

Dijkstra's algorithm

Douglas Wilhelm Harder, M.Math. LEL

Department of Electrical and Computer Engineering
University of Waterloo
Waterloo, Ontario, Canada

ece.uwaterloo.ca

dwharder@alumni.uwaterloo.ca

© 2006-2013 by Douglas Wilhelm Harder. Some rights reserved.

Outline

This topic will:

- Describe Dijkstra's algorithm for finding a shortest path from a single source vertex
- S

Dijkstra's algorithm

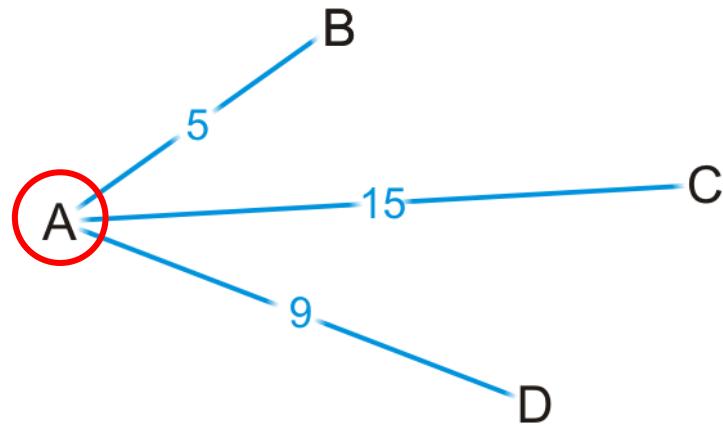
Dijkstra's algorithm solves the single-source shortest path problem

- It is very similar to Prim's algorithm
- Assumption: all the weights are positive

Strategy

Suppose you are at vertex A

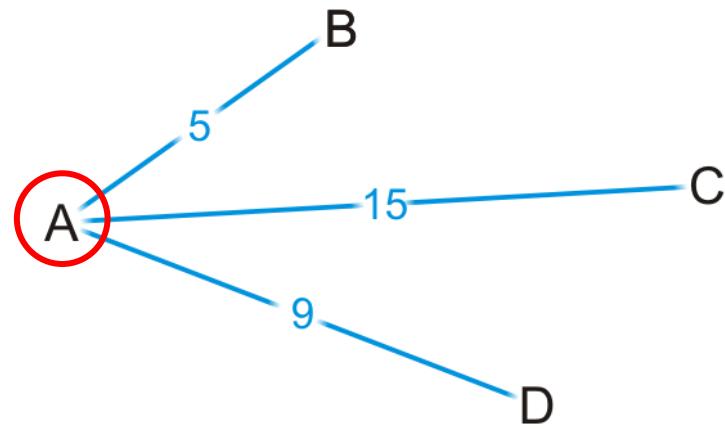
- You are aware of all vertices adjacent to it
- This information is either in an adjacency list or adjacency matrix



Strategy

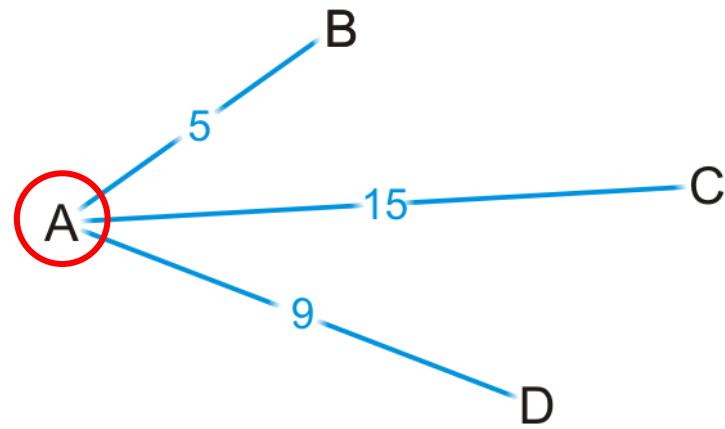
Is 5 the shortest distance to B via the edge (A, B)?

- Why or why not?



Strategy

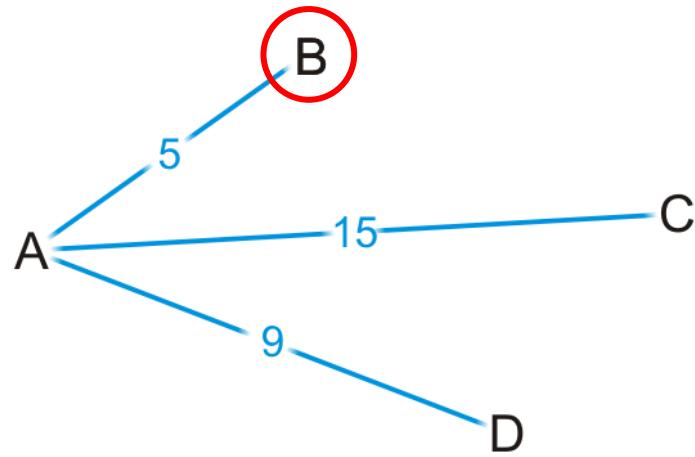
Are you guaranteed that the shortest path to C is (A, C), or that (A, D) is the shortest path to vertex D?



Strategy

We accept that (A, B) is the shortest path to vertex B from A

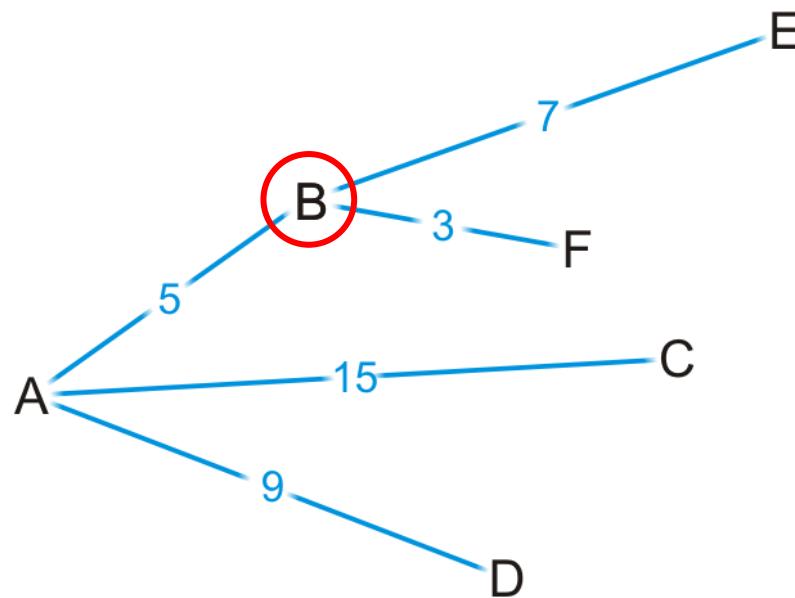
- Let's see where we can go from B



Strategy

By some simple arithmetic, we can determine that

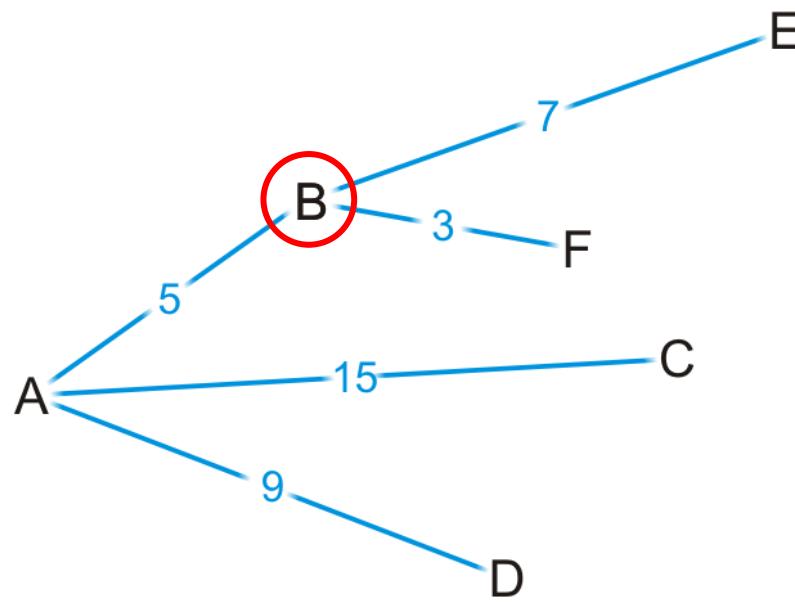
- There is a path (A, B, E) of length $5 + 7 = 12$
- There is a path (A, B, F) of length $5 + 3 = 8$



Strategy

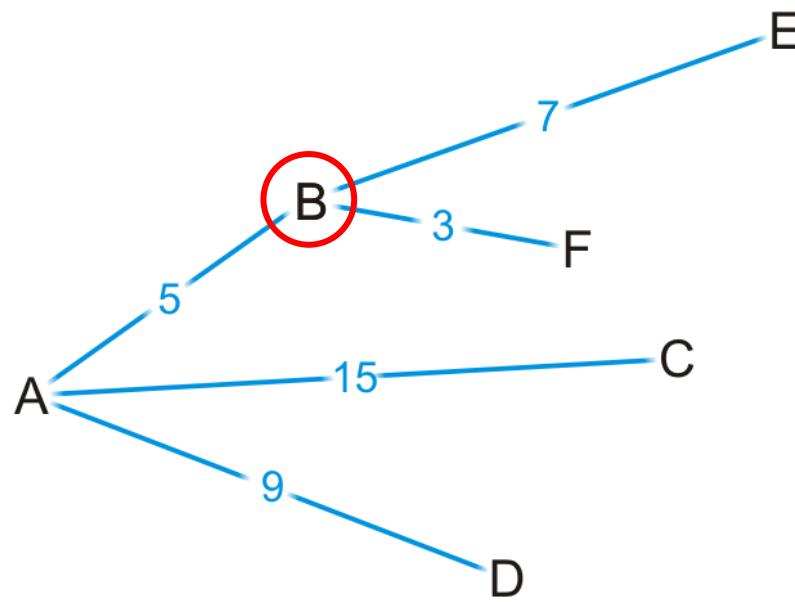
Is (A, B, F) is the shortest path from vertex A to F?

- Why or why not?



Strategy

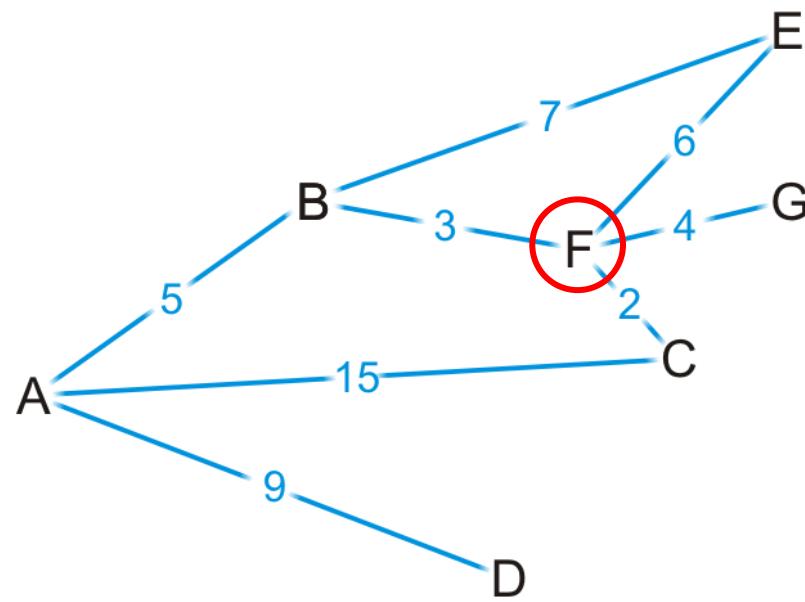
Are we guaranteed that any other path we are currently aware of is also going to be the shortest path?



Strategy

Okay, let's visit vertex F

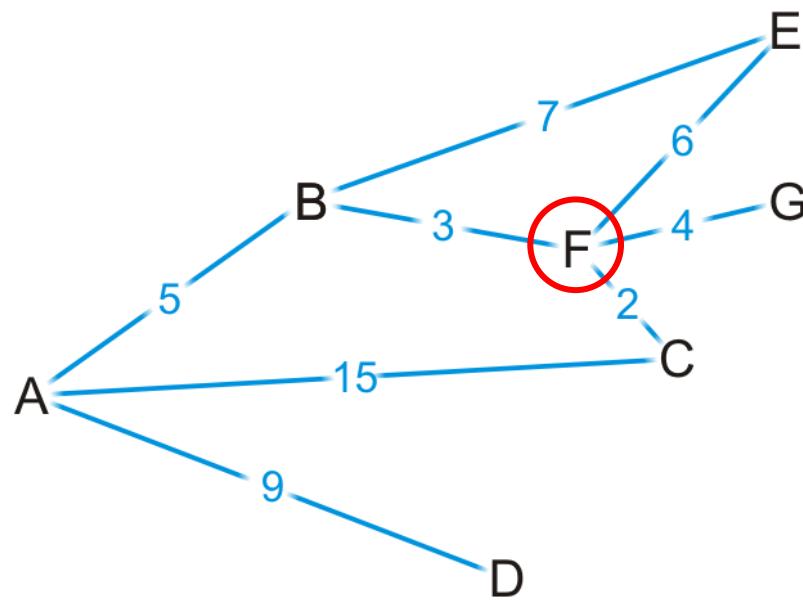
- We know the shortest path is (A, B, F) and it's of length 8



Strategy

There are three edges exiting vertex F, so we have paths:

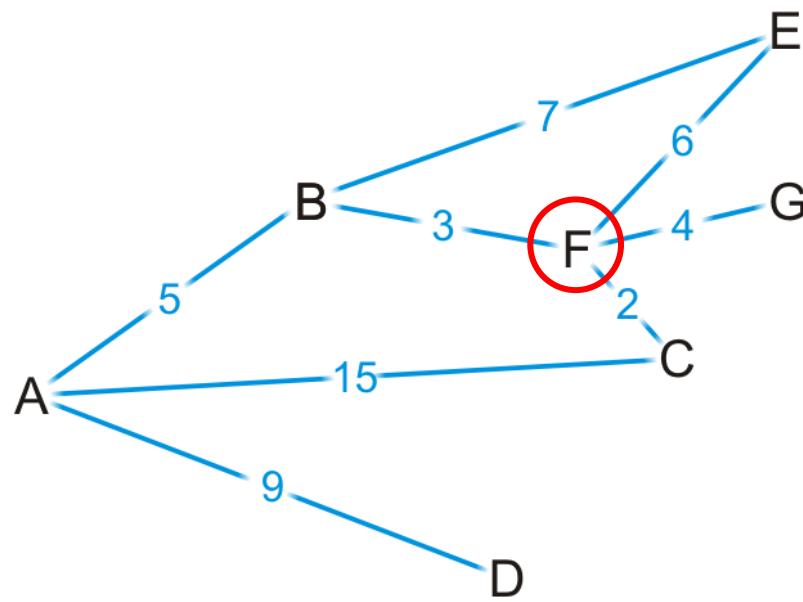
- (A, B, F, E) of length $8 + 6 = 14$
- (A, B, F, G) of length $8 + 4 = 12$
- (A, B, F, C) of length $8 + 2 = 10$



Strategy

By observation:

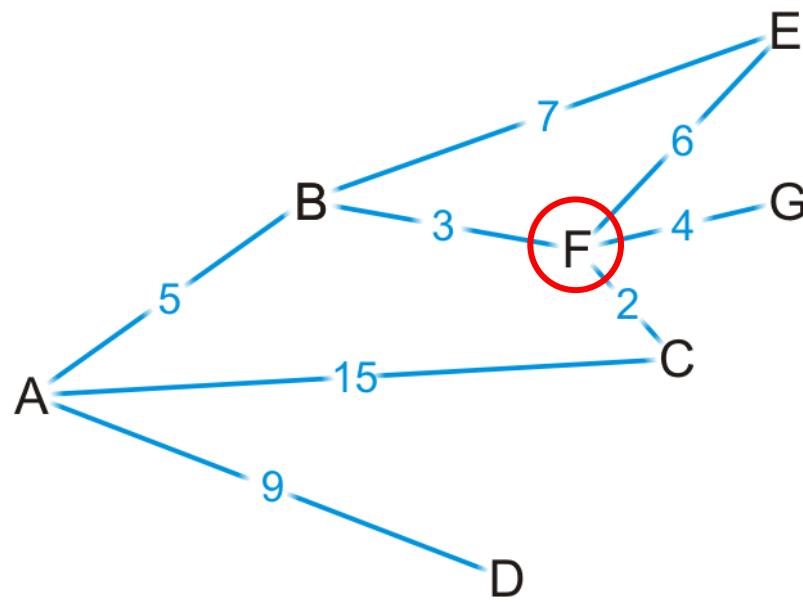
- The path (A, B, F, E) is longer than (A, B, E)
- The path (A, B, F, C) is shorter than the path (A, C)



Strategy

At this point, we've discovered the shortest paths to:

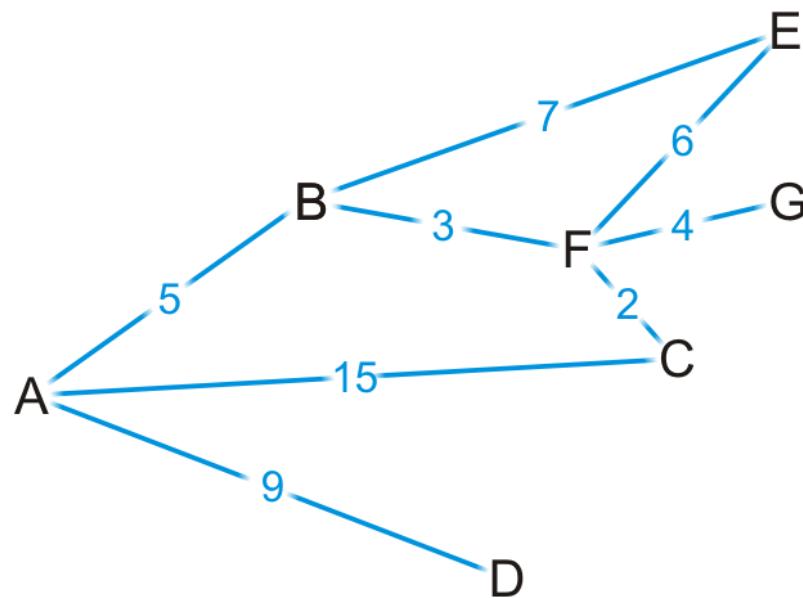
- Vertex B: (A, B) of length 5
- Vertex F: (A, B, F) of length 8



Strategy

At this point, we have the shortest distances to B and F

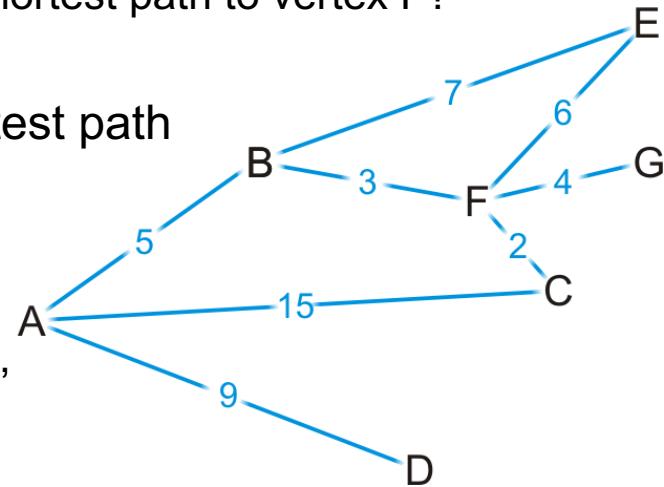
- Which remaining vertex are we currently guaranteed to have the shortest distance to?



Dijkstra's algorithm

Like Prim's algorithm, we initially don't know the distance to any vertex except the initial vertex

- We require an array of distances, all initialized to infinity except for the source vertex, which is initialized to 0
- Each time we visit a vertex, we will examine all adjacent vertices
 - We need to track visited vertices—a Boolean table of size $|V|$
- Do we need to track the shortest path to each vertex?
 - That is, do I have to store (A, B, F) as the shortest path to vertex F?
- We really only have to record that the shortest path to vertex F came from vertex B
 - We would then determine that the shortest path to vertex B came from vertex A
 - Thus, we need an array of previous vertices, all initialized to null



Dijkstra's algorithm

Thus, we will iterate $|V|$ times:

- Find that unvisited vertex v that has a minimum distance to it
- Mark it as having been visited
- Consider every adjacent vertex w that is unvisited:
 - Is the distance to v plus the weight of the edge (v, w) less than our currently known shortest distance to w
 - If so, update the shortest distance to w and record v as the previous pointer
- Continue iterating until all vertices are visited or all remaining vertices have a distance to them of infinity

Example

Consider the game of *Risk* from Parker Brothers

- A game of world domination
- The world is divided into 42 connected regions



Example

Consider the game of *Risk* from Parker Brothers

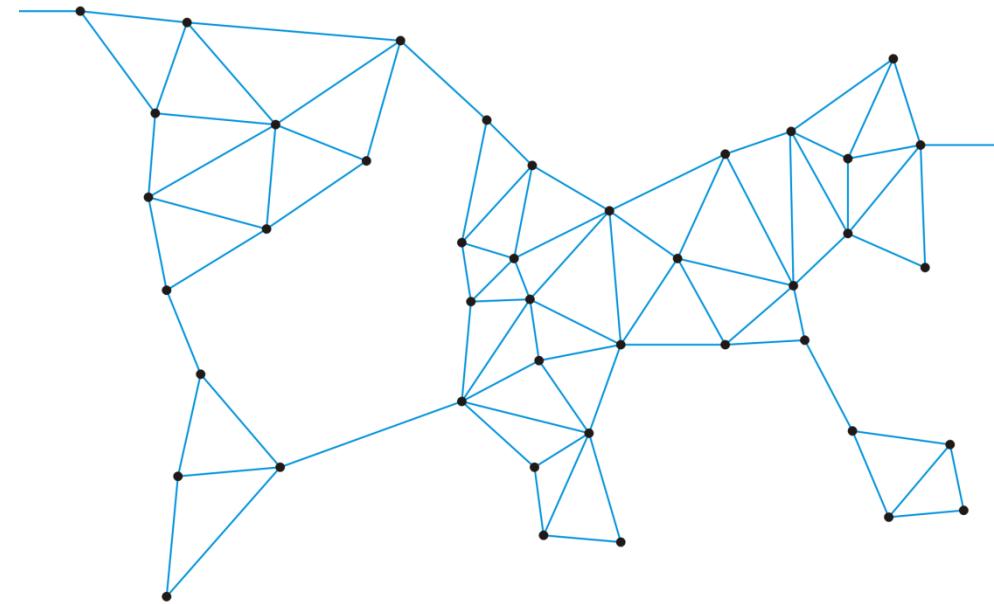
- A game of world domination
- The world is divided into 42 connected regions
- The regions are vertices and edges indicate adjacent regions



Example

Consider the game of *Risk* from Parker Brothers

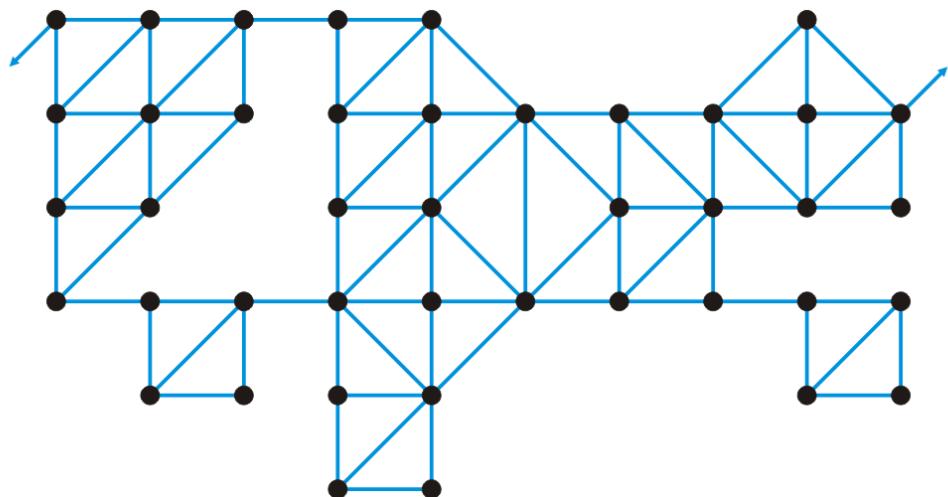
- A game of world domination
- The world is divided into 42 connected regions
- The regions are vertices and edges indicate adjacent regions
- The graph is sufficient to describe the game



Example

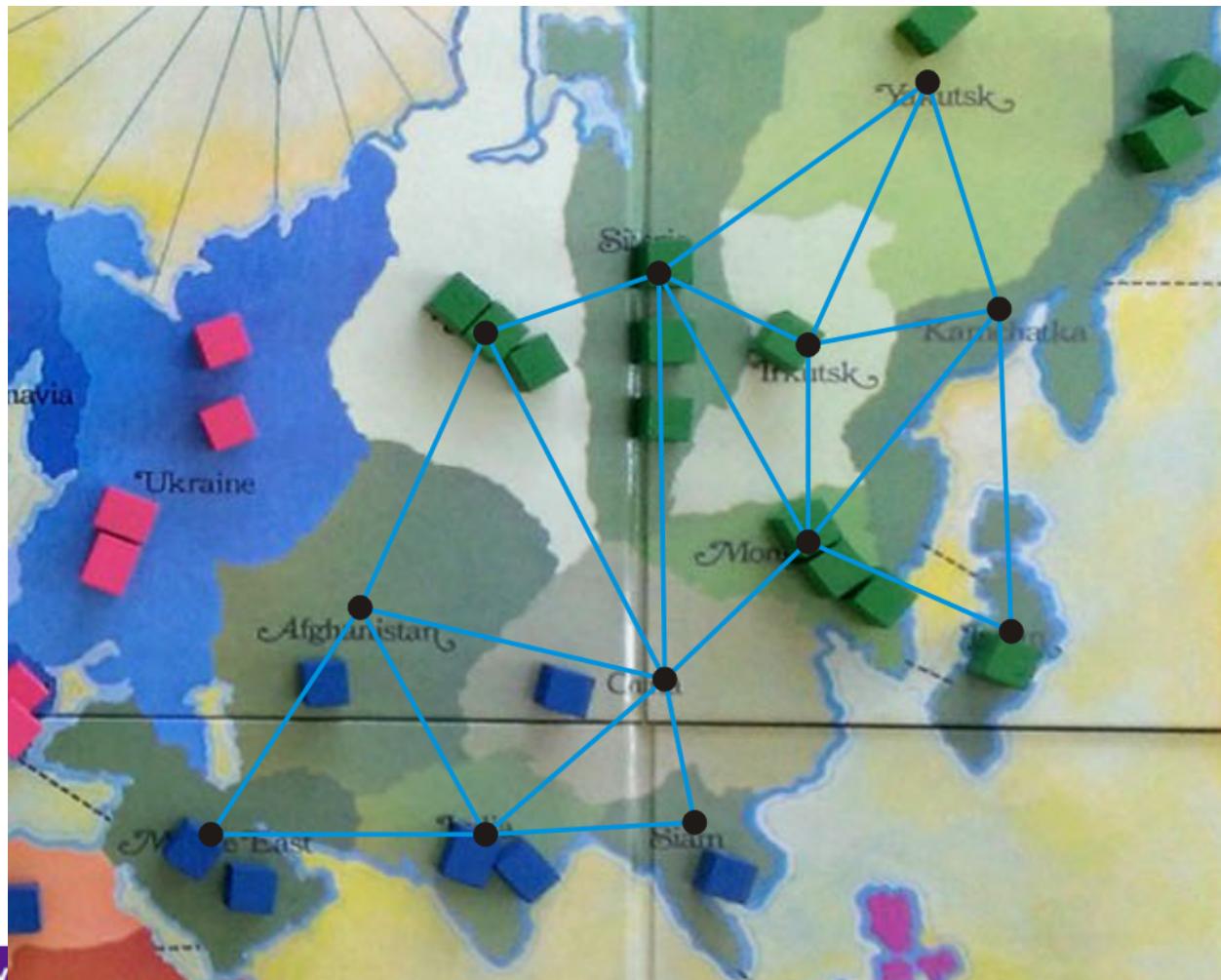
Consider the game of *Risk* from Parker Brothers

- Here is a more abstract representation of the game board
- Suddenly, it's less interesting: "I've conquered the graph!"



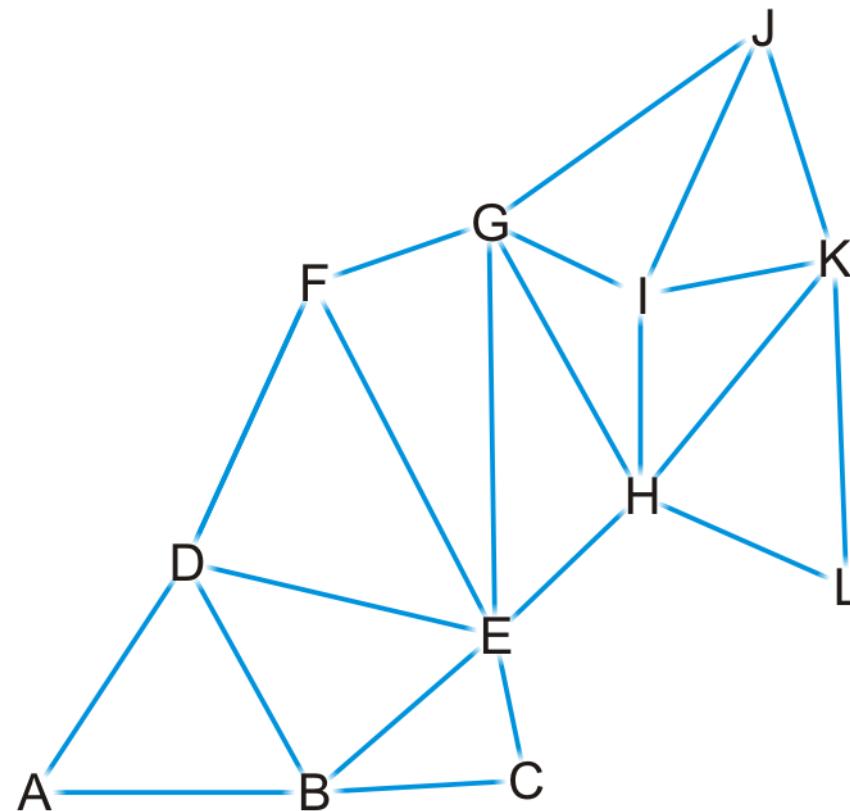
Example

We'll focus on Asia



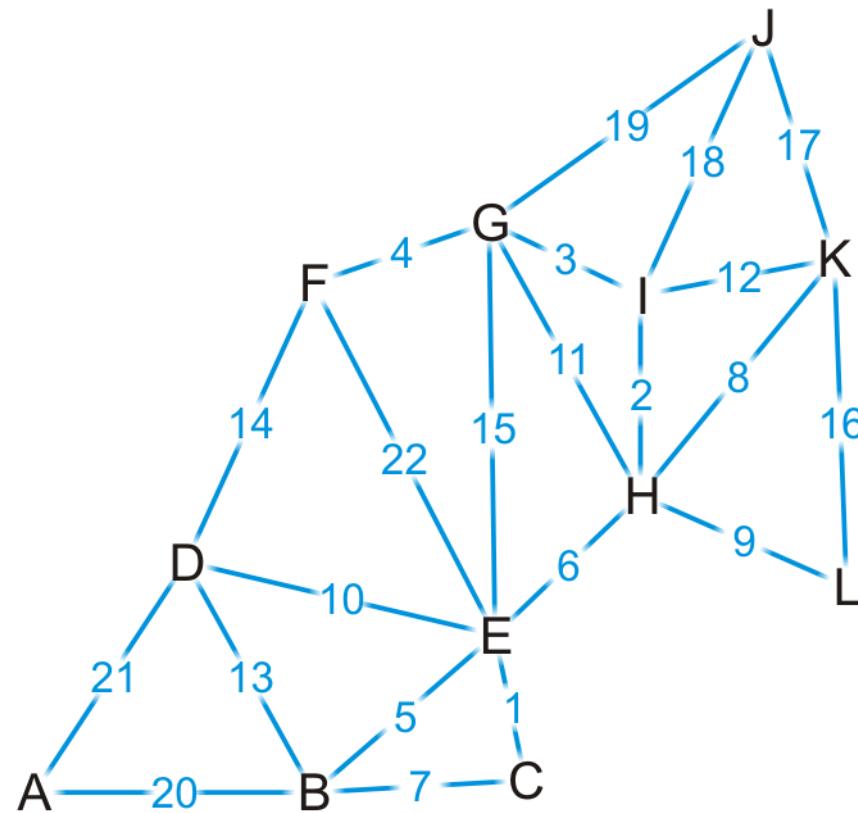
Example

Here is our abstract representation



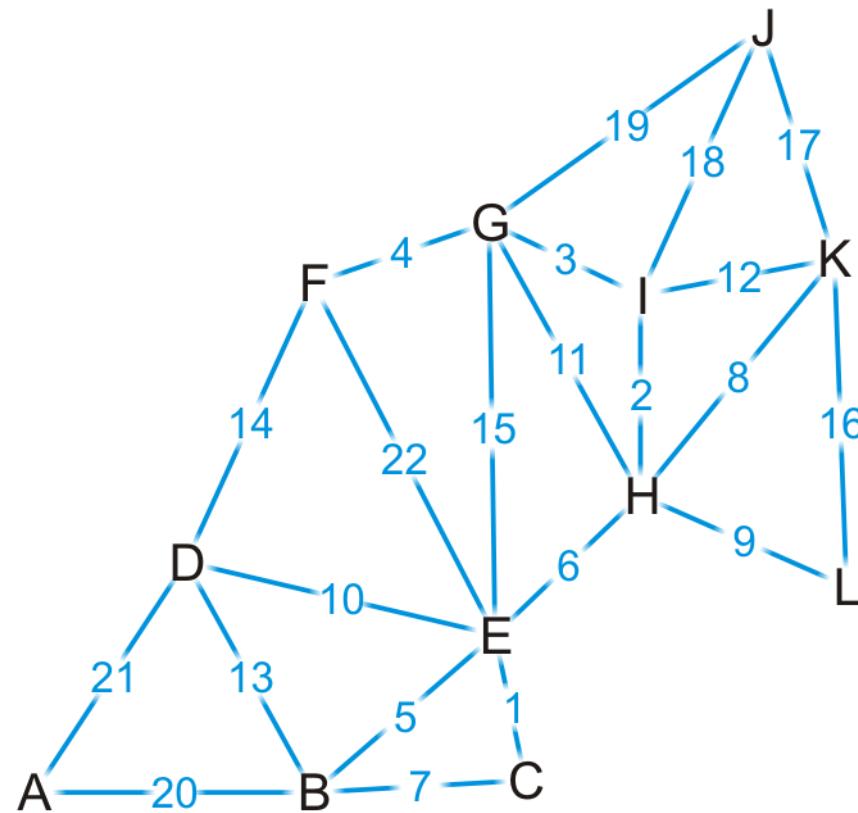
Example

Let us give a weight to each of the edges



Example

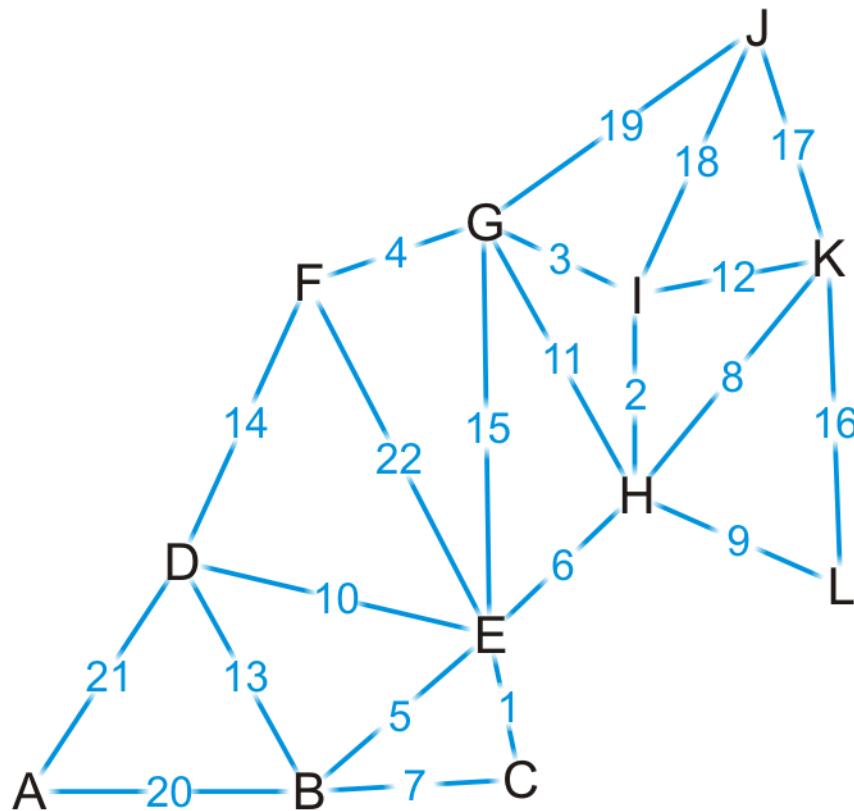
Find the shortest distance from Kamchatka (K) to every other region



Example

We set up our table

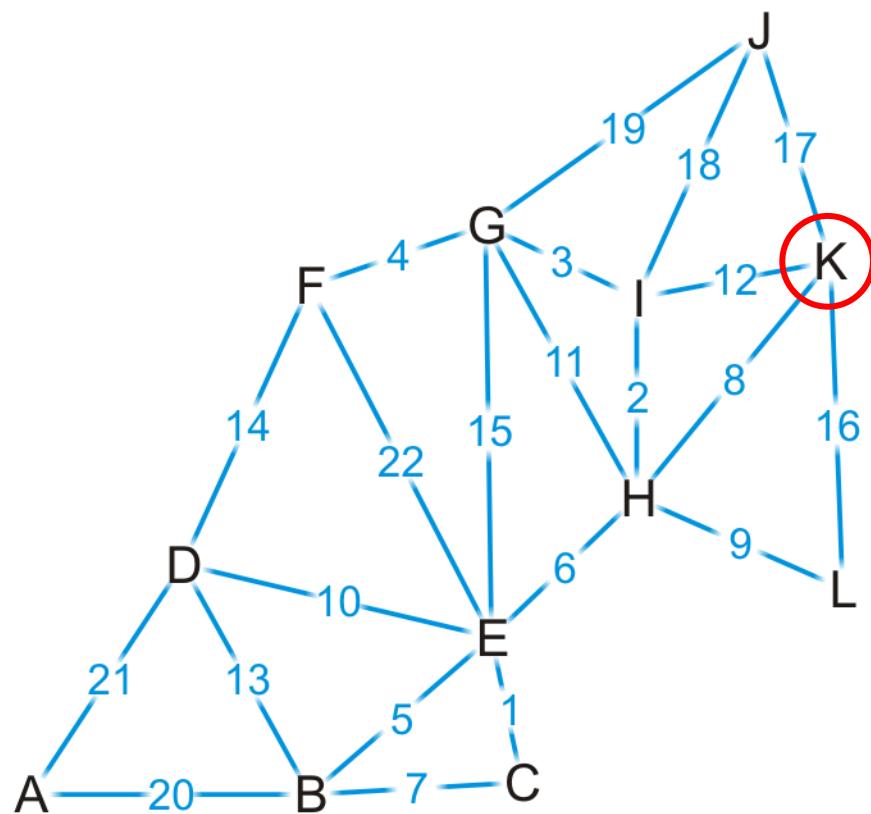
- Which unvisited vertex has the minimum distance to it?



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	F	∞	\emptyset
I	F	∞	\emptyset
J	F	∞	\emptyset
K	F	0	\emptyset
L	F	∞	\emptyset

Example

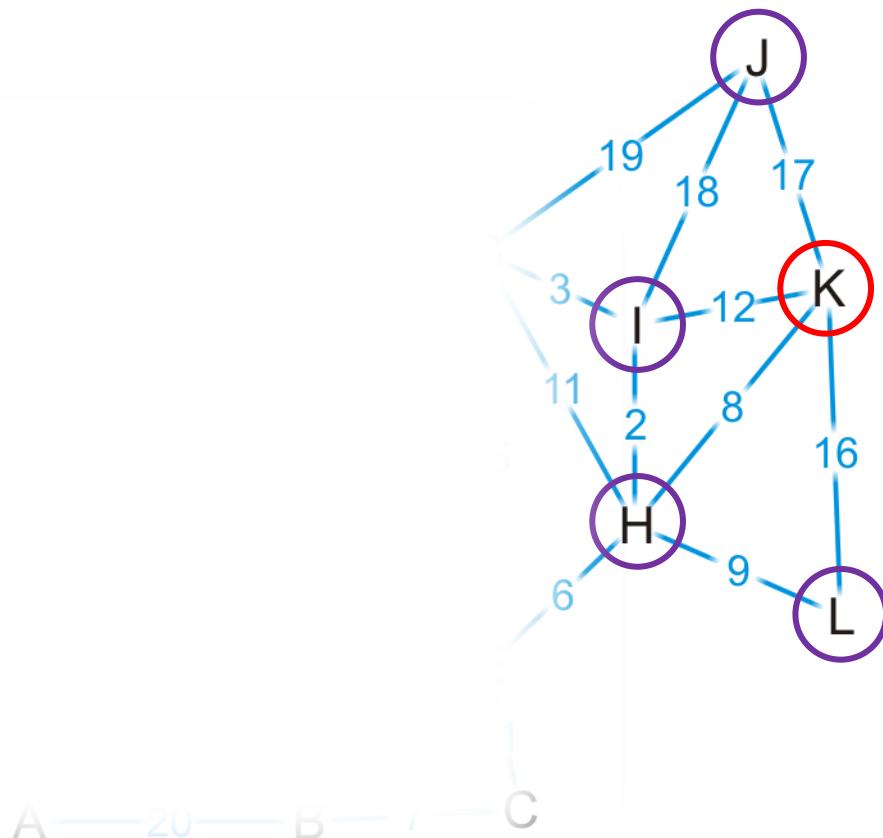
We visit vertex K



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	F	∞	\emptyset
I	F	∞	\emptyset
J	F	∞	\emptyset
K	T	0	\emptyset
L	F	∞	\emptyset

Example

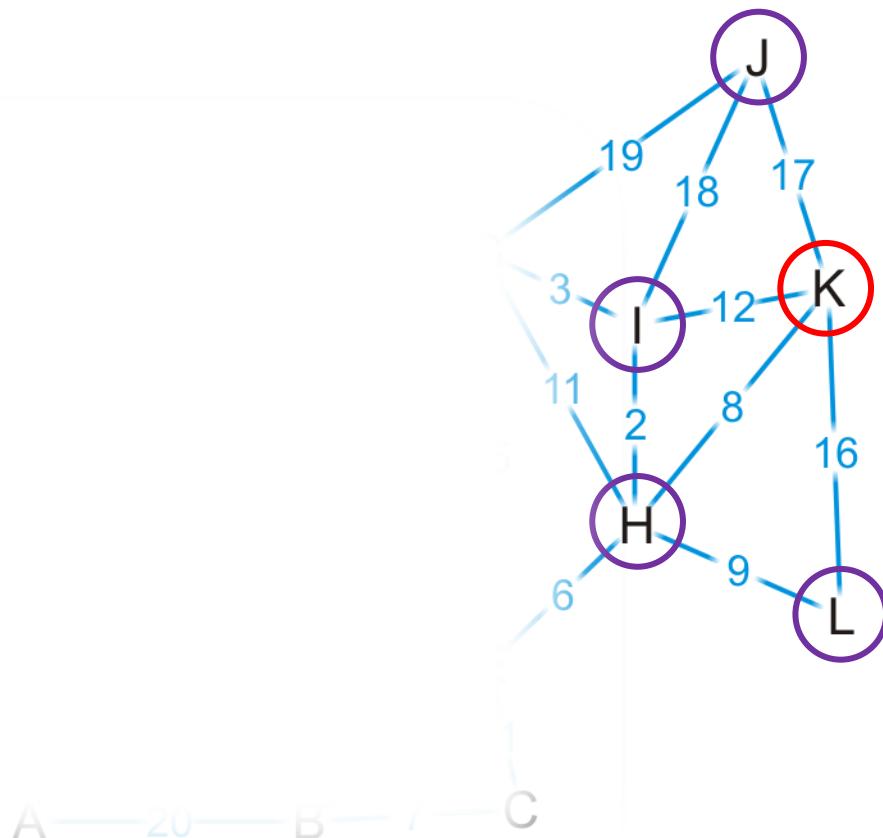
Vertex K has four neighbors: H, I, J and L



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	F	∞	\emptyset
I	F	∞	\emptyset
J	F	∞	\emptyset
K	T	0	\emptyset
L	F	∞	\emptyset

Example

We have now found at least one path to each of these vertices

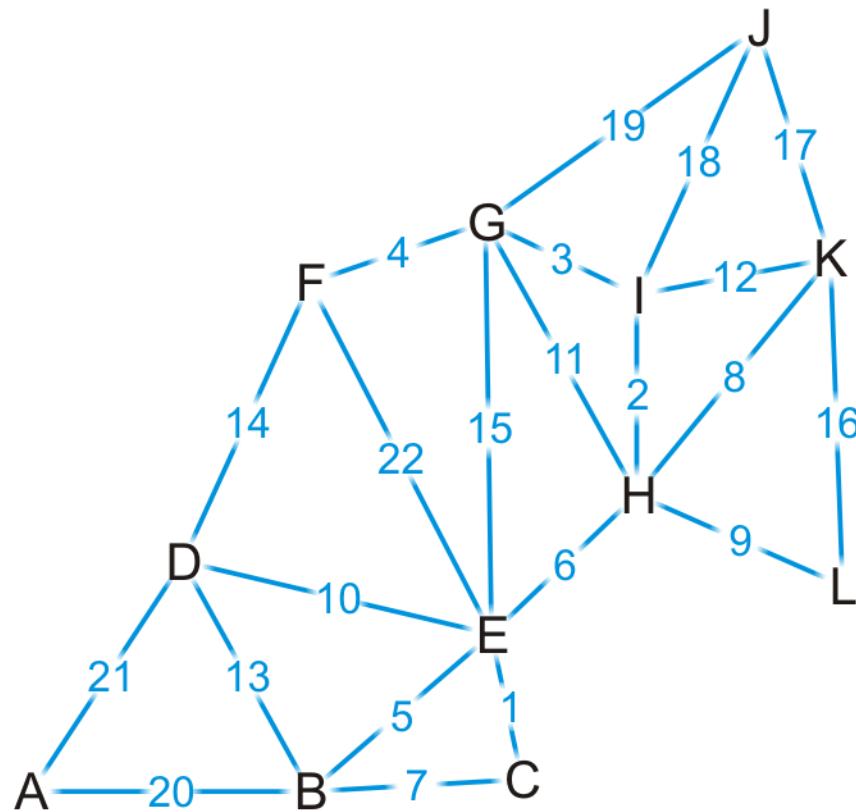


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	F	8	K
I	F	12	K
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

We're finished with vertex K

- To which vertex are we now guaranteed we have the shortest path?

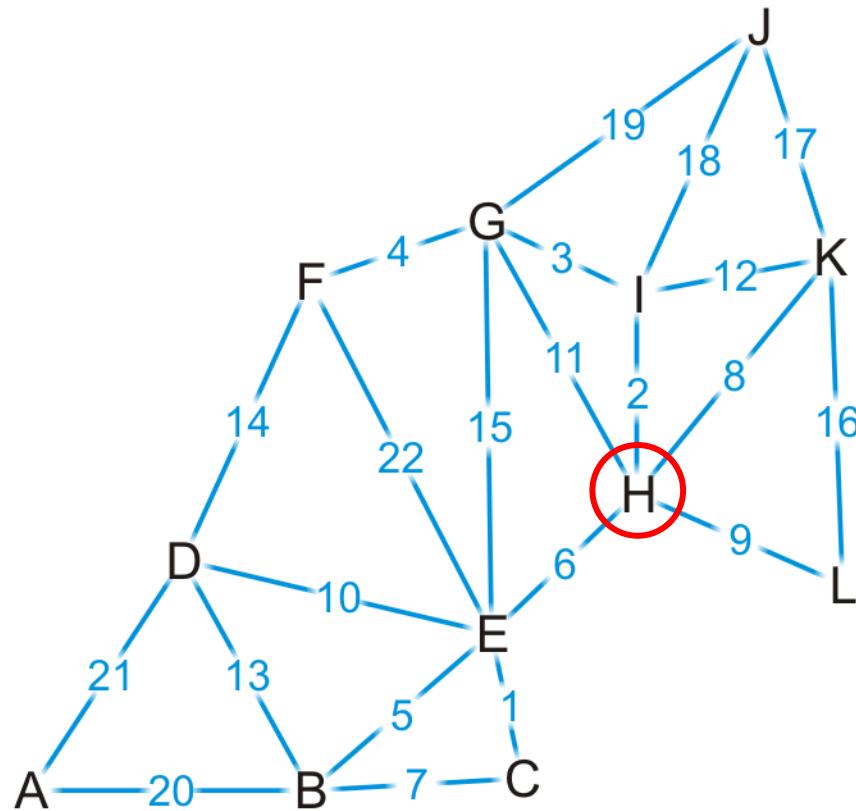


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	F	8	K
I	F	12	K
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

We visit vertex H: the shortest path is (K, H) of length 8

- Vertex H has four unvisited neighbors: E, G, I, L



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	T	8	K
I	F	12	K
J	F	17	K
K	T	0	\emptyset
L	F	16	K

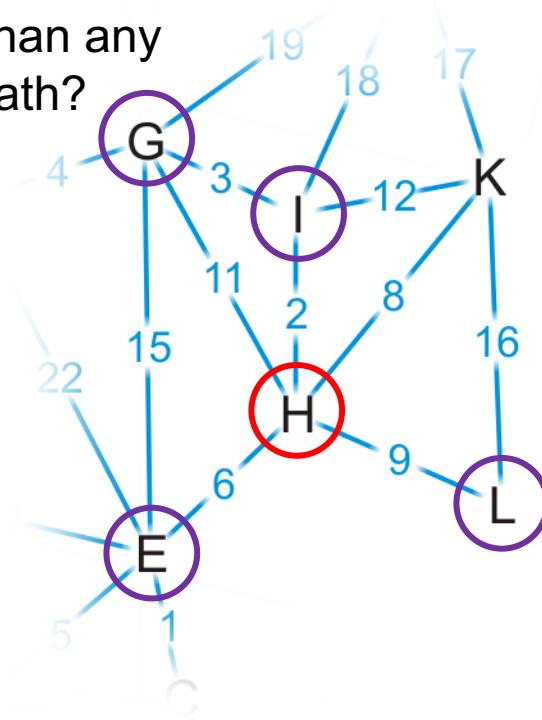
Example

Consider these paths:

(K, H, E) of length $8 + 6 = 14$

(K, H, I) of length $8 + 2 = 10$

- Which of these are shorter than any known path?



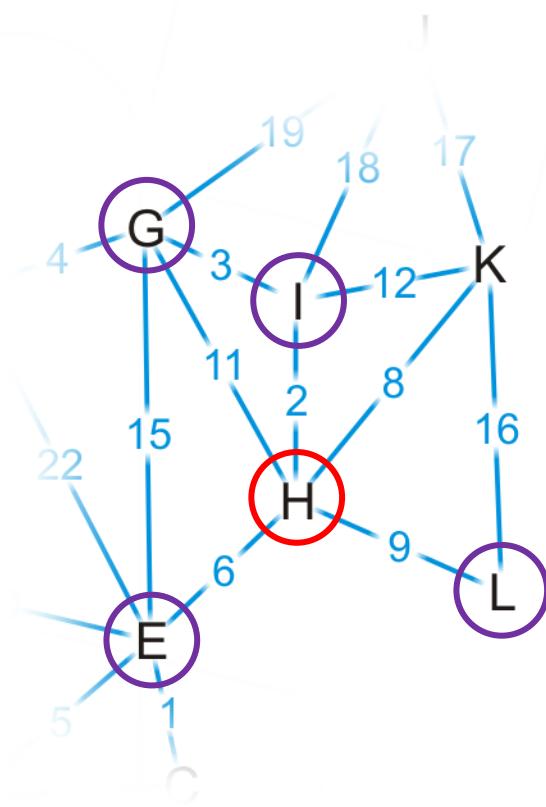
(K, H, G) of length $8 + 11 = 19$

(K, H, L) of length $8 + 9 = 17$

Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	∞	\emptyset
F	F	∞	\emptyset
G	F	∞	\emptyset
H	T	8	K
I	F	12	K
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

We already have a shorter path (K, L), but we update the other three

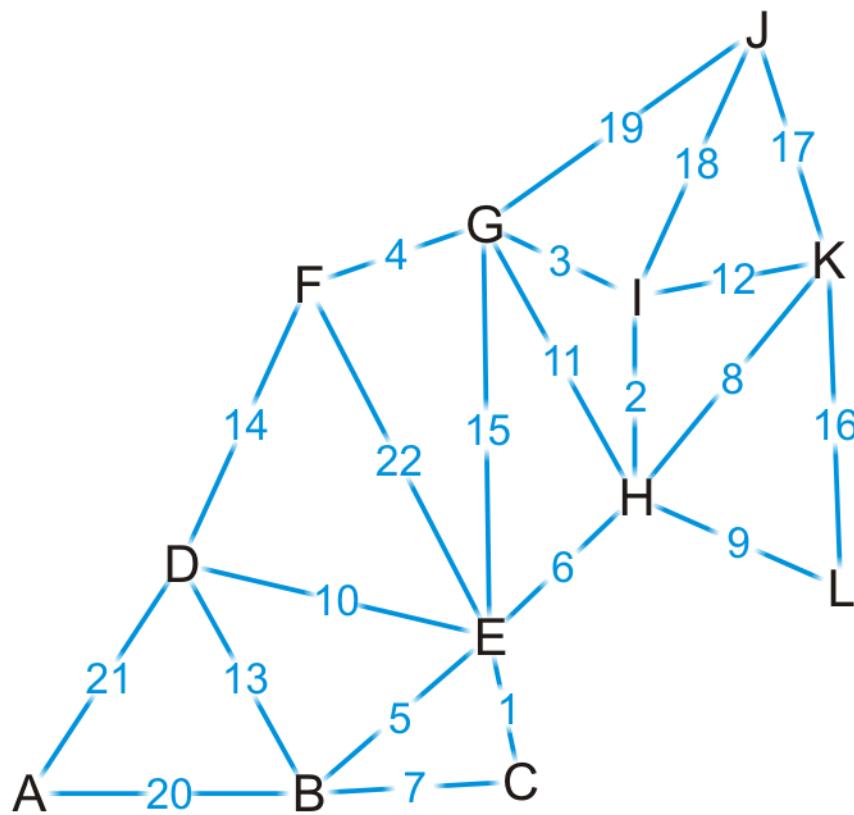


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	F	19	H
H	T	8	K
I	F	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

We are finished with vertex H

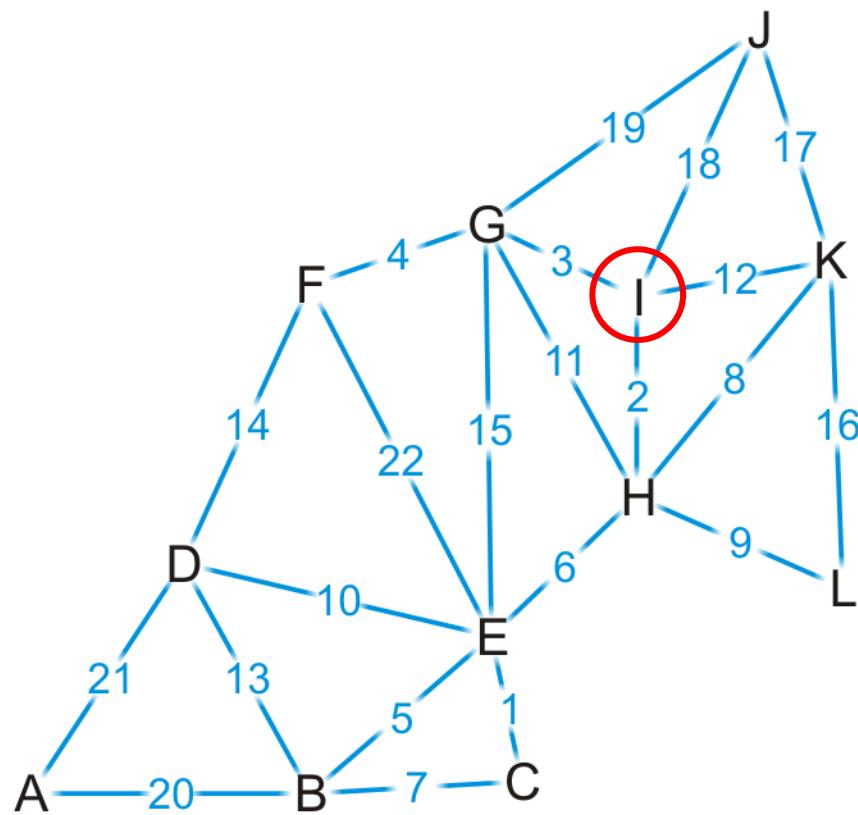
- Which vertex do we visit next?



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	F	19	H
H	T	8	K
I	F	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

The path (K, H, I) is the shortest path from K to I of length 10
– Vertex I has two unvisited neighbors: G and J



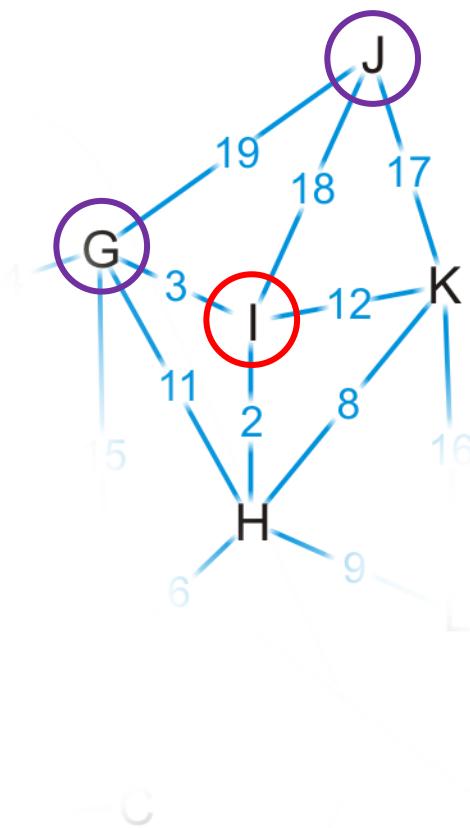
Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	F	19	H
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

Consider these paths:

(K, H, I, G) of length **10 + 3 = 13**

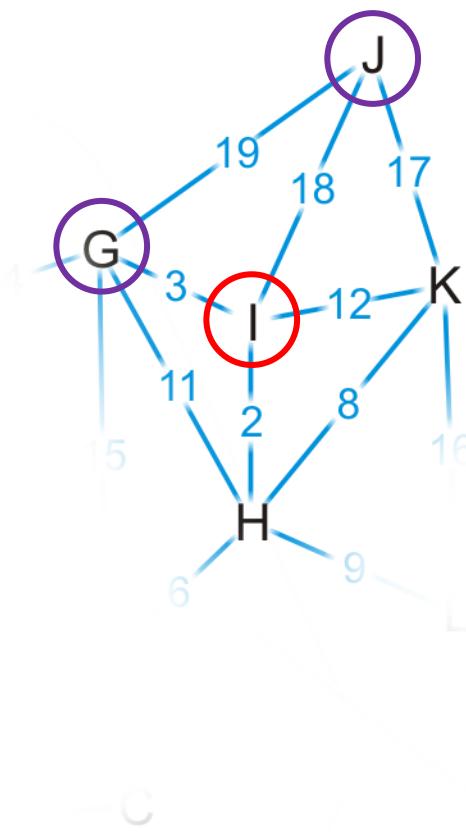
(K, H, I, J) of length **10 + 18 = 28**



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	F	19	H
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

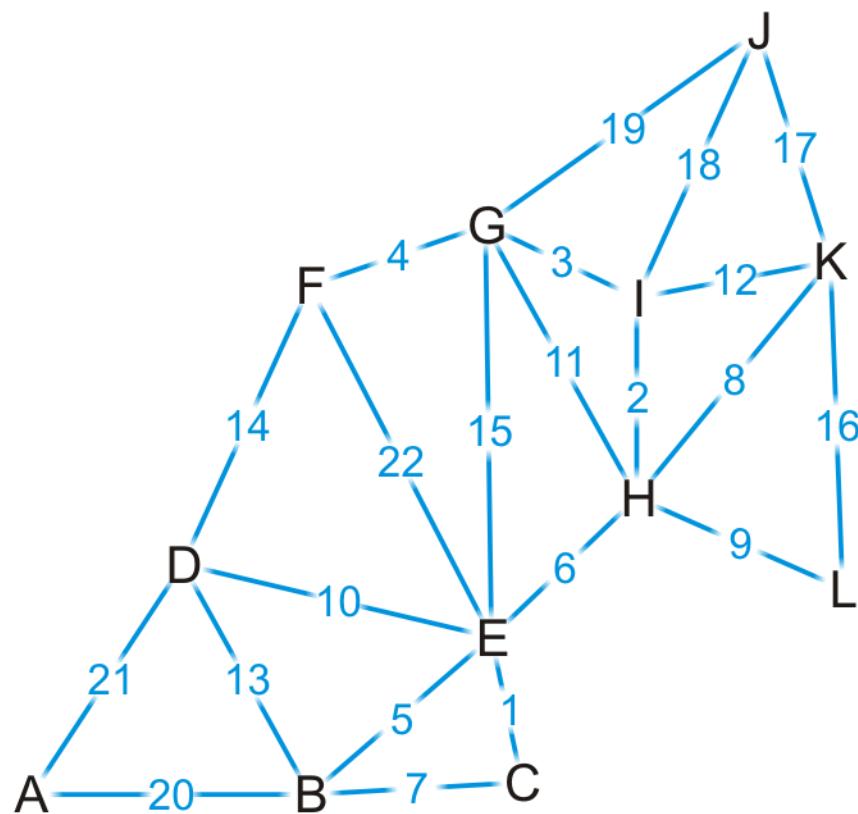
We have discovered a shorter path to vertex G, but (K, J) is still the shortest known path to vertex J



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	F	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

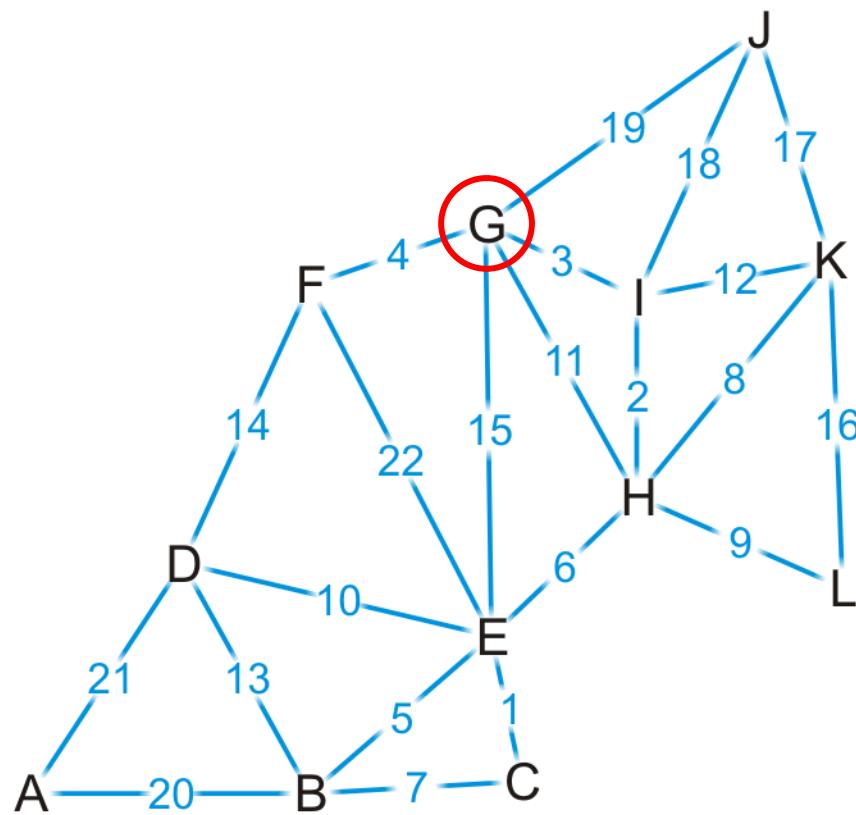
Which vertex can we visit next?



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	F	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

The path (K, H, I, G) is the shortest path from K to G of length 13
– Vertex G has three unvisited neighbors: E, F and J



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

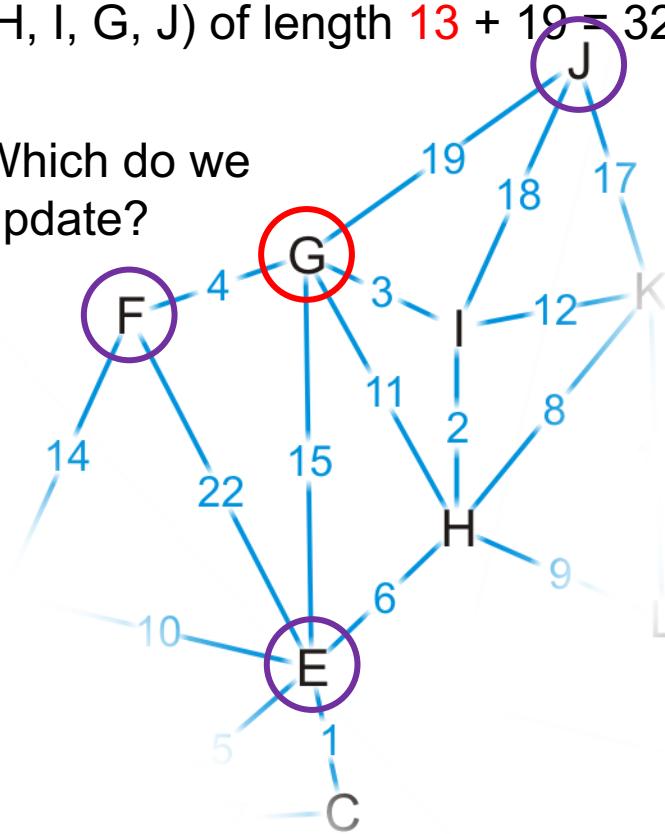
Example

Consider these paths:

(K, H, I, G, E) of length $13 + 15 = 28$

(K, H, I, G, J) of length $13 + 19 = 32$

- Which do we update?

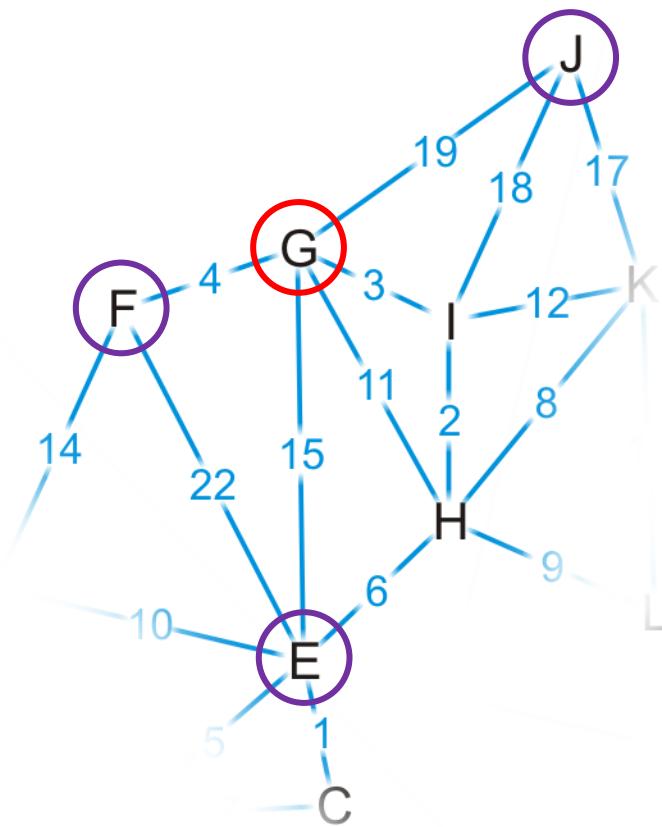


(K, H, I, G, F) of length $13 + 4 = 17$

Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	∞	\emptyset
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

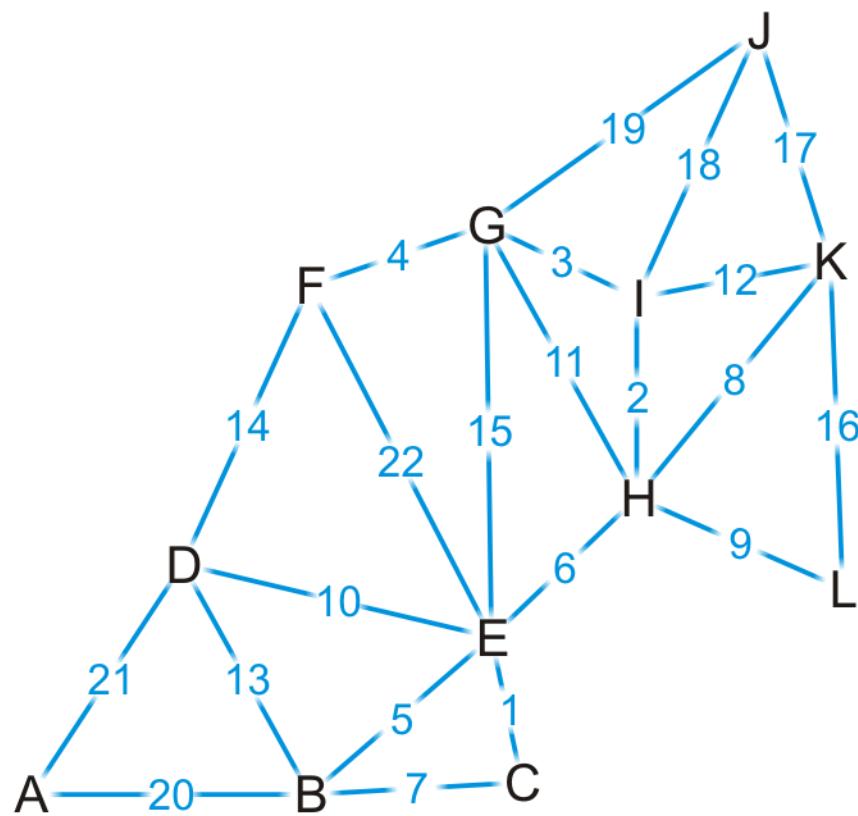
We have now found a path to vertex F



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

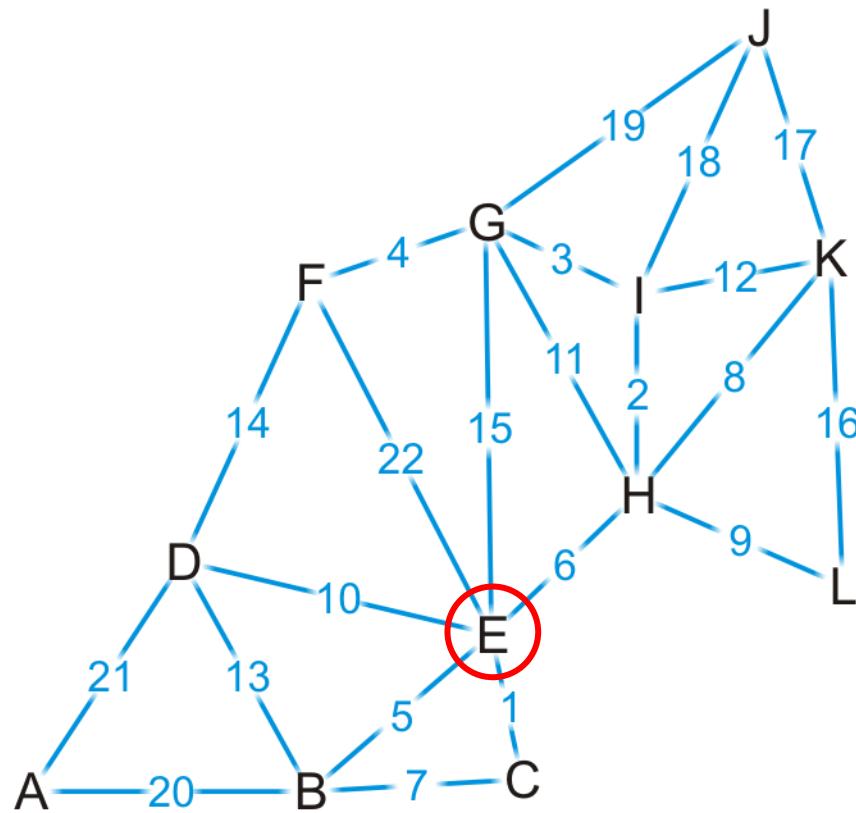
Where do we visit next?



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	F	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

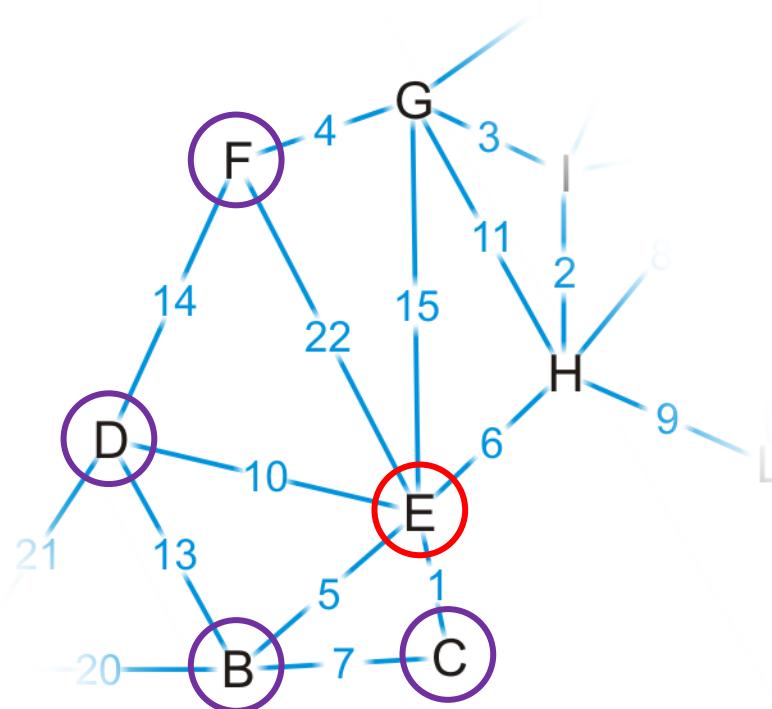
The path (K, H, E) is the shortest path from K to E of length 14
– Vertex G has four unvisited neighbors: B, C, D and F



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

The path (K, H, E) is the shortest path from K to E of length 14
– Vertex G has four unvisited neighbors: B, C, D and F



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

Consider these paths:

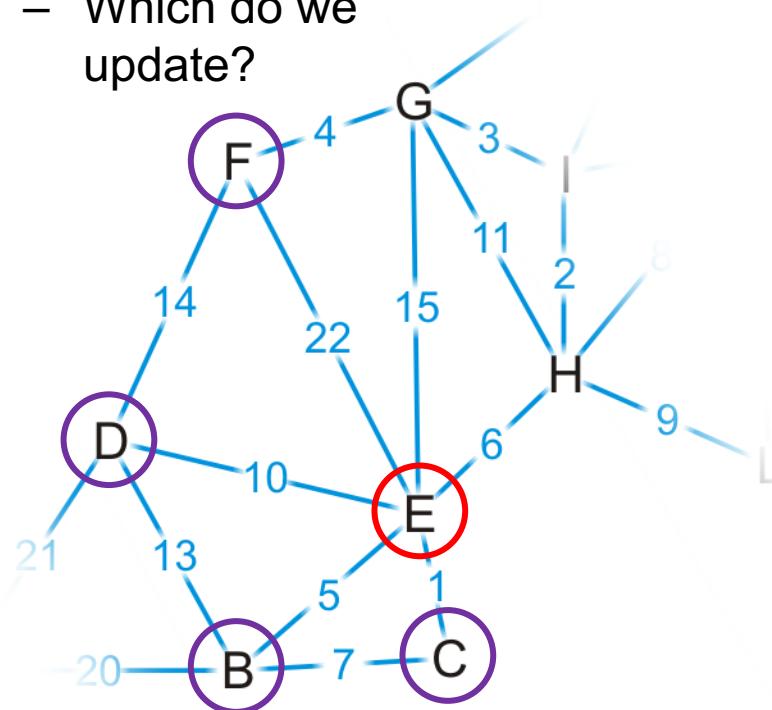
(K, H, E, B) of length **14** + 5 = 19

(K, H, E, D) of length **14** + 10 = 24

(K, H, E, C) of length **14** + 1 = 15

(K, H, E, F) of length **14** + 22 = 36

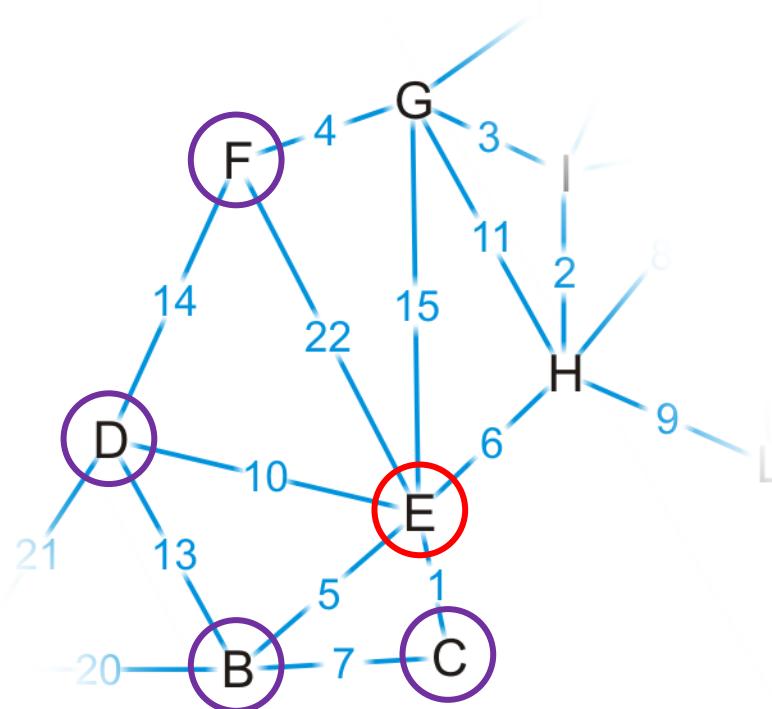
- Which do we update?



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	∞	\emptyset
C	F	∞	\emptyset
D	F	∞	\emptyset
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

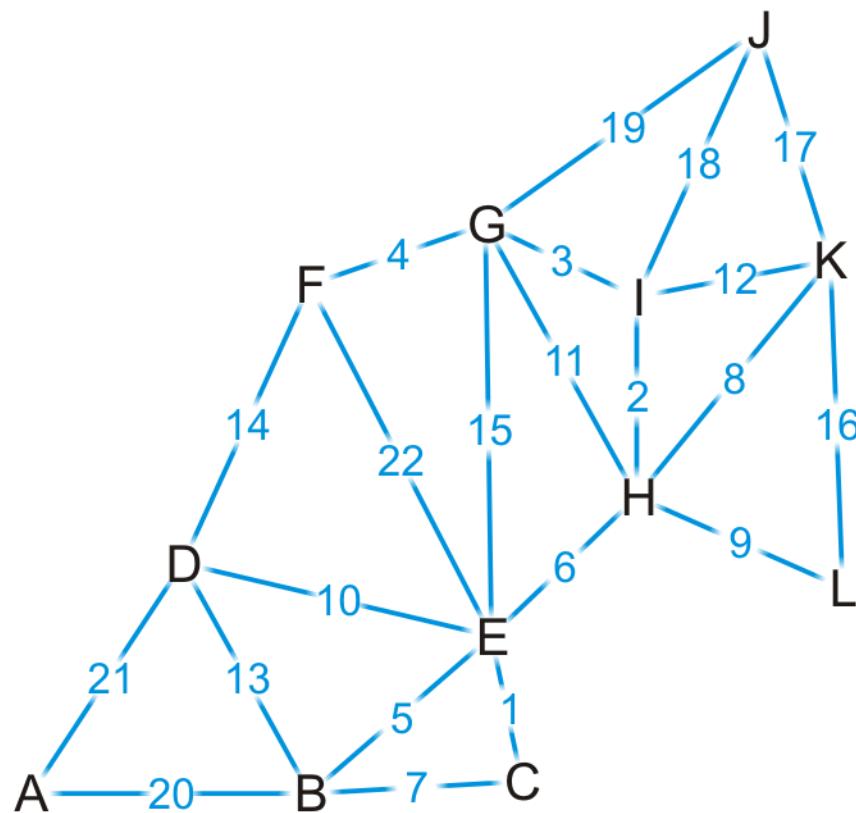
We've discovered paths to vertices B, C, D



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	F	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

Which vertex is next?

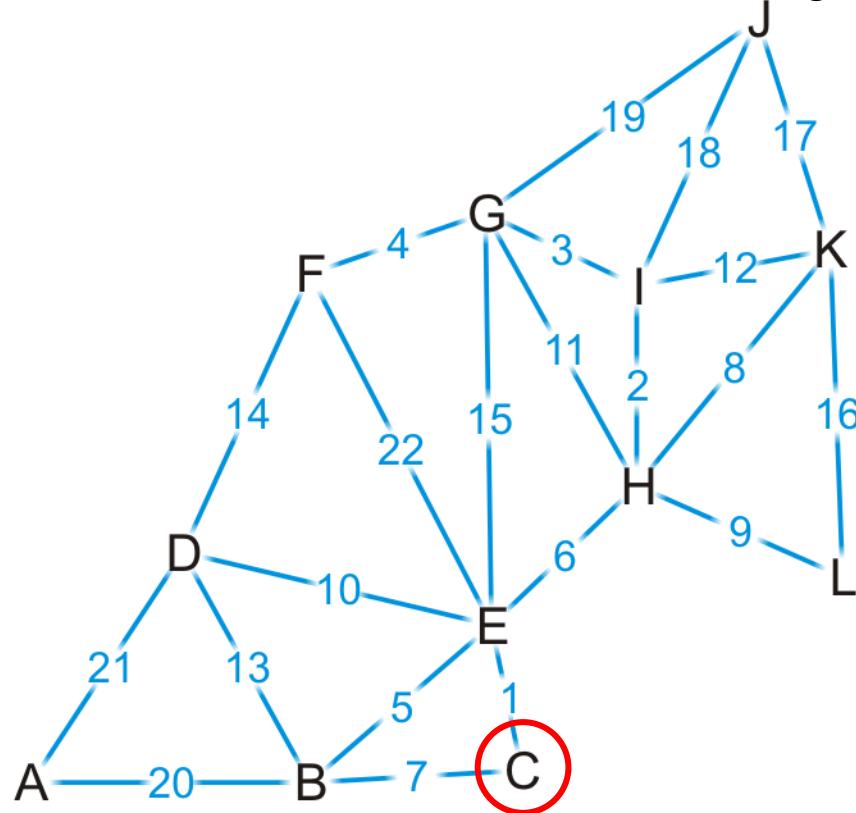


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	F	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

We've found that the path (K, H, E, C) of length 15 is the shortest path from K to C

- Vertex C has one unvisited neighbor, B

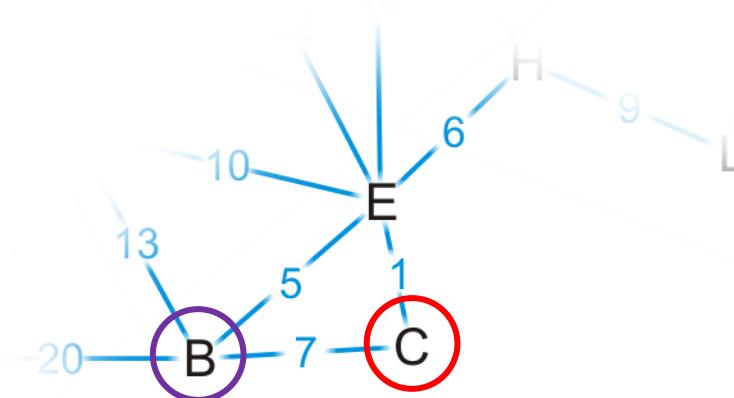


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

The path (K, H, E, C, B) is of length $15 + 7 = 22$

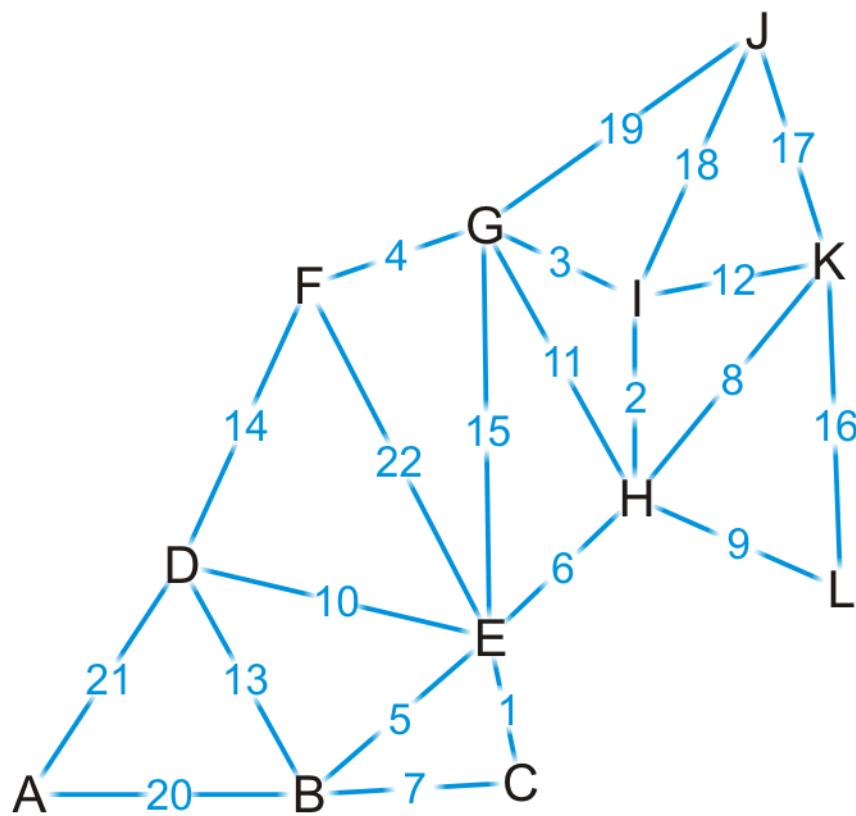
- We have already discovered a shorter path through vertex E



Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

Where to next?

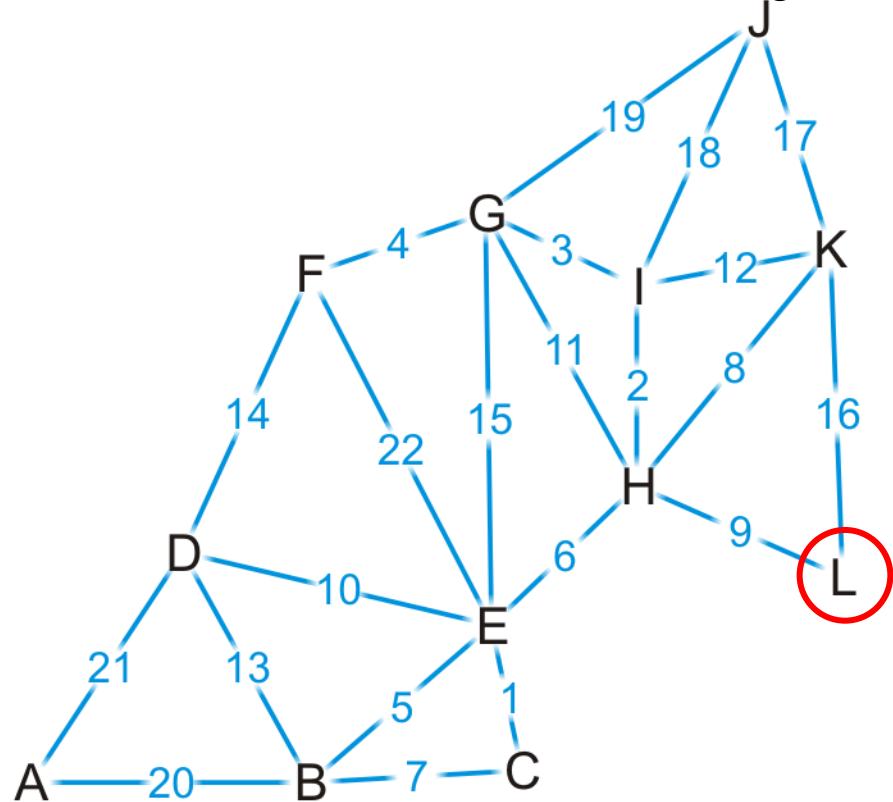


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	F	16	K

Example

We now know that (K, L) is the shortest path between these two points

- Vertex L has no unvisited neighbors

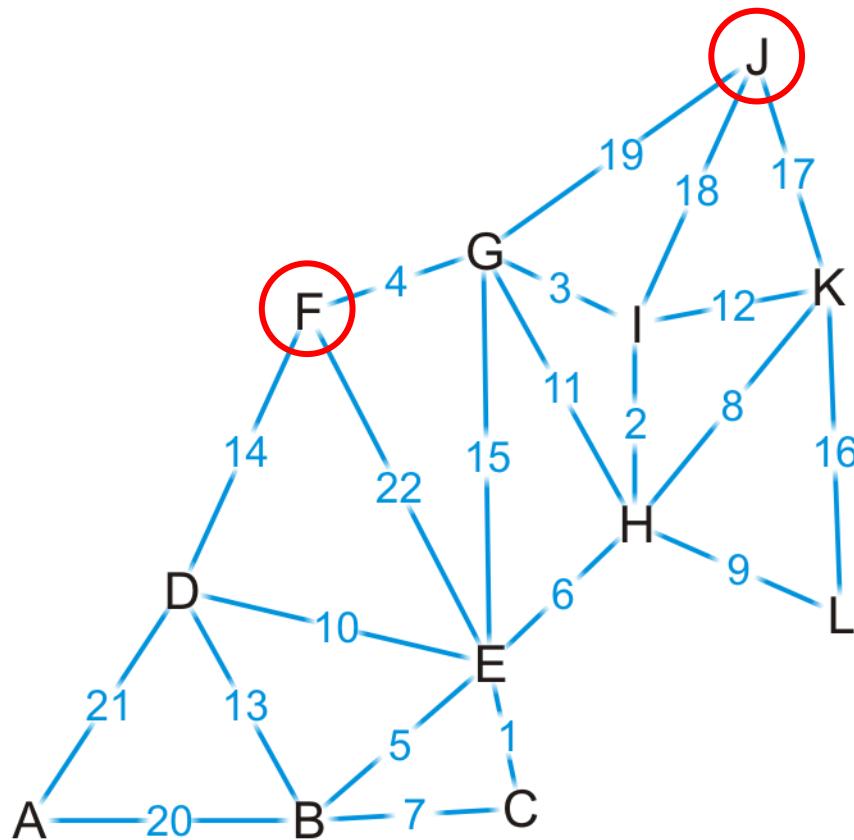


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	T	16	K

Example

Where to next?

- Does it matter if we visit vertex F first or vertex J first?

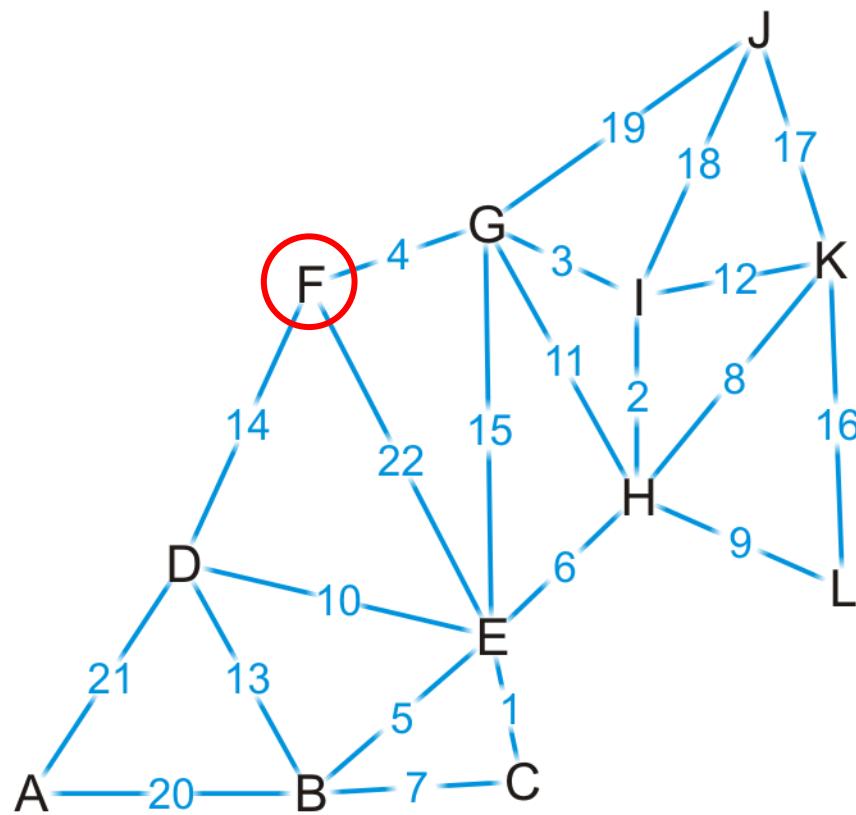


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	F	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	T	16	K

Example

Let's visit vertex F first

- It has one unvisited neighbor, vertex D

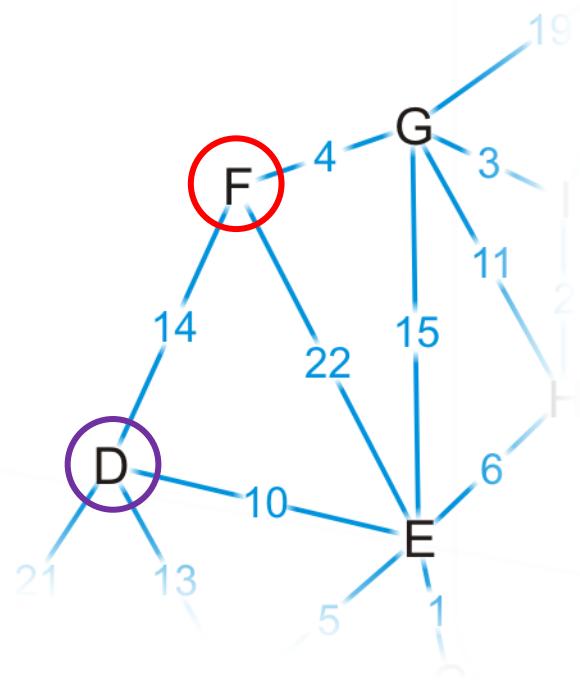


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	T	16	K

Example

The path (K, H, I, G, F, D) is of length $17 + 14 = 31$

- This is longer than the path we've already discovered

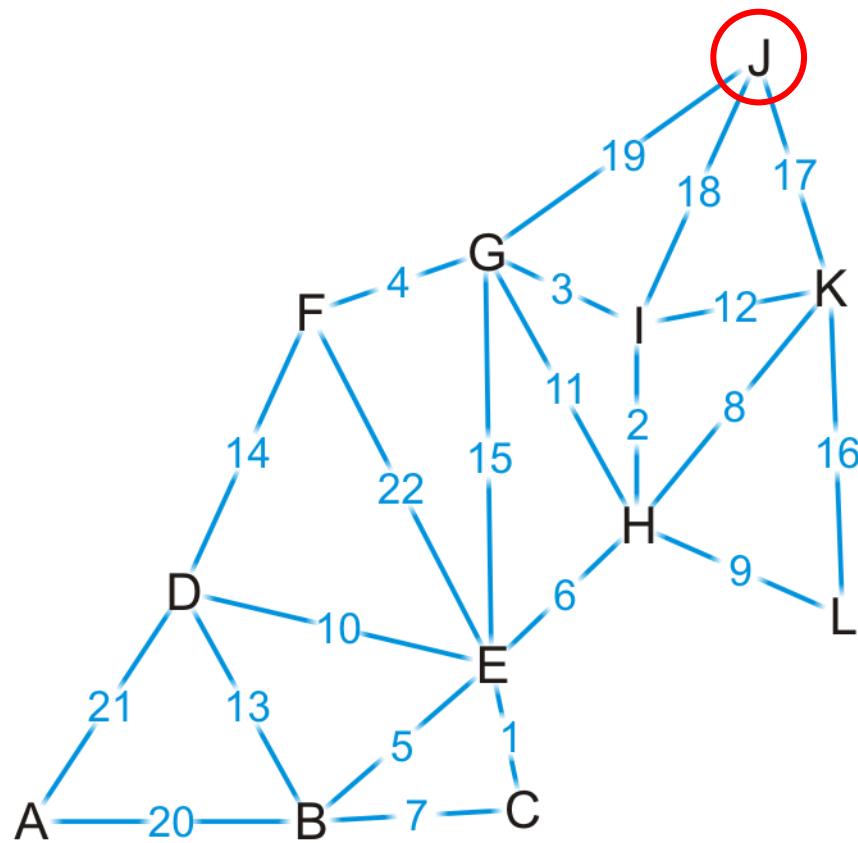


Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	F	17	K
K	T	0	\emptyset
L	T	16	K

Example

Now we visit vertex J

- It has no unvisited neighbors



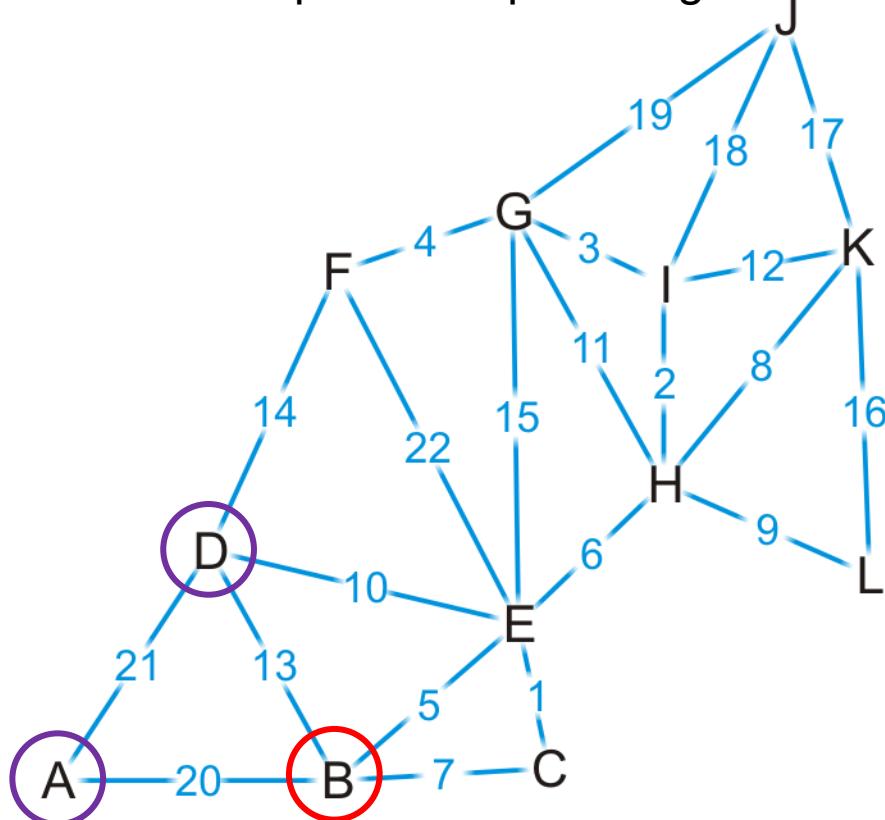
Vertex	Visited	Distance	Previous
A	F	∞	\emptyset
B	F	19	E
C	T	15	E
D	F	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	T	17	K
K	T	0	\emptyset
L	T	16	K

Example

Next we visit vertex B, which has two unvisited neighbors:

(K, H, E, B, A) of length **19 + 20 = 39** (K, H, E, B, D) of length **19 + 13 = 32**

- We update the path length to A

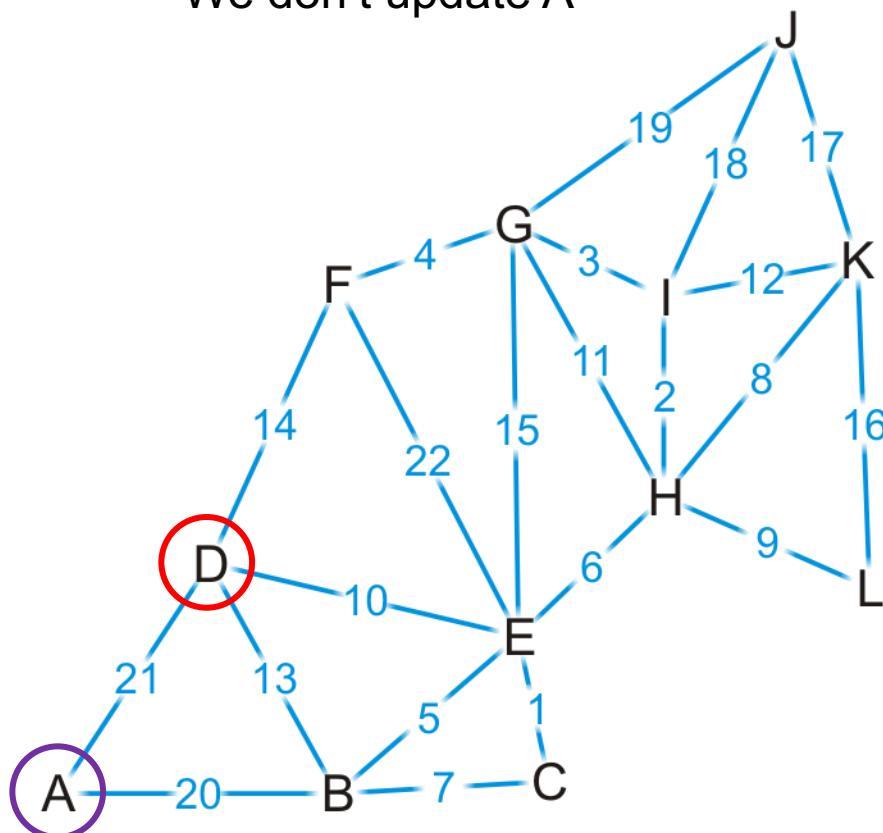


Vertex	Visited	Distance	Previous
A	F	39	B
B	T	19	E
C	T	15	
D	F	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	T	17	K
K	T	0	Ø
L	T	16	K

Example

Next we visit vertex D

- The path (K, H, E, D, A) is of length $24 + 21 = 45$
- We don't update A

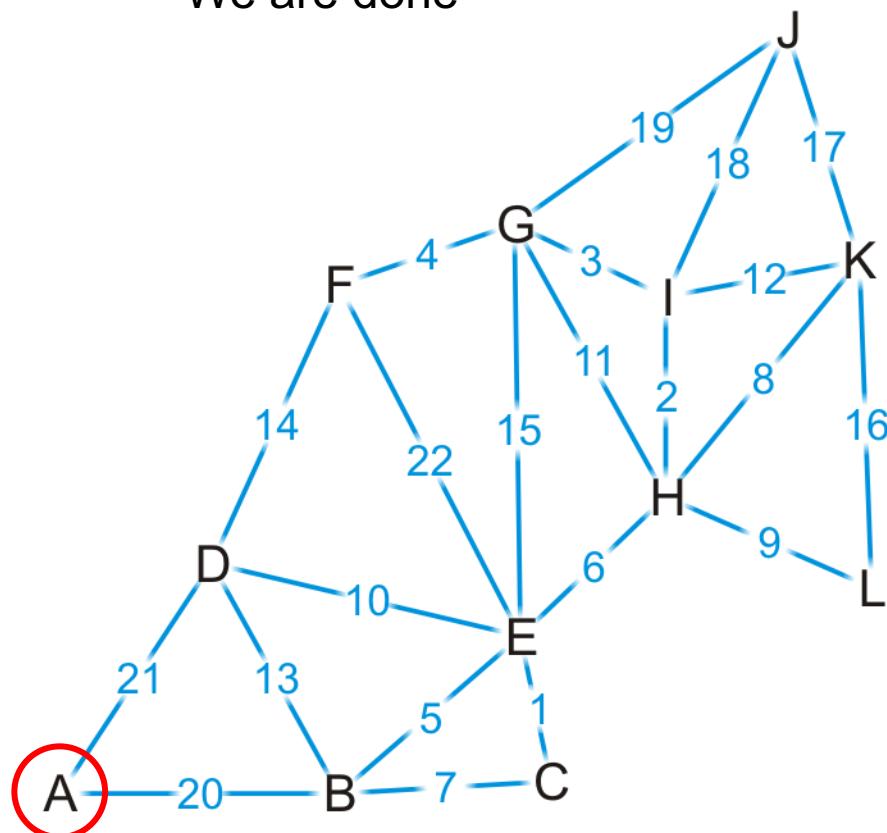


Vertex	Visited	Distance	Previous
A	F	39	B
B	T	19	
C	T	15	E
D	T	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	T	17	K
K	T	0	Ø
L	T	16	K

Example

Finally, we visit vertex A

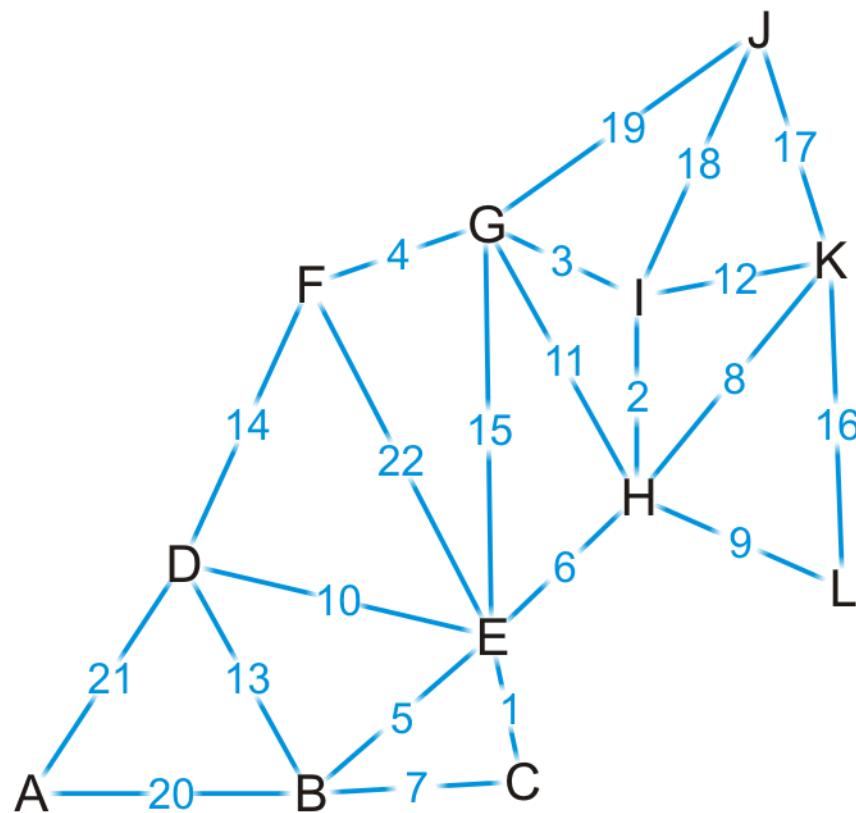
- It has no unvisited neighbors and there are no unvisited vertices left
- We are done



Vertex	Visited	Distance	Previous
A	T	39	B
B	T	19	E
C	T	15	E
D	T	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	T	17	K
K	T	0	Ø
L	T	16	K

Example

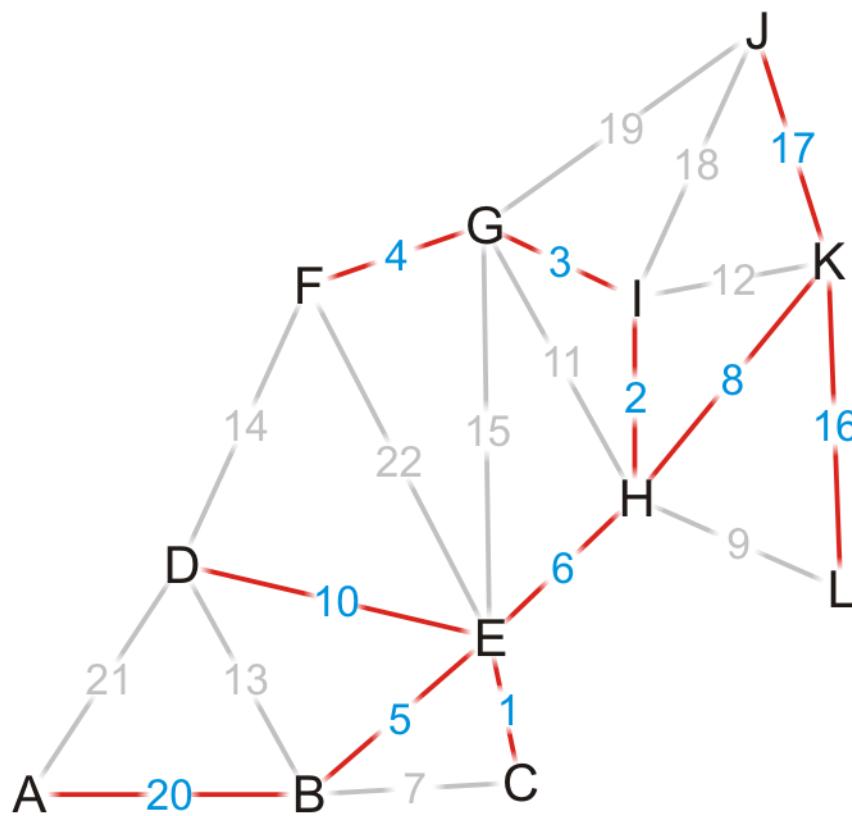
Thus, we have found the shortest path from vertex K to each of the other vertices



Vertex	Visited	Distance	Previous
A	T	39	B
B	T	19	E
C	T	15	E
D	T	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	T	17	K
K	T	0	Ø
L	T	16	K

Example

Using the *previous* pointers, we can reconstruct the paths

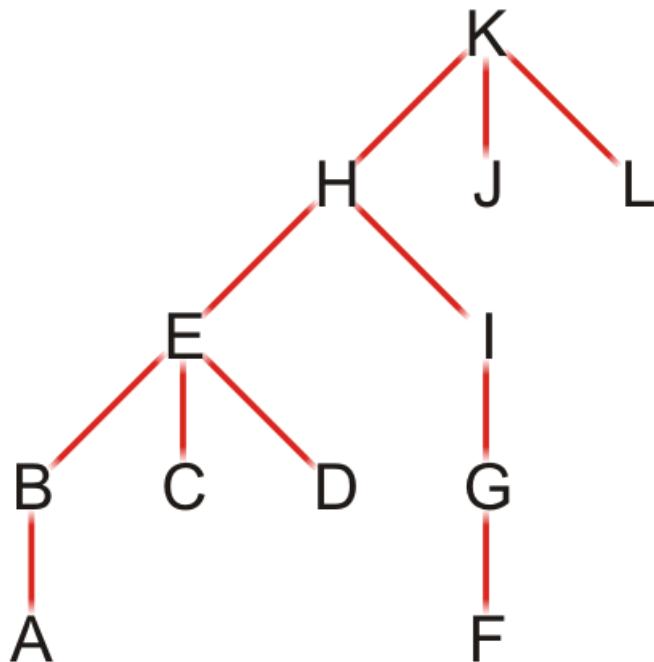


Vertex	Visited	Distance	Previous
A	T	39	B
B	T	19	E
C	T	15	E
D	T	24	E
E	T	14	H
F	T	17	G
G	T	13	I
H	T	8	K
I	T	10	H
J	T	17	K
K	T	0	Ø
L	T	16	K

Example

Note that this table defines a rooted parental tree

- The source vertex K is at the root
- The previous pointer is the *parent* of the vertex in the tree



Vertex	Previous
A	B
B	E
C	E
D	E
E	H
F	G
G	I
H	K
I	H
J	K
K	∅
L	K

References

Cormen, Leiserson, Rivest and Stein,
Introduction to Algorithms, The MIT Press, 2001, §25.2, pp.629-35.

Mark A. Weiss,
Data Structures and Algorithm Analysis in C++, 3rd Ed., Addison Wesley, 2006, Ch.?, p.?.

Joh Kleinberg and Eva Tardos,
Algorithm Design, Pearson, 2006.

Elliot B. Koffman and Paul A.T. Wolfgang,
Objects, Abstractions, Data Structures and Design using C++, Wiley, 2006.

These slides are provided for the ECE 250 *Algorithms and Data Structures* course. The material in it reflects Douglas W. Harder's best judgment in light of the information available to him at the time of preparation. Any reliance on these course slides by any party for any other purpose are the responsibility of such parties. Douglas W. Harder accepts no responsibility for damages, if any, suffered by any party as a result of decisions made or actions based on these course slides for any other purpose than that for which it was intended.