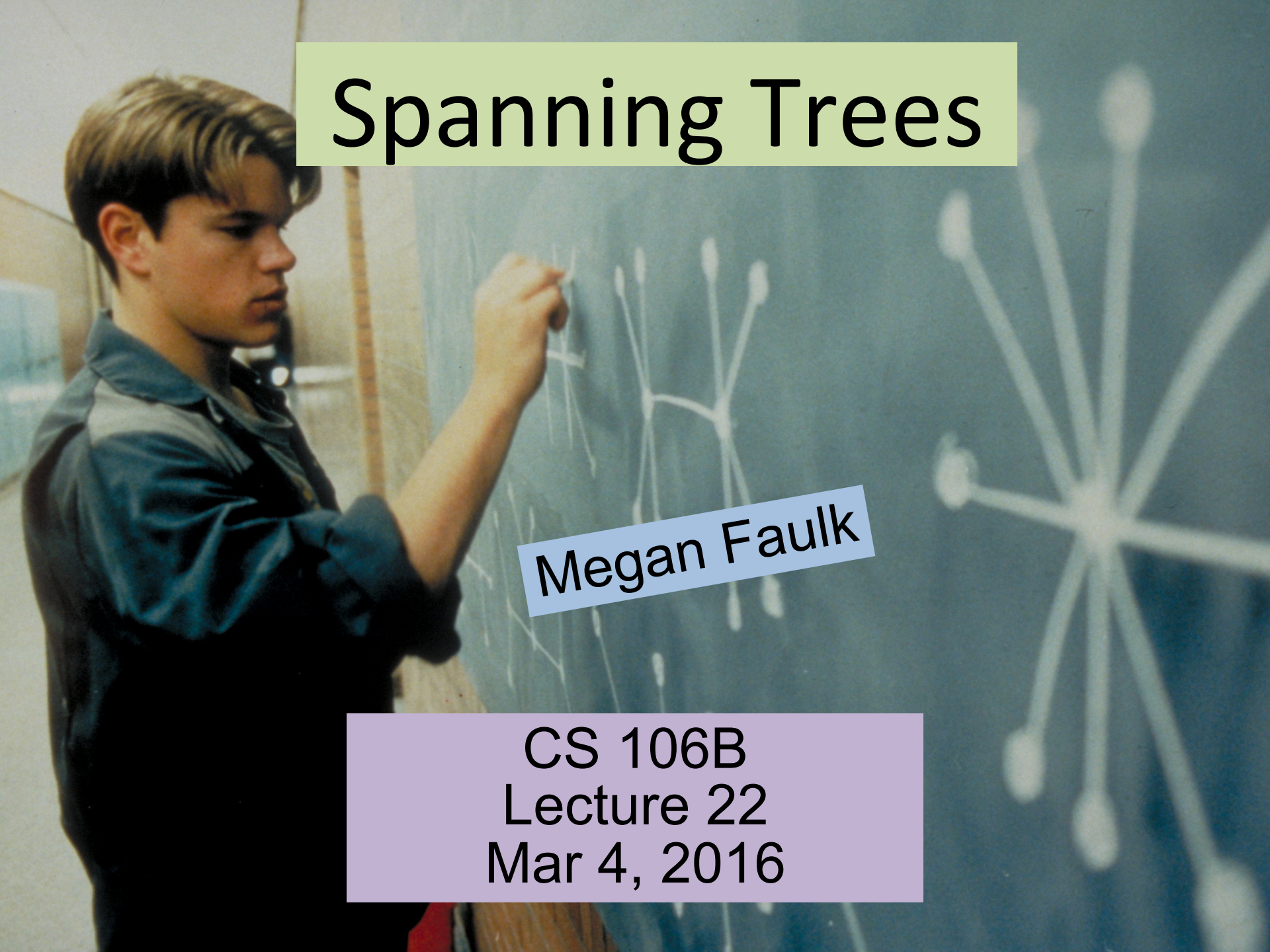


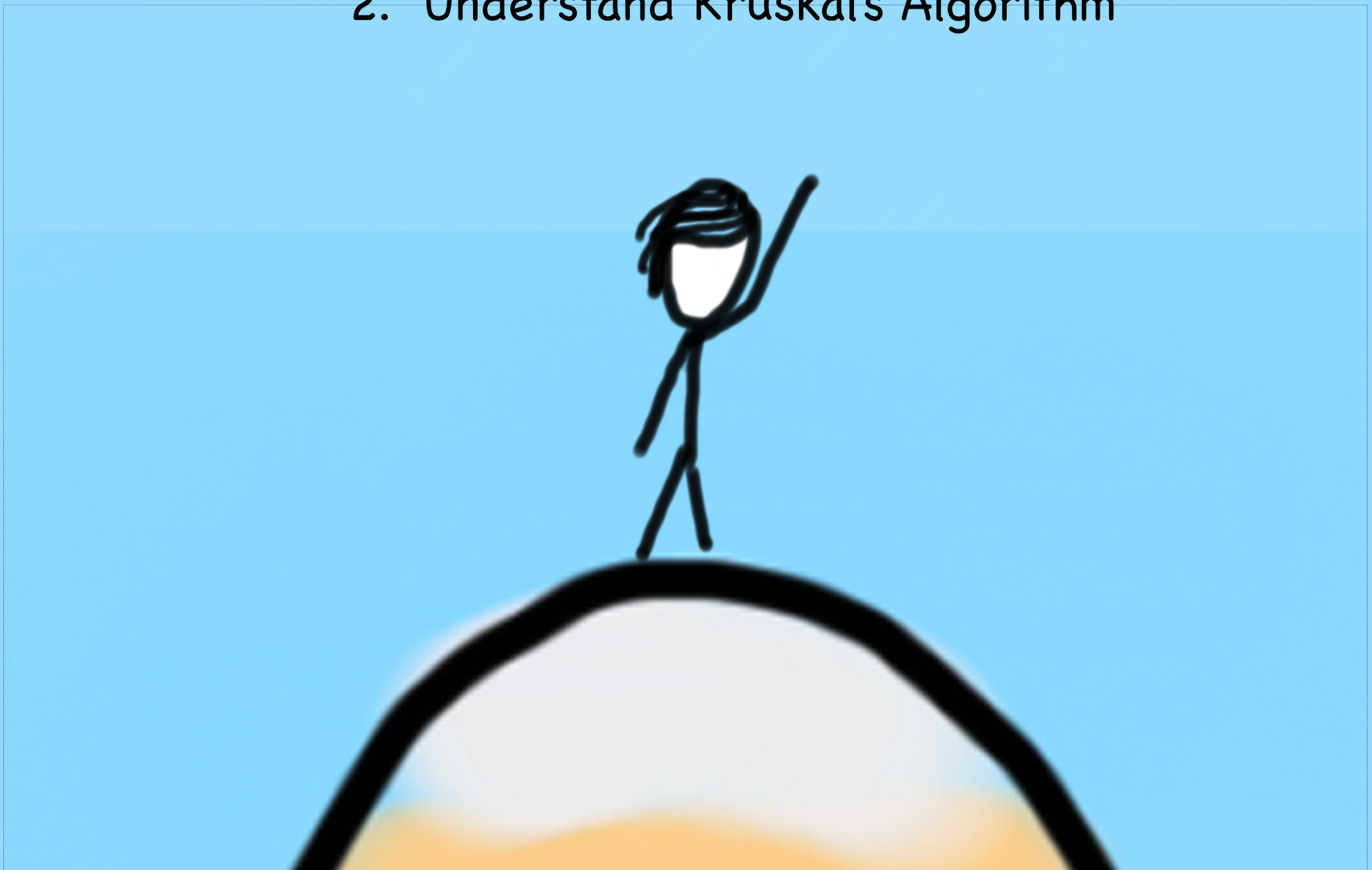
Spanning Trees

Megan Faulk

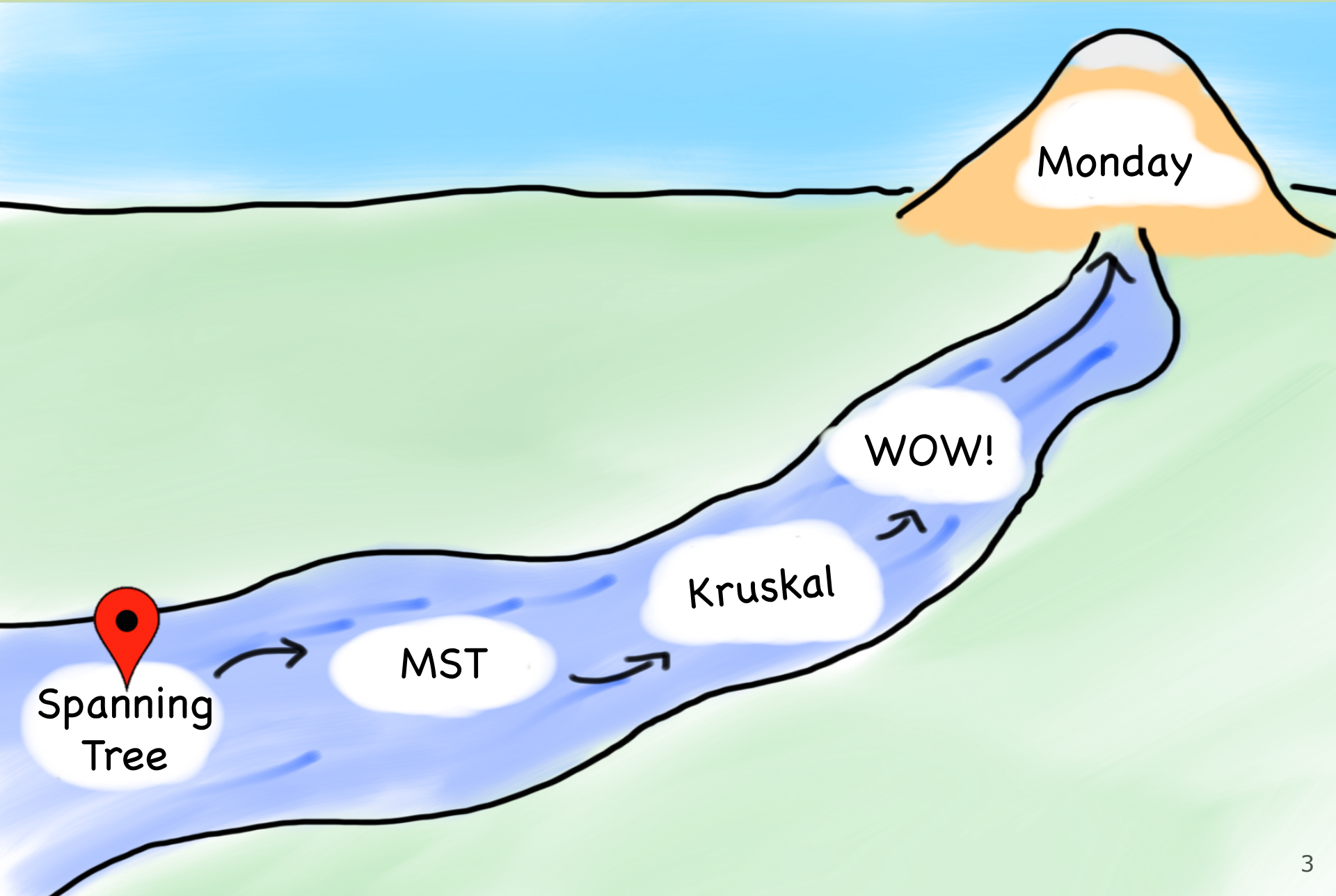
CS 106B
Lecture 22
Mar 4, 2016

Today's Goal

1. Learn what a Spanning Tree is
2. Understand Kruskal's Algorithm

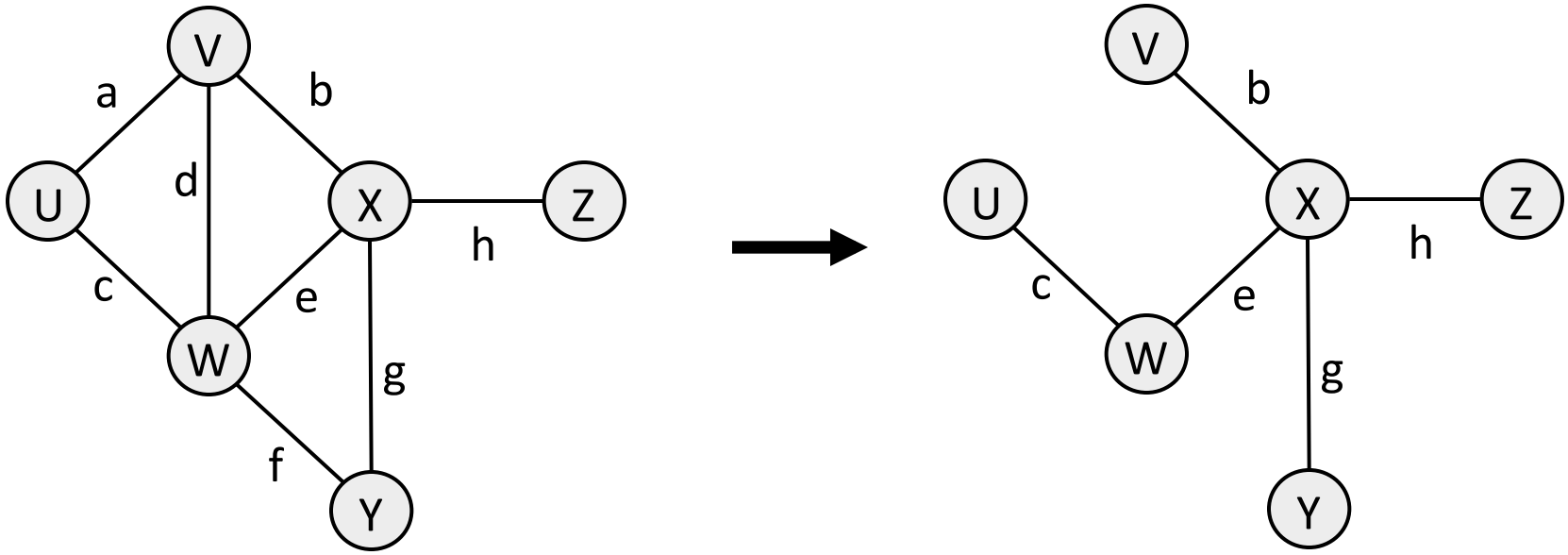


Today's Route



Spanning trees

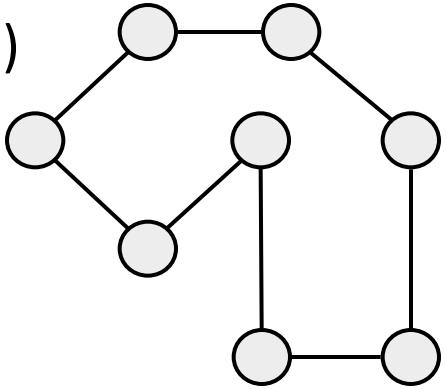
- A **spanning tree** of a graph is a set of edges that connects all vertices in the graph with no cycles.
  - What is a spanning tree for the graph below?



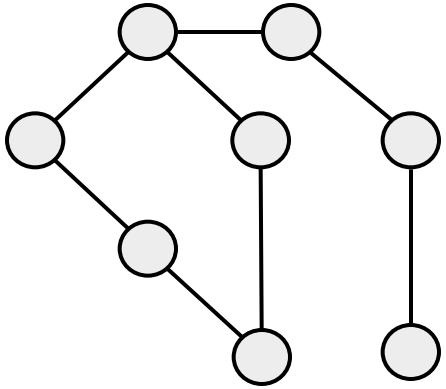
Span tree examples

- **Q:** How many of the graphs shown are legal spanning trees?

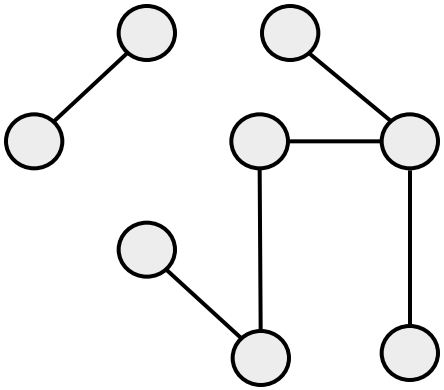
(1)



(2)



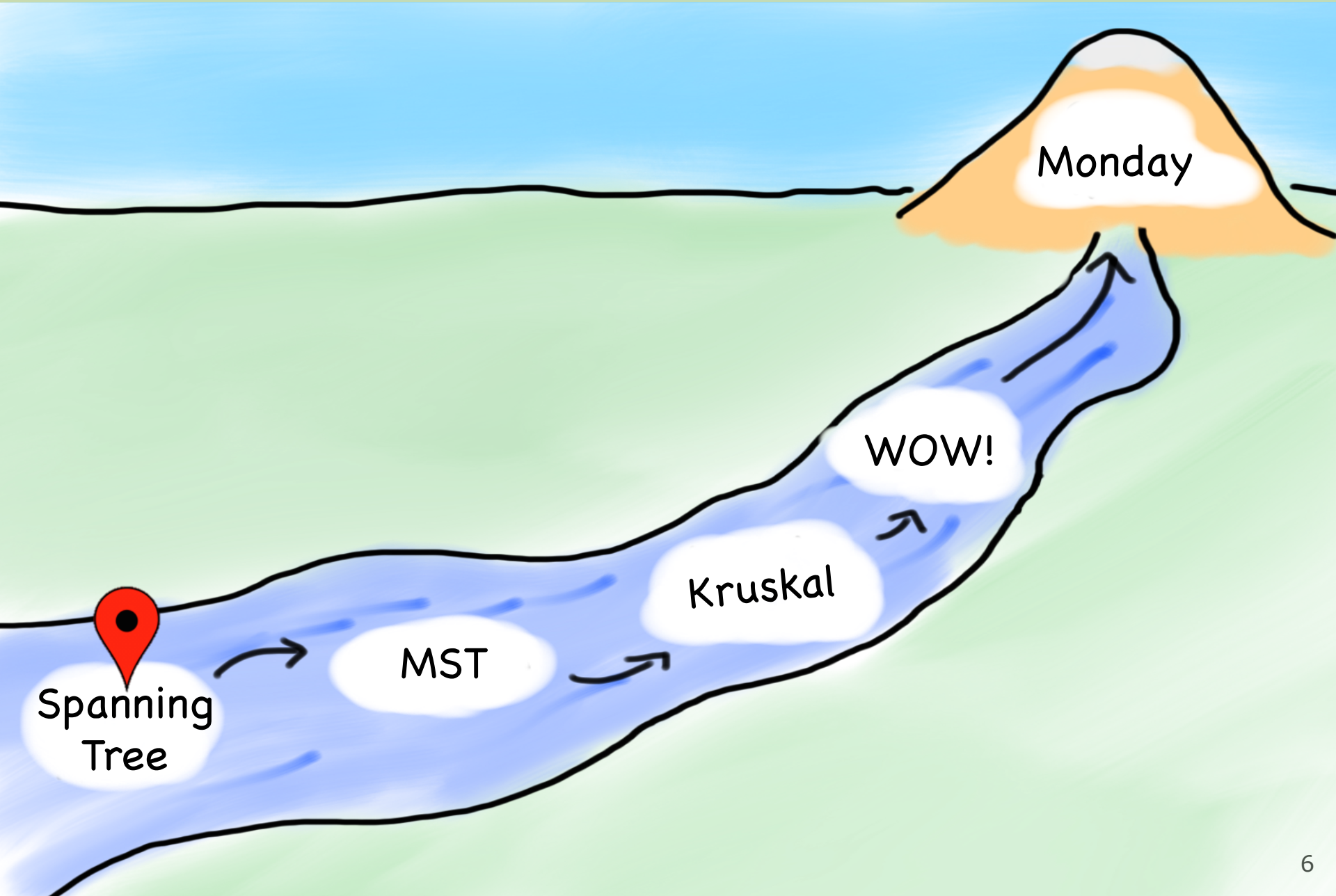
(3)



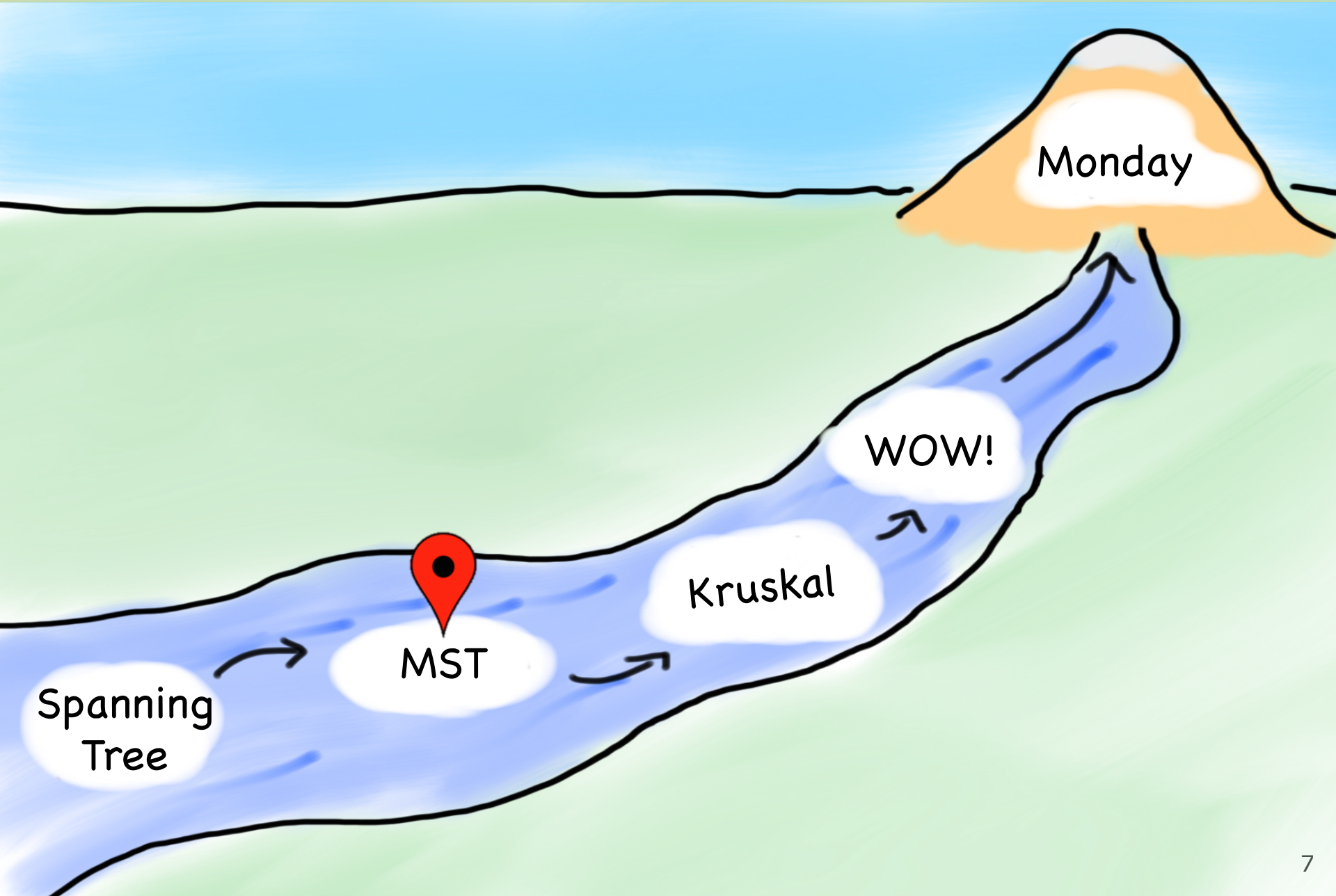
A. none
B. one
C. two
D. all three

STUDENT

Today's Route

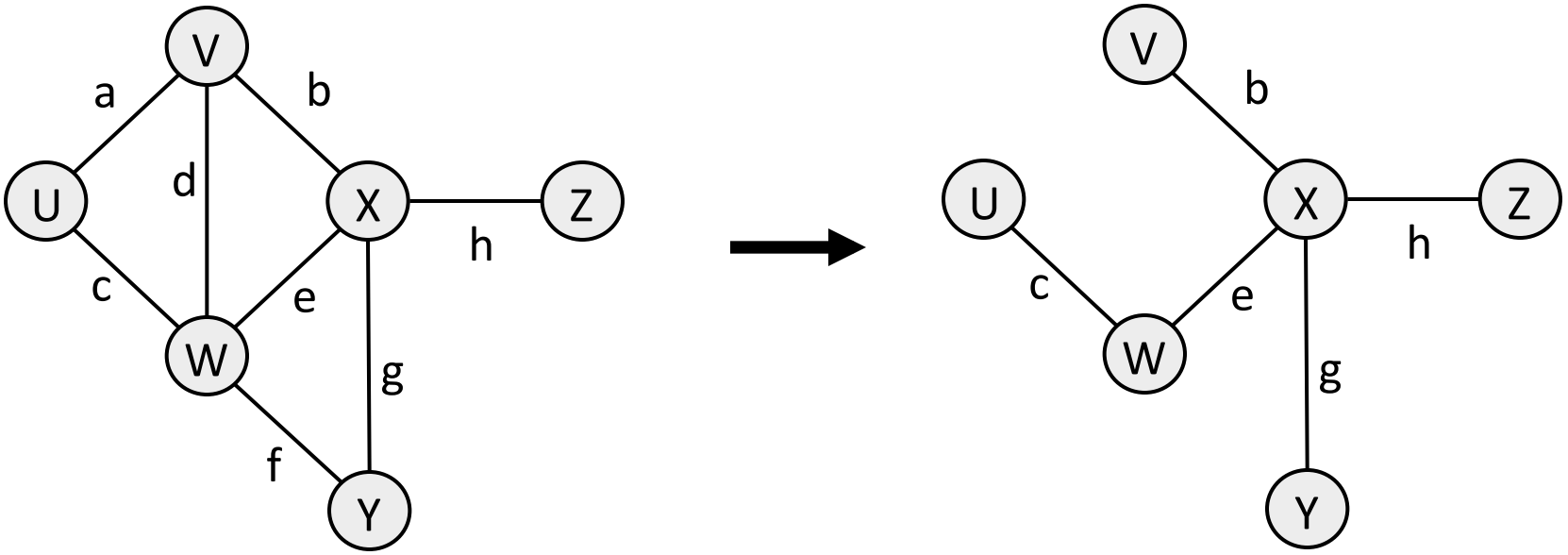


Today's Route



Minimum spanning tree

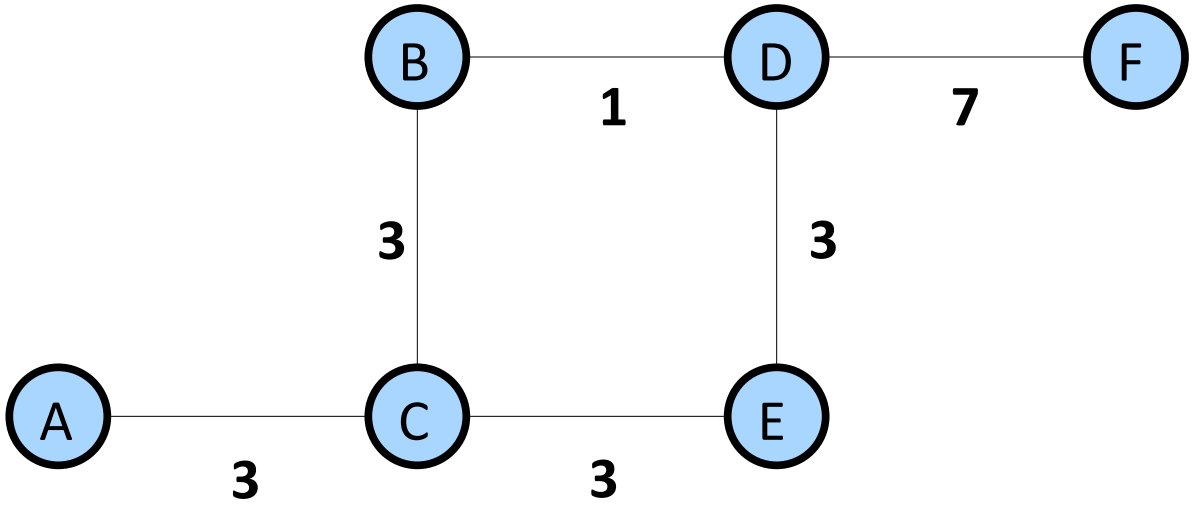
- **minimum spanning tree** (MST): A spanning tree that has the lowest combined edge weight (cost), or minimum # edges if unweighted.



MST examples

- **Q:** How many minimum spanning trees does this graph have?

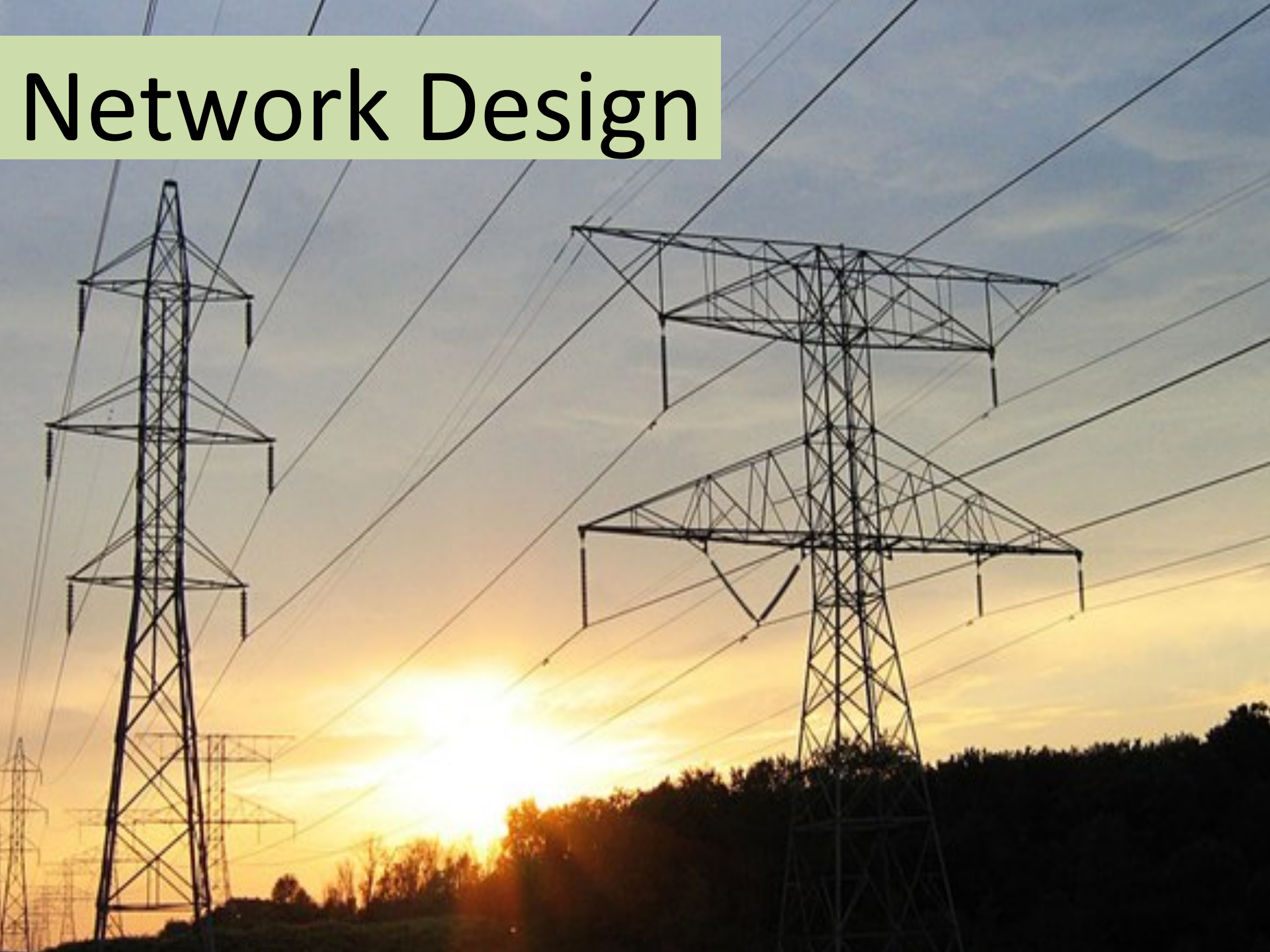
A. 0-1
B. 2-3
C. 4-5
D. 6-7
E. > 7



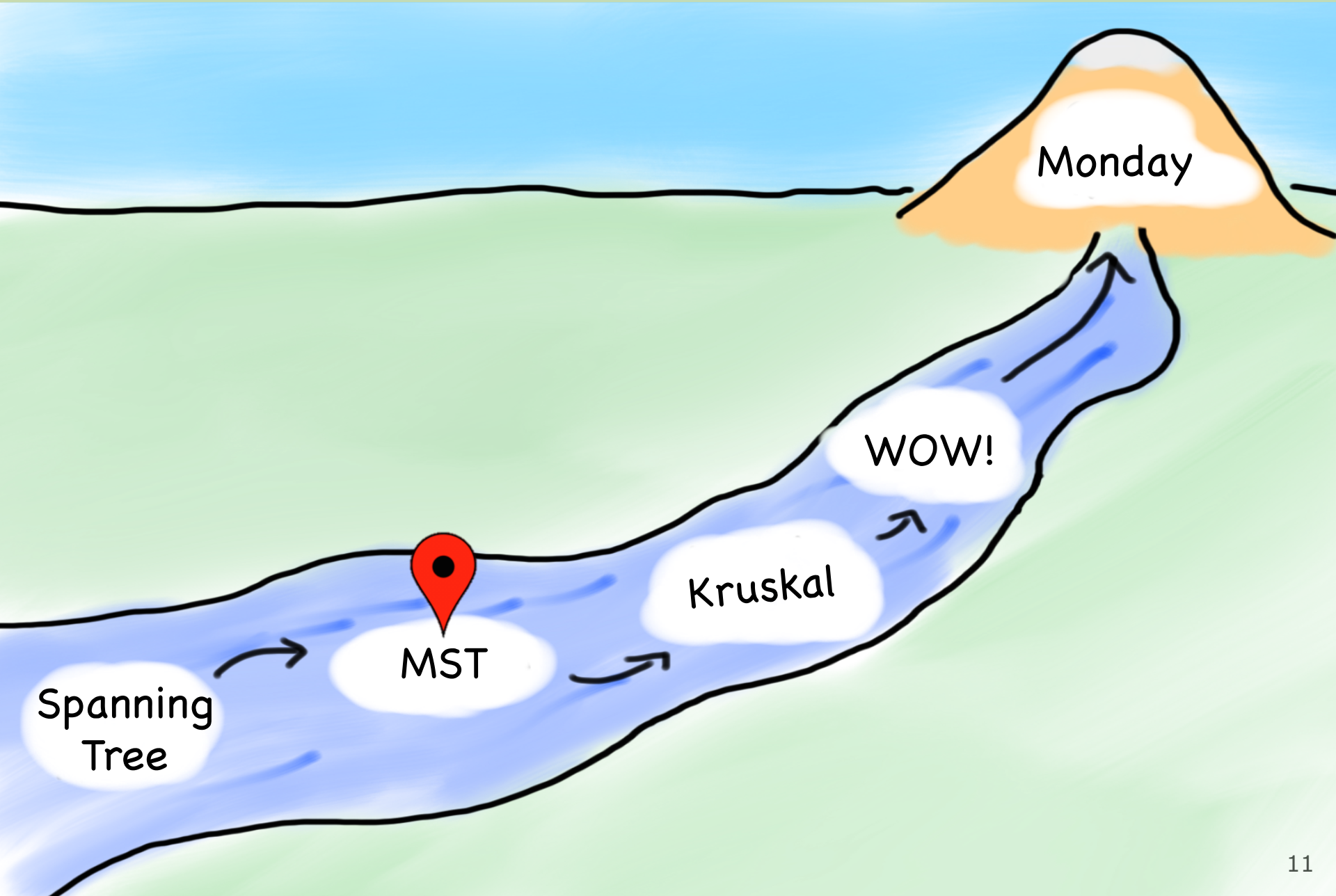
(question courtesy Cynthia Lee)

STUDENT

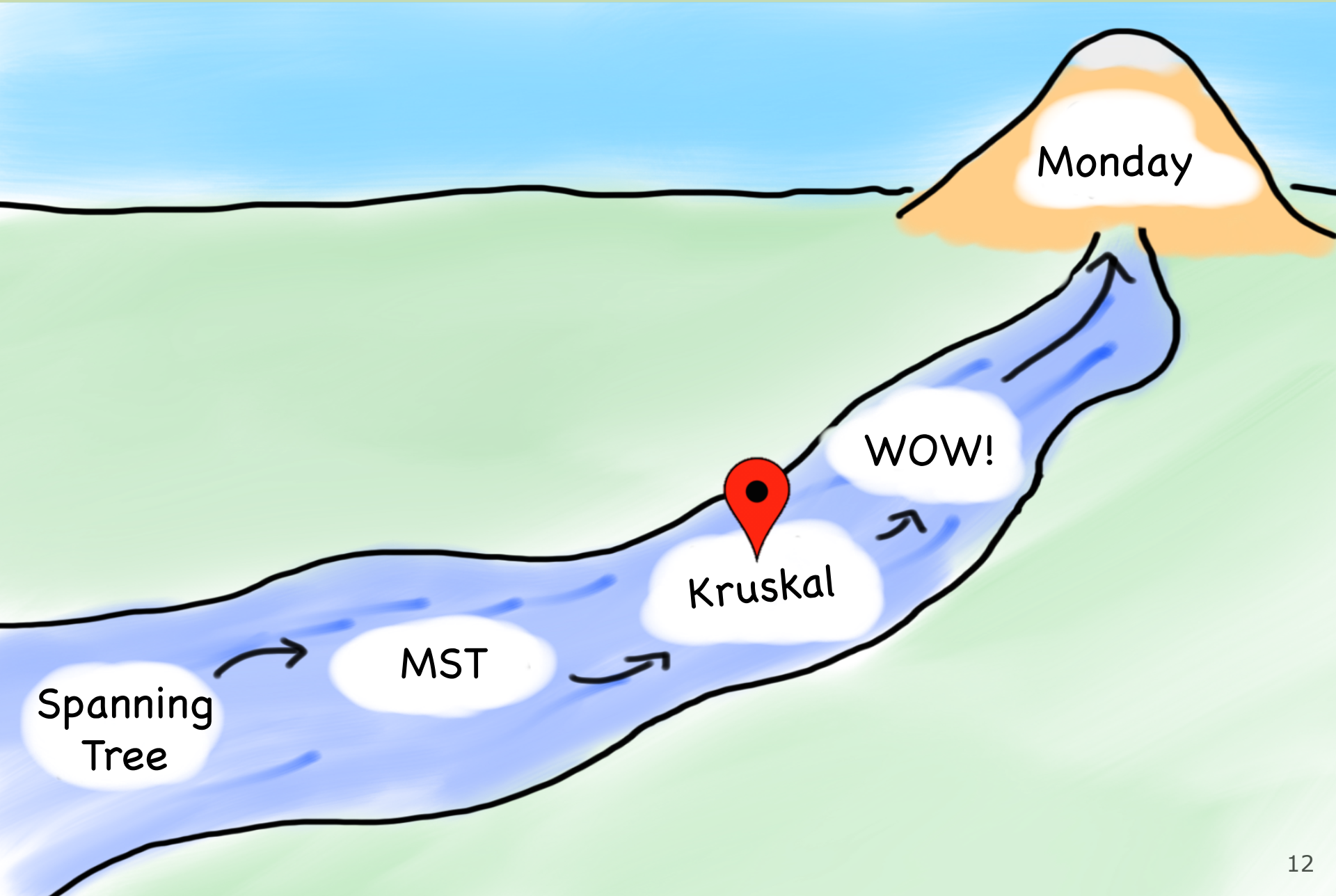
Network Design



Today's Route



Today's Route



Kruskal's algorithm

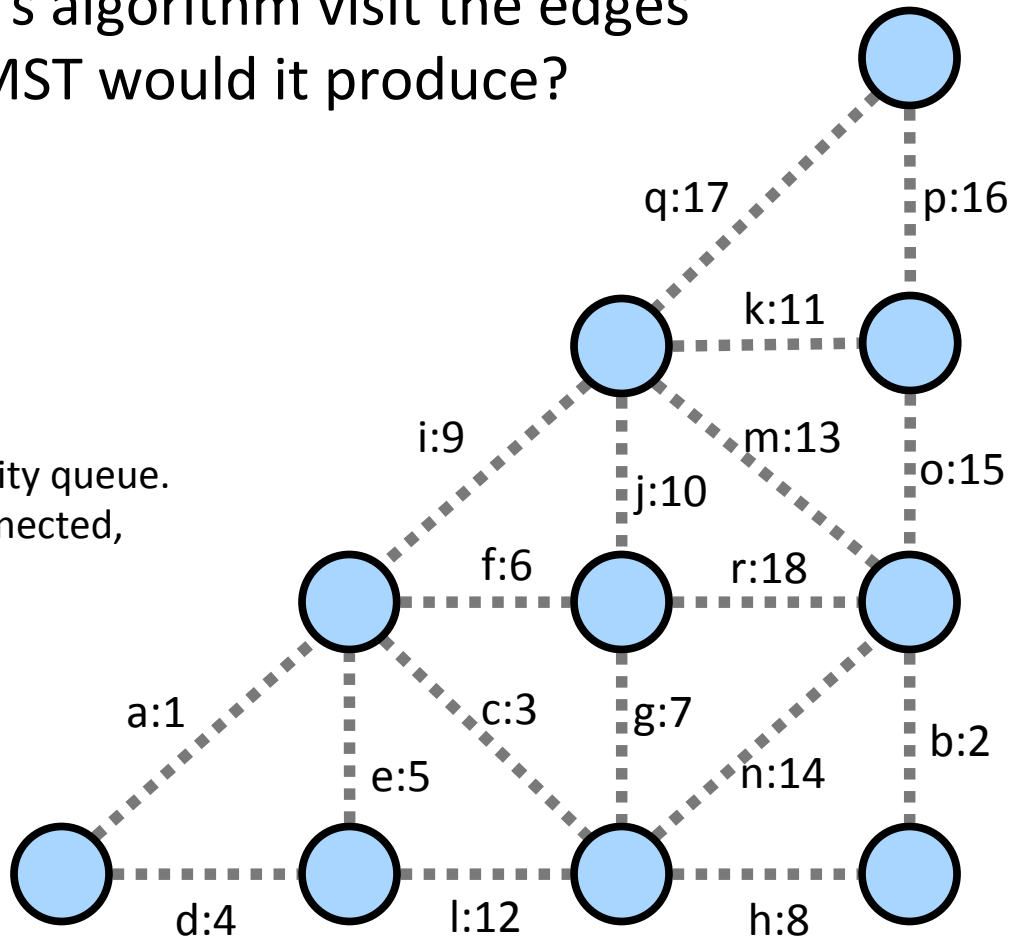
- **Kruskal's algorithm**: Finds a MST in a given graph.

```
function kruskal(graph):
    Remove all edges from the graph.
    Place all edges into a priority queue based on their weight (cost).
    While the priority queue is not empty:
        Dequeue an edge e from the priority queue.
        If e's endpoints aren't already connected to one another,
                add that edge into the graph.
        Otherwise, skip the edge.
```

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

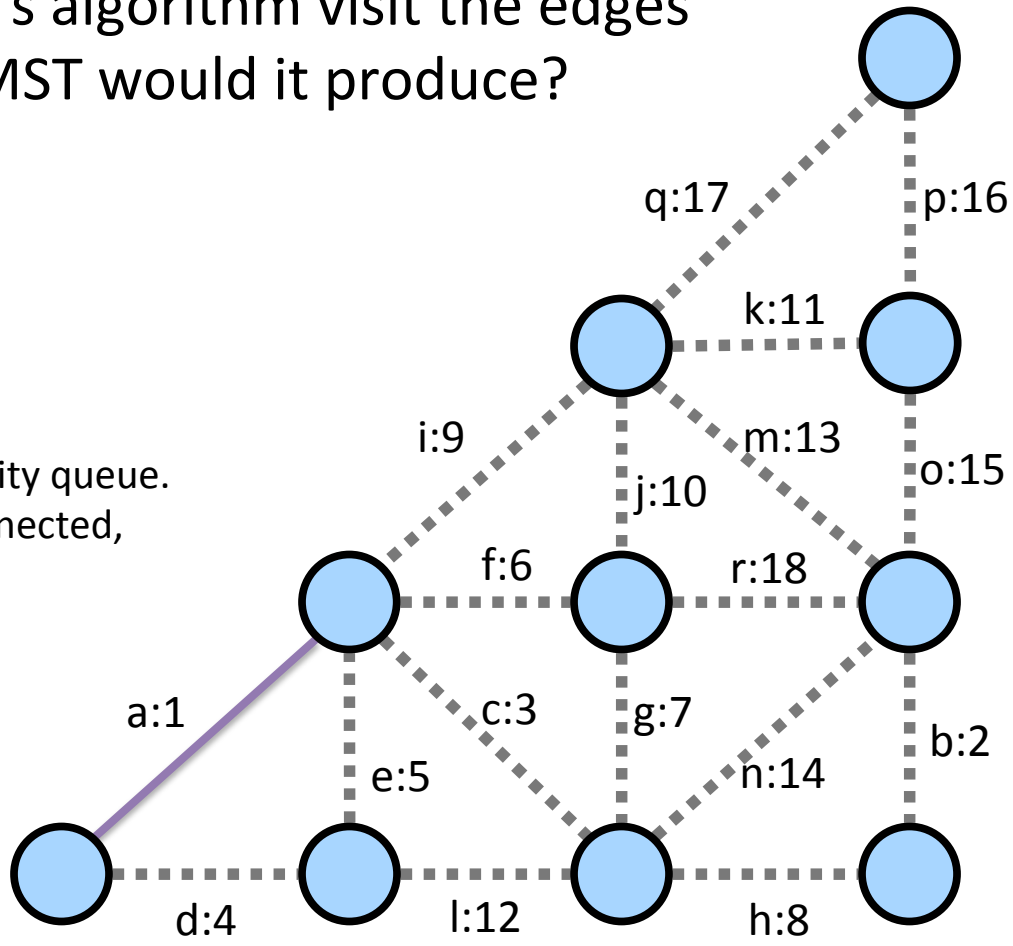


pq = {a:1, b:2, c:3, d:4, e:5, f:6, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

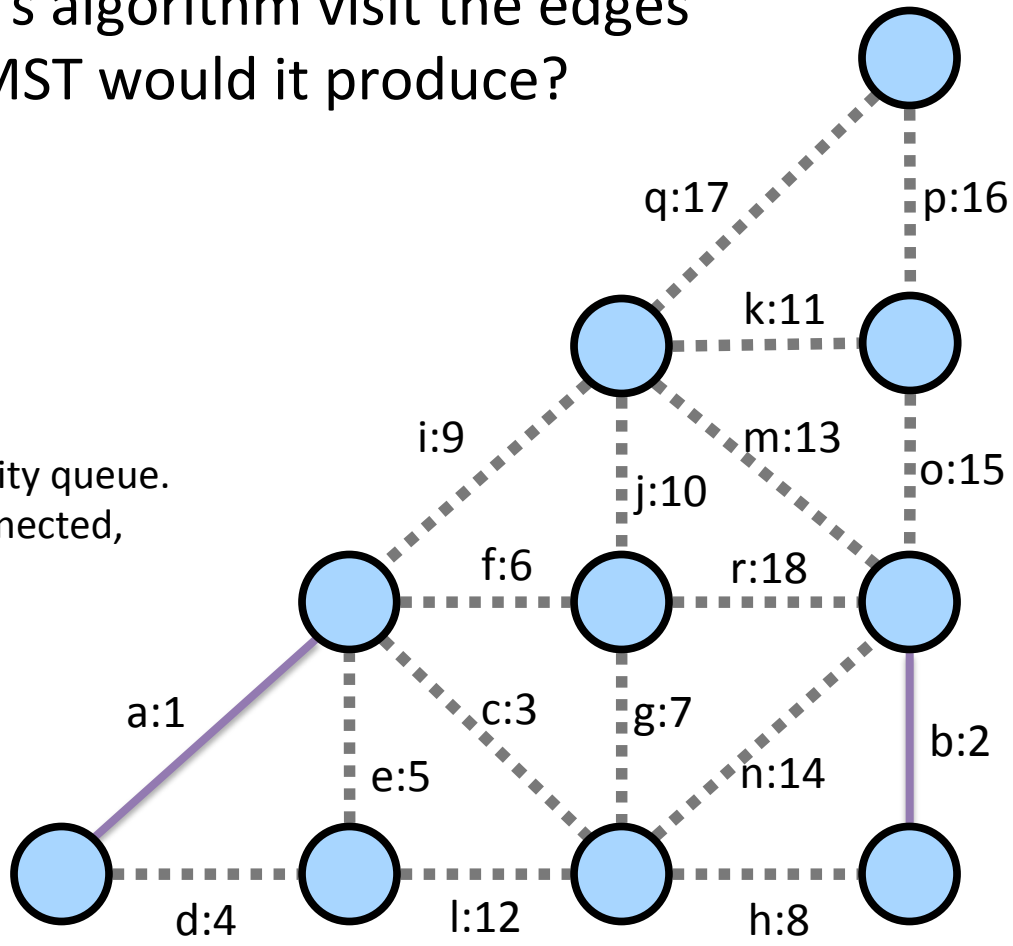


pq = {**a:1**, b:2, c:3, d:4, e:5, f:6, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

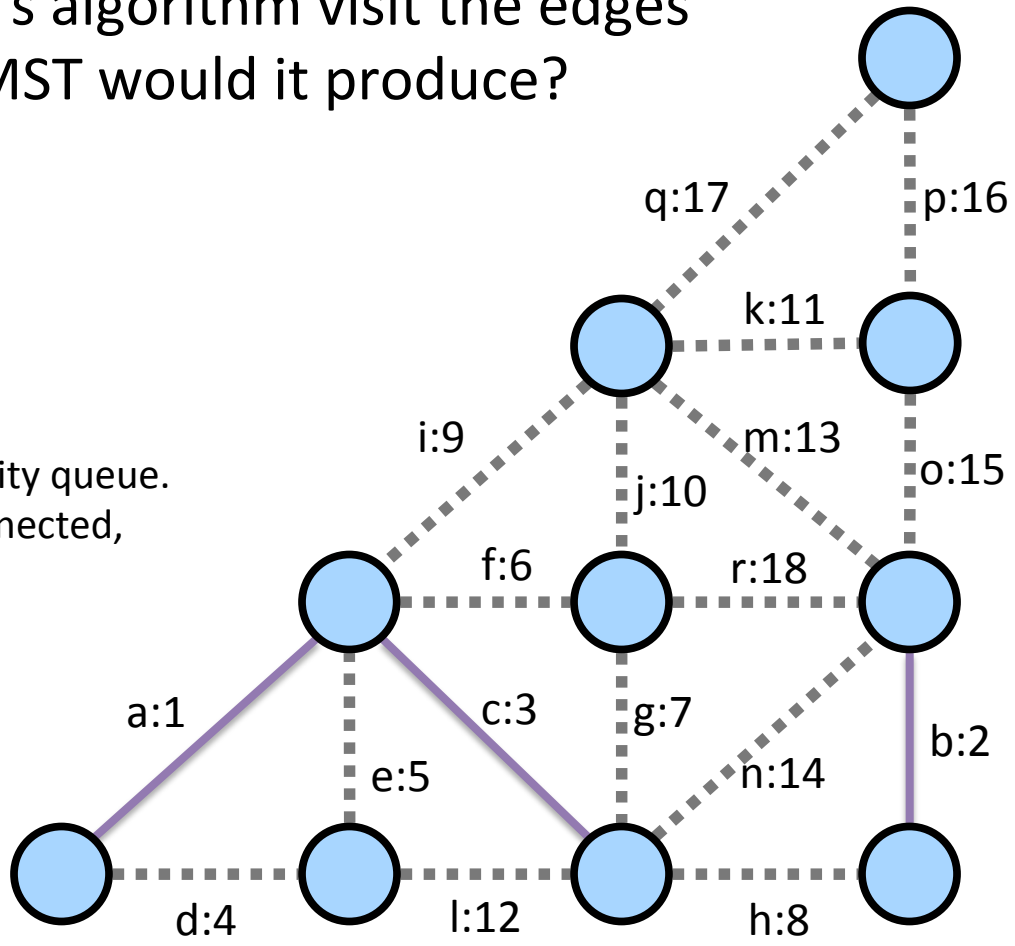


pq = {**b:2**, c:3, d:4, e:5, f:6, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

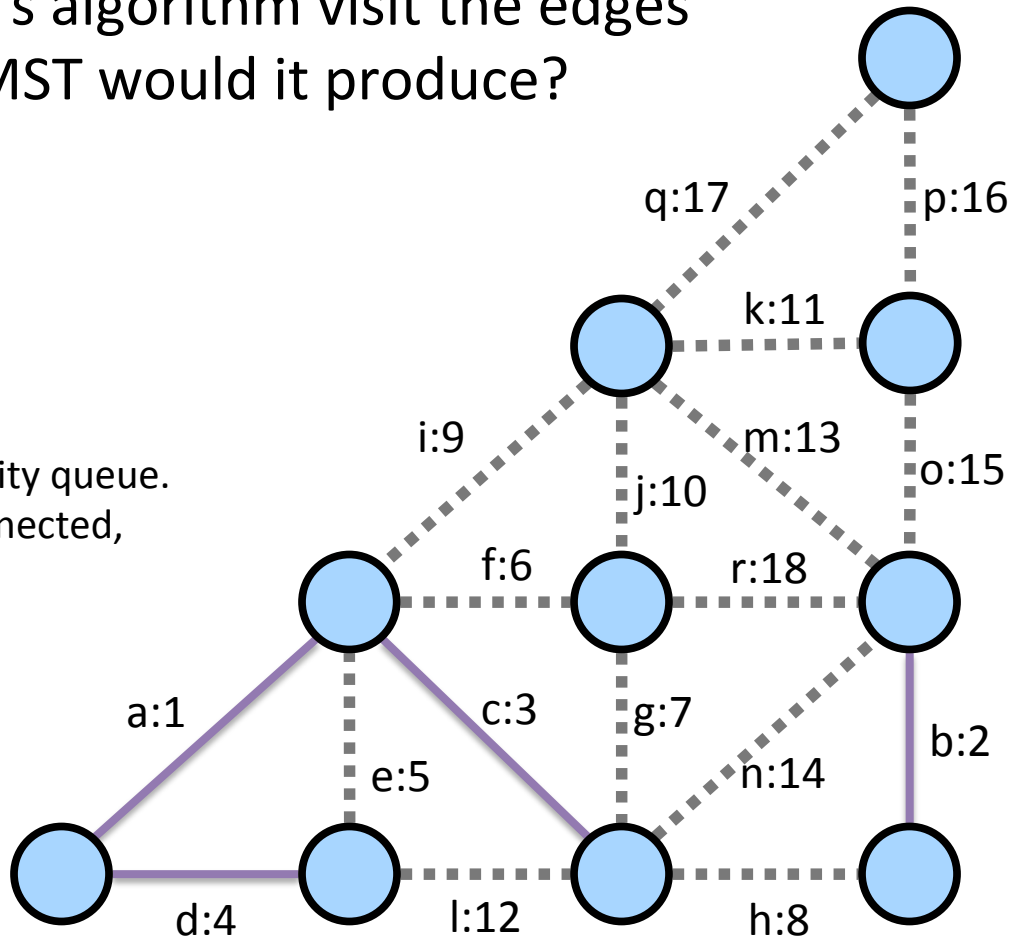


pq = {**c:3**, d:4, e:5, f:6, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

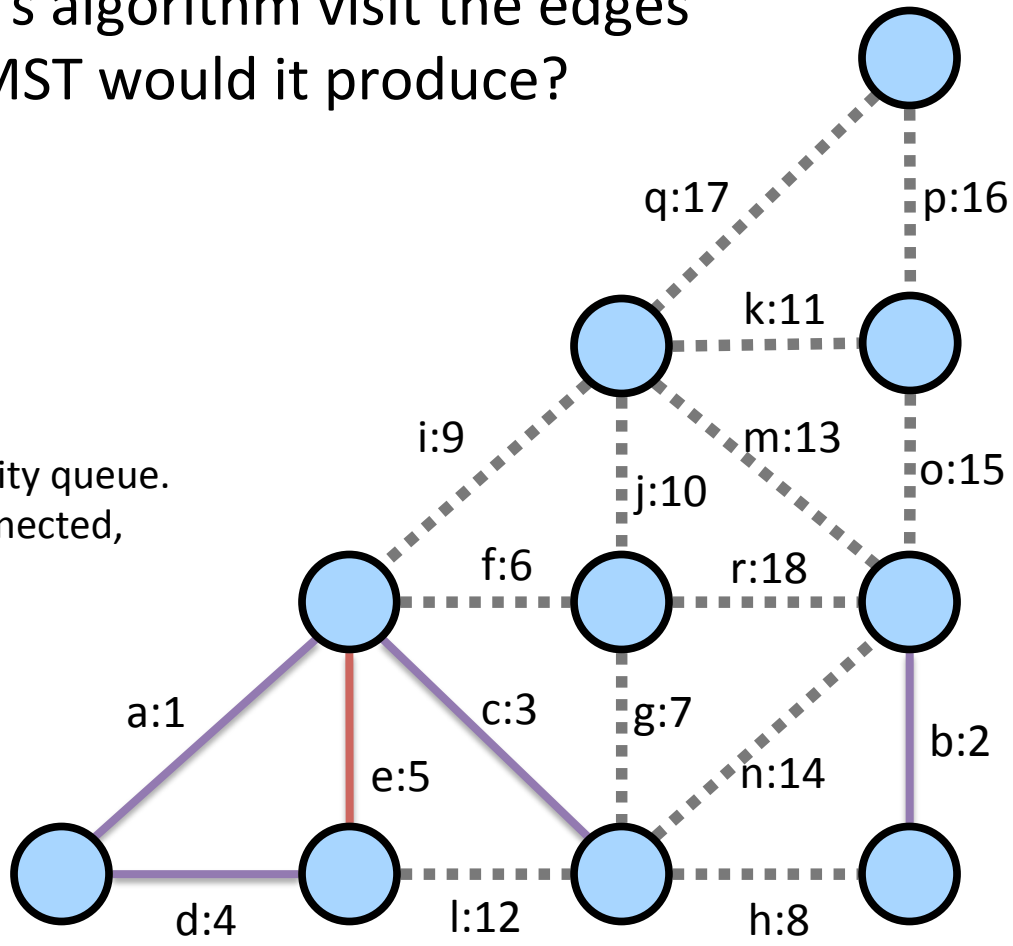


pq = {**d:4**, e:5, f:6, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

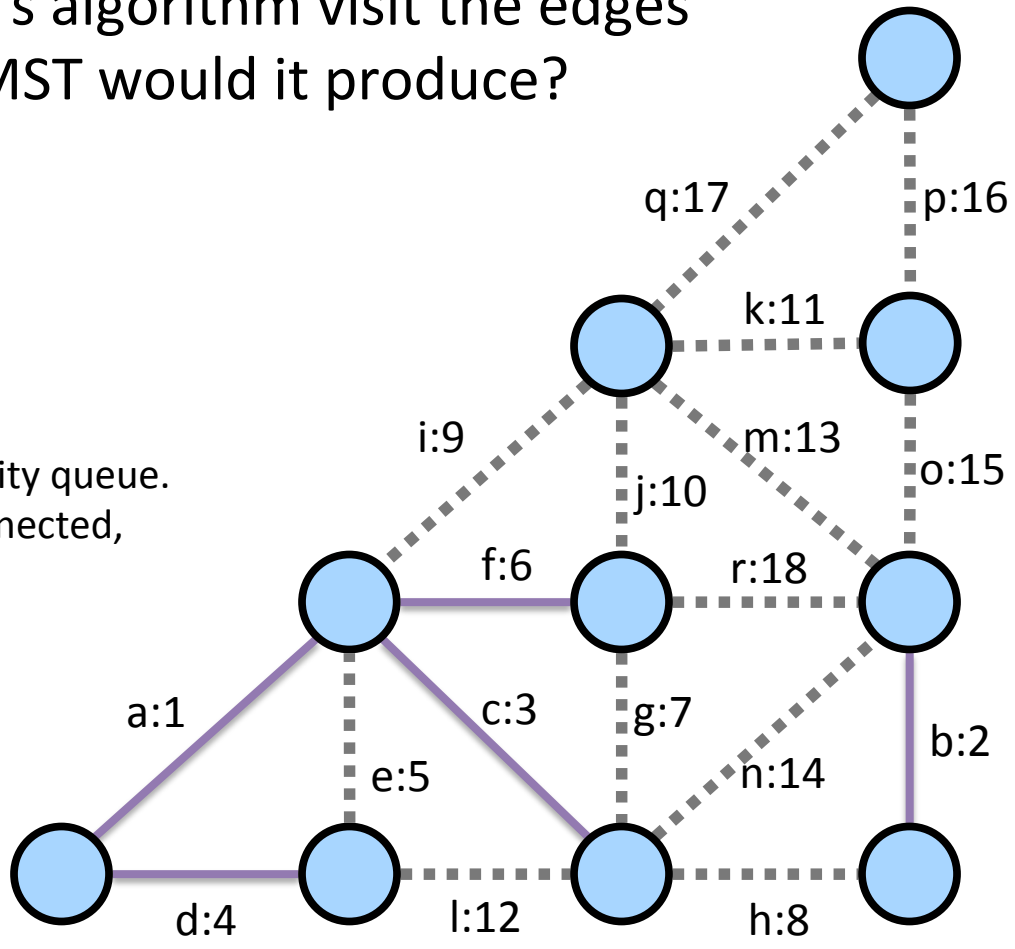


pq = {**e:5**, f:6, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

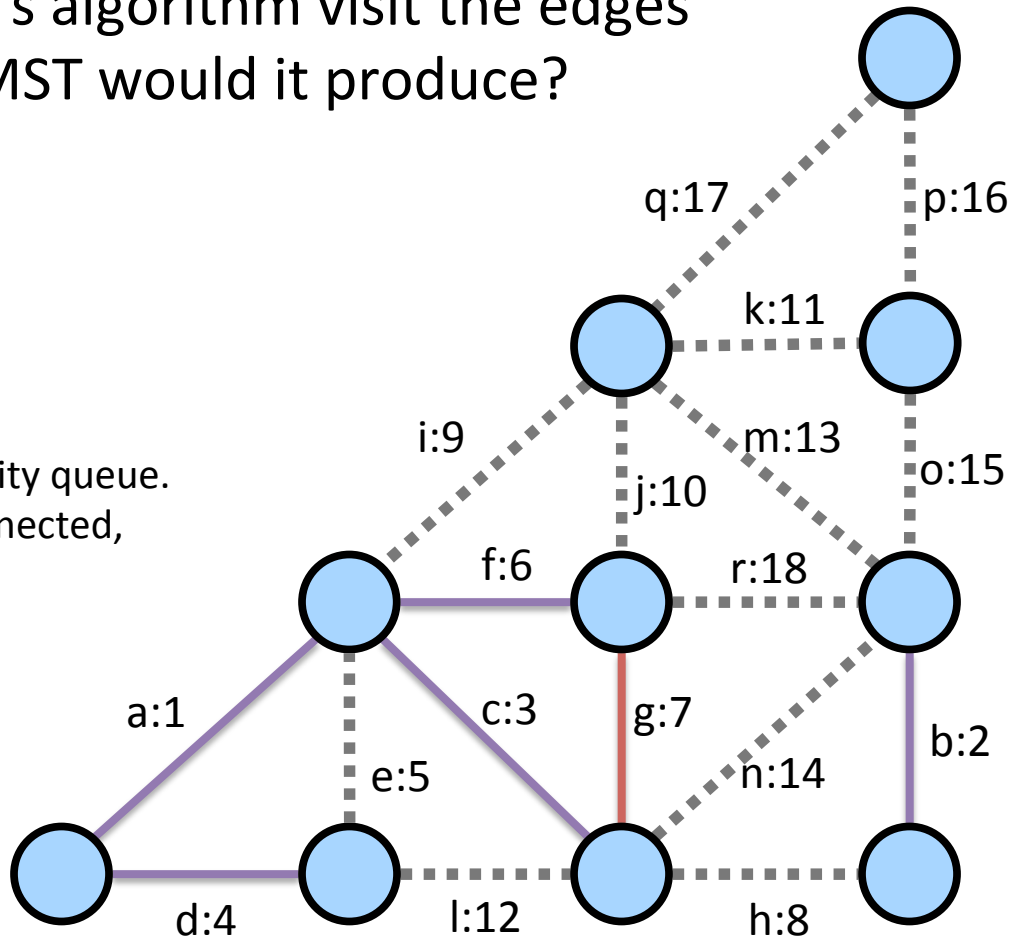


pq = {**f:6**, g:7, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

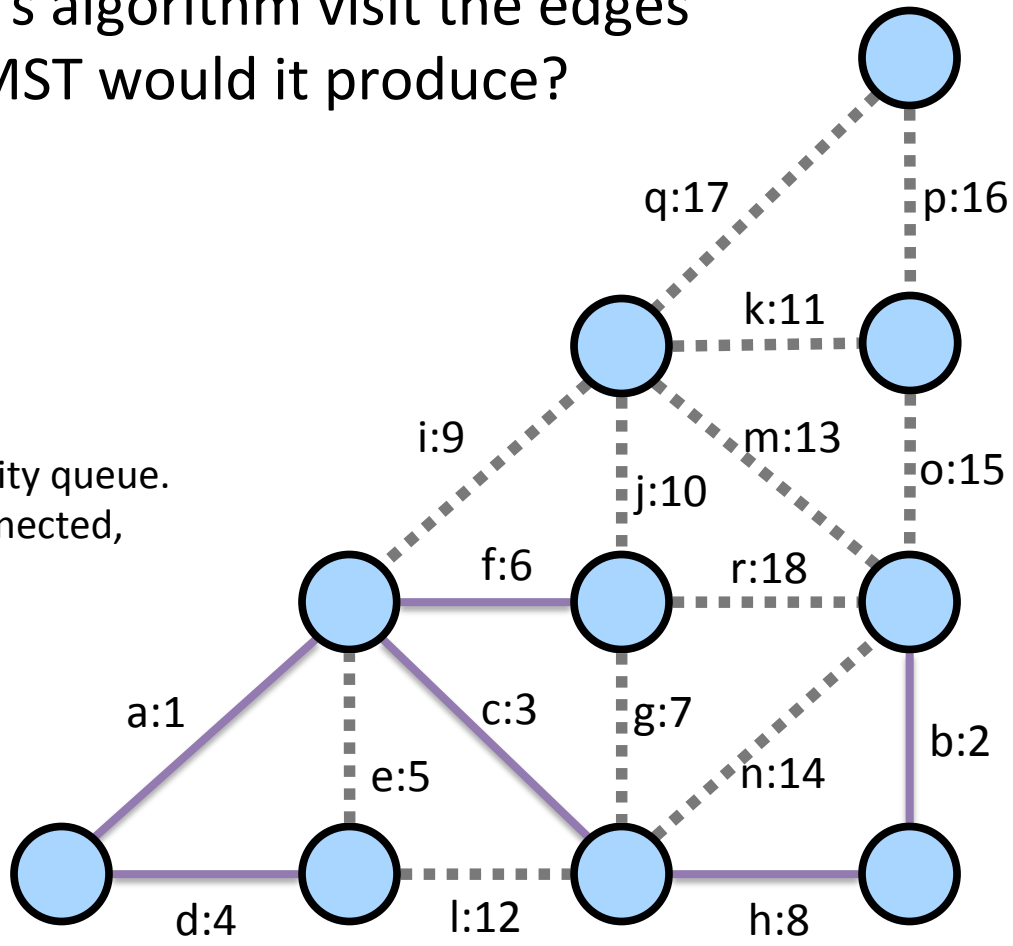


pq = {**g:7**, h:8, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

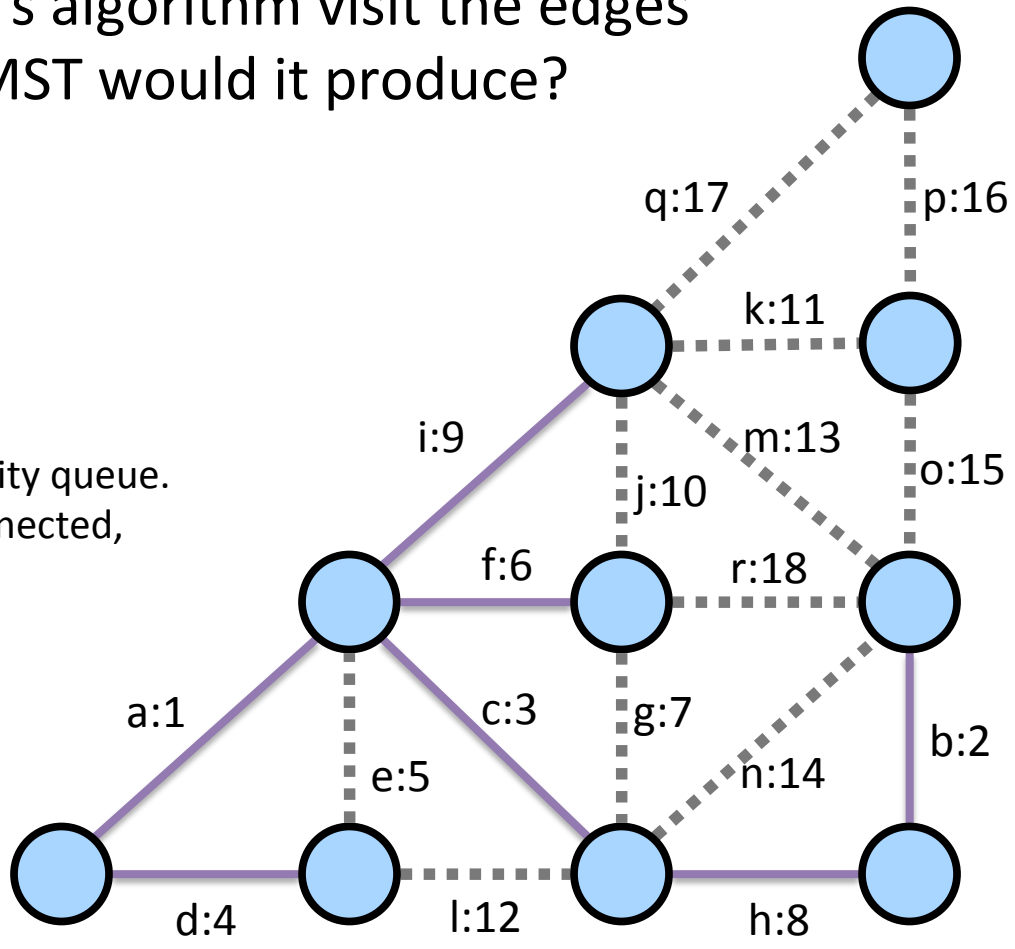


pq = {**h:8**, i:9, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

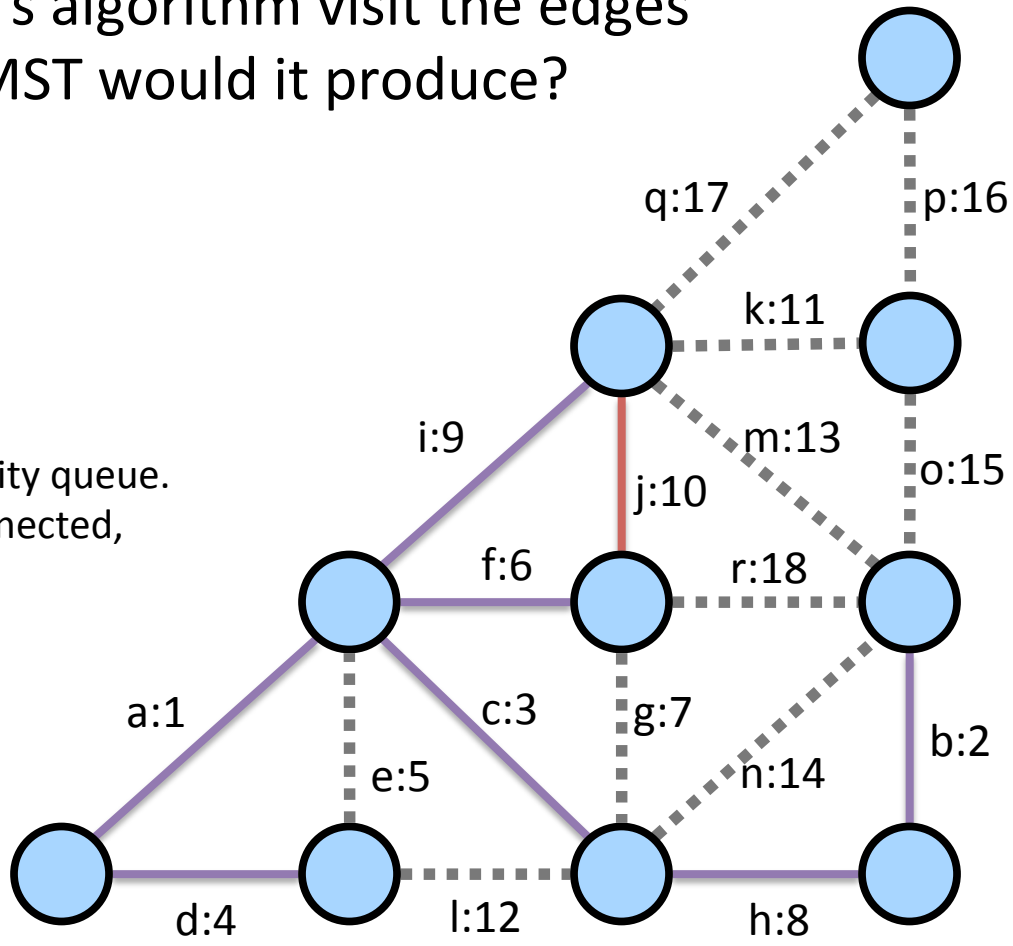


pq = {**i:9**, j:10, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

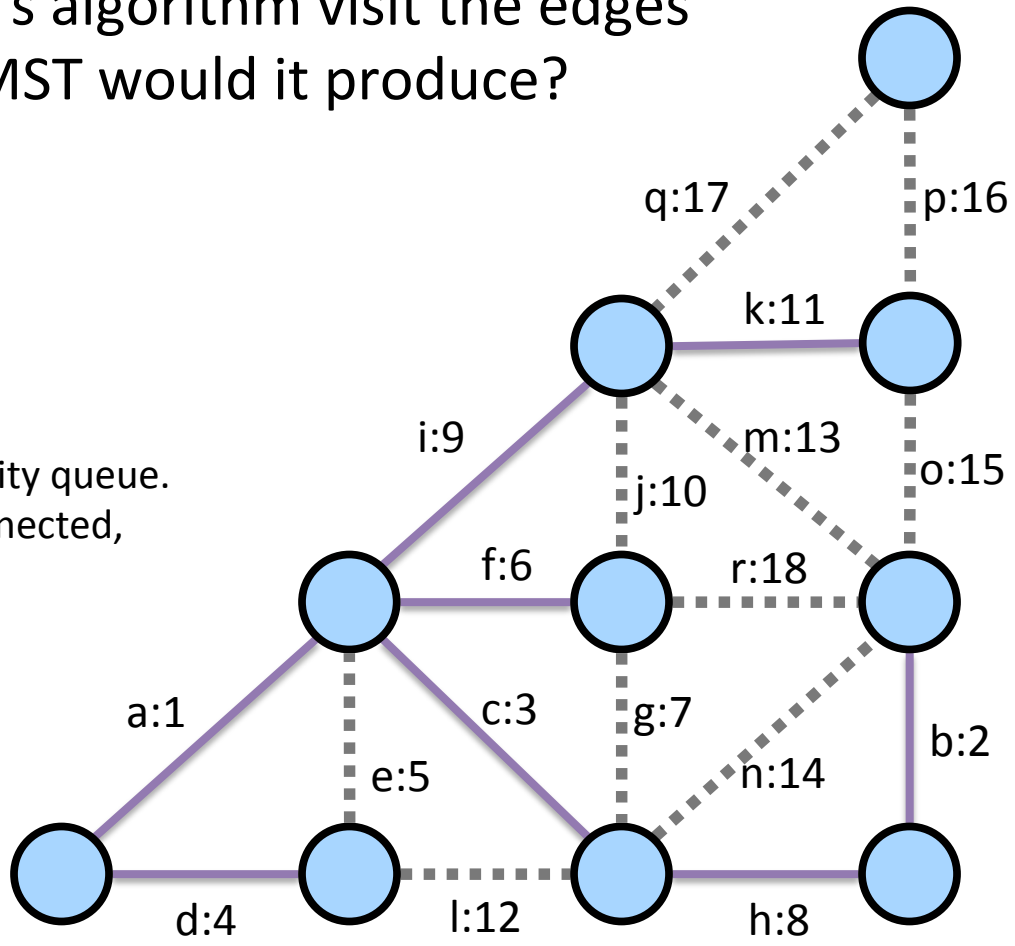


pq = {**j:10**, k:11, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

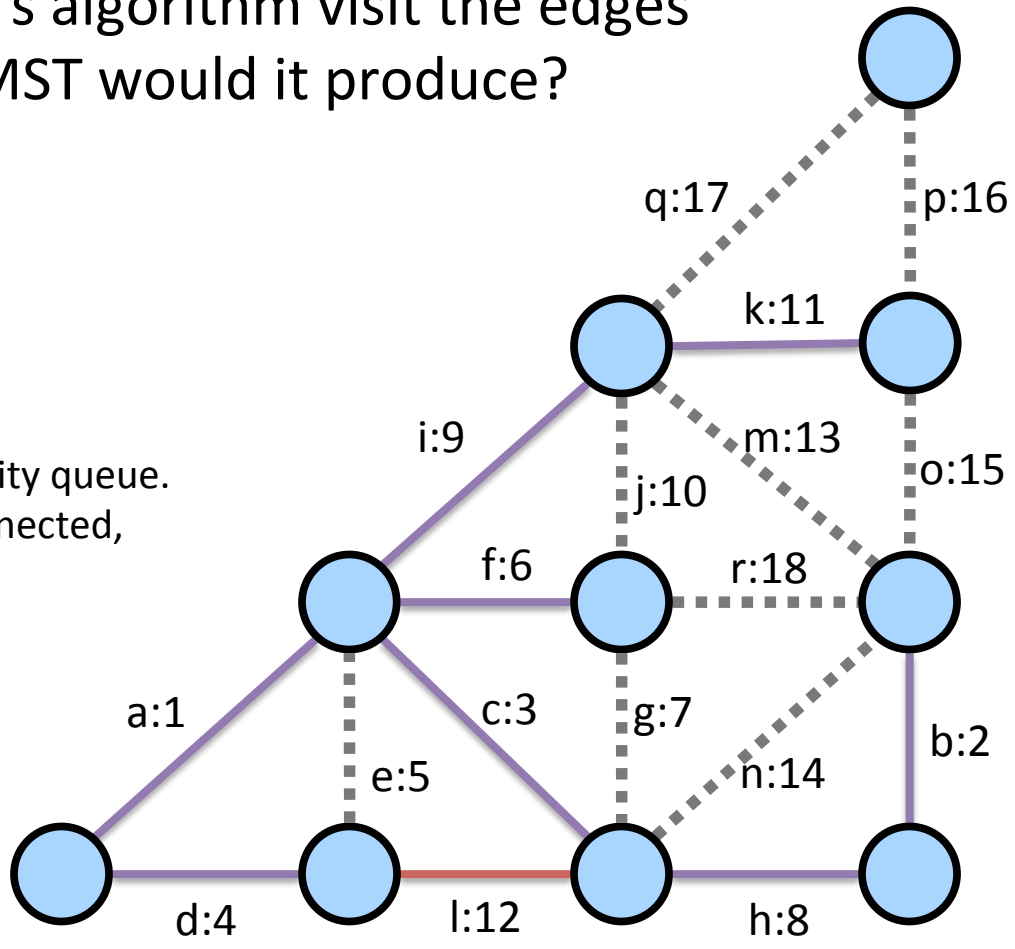


pq = {**k:11**, l:12, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

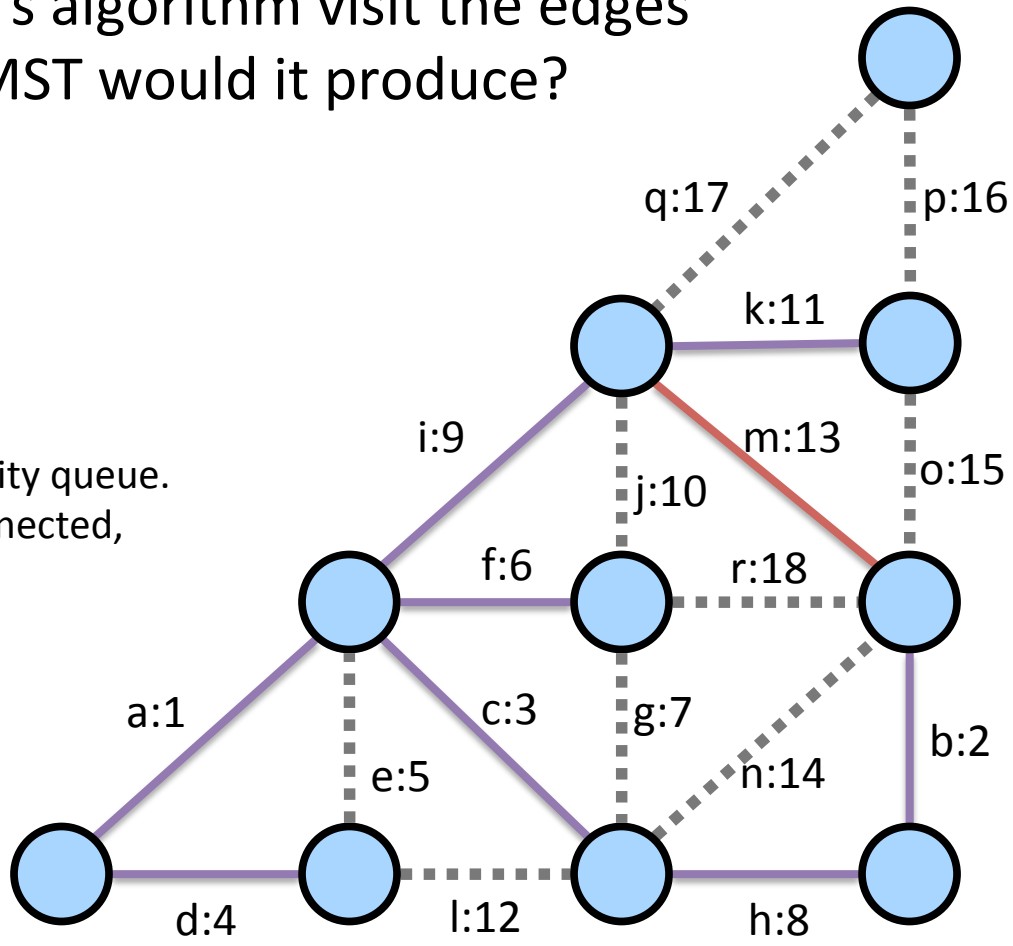


pq = {**l:12**, m:13, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

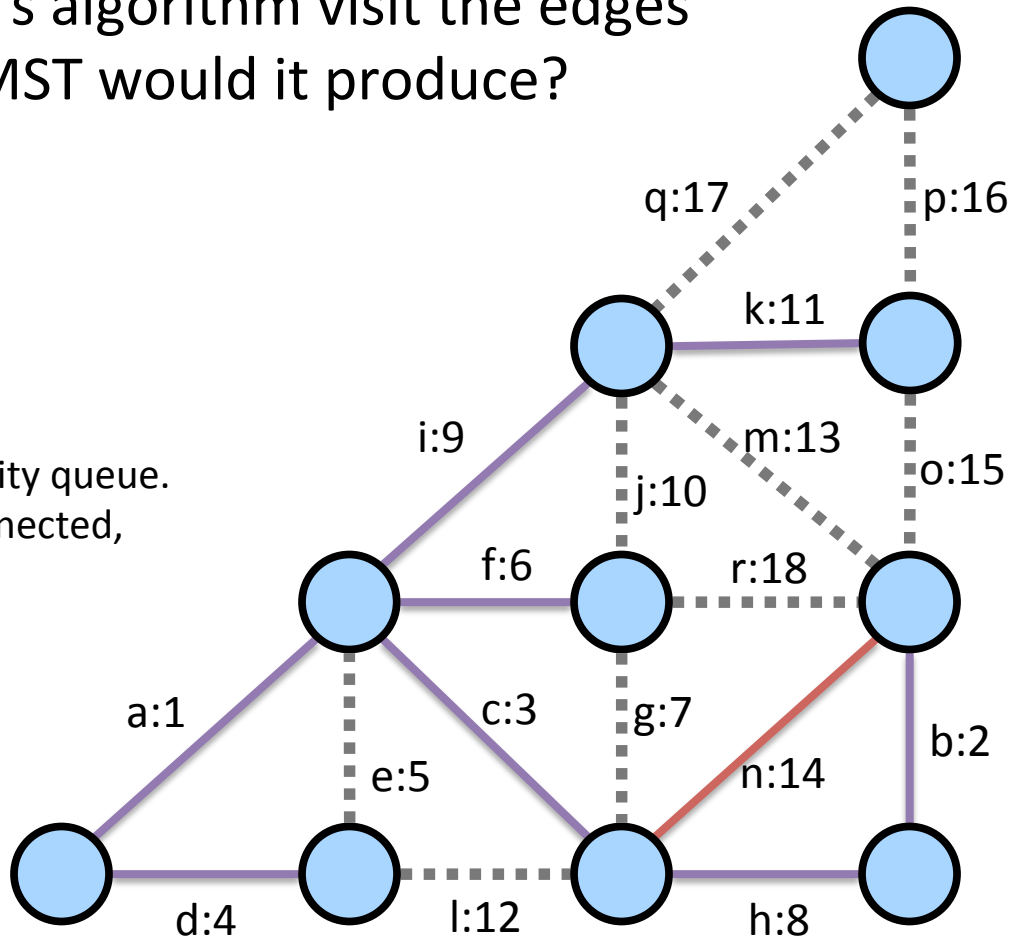


pq = {**m:13**, n:14, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

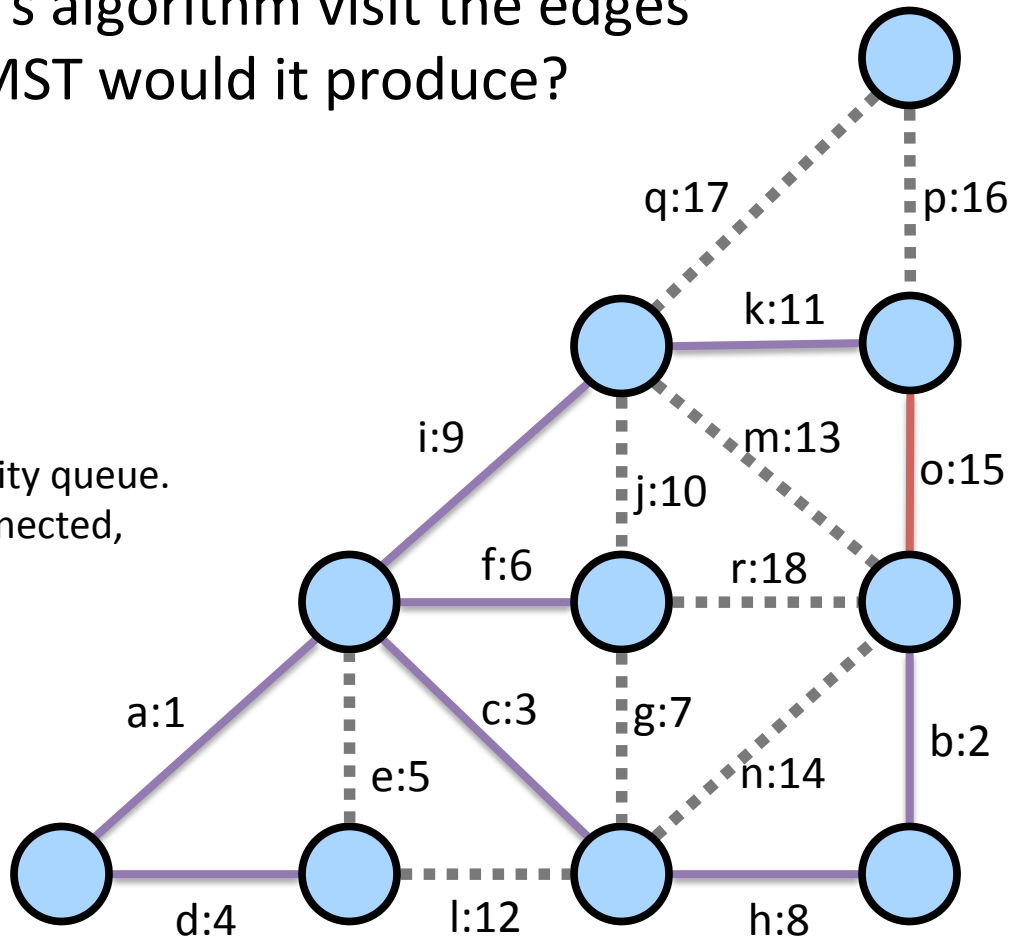


pq = {**n:14**, o:15, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

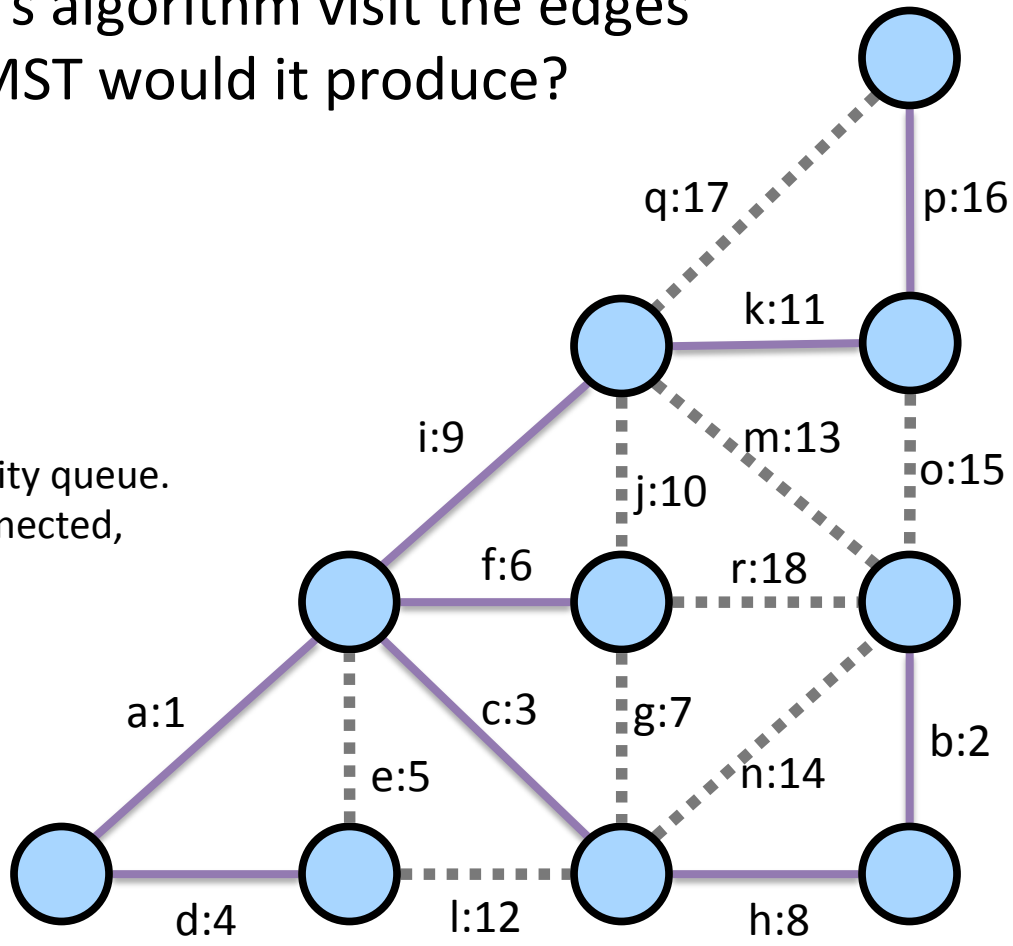


pq = {**o:15**, p:16, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

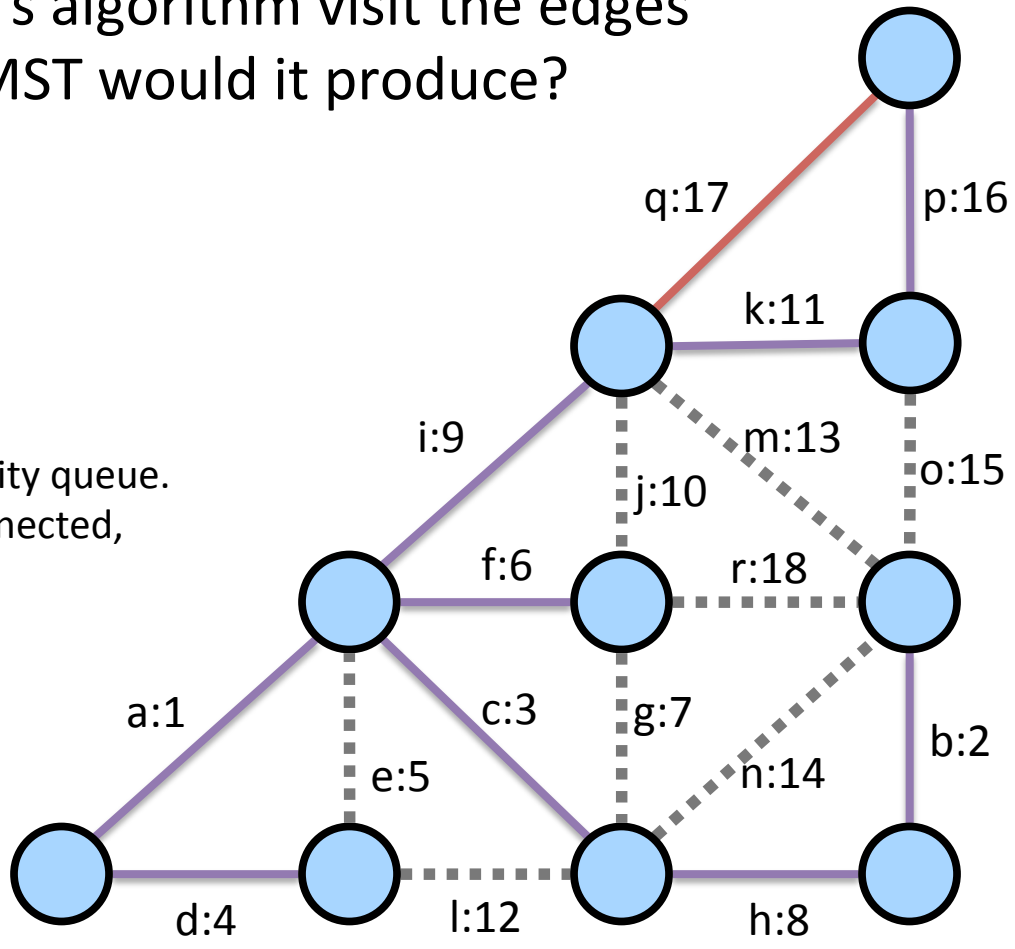


pq = {**p:16**, q:17, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

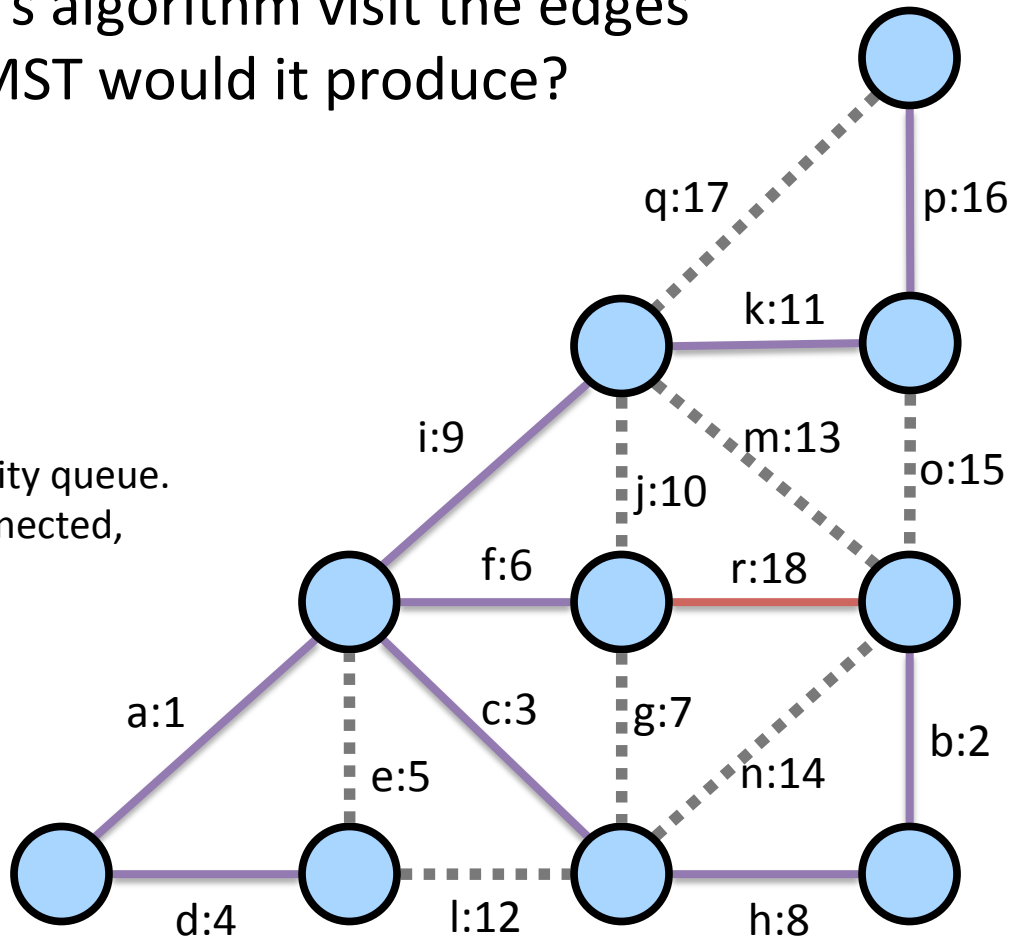


pq = {**q:17**, r:18}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.



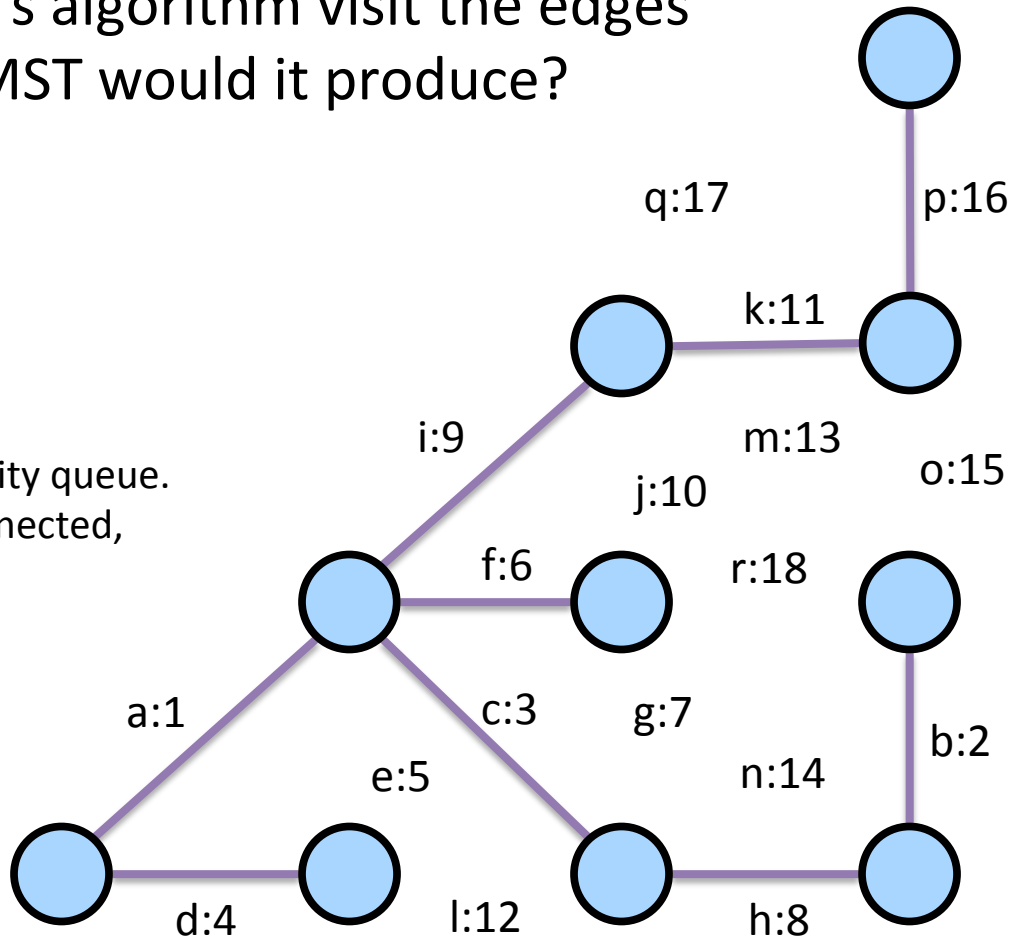
pq = {**r:18**}

Kruskal example

- In what order would Kruskal's algorithm visit the edges in the graph below? What MST would it produce?

function **kruskal**(graph):
  Remove all edges from the graph.
  Place all edges into a priority queue based on their weight (cost).
  While the priority queue is not empty:
    Dequeue an edge *e* from the priority queue.
    If *e*'s endpoints aren't already connected, add that edge into the graph.
    Otherwise, skip the edge.

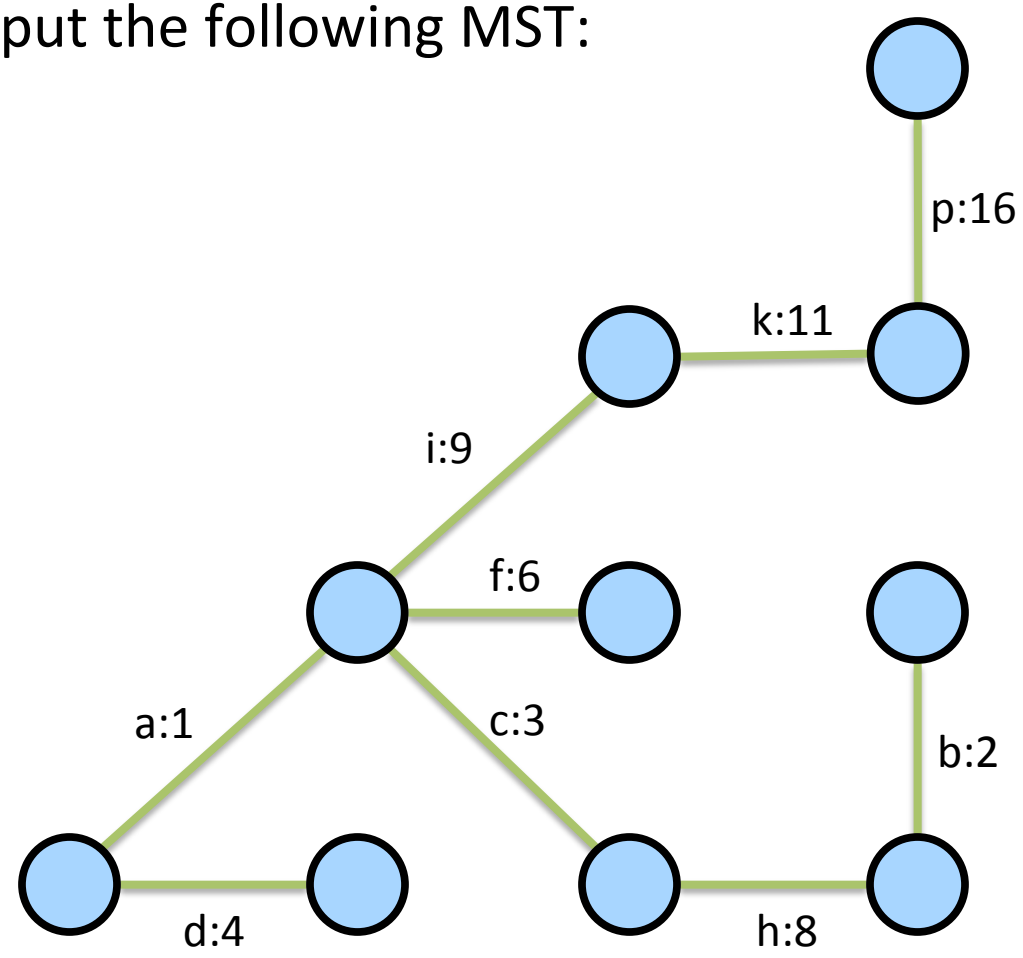
pq = {}



Kruskal example

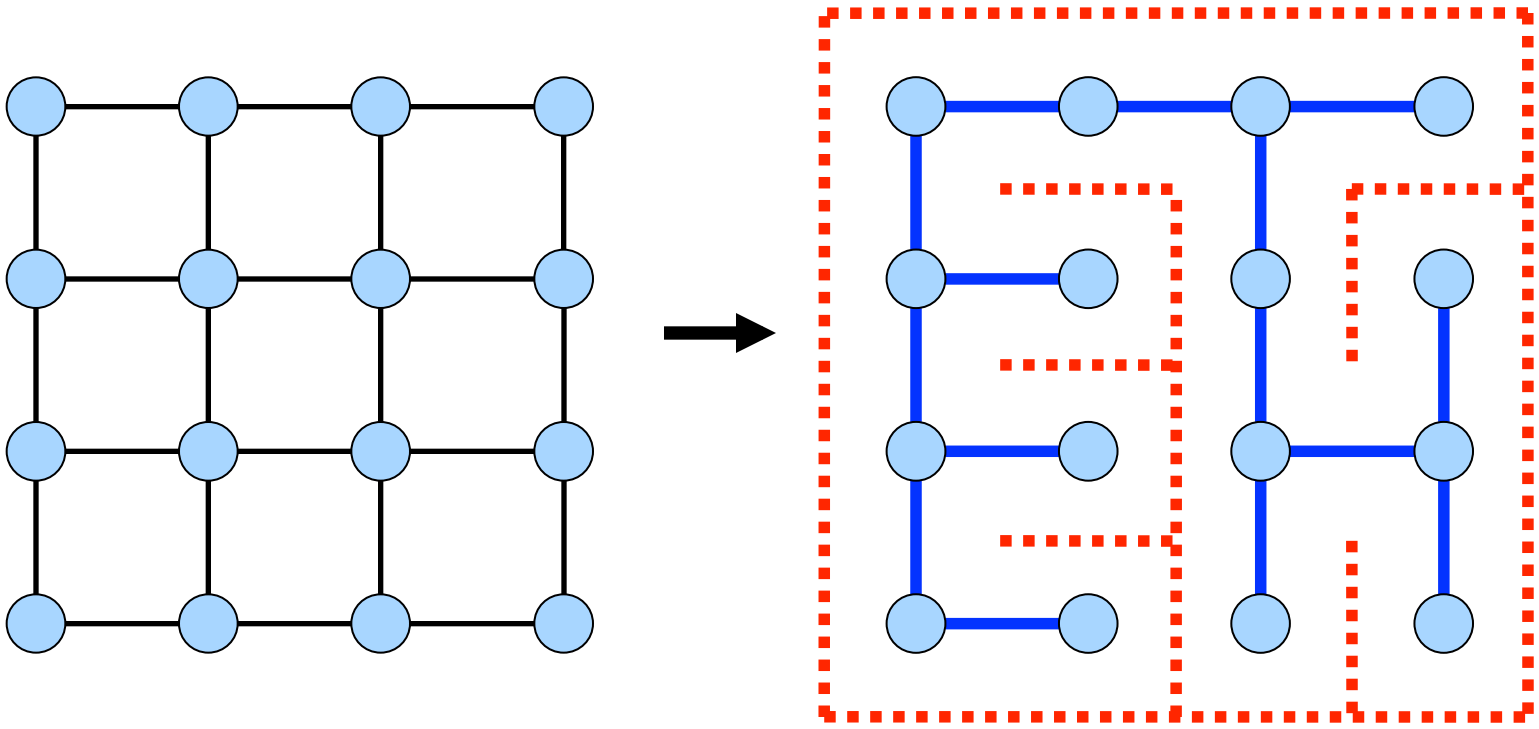
- Kruskal's algorithm would output the following MST:
  - {a, b, c, d, f, h, i, k, p}
- The MST's total cost is:

  1+2+3+4+6+8+9+11+16 = 60



Mazes with Kruskal's

- Start with a fully connected maze, and then run Kruskal's to add walls (edges) back in until you have a MST:



Implementing Kruskal

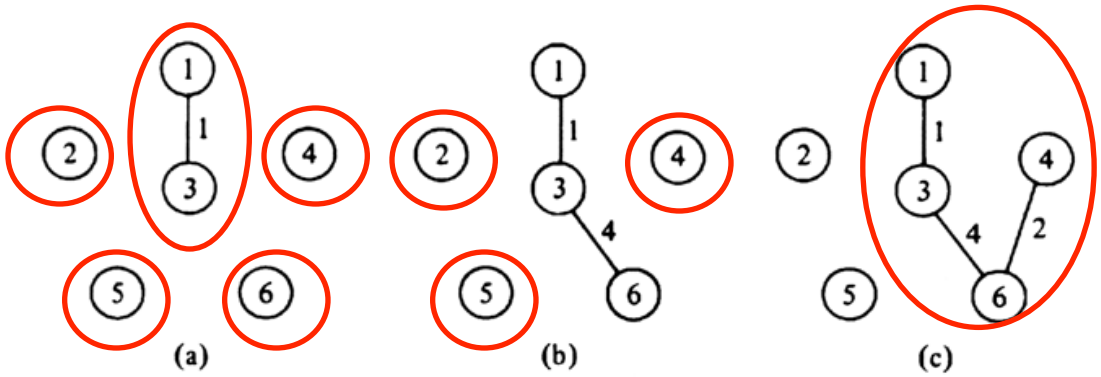
- What data structures should we use to implement this algorithm?

function **kruskal**(graph):
    Remove all edges from the graph.
    Place all edges into a **priority queue**
        based on their weight (cost).
    While the priority queue is not empty:
        Dequeue an edge *e* from the priority queue.
        If *e*'s endpoints aren't already connected,
            add that edge into the graph.
        Otherwise, skip the edge.

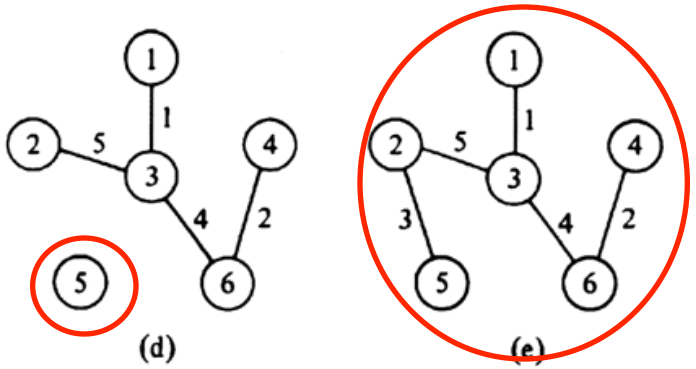
Vertex clusters

- Need some way to identify which vertexes are "connected" to which other ones
  - we call these "**clusters**" of vertices

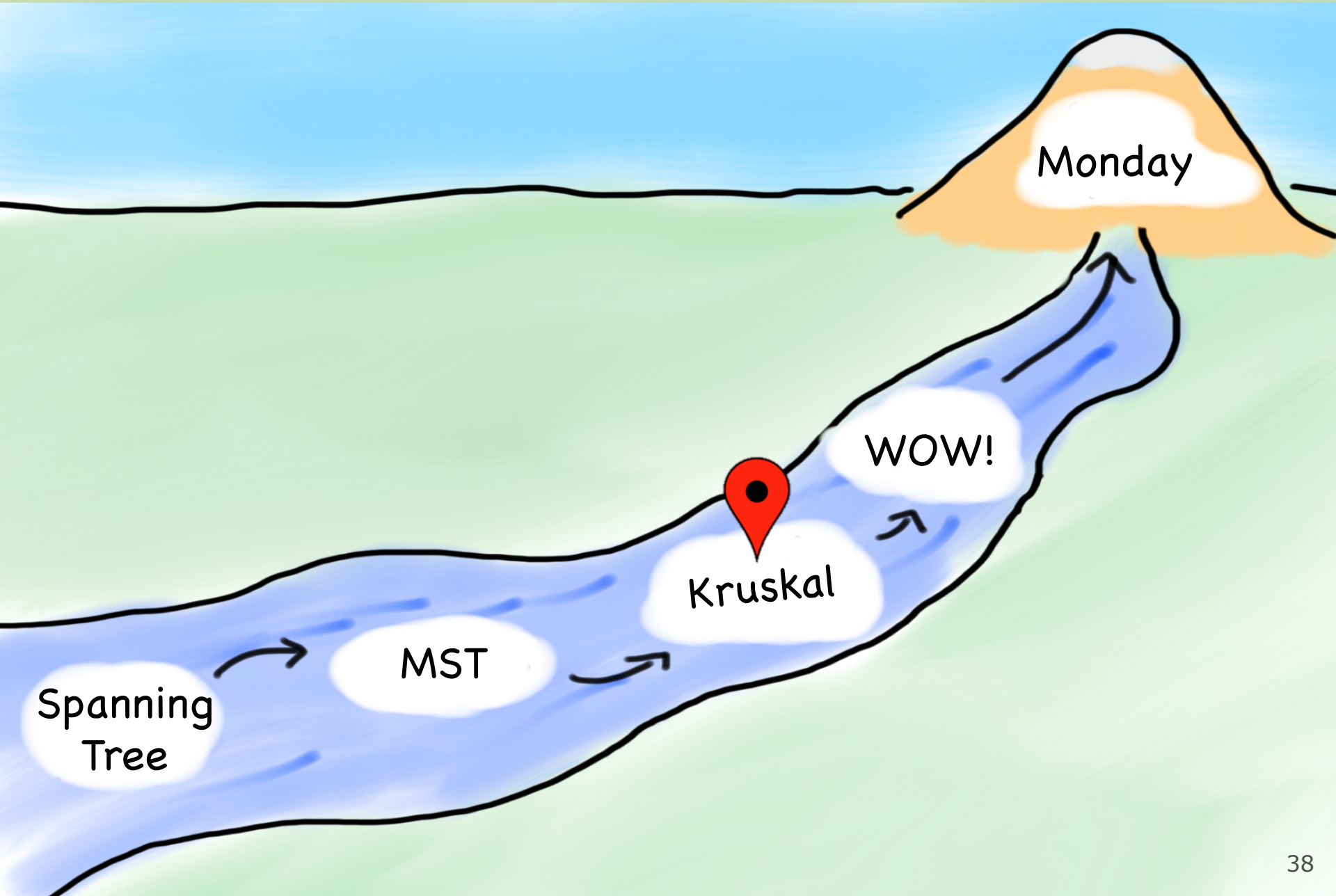
- Also need an efficient way to figure out which cluster a given vertex is in.



- Also need to **merge clusters** when adding an edge.



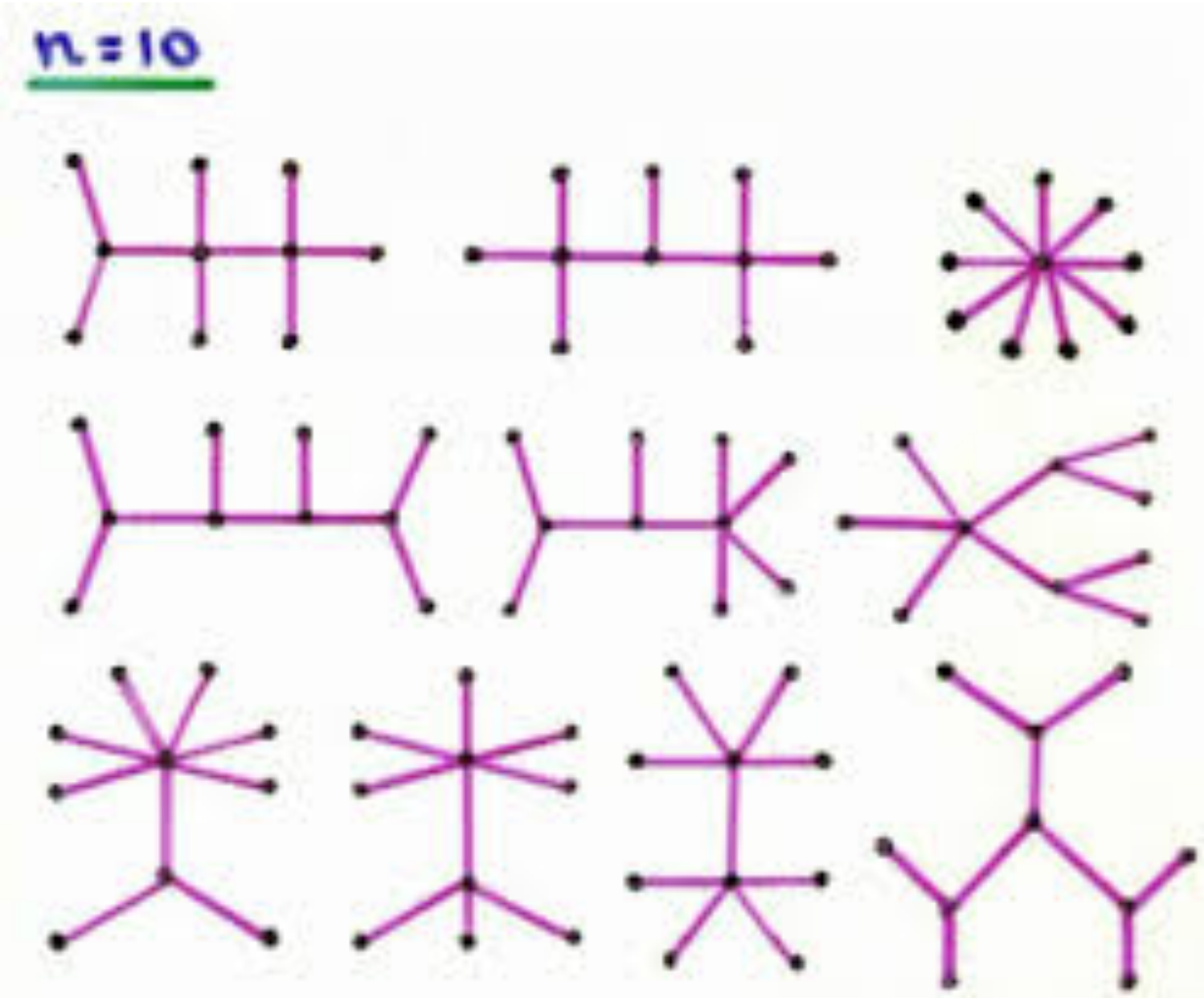
Today's Route



Today's Route

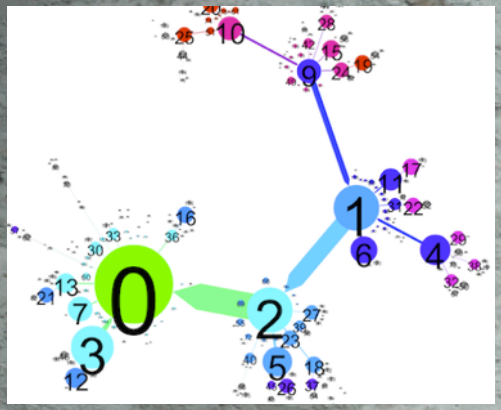

These are spanning trees!
…well homeomorphically irreducible trees

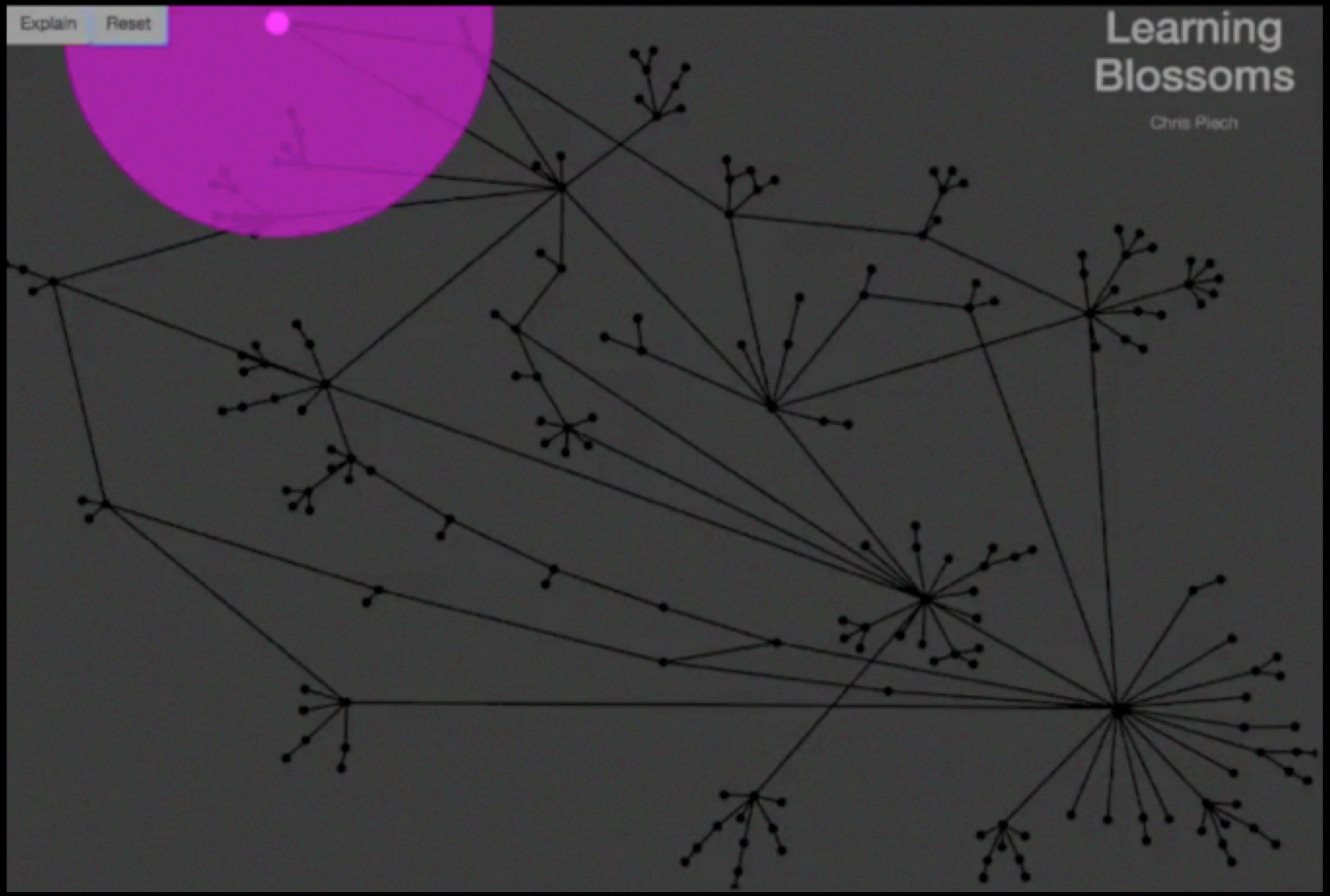
n = 10

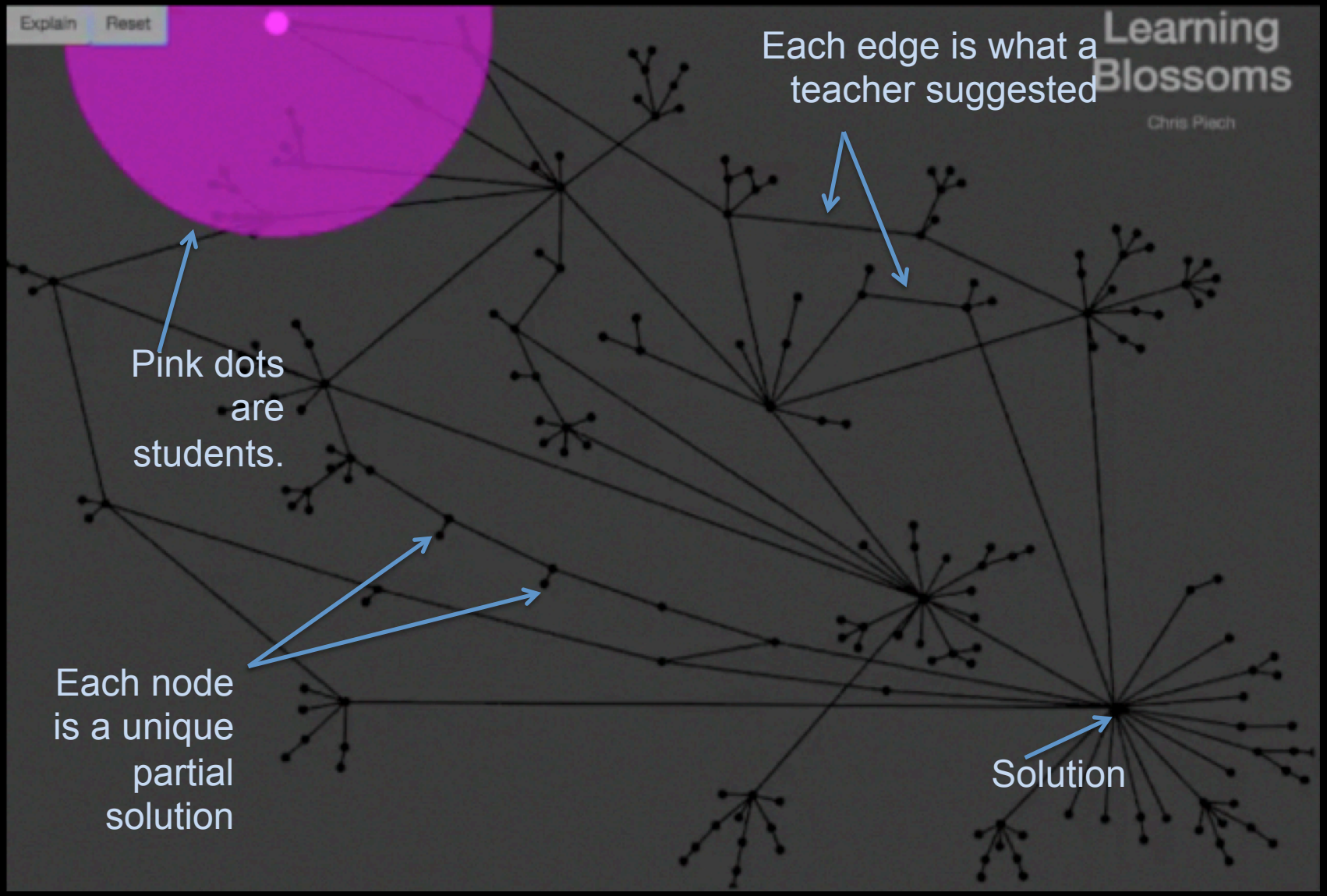


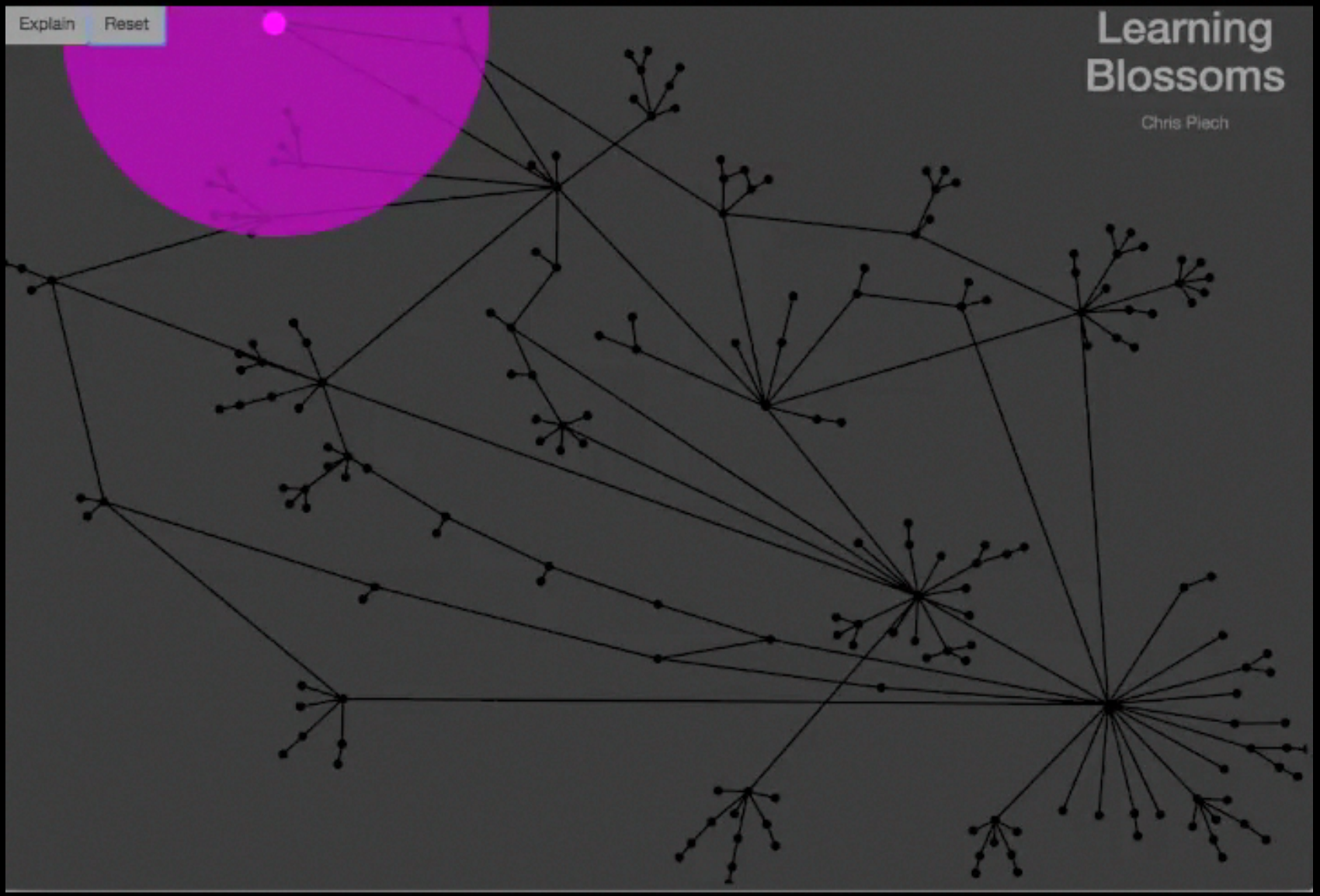
Did you know Chris is awesome?

Autonomously Generating Hints by Inferring Problem Solving Policies

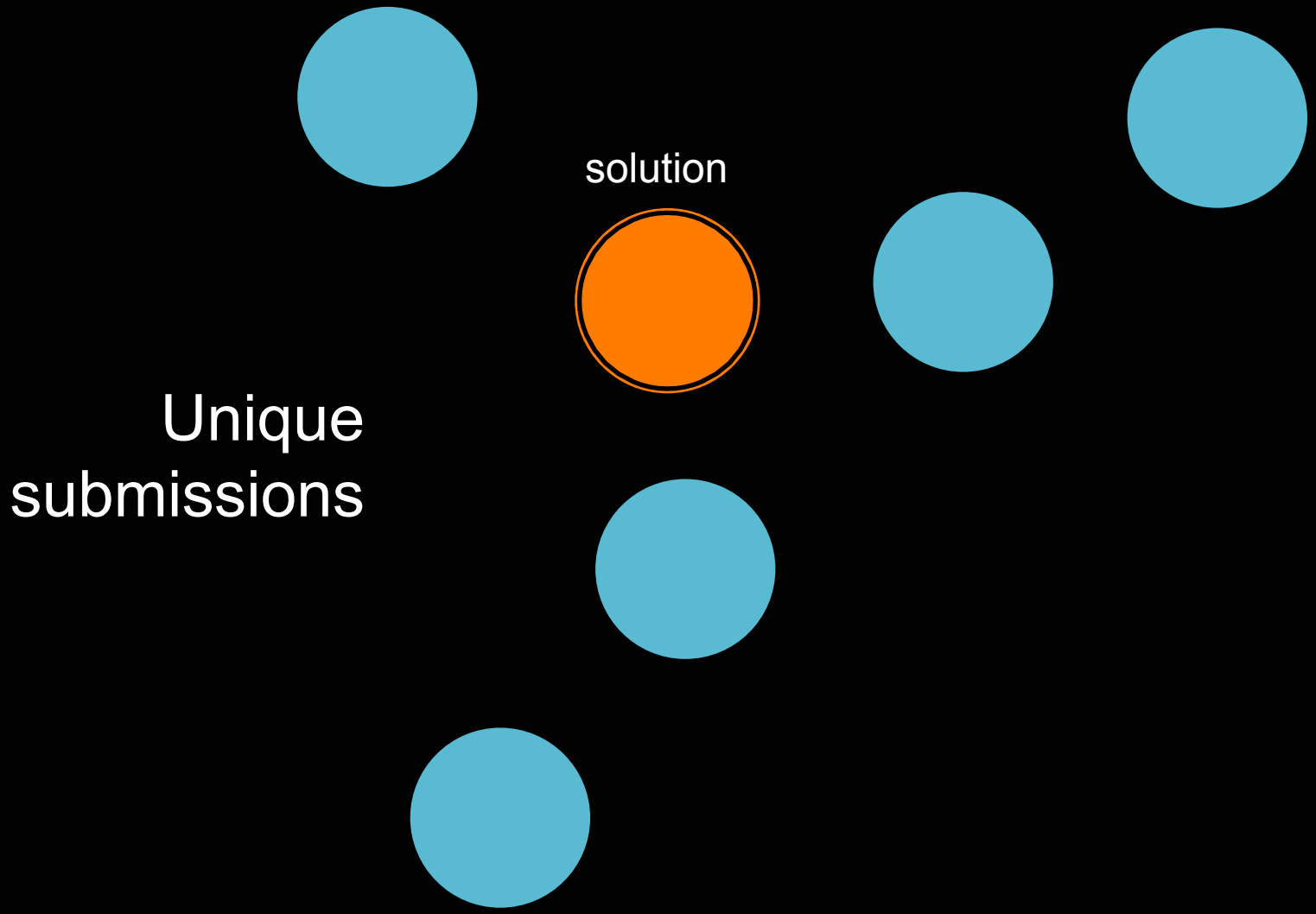



Explain
Reset
Learning
Blossoms
Chris Piech

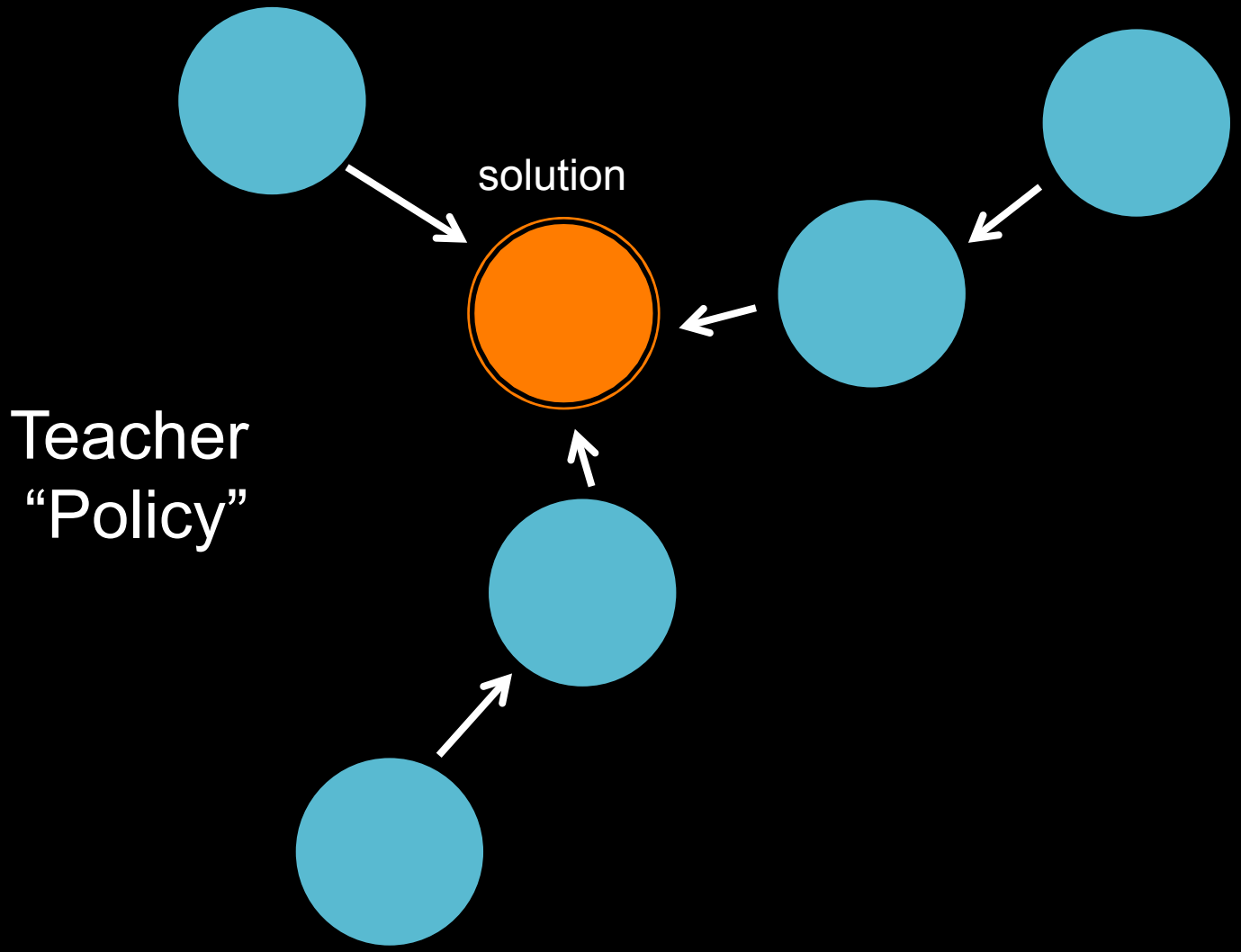

Explain
Reset
Learning Blossoms
Chris Piech
Each edge is what a teacher suggested
Pink dots are students.
Each node is a unique partial solution
Solution


Explain
Reset
Learning
Blossoms
Chris Piech

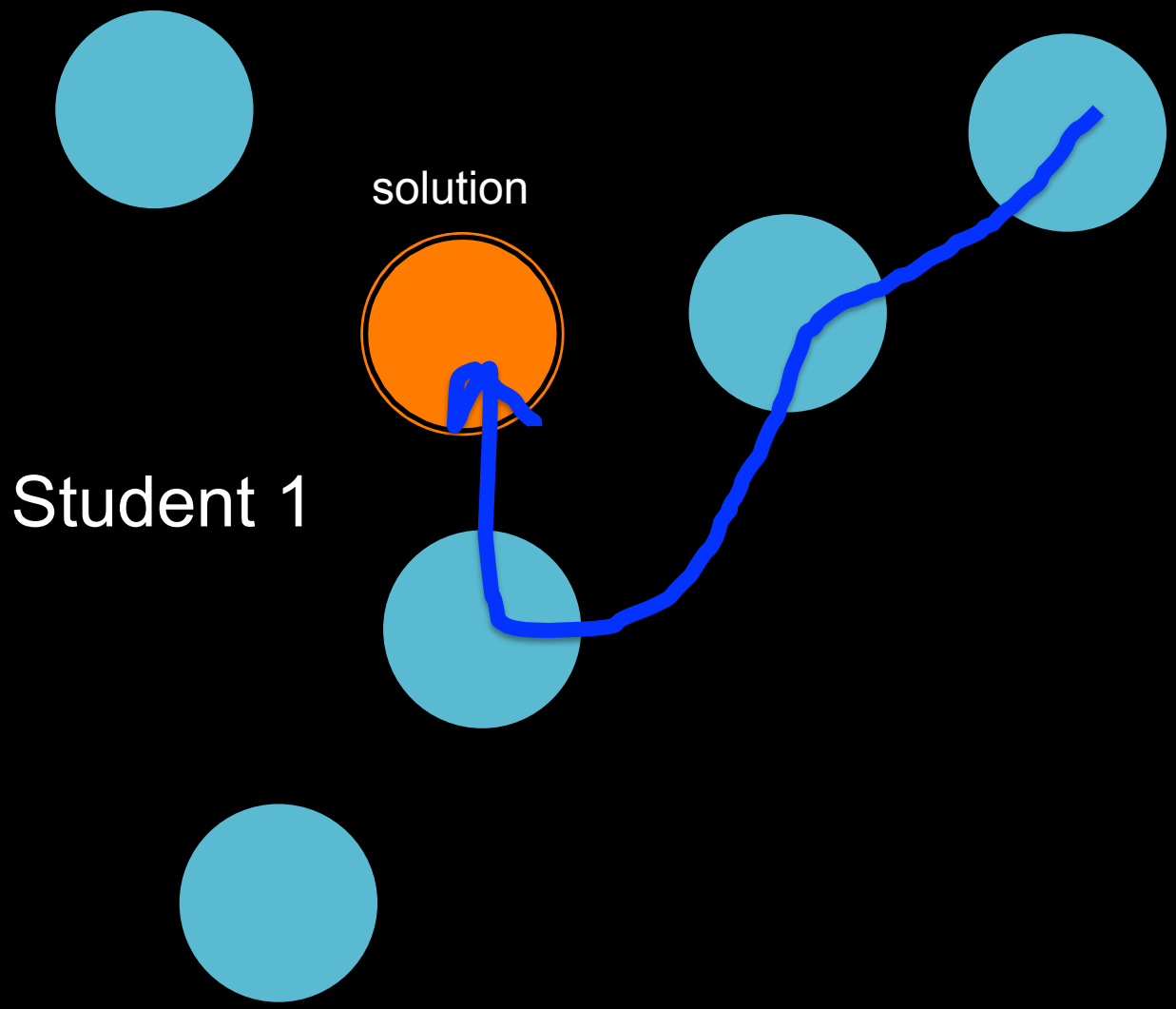
Machine Learning Problem



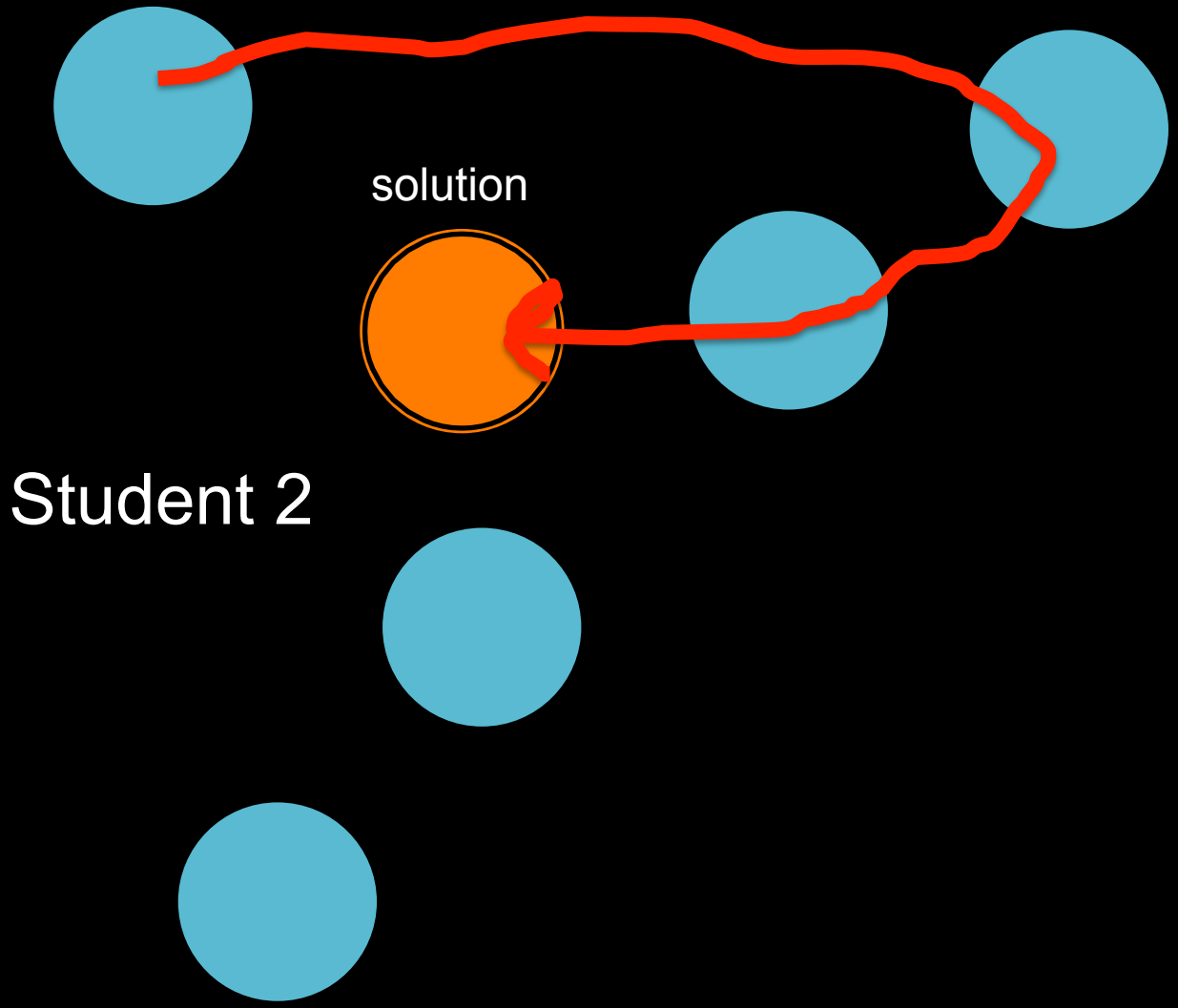
Machine Learning Problem



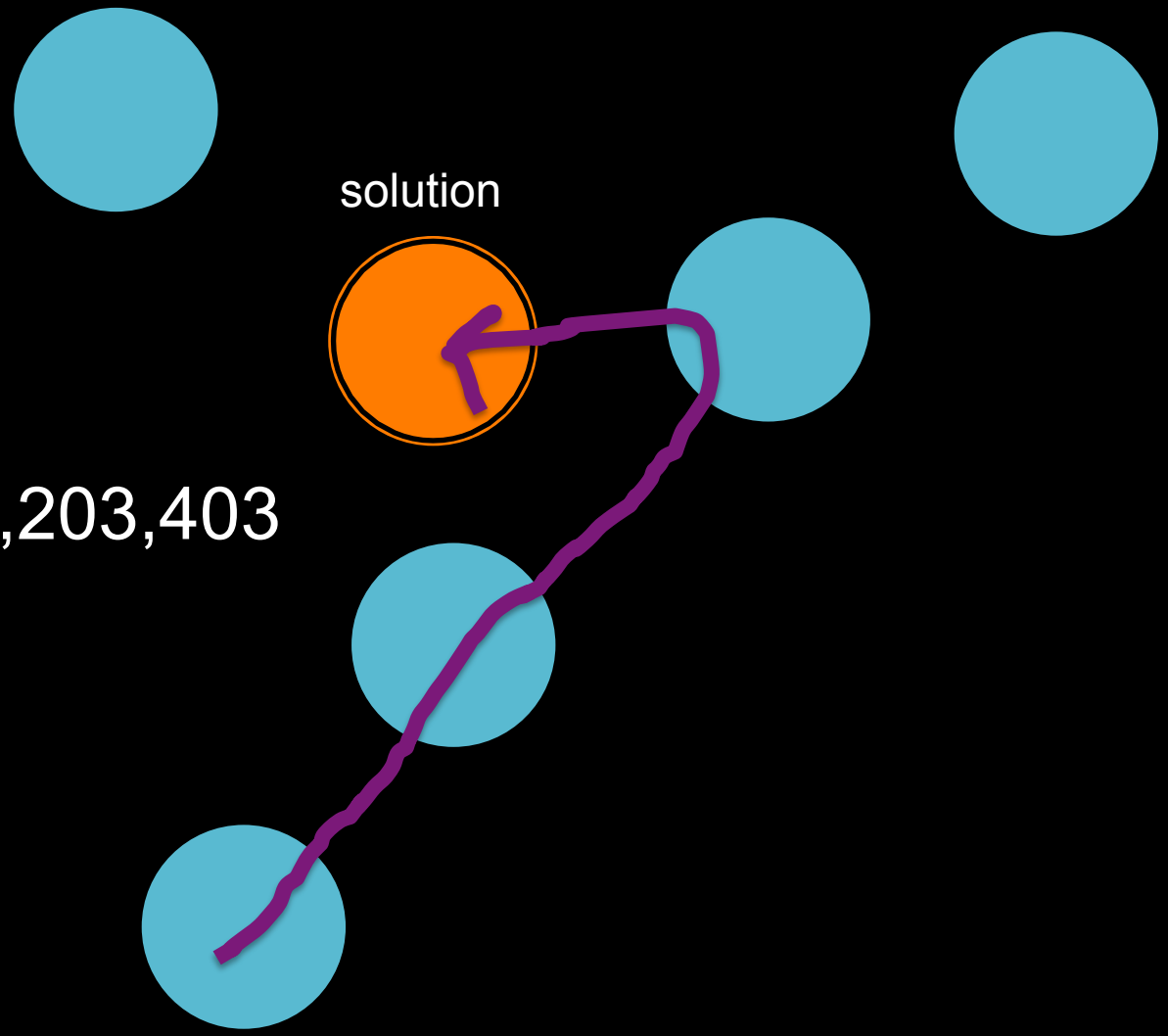
Example Student



Example Student

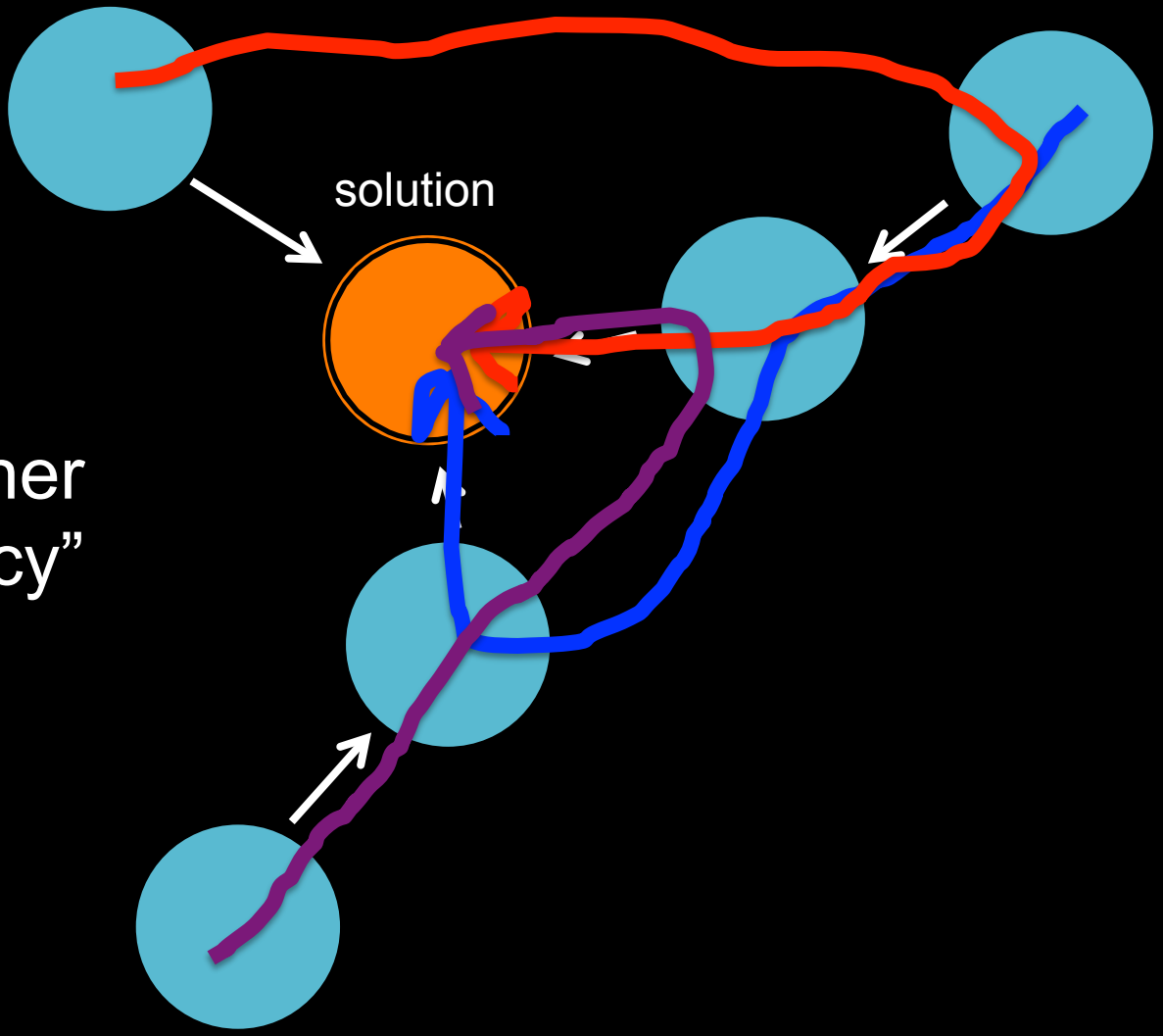


Example Student

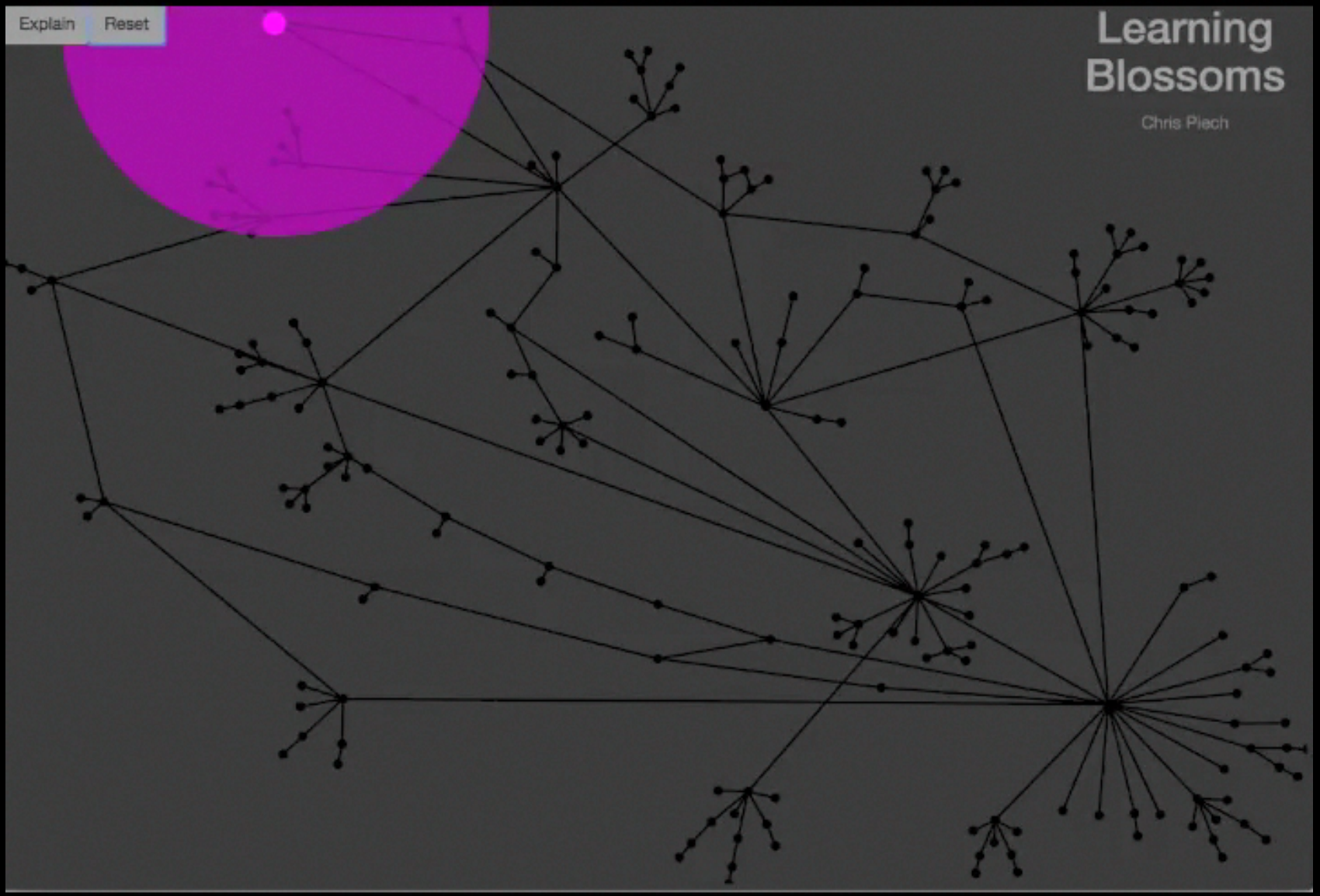


Student 1,203,403

Machine Learning Problem



Teacher
"Policy"


Explain
Reset
Learning
Blossoms
Chris Piech