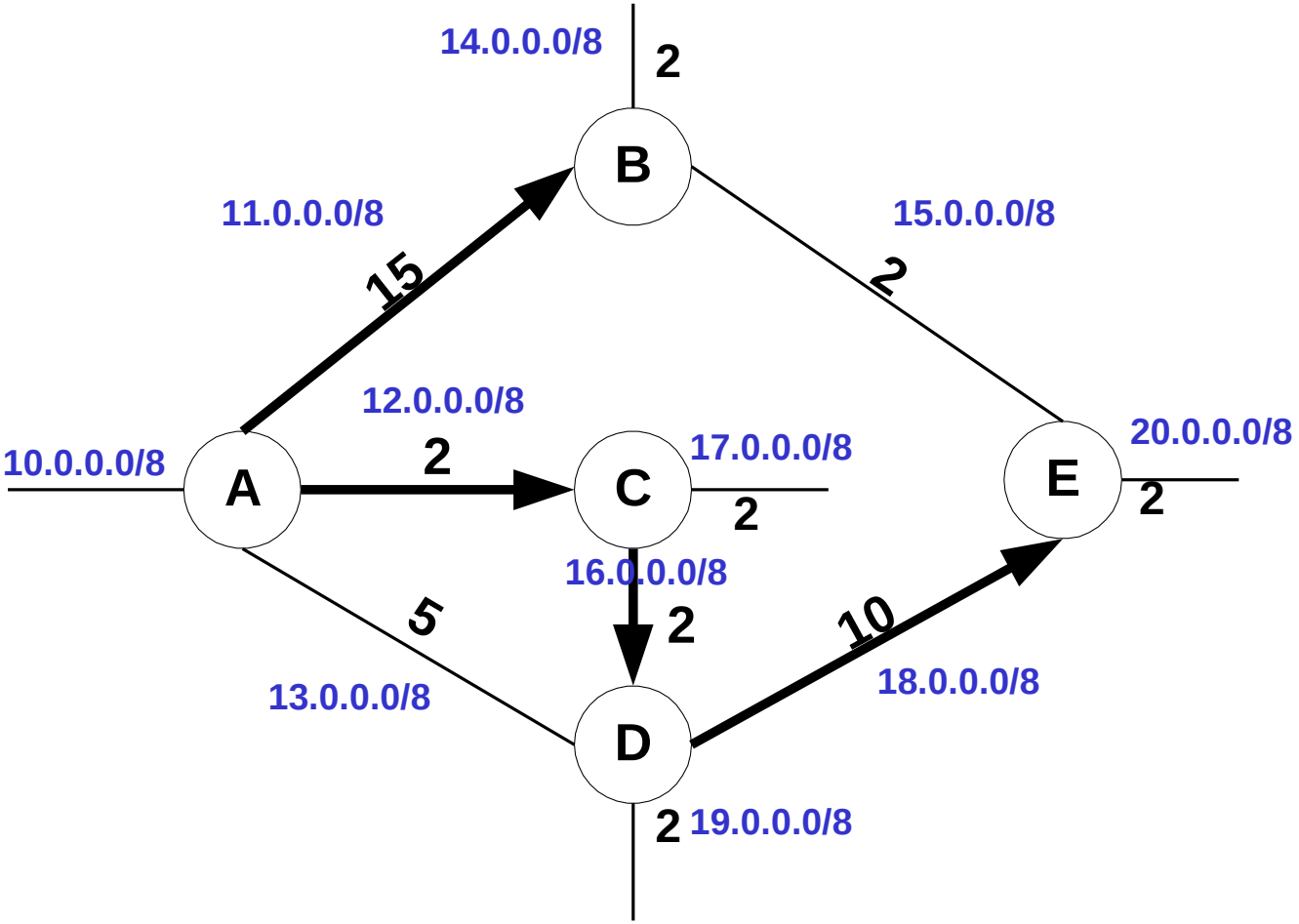
Choosing the Best Path

- Using the link-state algorithm RouterA can now proceed to find the shortest path to each leaf network.



Choosing the Best Path

Now RouterA knows the best path to each network, creating an SPT (Shortest Path Tree).

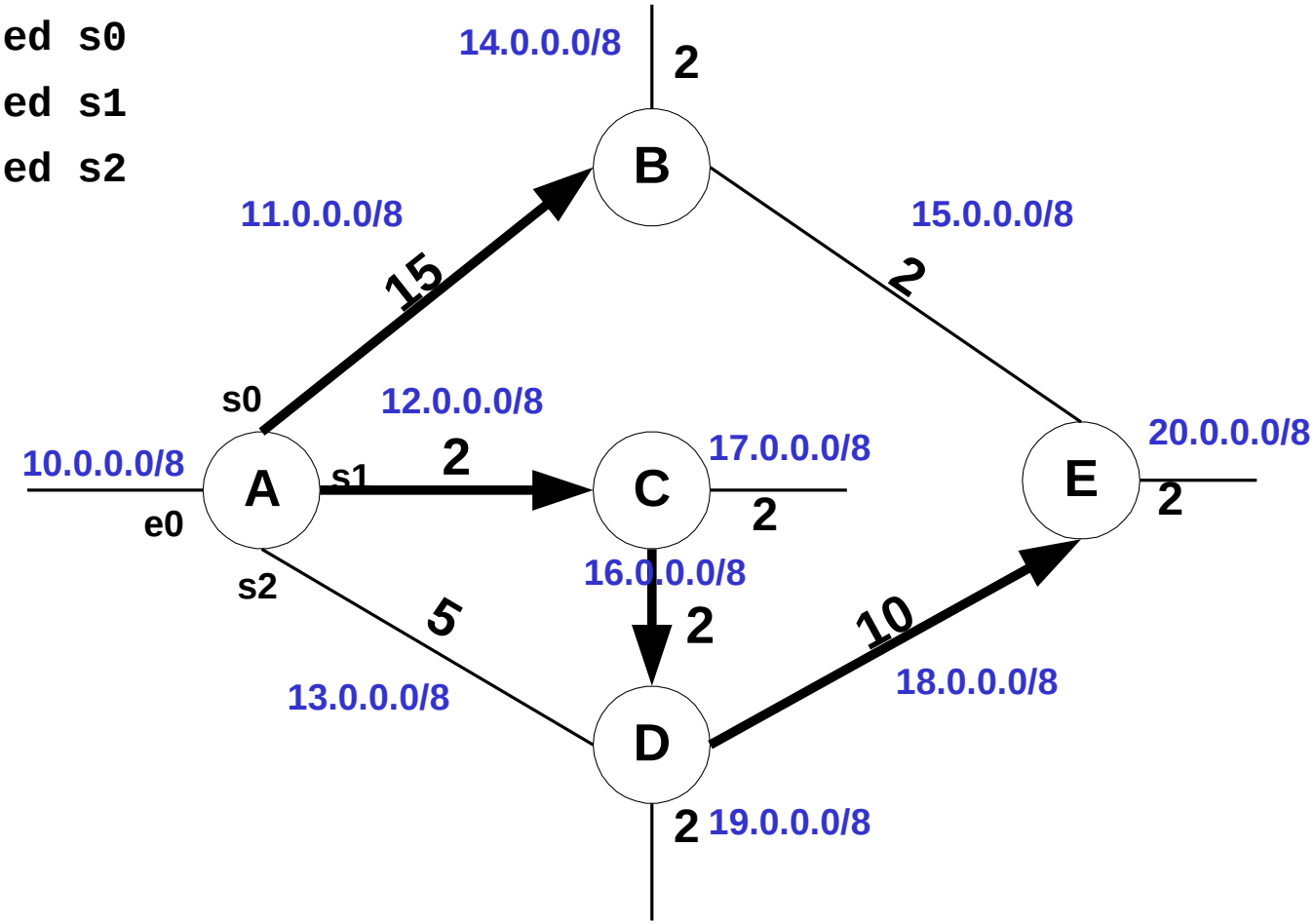


SPT Results Get Put into the Routing Table

RouterA's Routing Table

```
10.0.0.0/8   connected e0
11.0.0.0/8   connected s0
12.0.0.0/8   connected s1
13.0.0.0/8   connected s2

14.0.0.0/8   17 s0
15.0.0.0/8   17 s1
16.0.0.0/8    4 s1
17.0.0.0/8    4 s1
18.0.0.0/8   14 s1
19.0.0.0/8    6 s1
20.0.0.0/8   16 s1
```



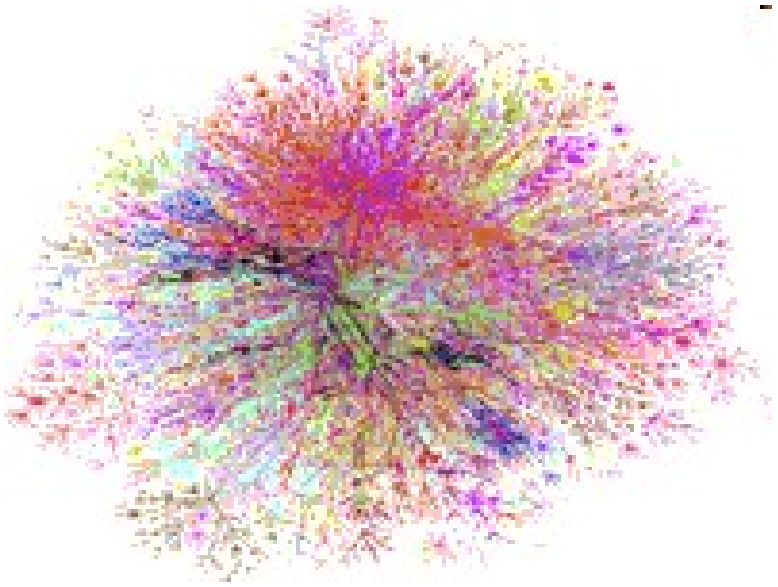
References

- Rip vs. OSPF

  http://www.networkcomputing.com/unixworld/feature/002.html

- Routing Overview
  - http://networking.ringofsaturn.com/IP/Routing.php

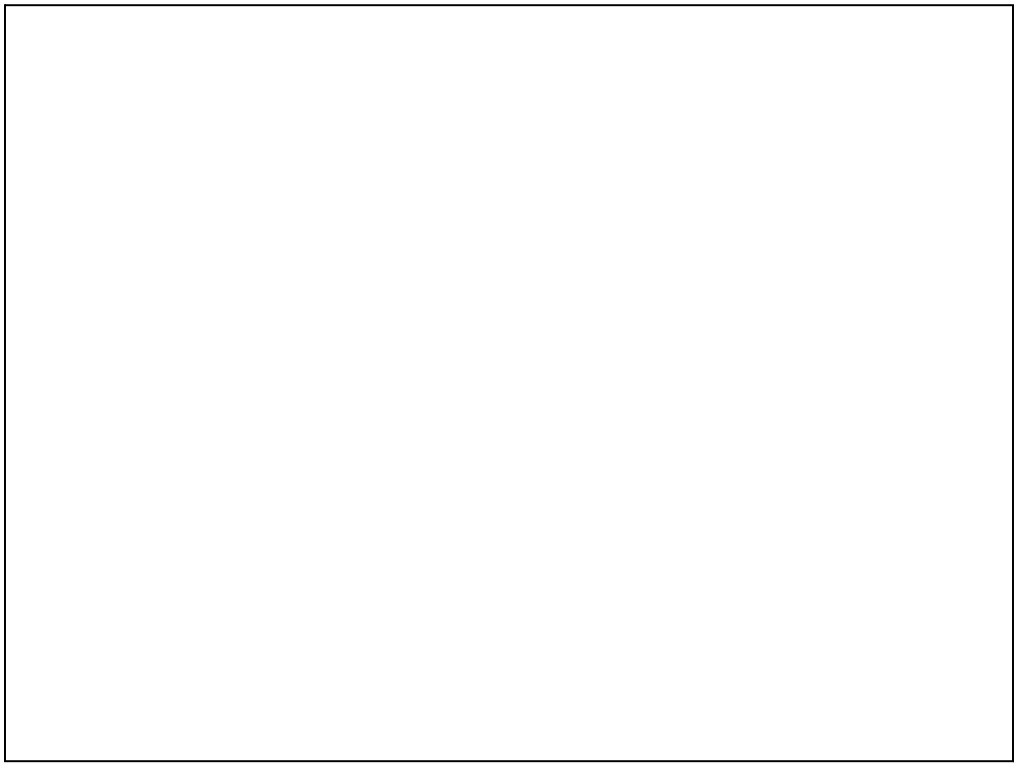
INTERNET MAPPING PROJECT

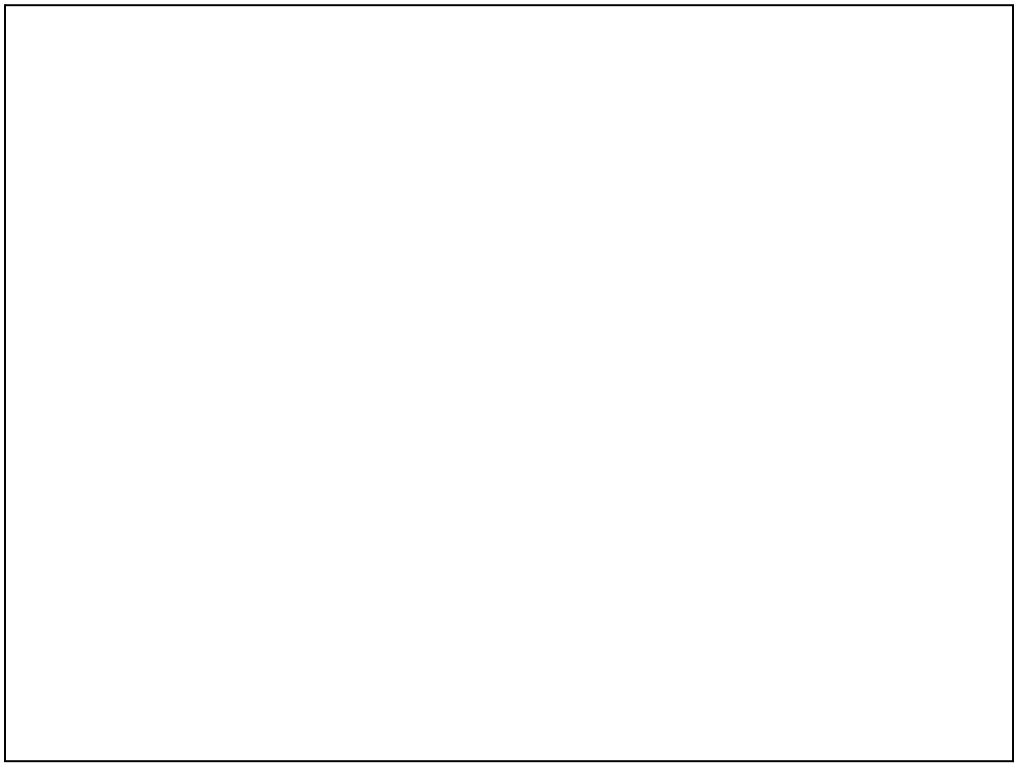


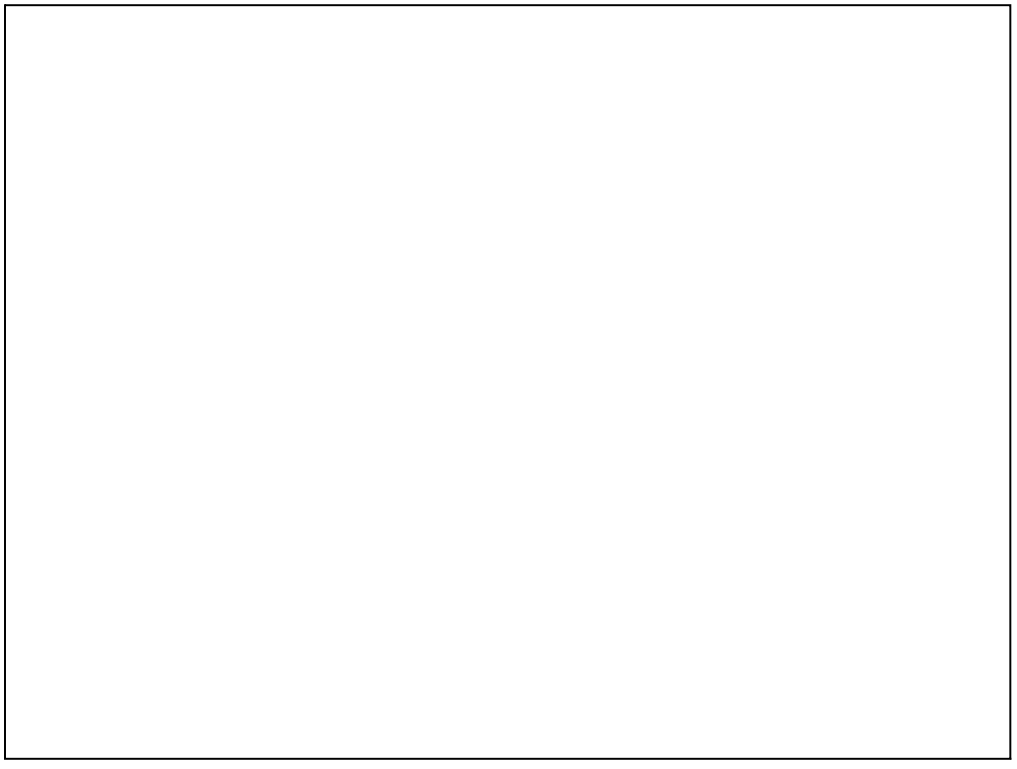
http://www.cheswick.com/ches/map/

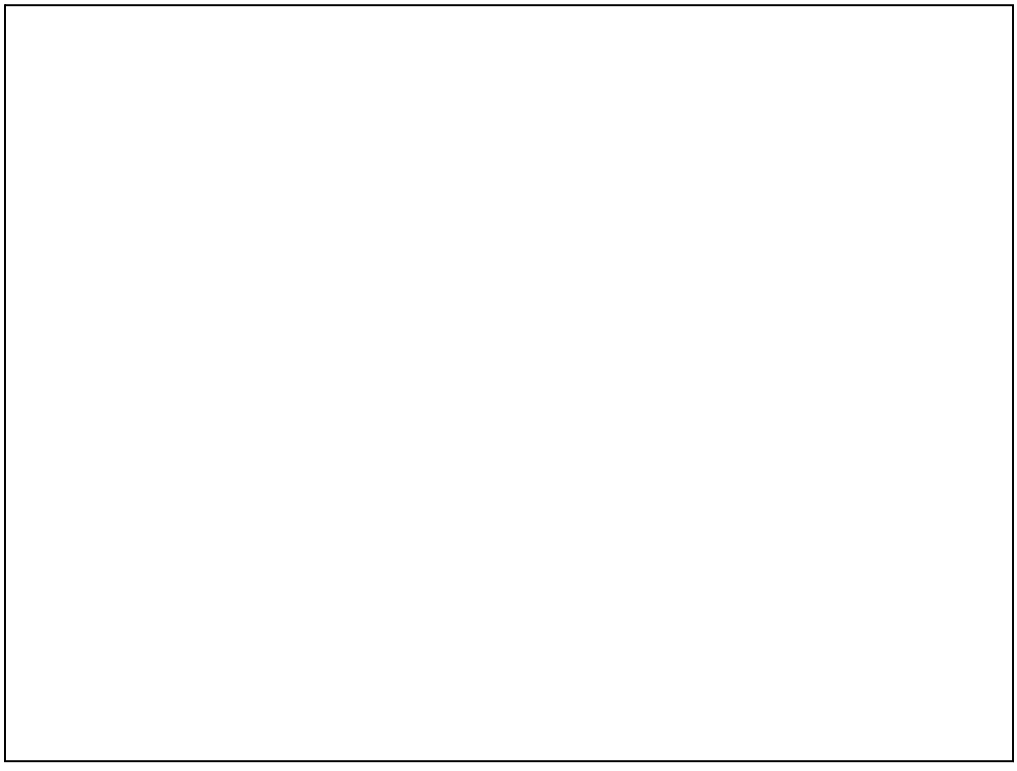
Next

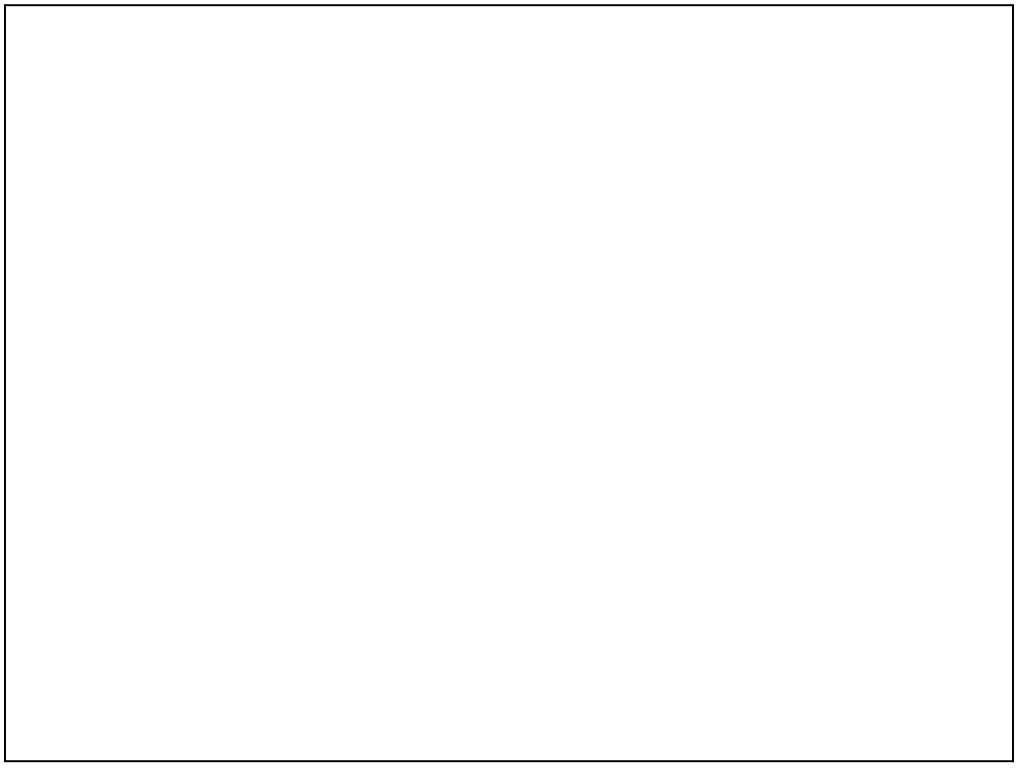
Reading: Moving on to Chapter 5!!!

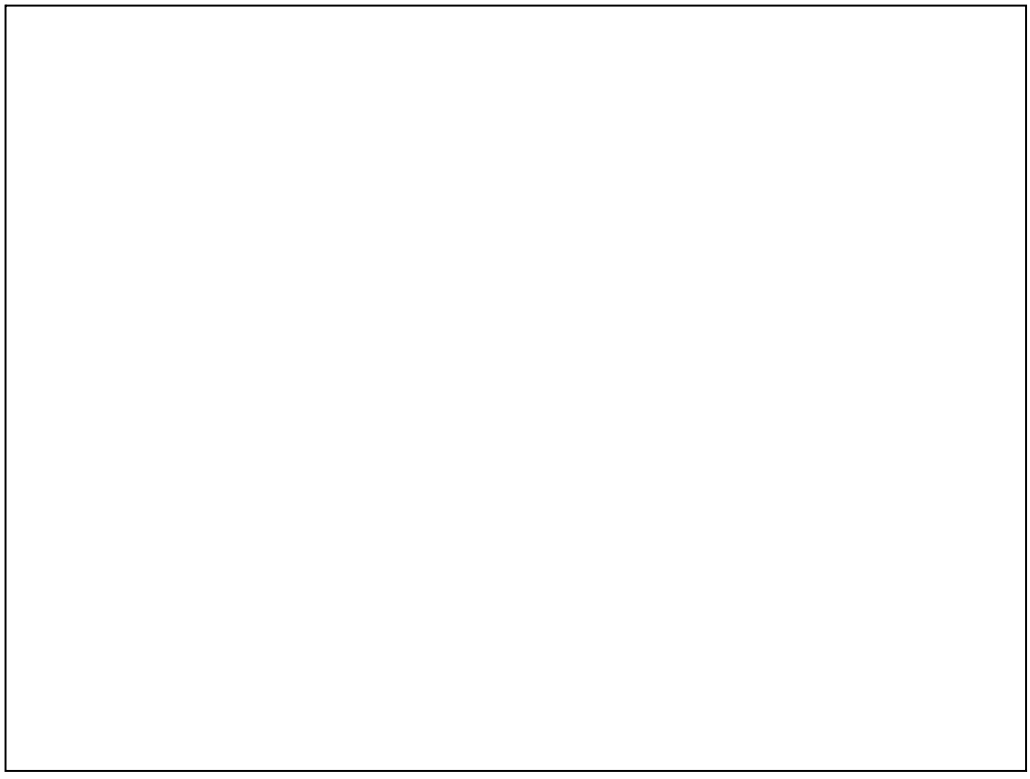










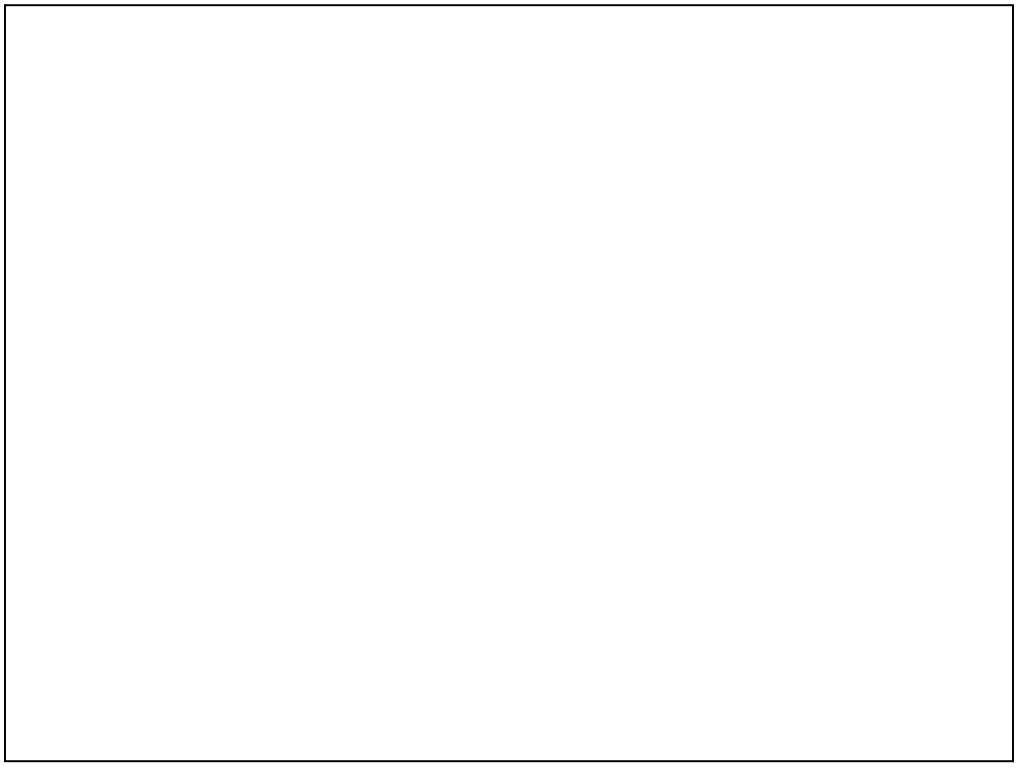


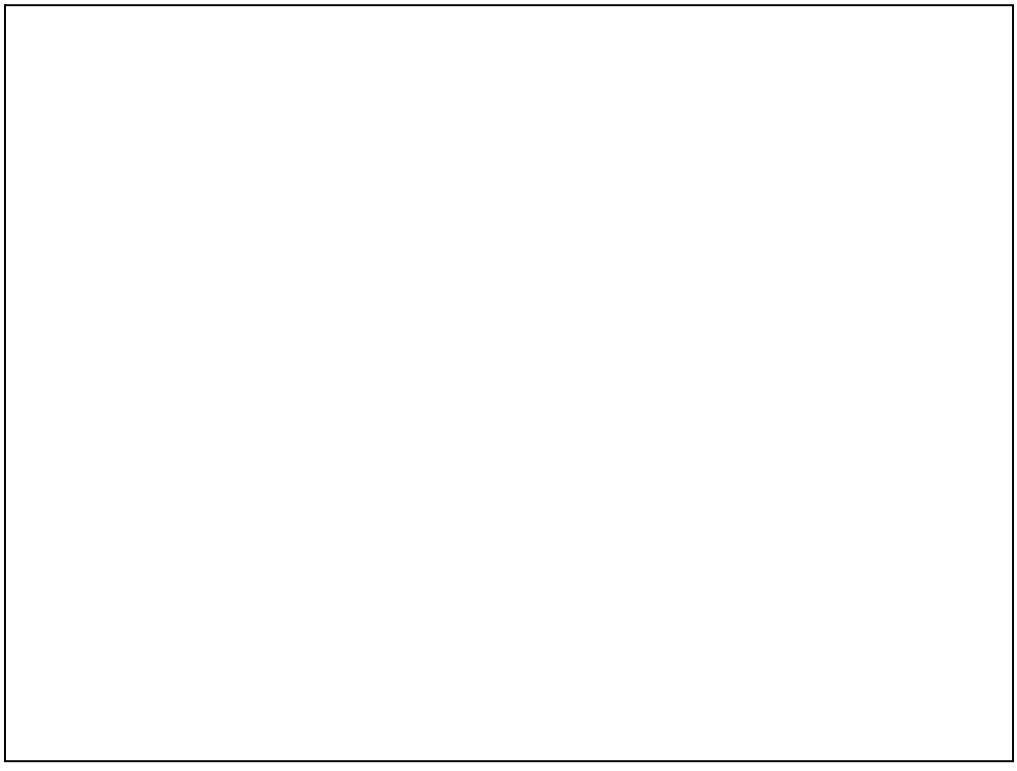
Routing Algorithms

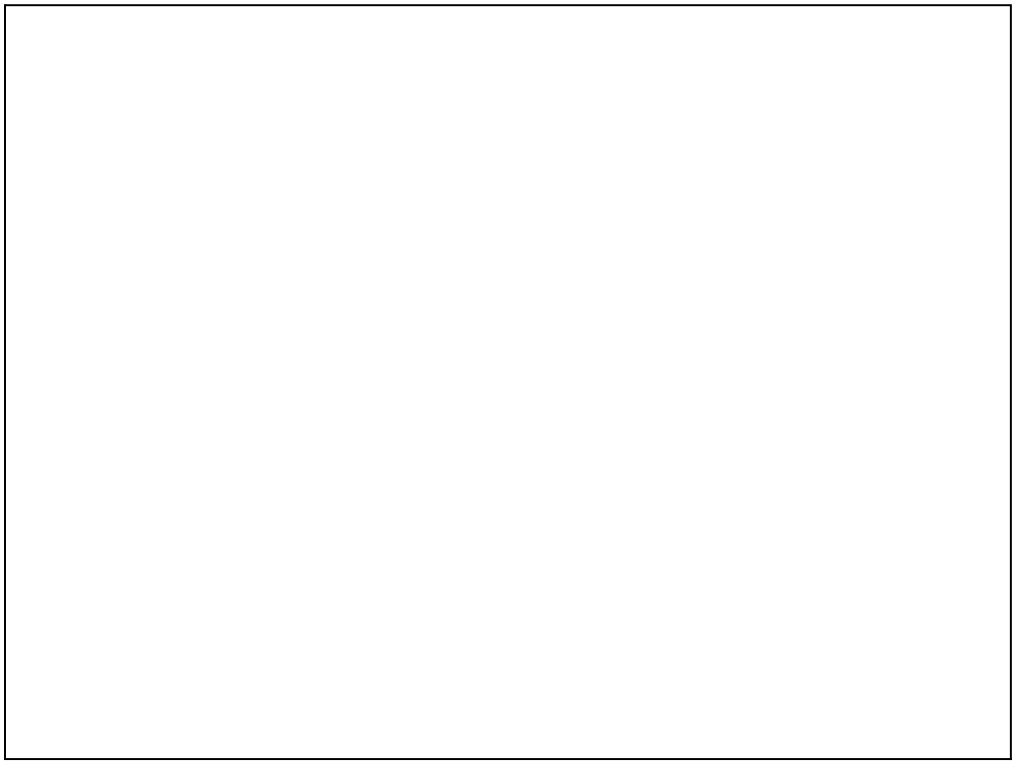
Purpose of Routing Protocols

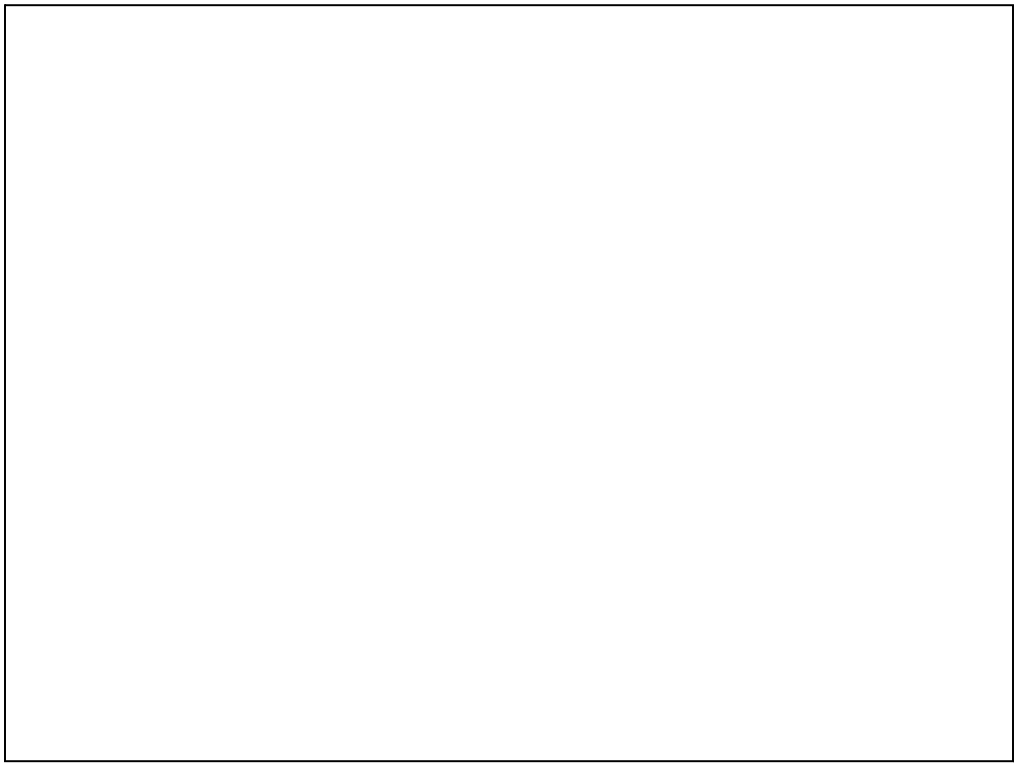
- Routing protocols are used to exchange routing information between routers.
- A routing protocol is: ***"Set of processes, algorithms, and messages used to exchange routing information and populate the routing table with the best paths"***

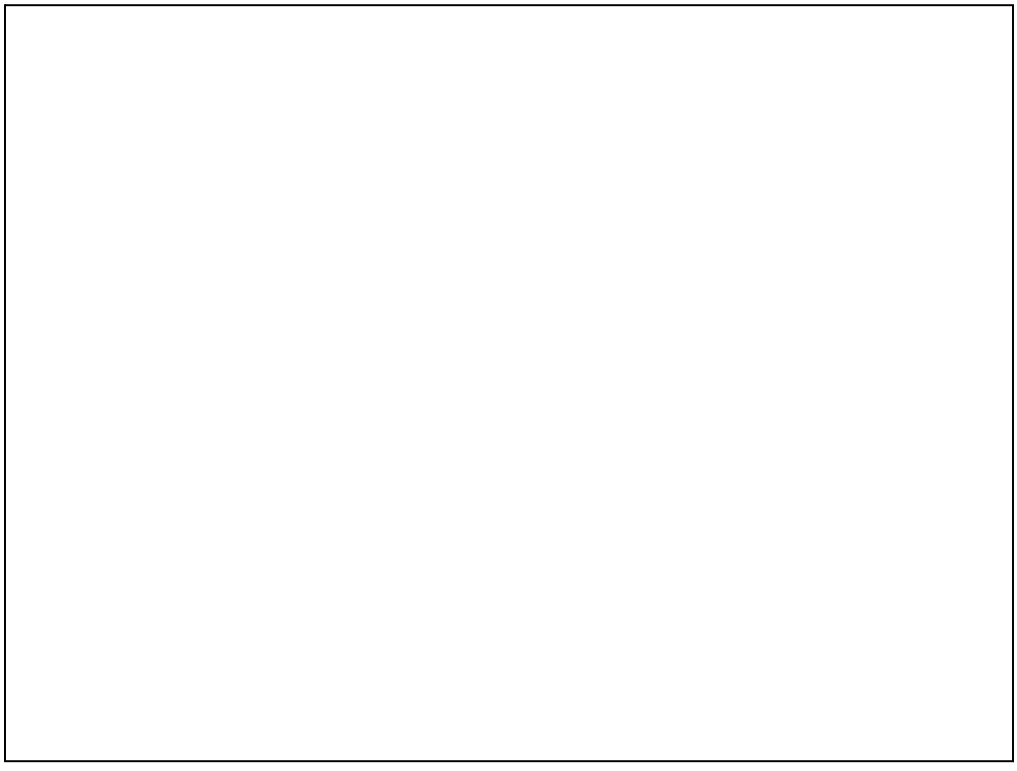
- The purpose of dynamic routing protocols includes:
- Discovery of remote networks
- Maintaining up-to-date routing information
- Choosing the best path to destination networks
- Ability to find new best path if current path is no longer available

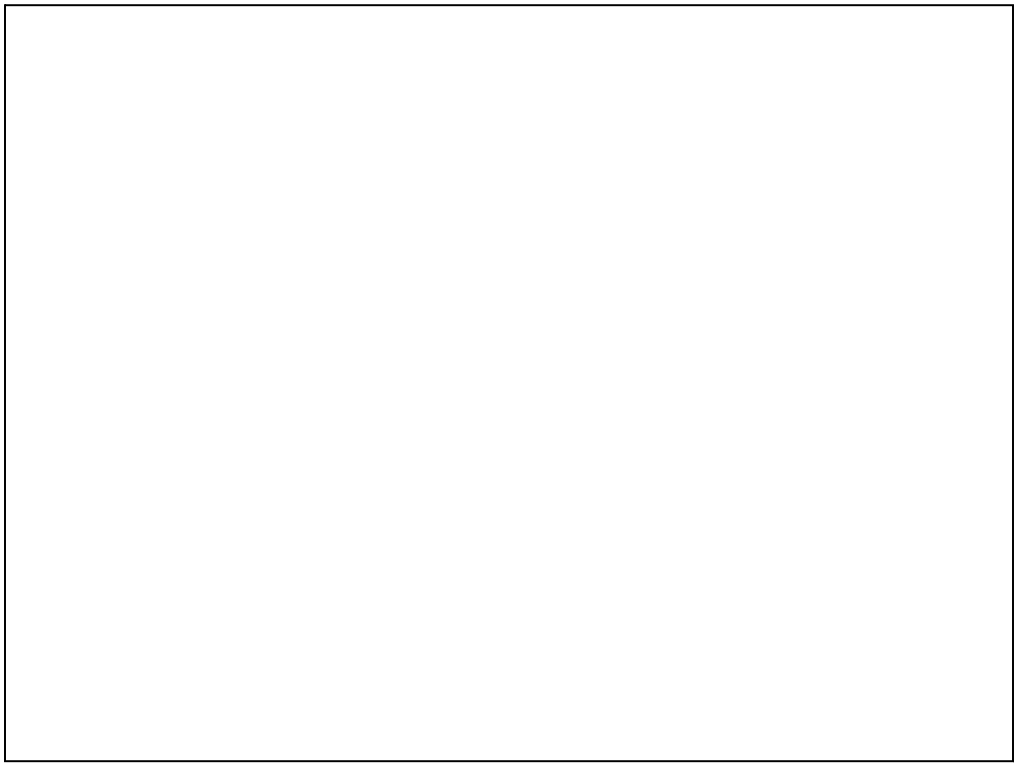


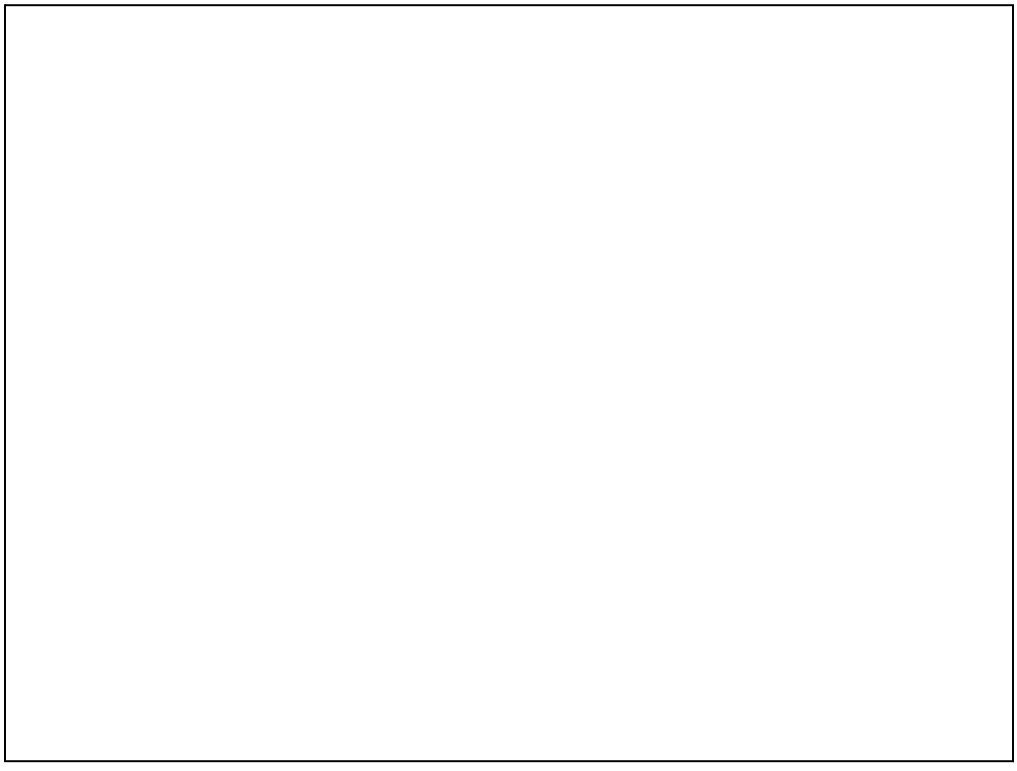


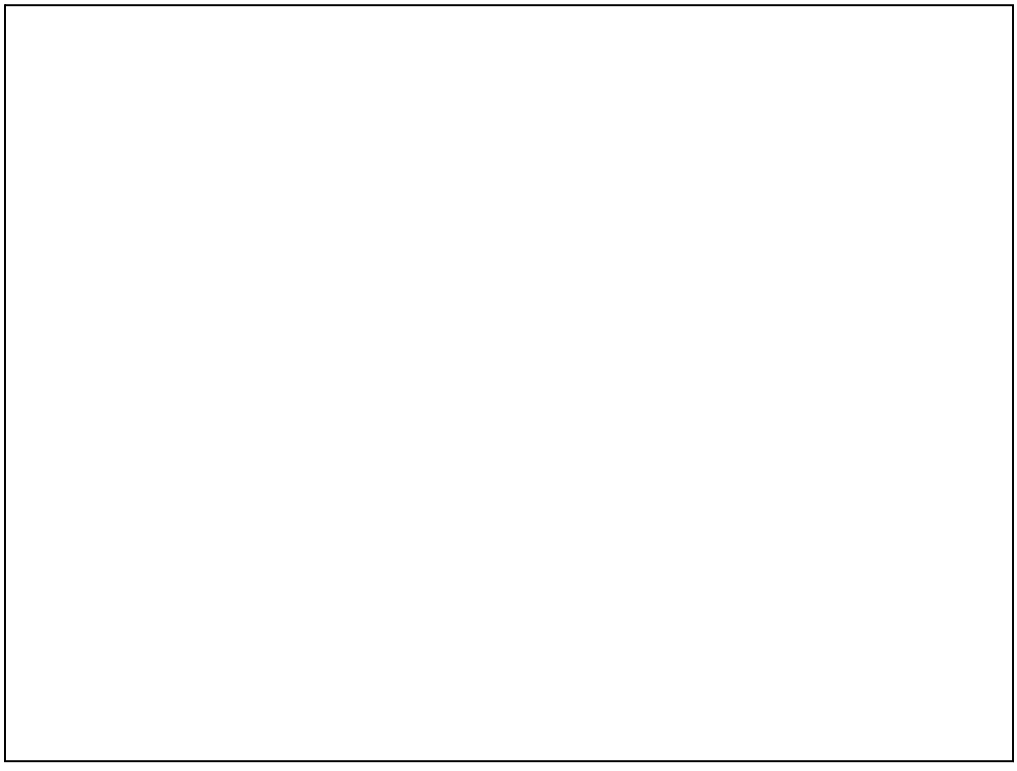


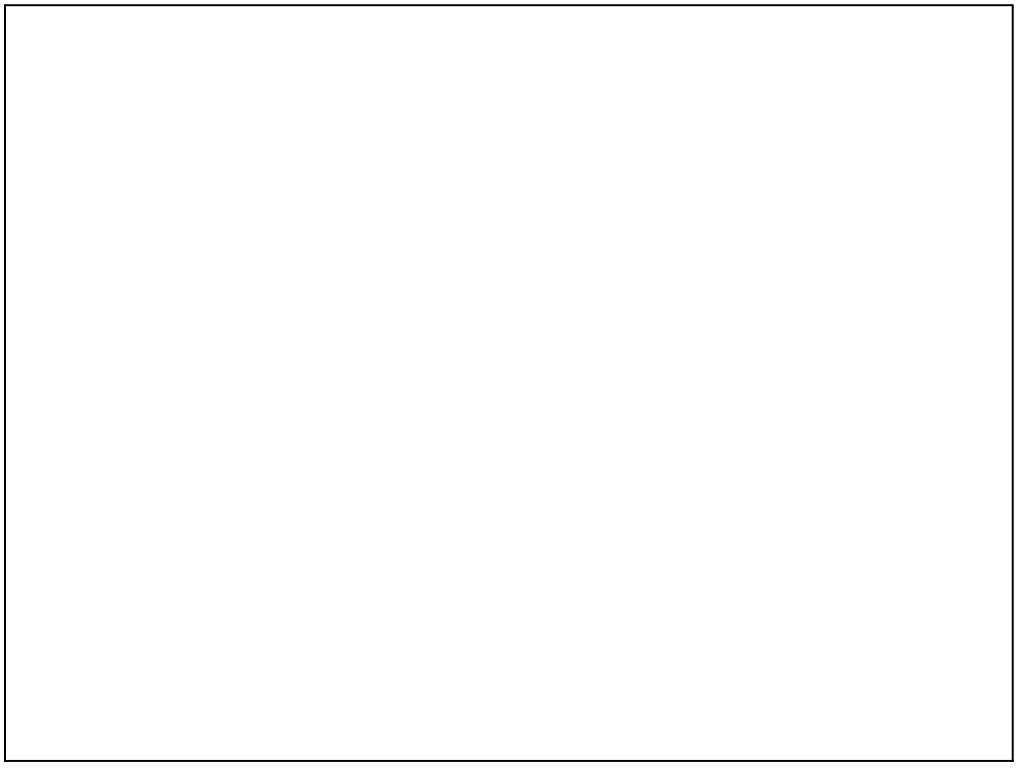


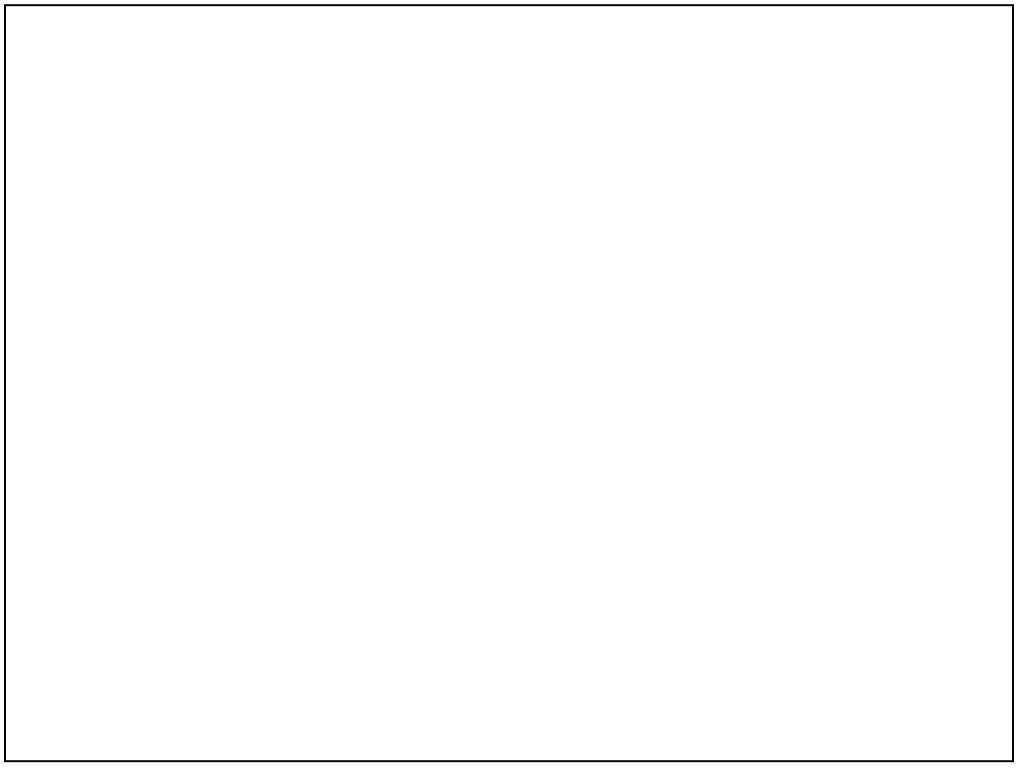


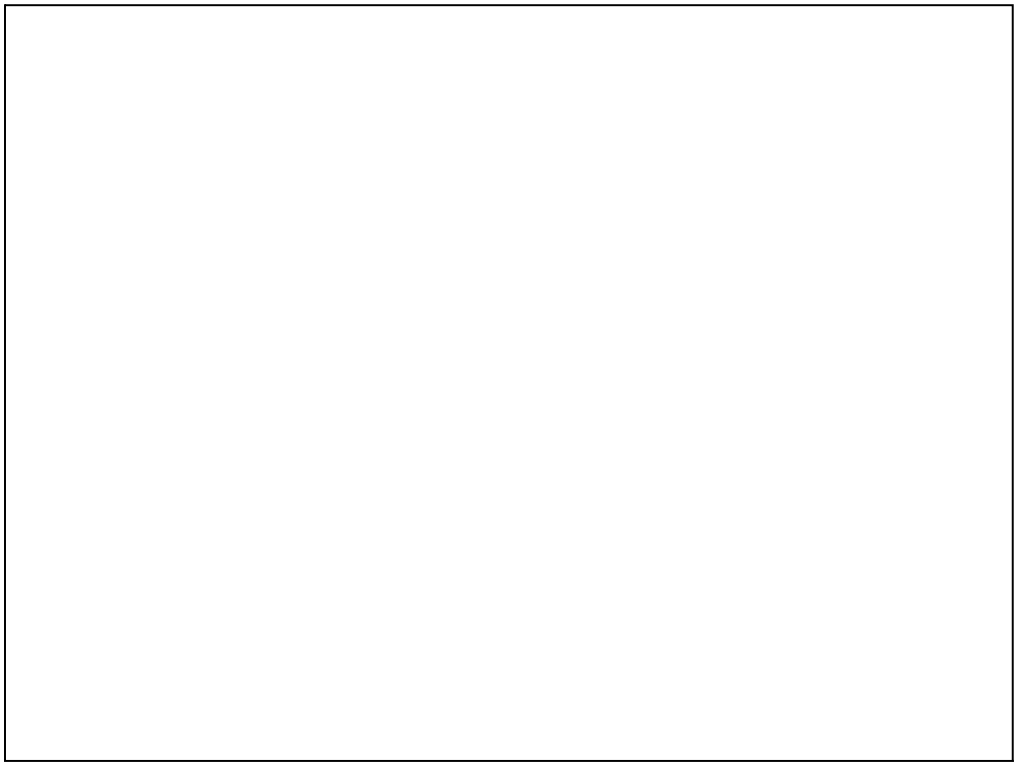


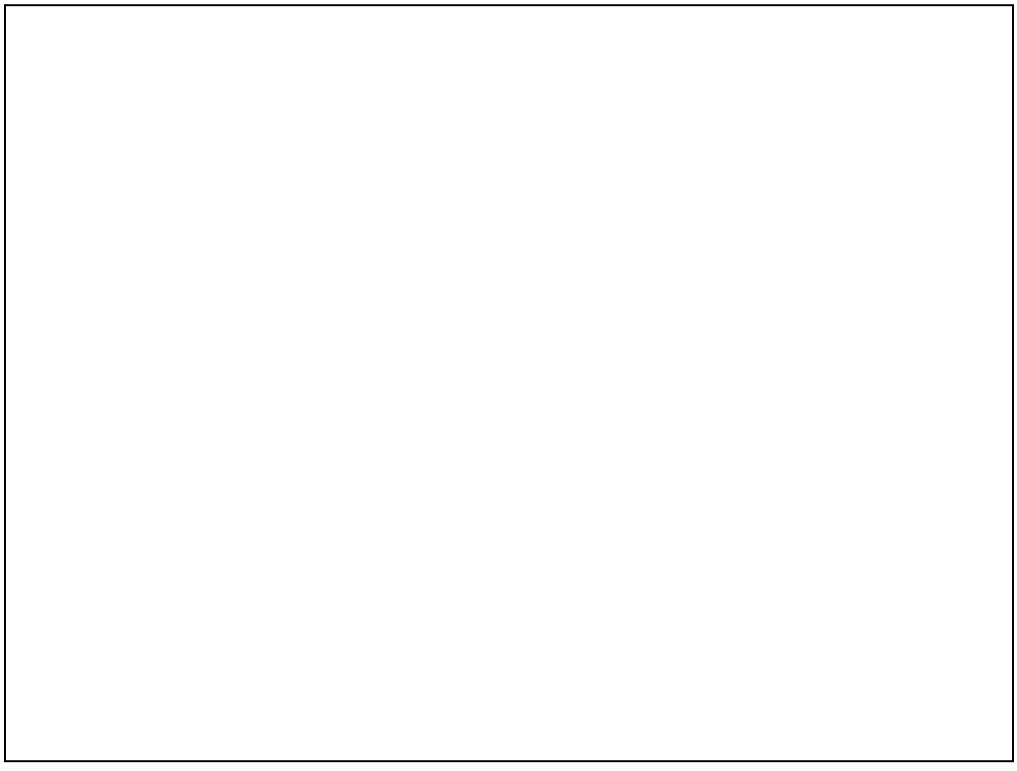






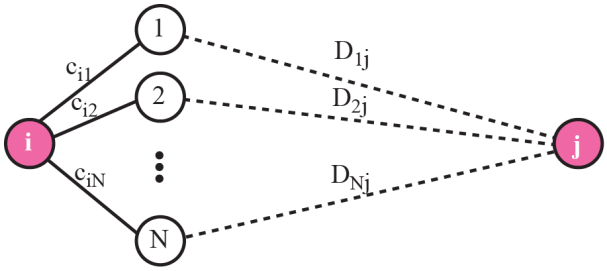






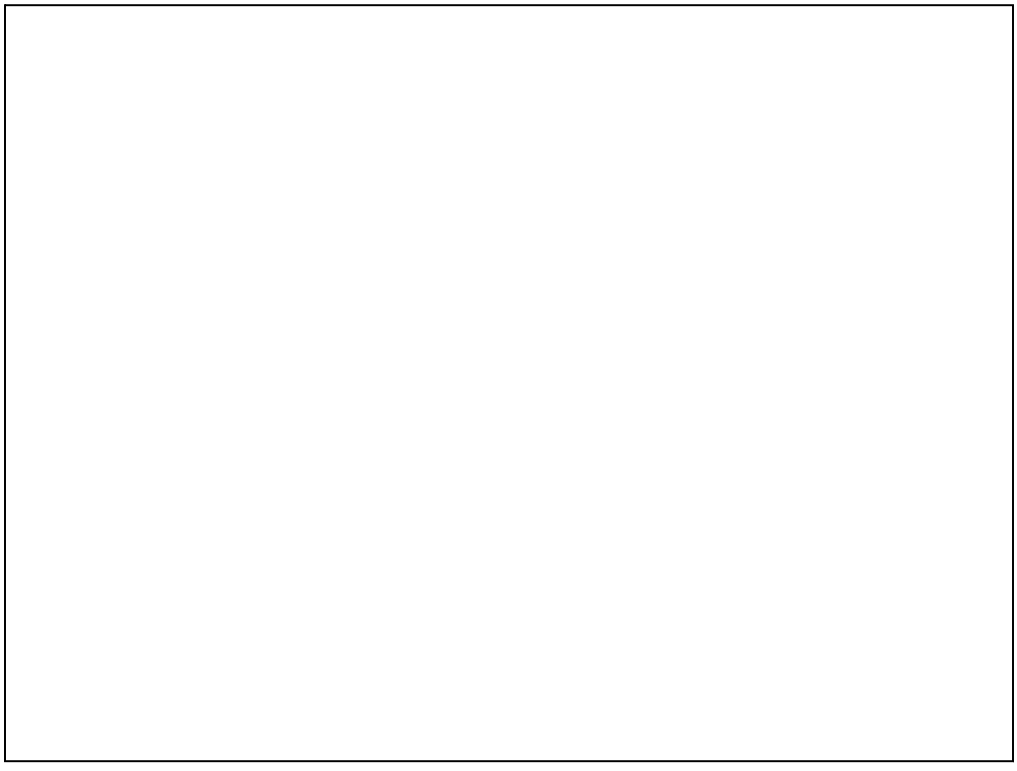
Bellman-Ford algorithm

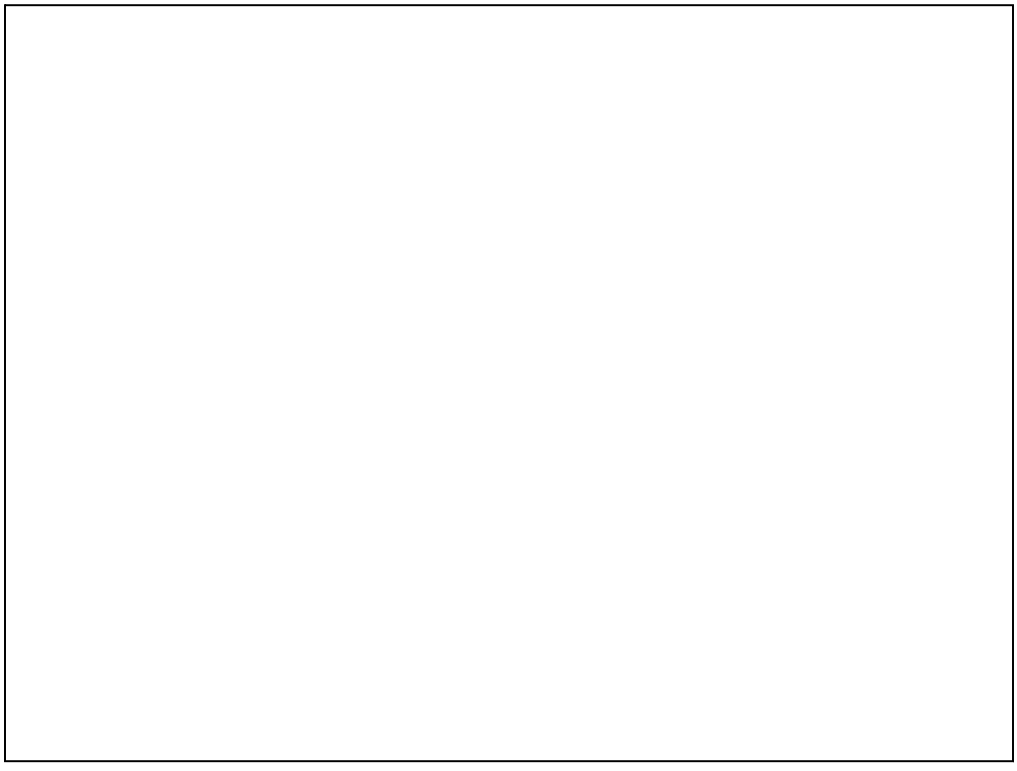
$$D_{ij} = \text{minimum } \{(c_{i1} + D_{1j}), (c_{i2} + D_{2j}), \ldots (c_{iN} + D_{Nj})$$

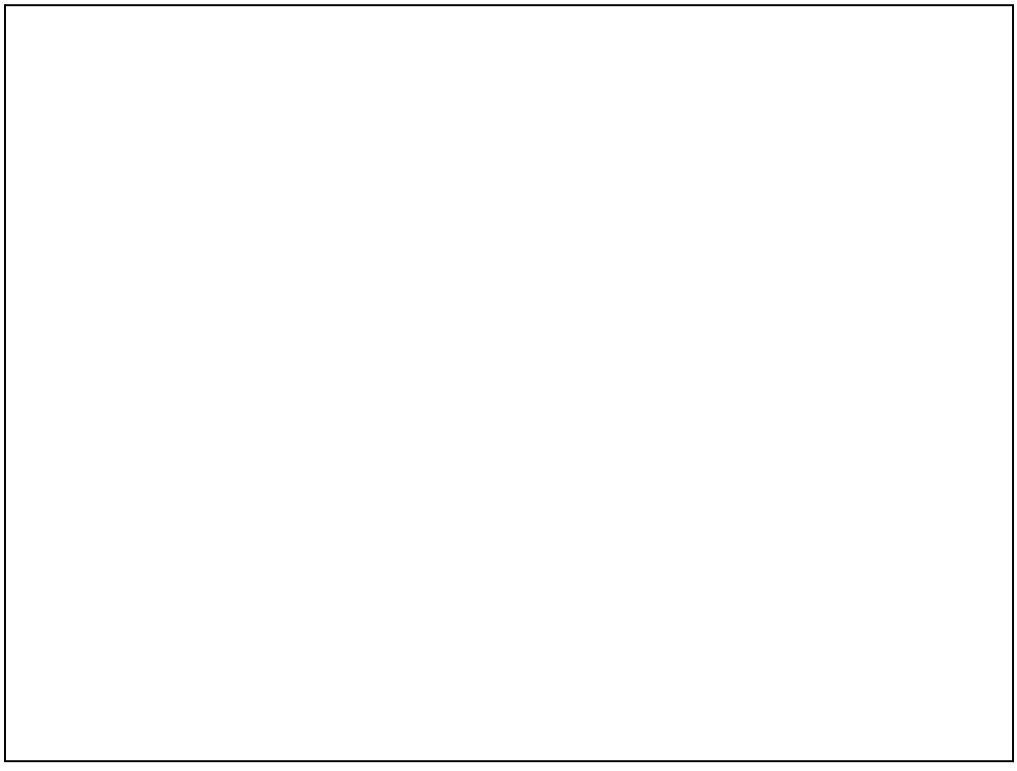


Legend

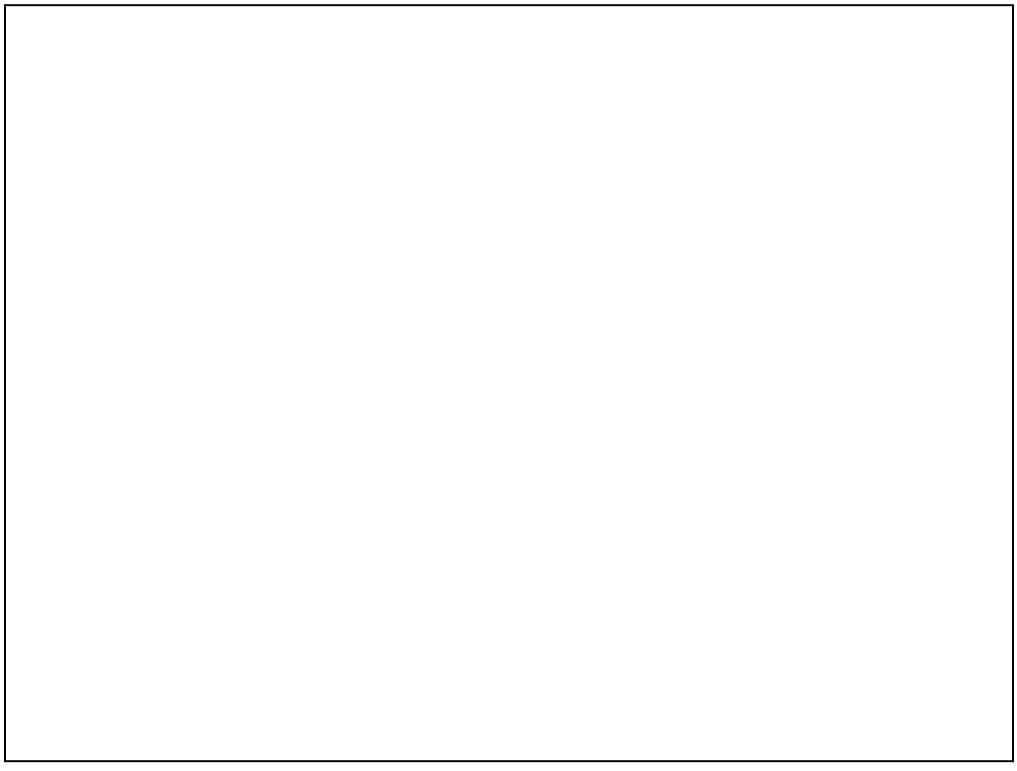
D_{ij} Shortest distance between i and j
c_{ij} Cost between i and j
N Number of nodes

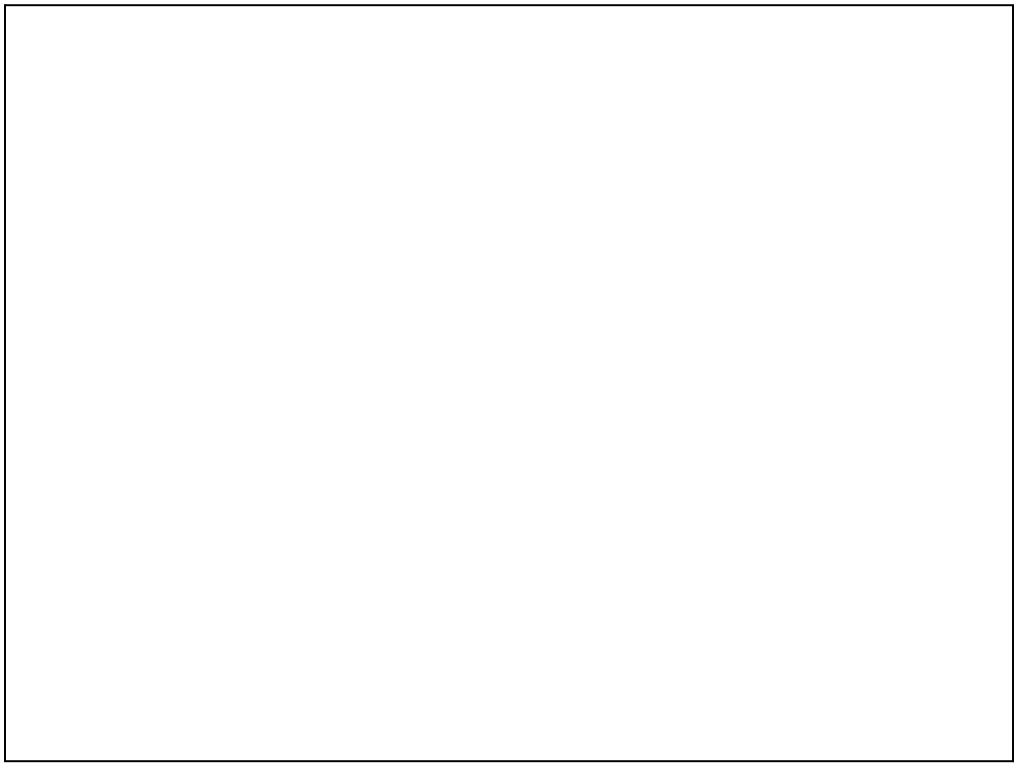


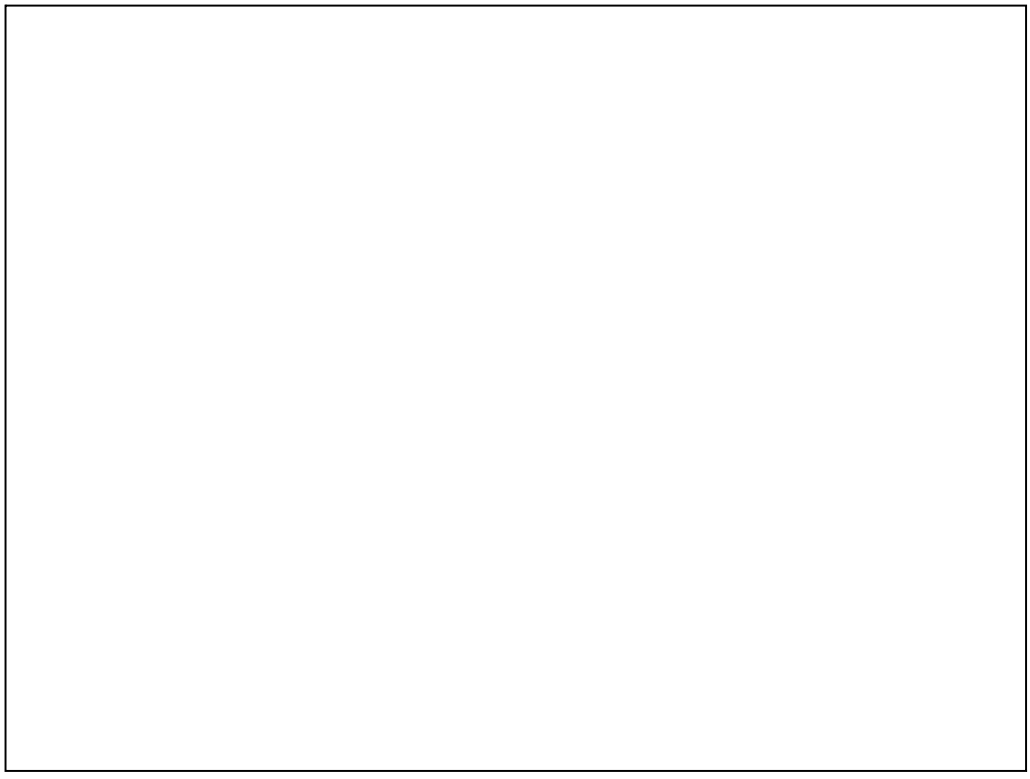


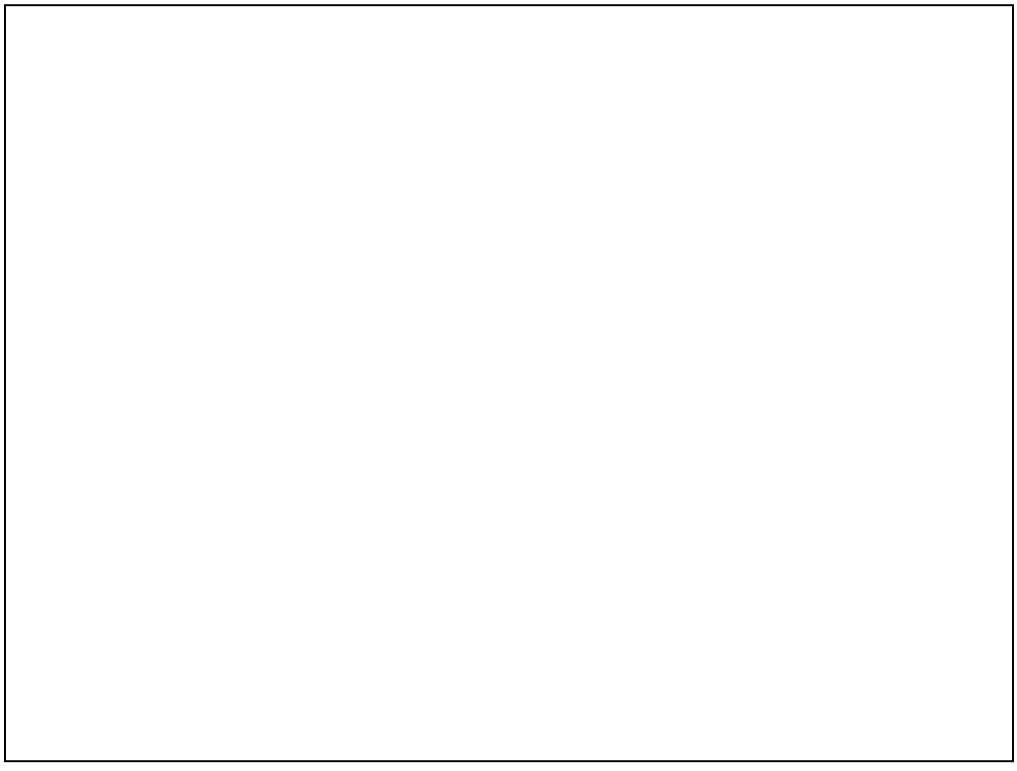


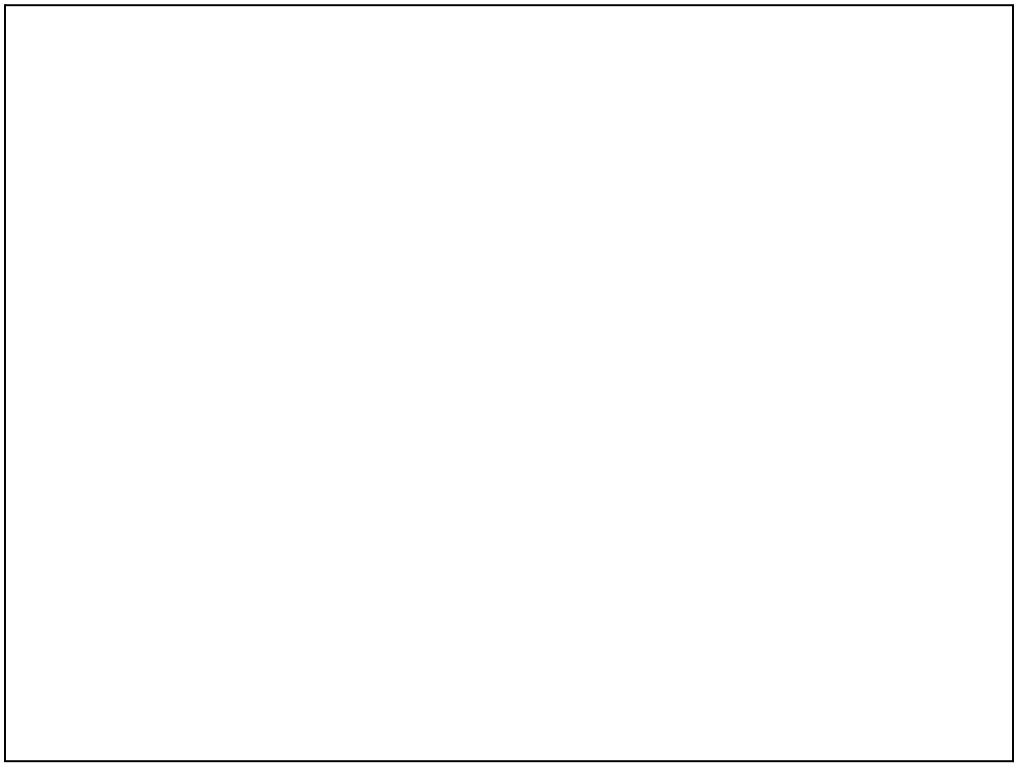
Routing Information Protocol
As an Example of Distance Vector

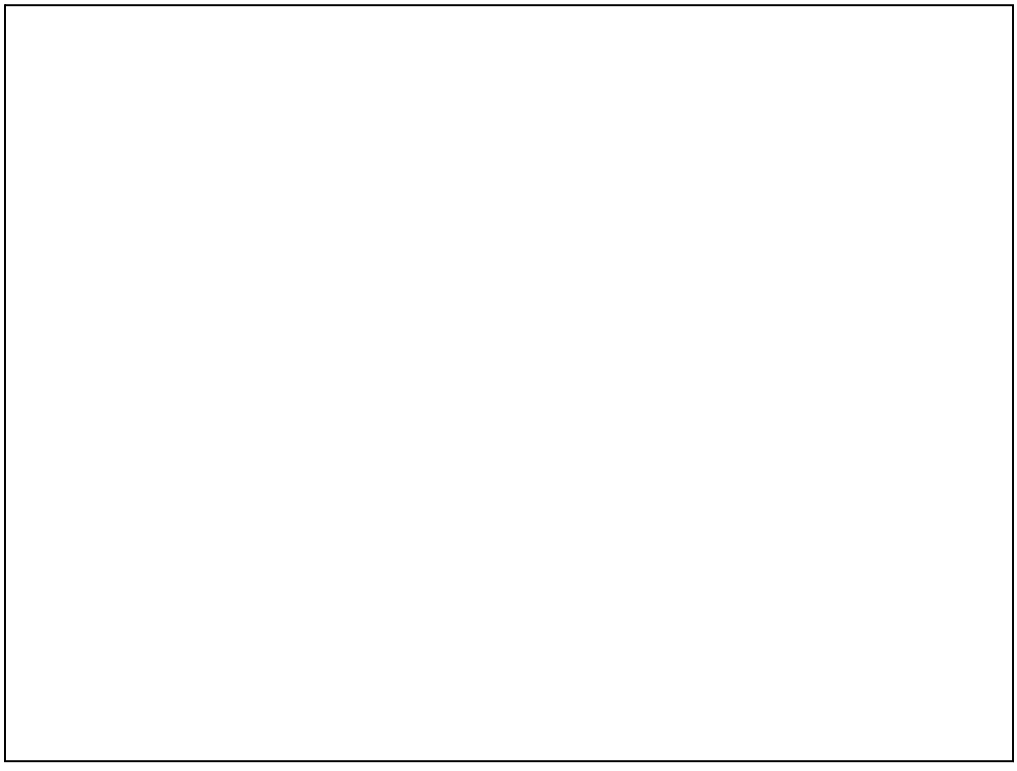


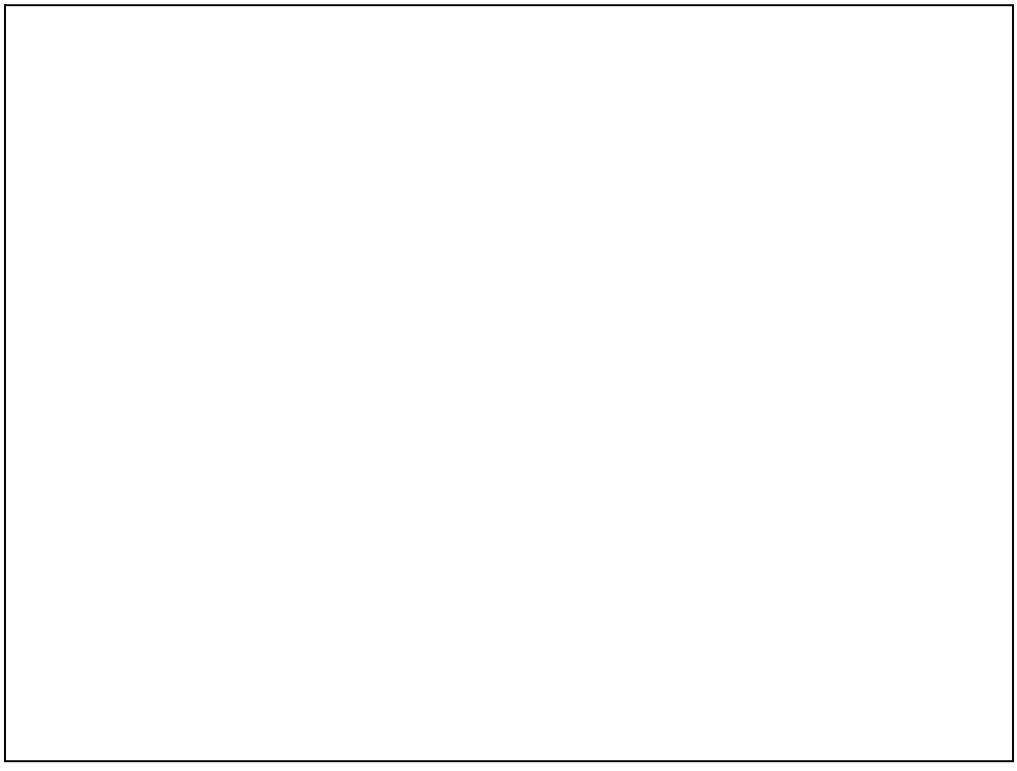


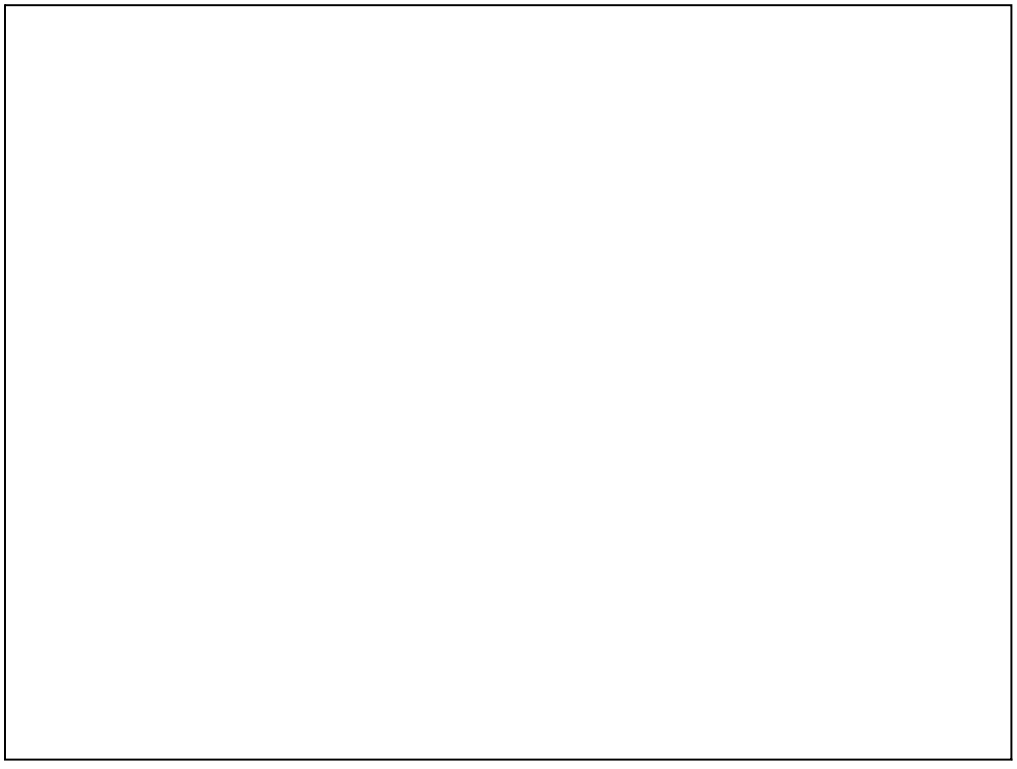


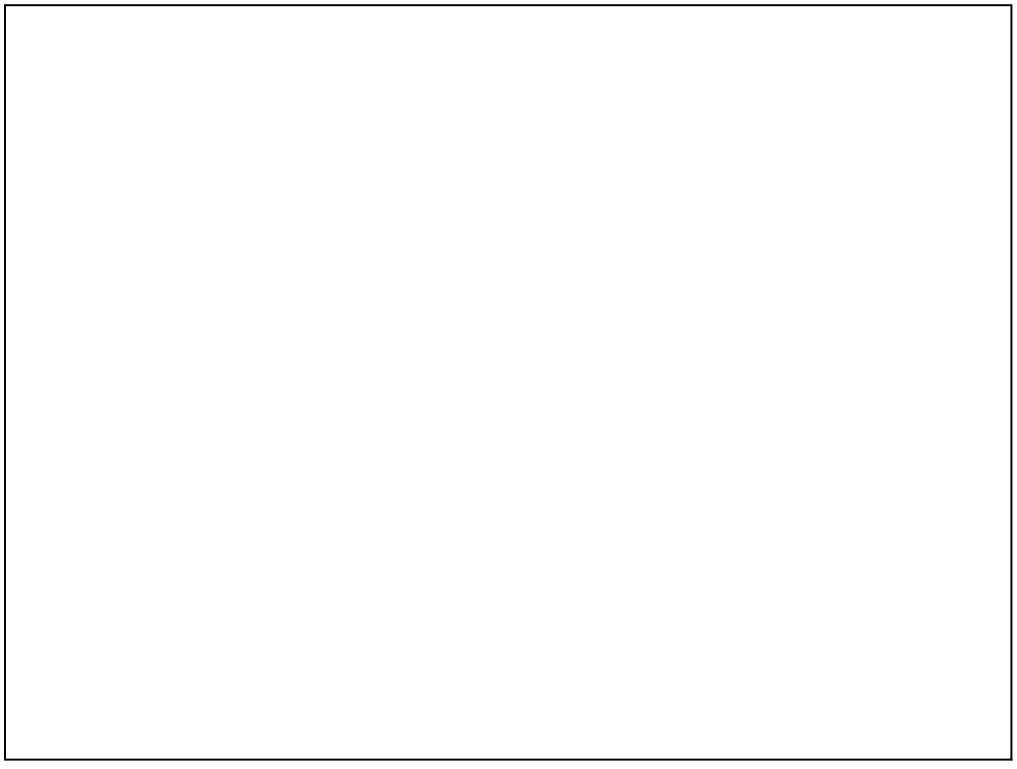


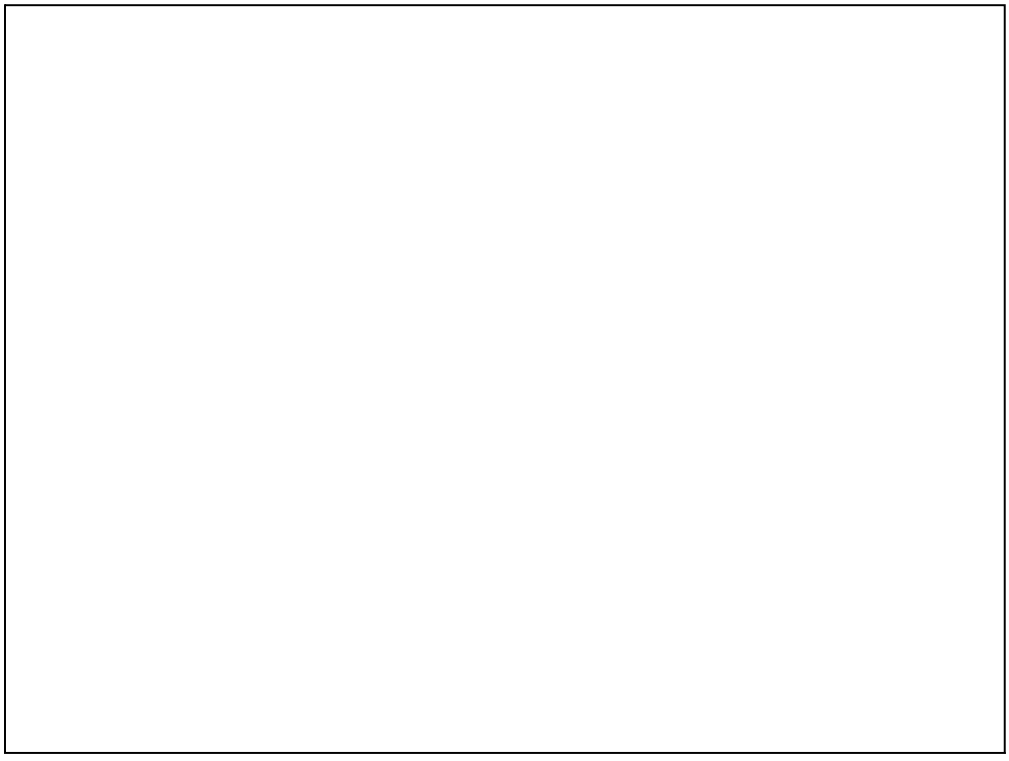


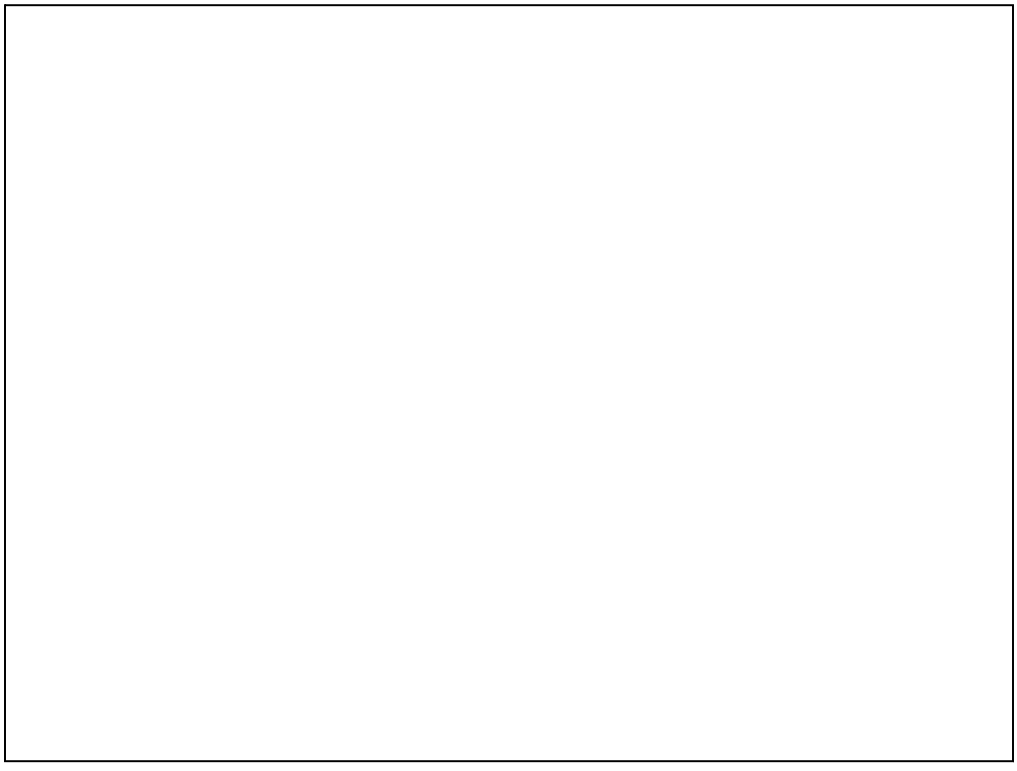


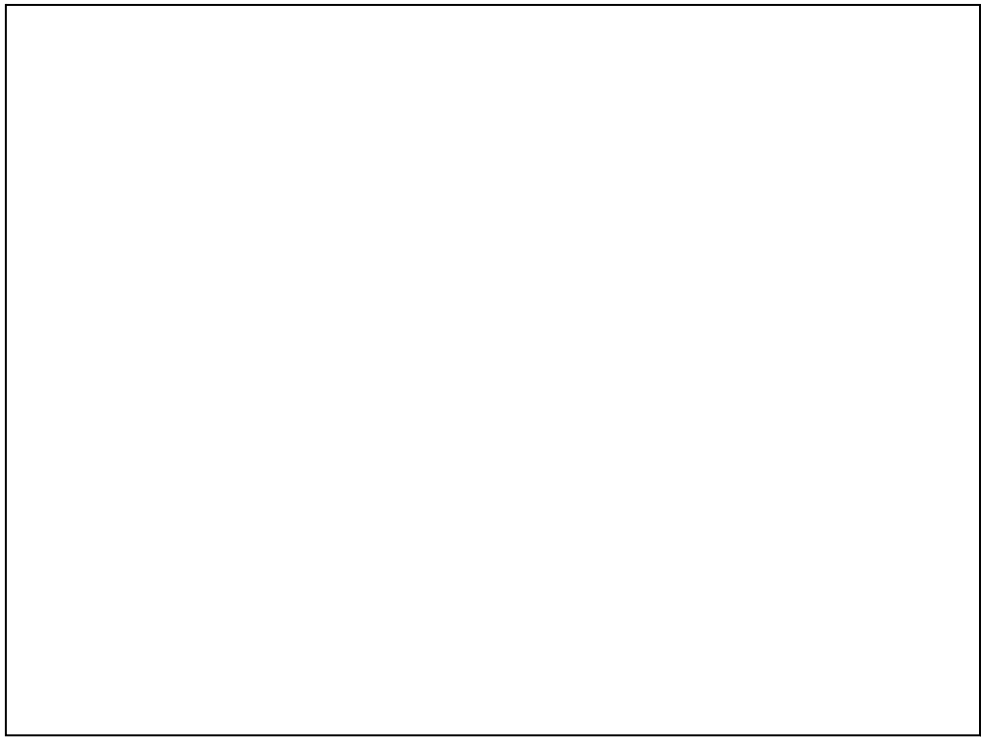


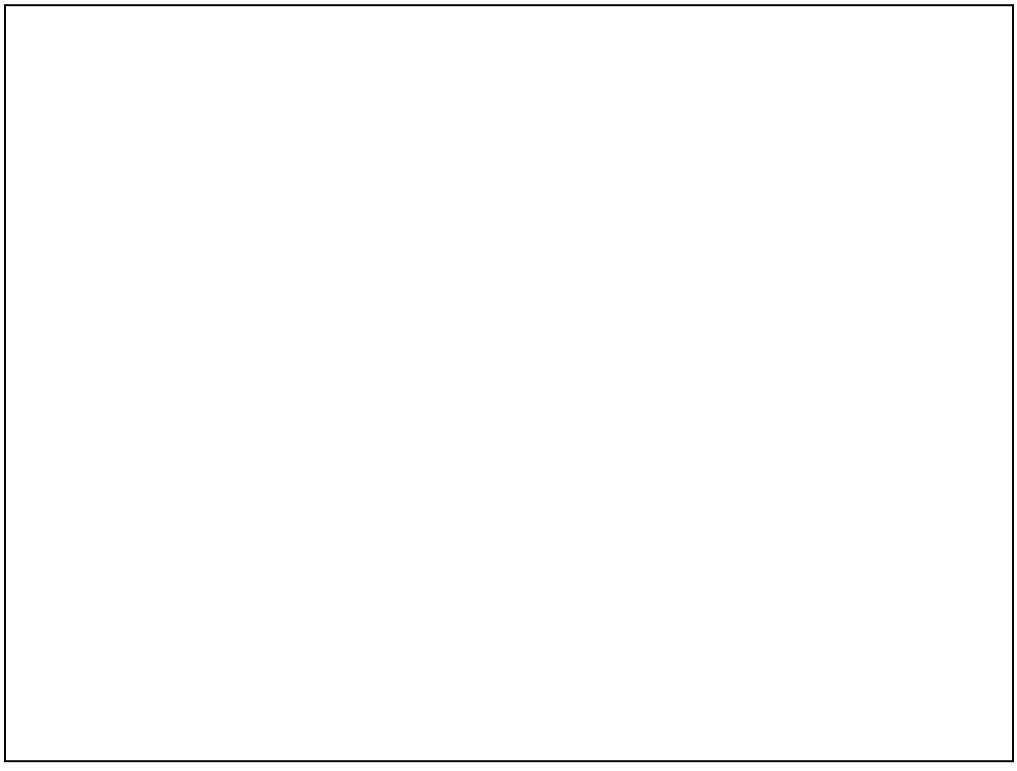


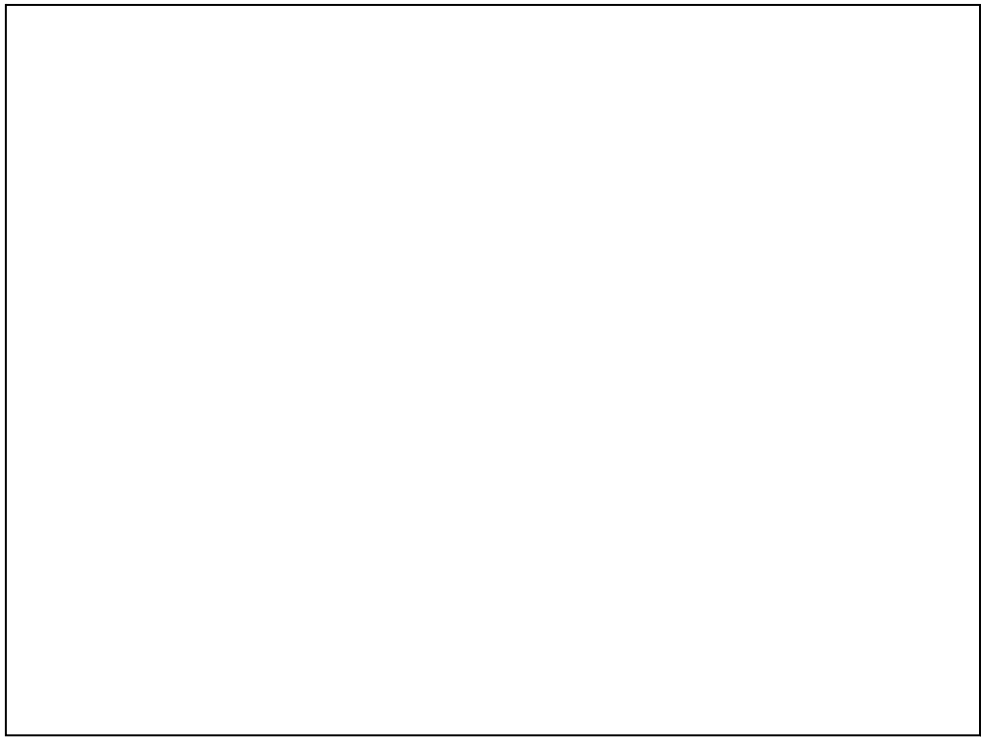


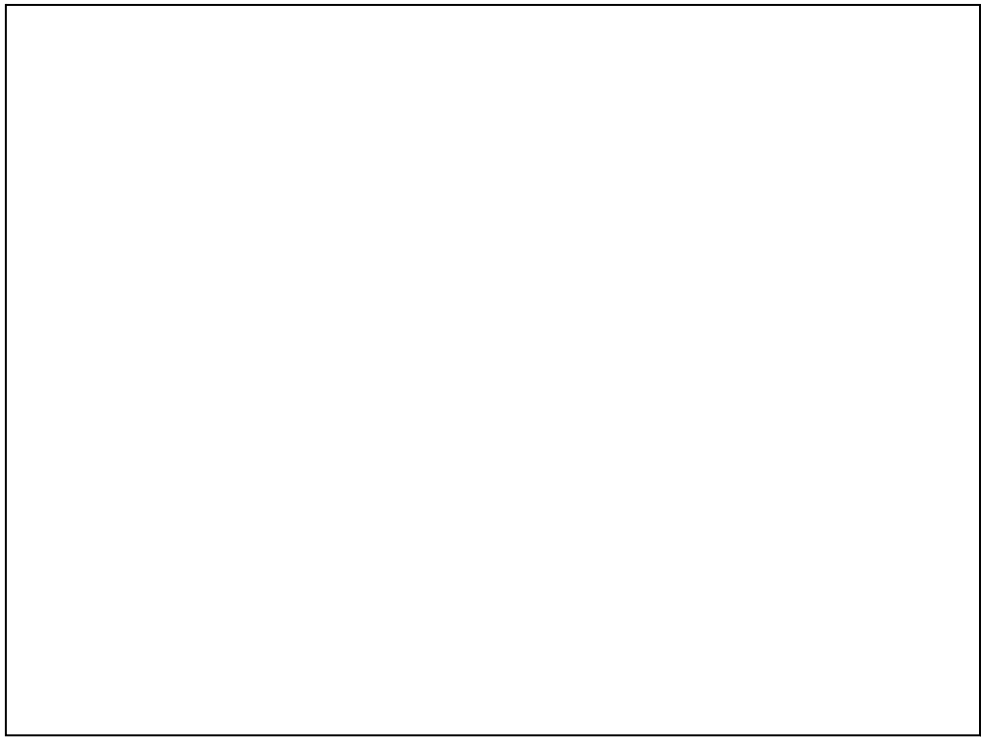


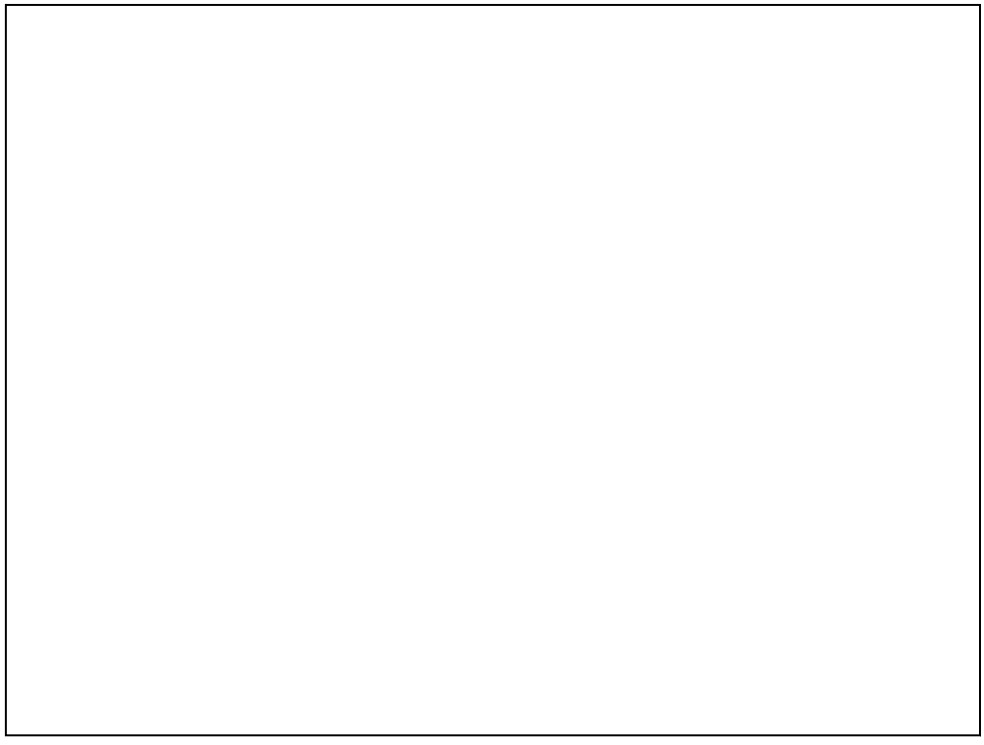


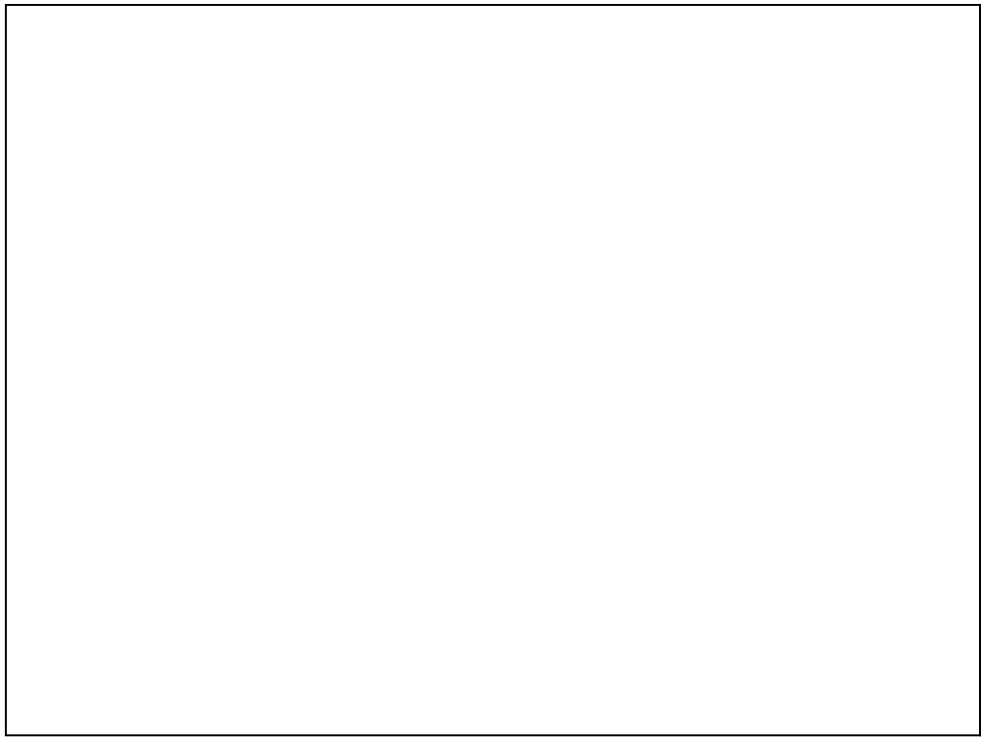


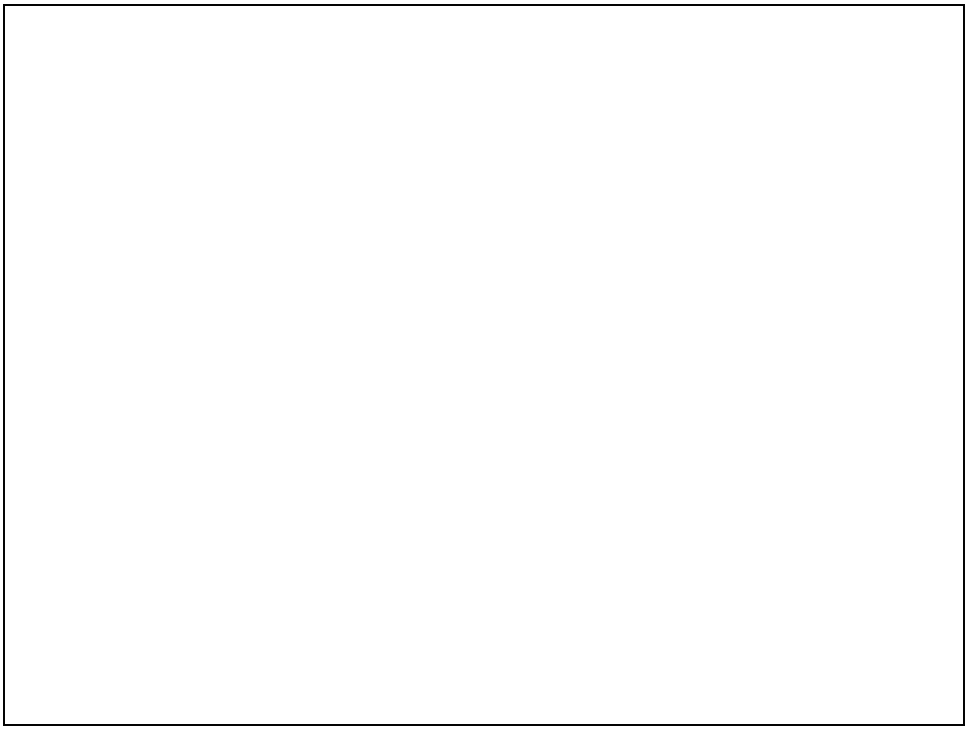


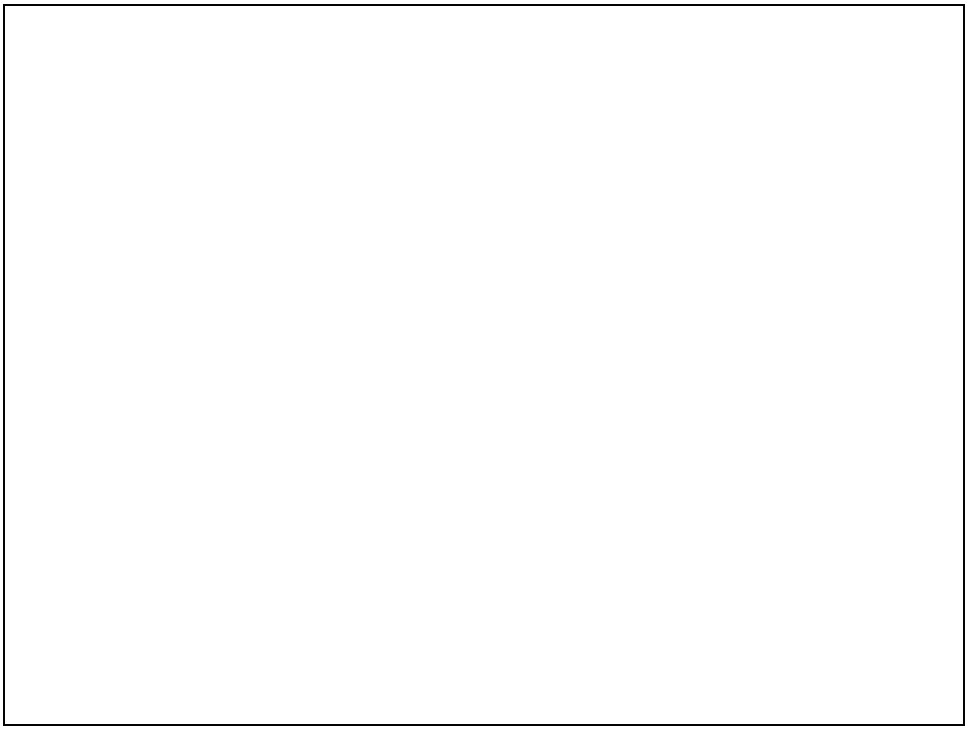




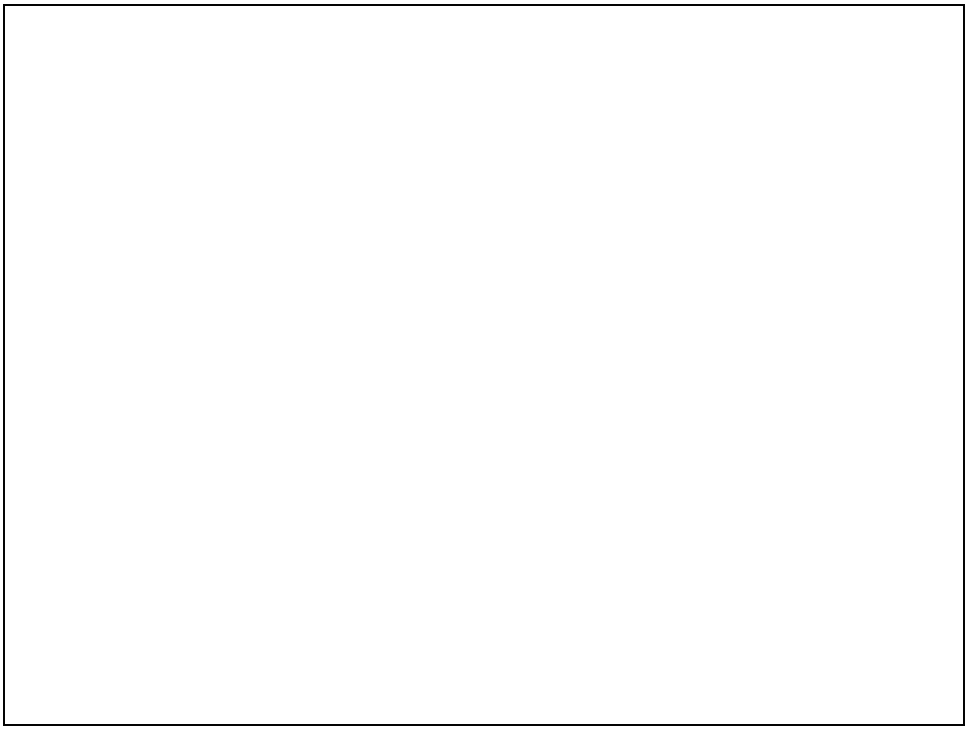




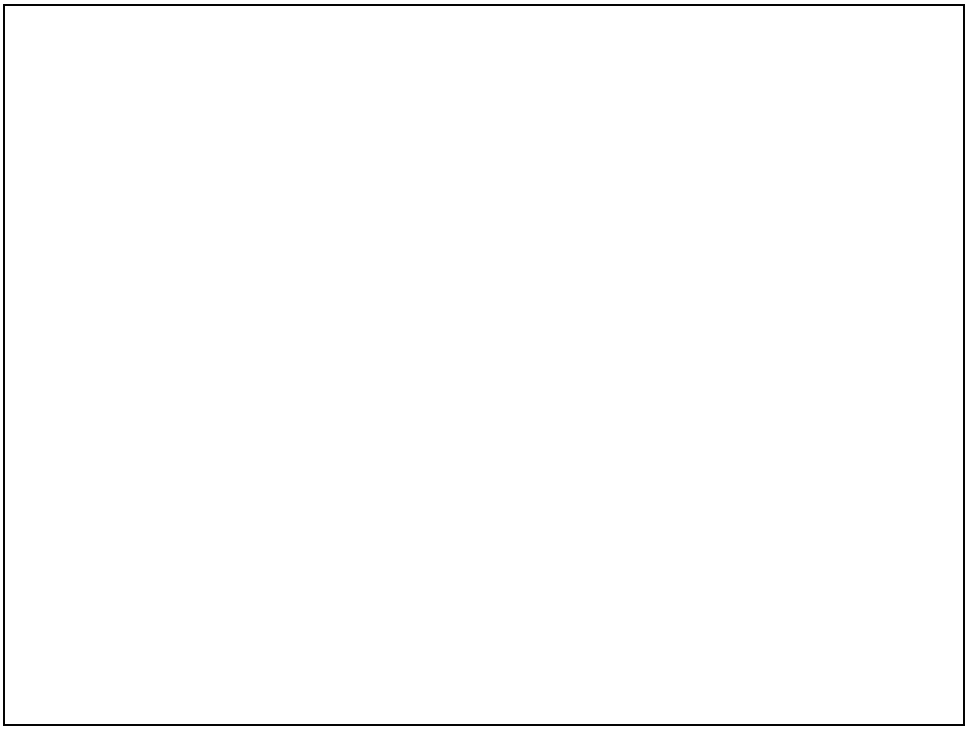




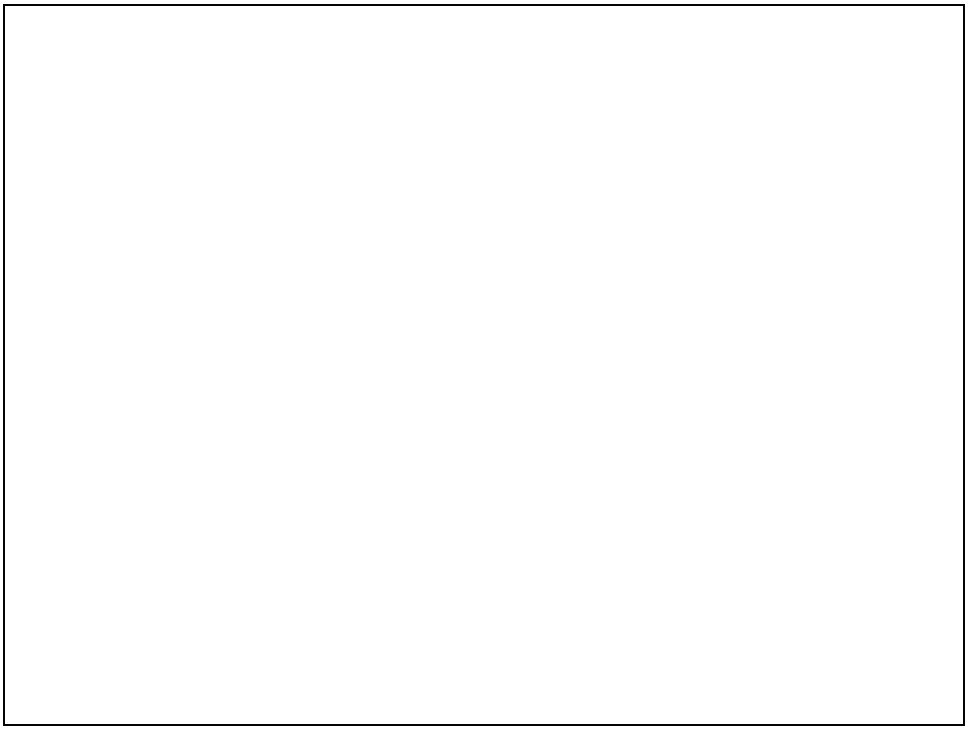
A



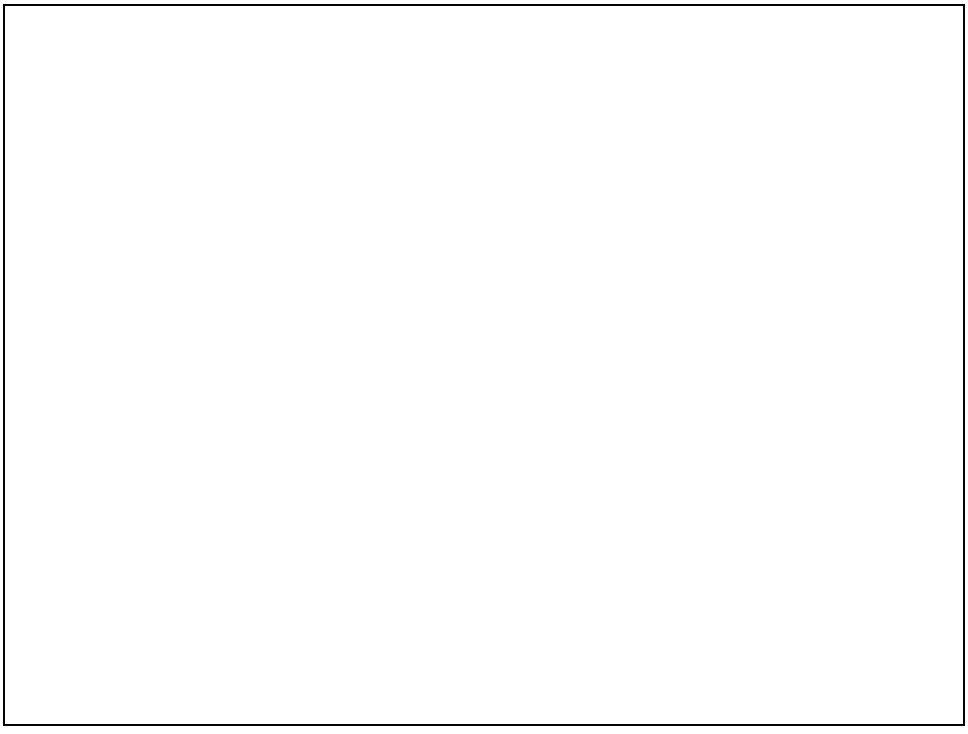
A



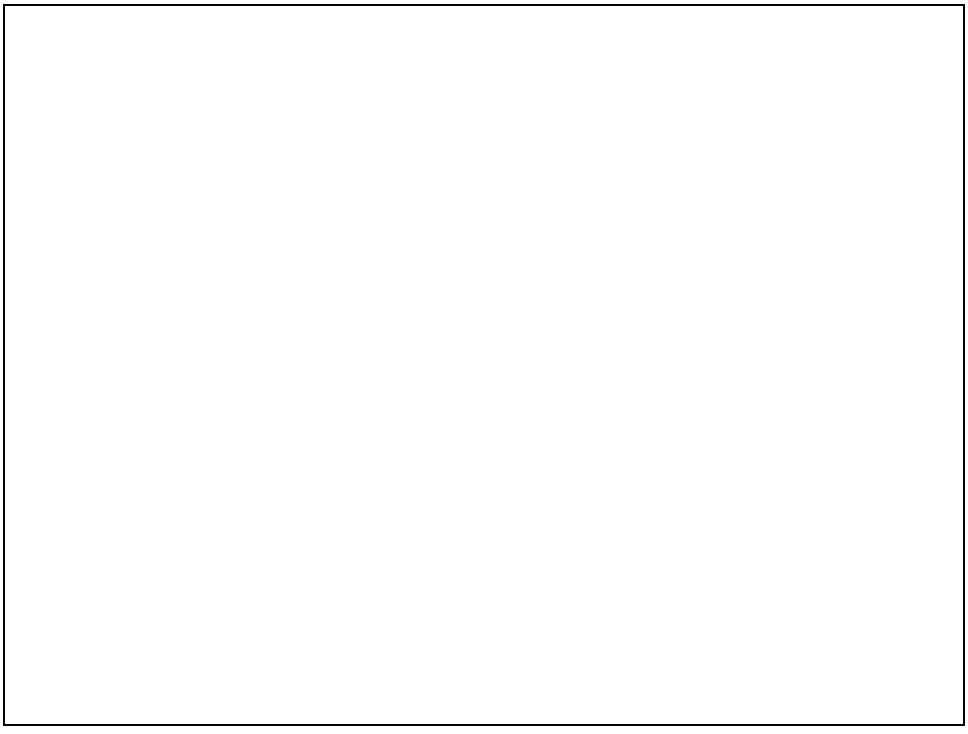
A

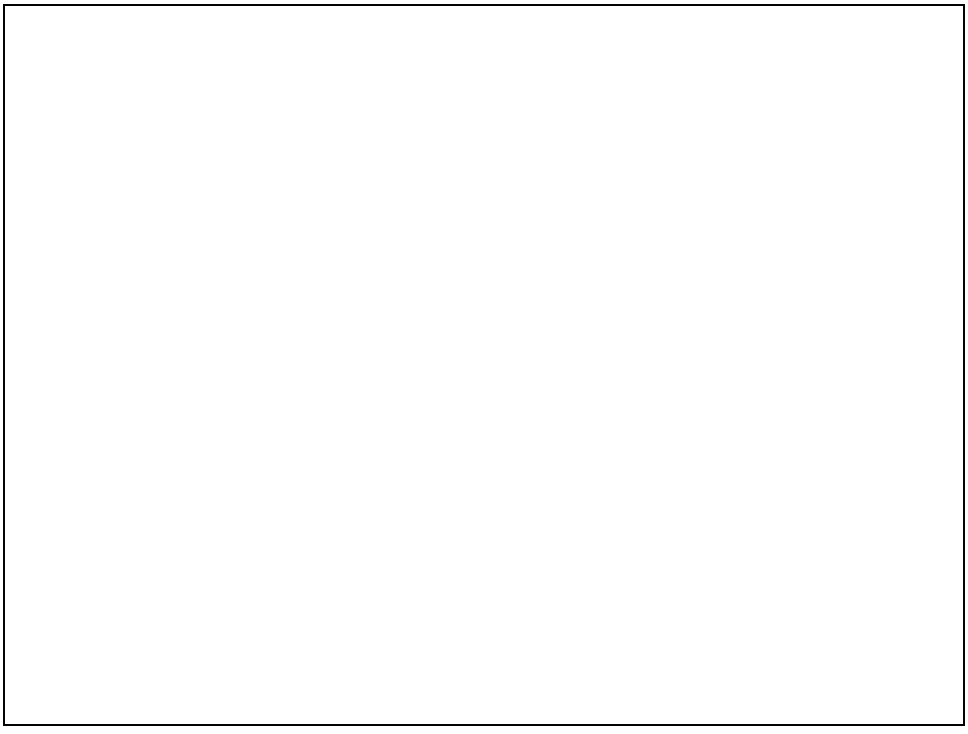


A

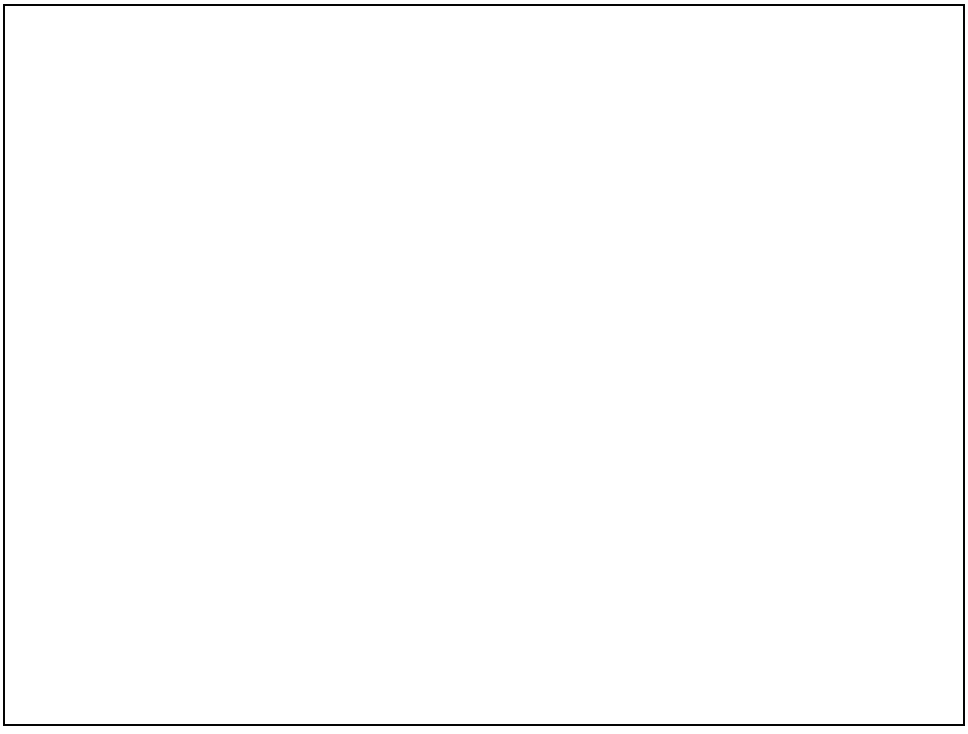


A





A



A