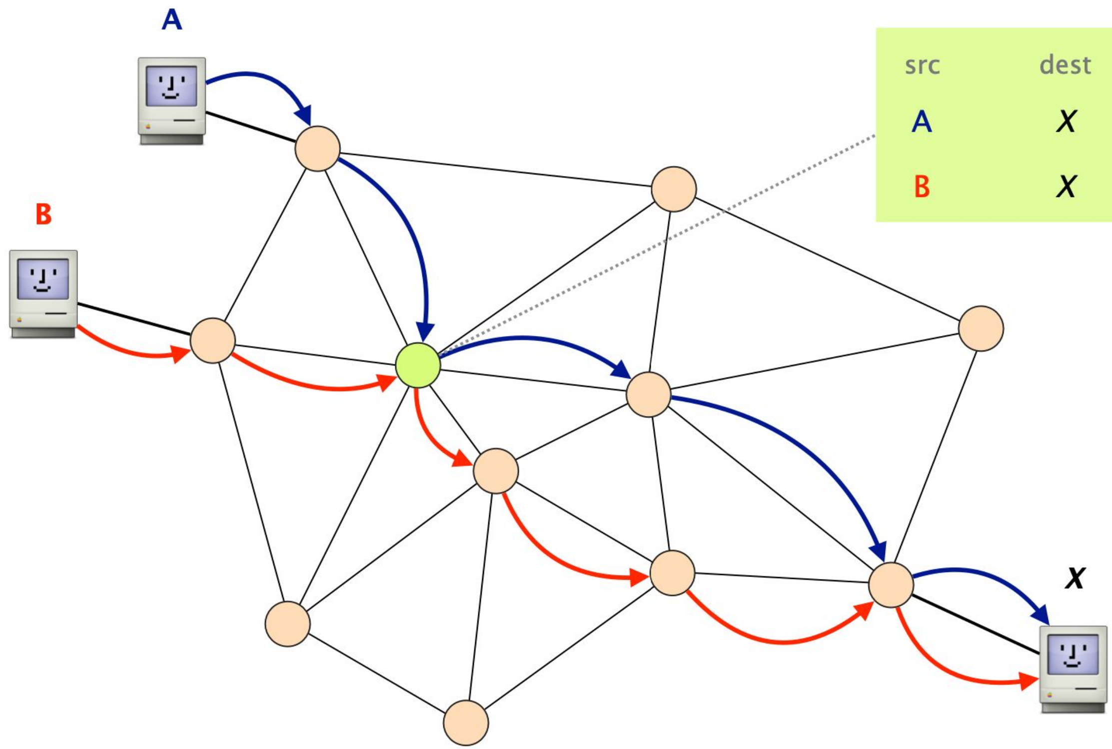
Forwarding vs Routing

	forwarding	routing
goal	directing packet to an outgoing link	computing the paths packets will follow
scope	local	network-wide
implem.	hardware usually	software usually
timescale	nanoseconds	milliseconds (hopefully)

Forwarding Depends on Destination, but Can Also Consider Other Criteria

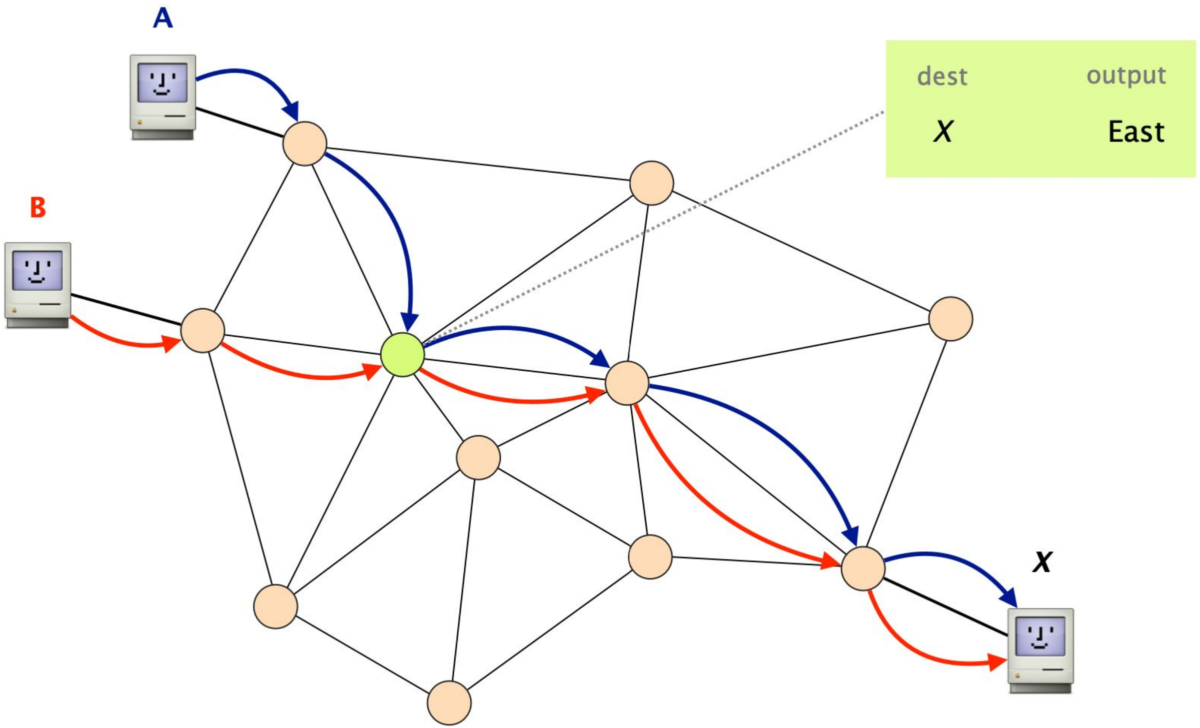
- Destination - Why is this mandatory?
- Source
- Input port
- Any other header field

Forwarding on Both Source and Destination - Paths from Different Sources can Differ

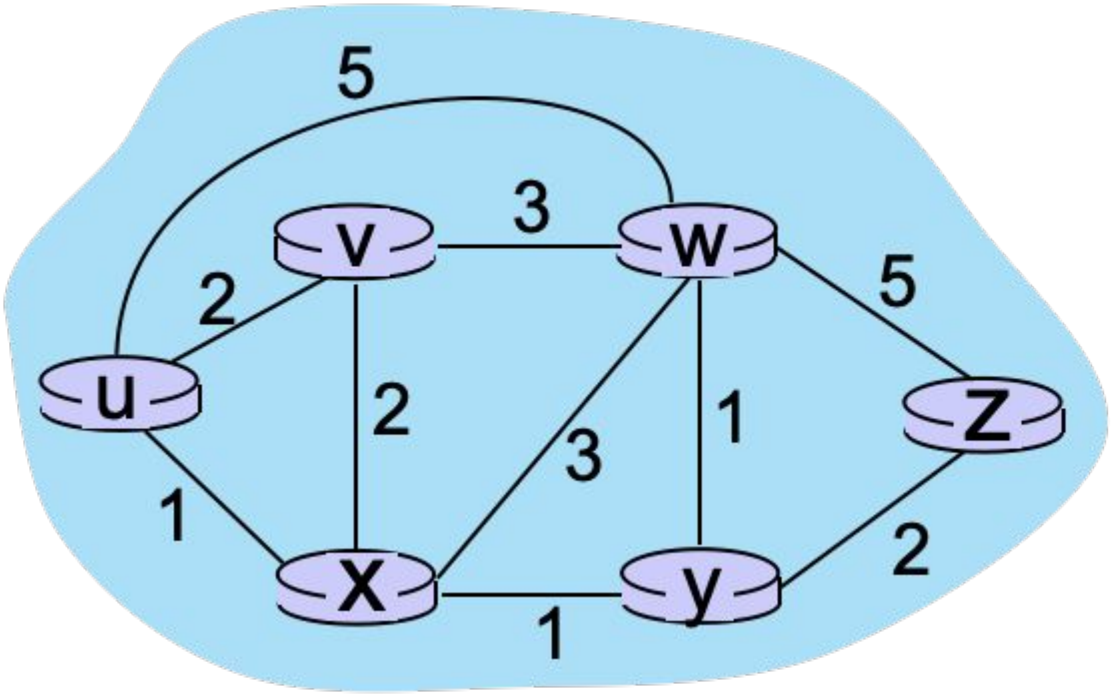


src	dest	output
A	*X*	East
B	*X*	South-East

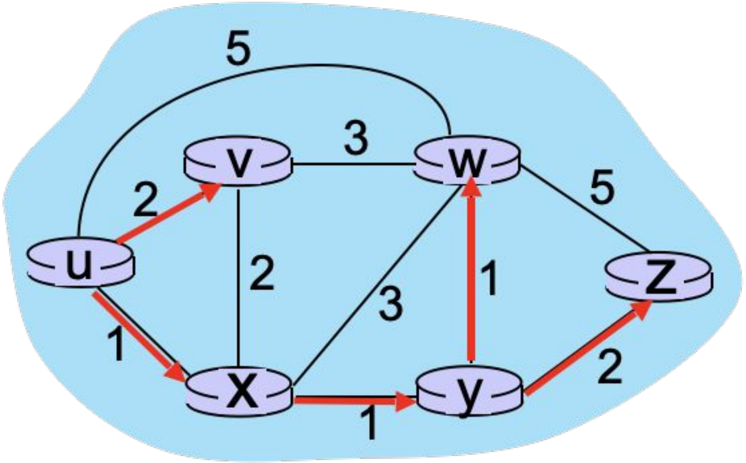
Destination-Based Routing, Once Paths from Sources Overlap They Remain the Same



Dijkstra’s Algorithm for Shortest Path Search



Forwarding Table Comes from Dijkstra’s Algorithm Results



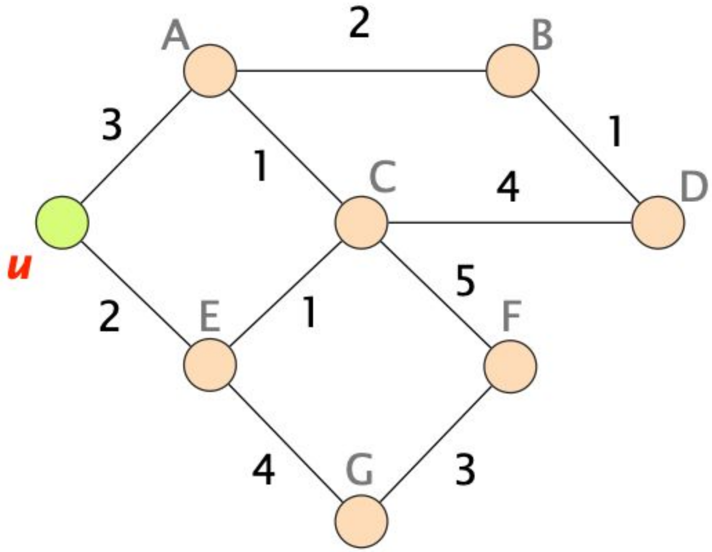
resulting forwarding table in u:

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

route from *u* to *v* directly

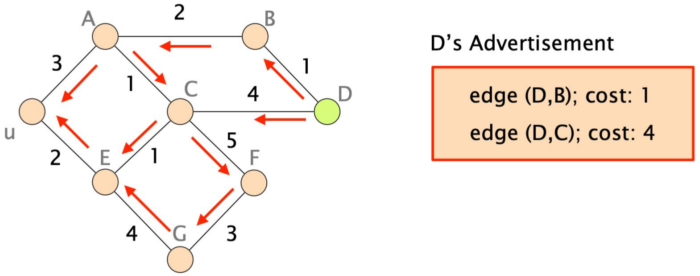
route from u to all other destinations via *x*

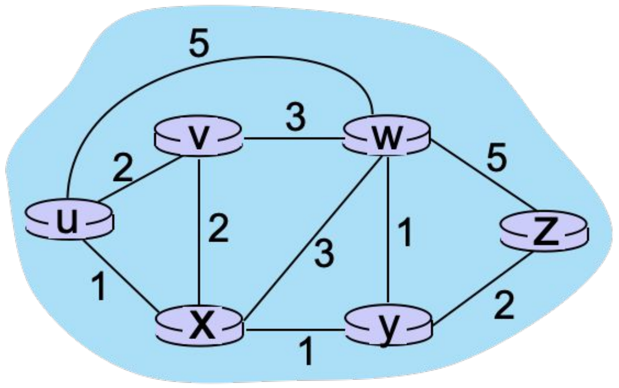
Dijkstra’s Algorithm for Shortest Path Search



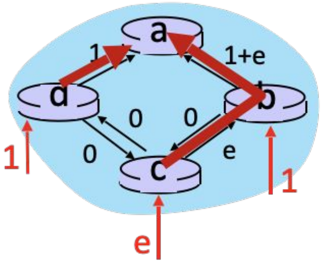
Dijkstra's Algorithm -> Link State Routing

- Each router floods its link state information to other n routers in order to generate a global view
- Updates are sent when things change, and only the difference is sent, not everything
- Any drawbacks you can think of?
- ```
  U: {v=2, x=1, w=5}
  V: {u=2, x=2, w=3}
  W: {v=3, u=5, x=3, y=1, z=5}
  X: {u=1, v=2, w=3, y=1}
  Y: {x=1, w=1, z=2}
  Z: {w=5, y=2}
  ```

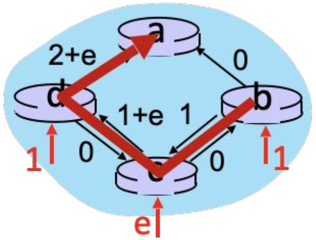



  ```

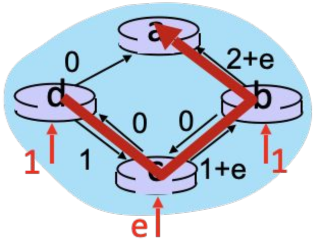
Dynamic Weights -> Route Oscillations



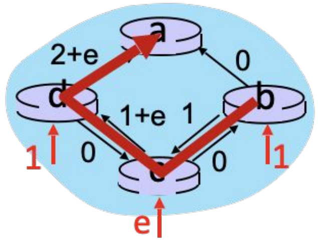
initially



given these costs,
find new routing....
resulting in new costs



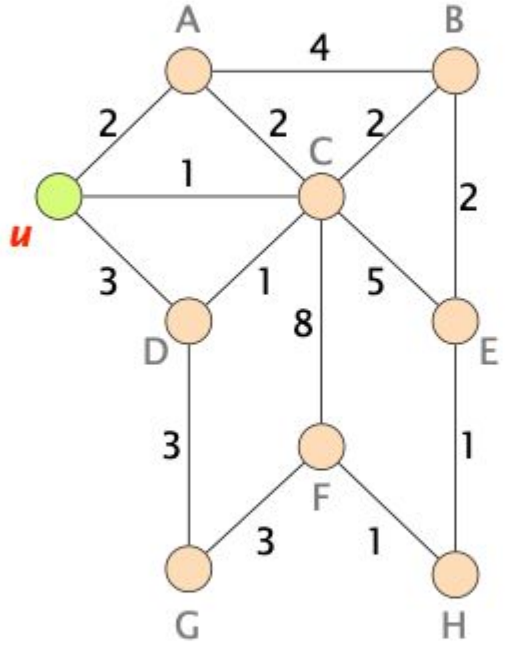
given these costs,
find new routing....
resulting in new costs



given these costs,
find new routing....
resulting in new costs

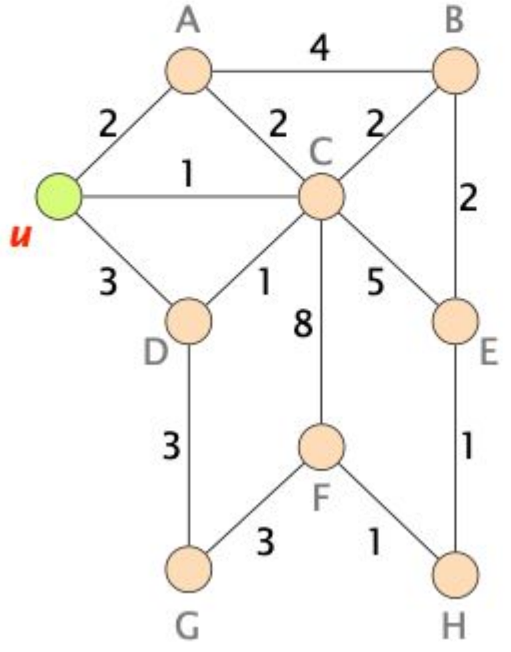
Dijkstra’s Example

Starting from node u, (i) manually compute Dijkstra’s algorithm, and then (ii) list the obtained shortest-paths from u to each of the other nodes. For computing Dijkstra’s algorithm, you can use the table below. The algorithm follows the one discussed in the lecture. If several nodes could next be added to node set S, select the node that comes first in the alphabet.



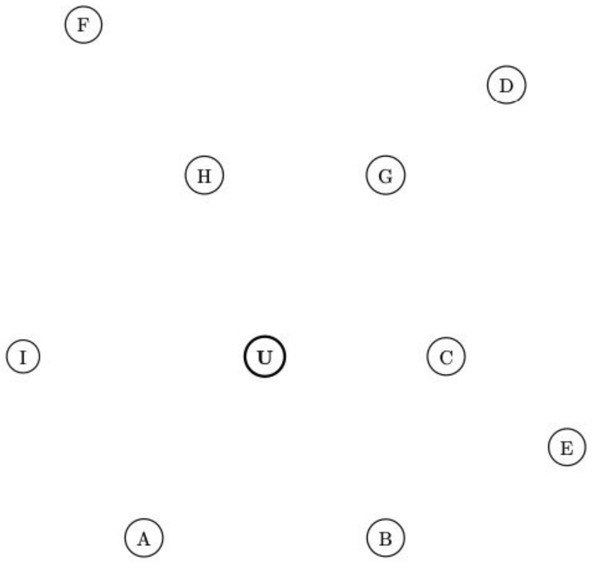
Dijkstra's Example

Starting from node u, (i) manually compute Dijkstra's algorithm, and then (ii) list the obtained shortest-paths from u to each of the other nodes. For computing Dijkstra's algorithm, you can use the table below. The algorithm follows the one discussed in the lecture. If several nodes could next be added to node set S, select the node that comes first in the alphabet.



Node	Path	∑(weights)
A	u - A	2
B	u - C - B	3
C	u - C	1
D	u - C - D	2
E	u - C - B - E	5
F	u - C - B - E - H - F	7
G	u - C - D - G	5
H	u - C - B - E - H	6

Reverse Dijkstra is Possible From Results

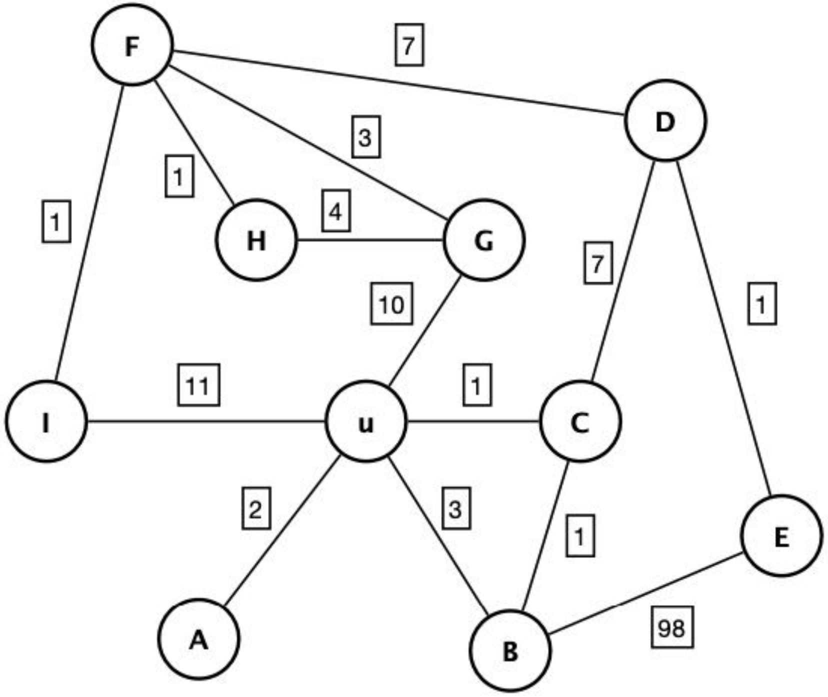


A network consisting of 10 nodes with unknown links and link weights.

#	U	A	B	C	D	E	F	G	H	I
1	0	2	3	1	-	-	-	10	-	11
2	0	2	2	1	8	-	-	10	-	11
3	0	2	2	1	8	-	-	10	-	11
4	0	2	2	1	8	100	-	10	-	11
5	0	2	2	1	8	9	15	10	-	11
6	0	2	2	1	8	9	15	10	-	11
7	0	2	2	1	8	9	13	10	14	11
8	0	2	2	1	8	9	12	10	14	11
9	0	2	2	1	8	9	12	10	13	11
10	0	2	2	1	8	9	12	10	13	11

For each iteration (1 to 10) the table shows the shortest path found by Dijkstra's algorithm performed on node **U** towards all other nodes.

Reverse Dijkstra is Possible From Results - Solution



#	U	A	B	C	D	E	F	G	H	I
1	0	2	3	1	-	-	-	10	-	11
2	0	2	2	1	8	-	-	10	-	11
3	0	2	2	1	8	-	-	10	-	11
4	0	2	2	1	8	100	-	10	-	11
5	0	2	2	1	8	9	15	10	-	11
6	0	2	2	1	8	9	15	10	-	11
7	0	2	2	1	8	9	13	10	14	11
8	0	2	2	1	8	9	12	10	14	11
9	0	2	2	1	8	9	12	10	13	11
10	0	2	2	1	8	9	12	10	13	11

For each iteration (1 to 10) the table shows the shortest path found by Dijkstra's algorithm performed on node **U** towards all other nodes.