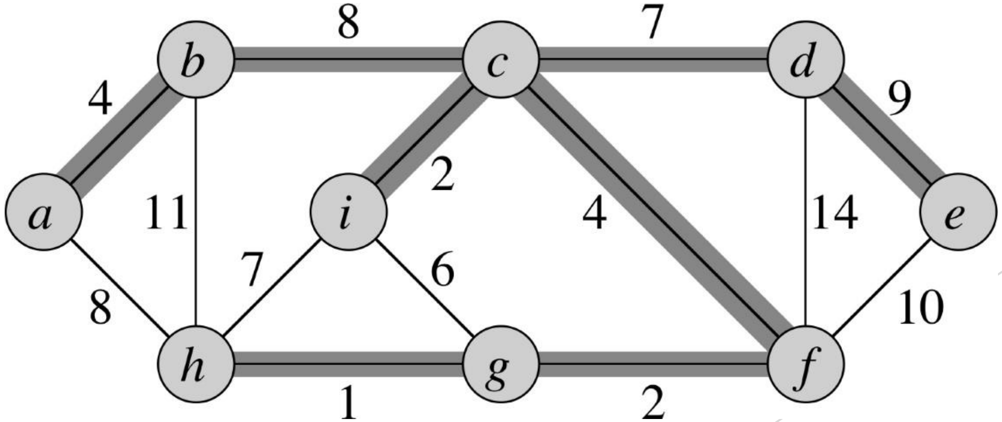
01418231 Data Structure

Minimum Spanning Tree

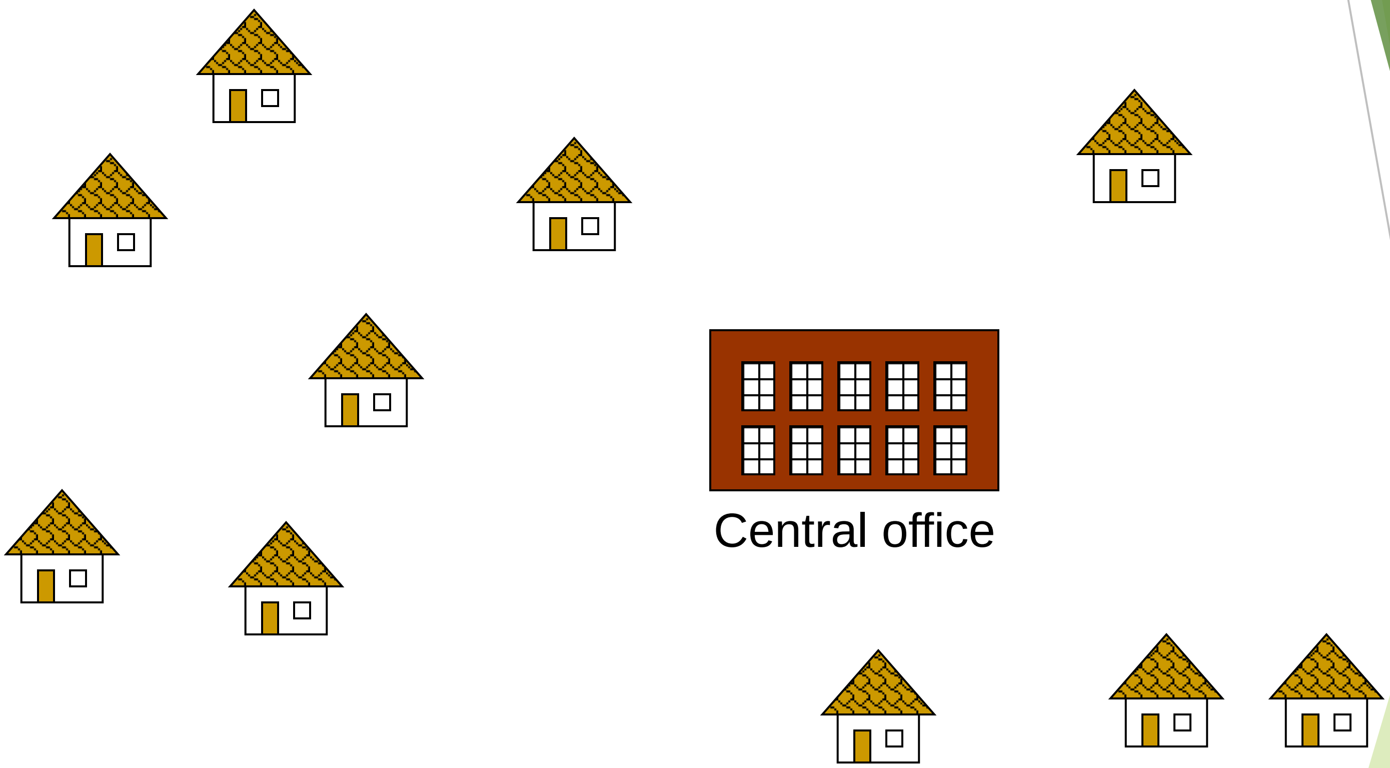


https://www.cs.indiana.edu/~achauhan/Teaching/B403/LectureNotes/09-mst.html

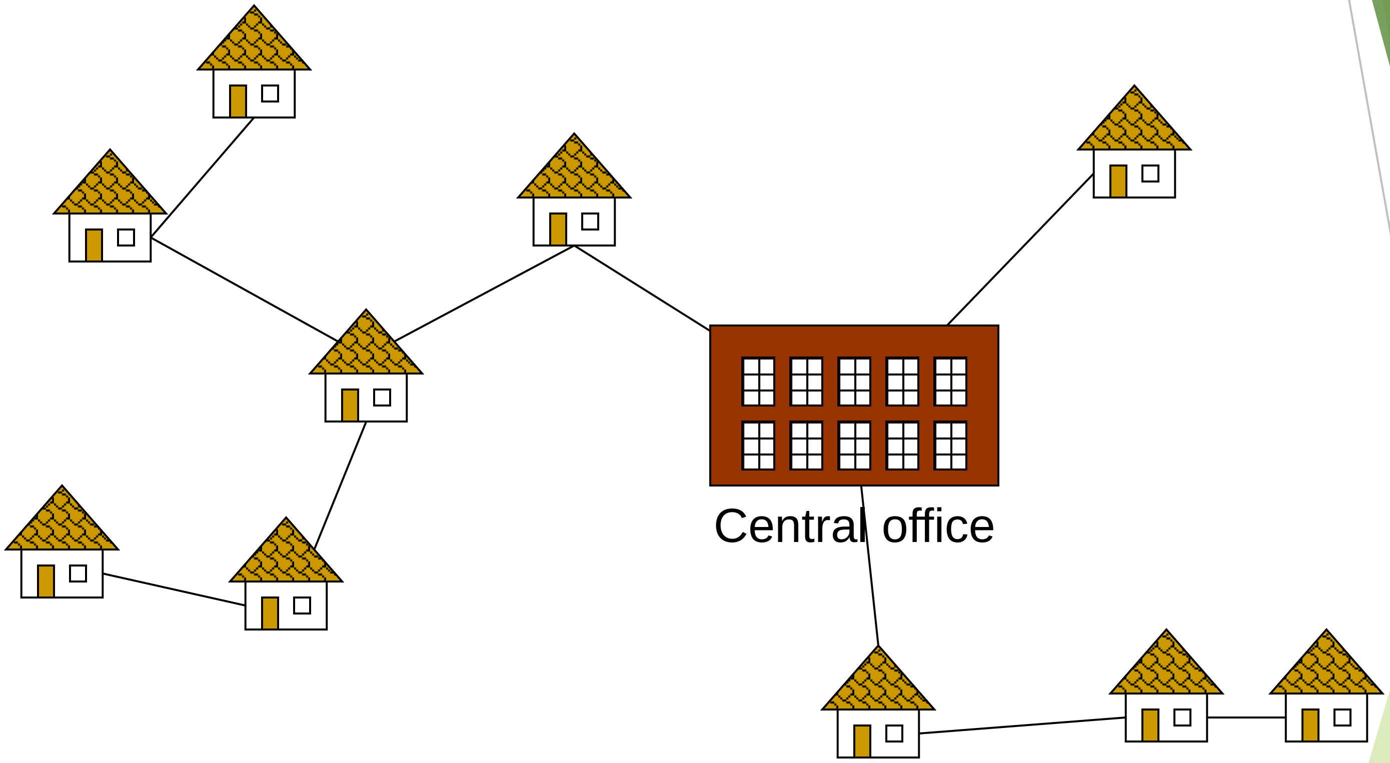
Agenda

- **What is Minimum Spanning Tree?**
- **Minimum Spanning Tree Algorithm**
  - Kruskal's algorithm
  - Prim algorithm
  - Dijkstra's algorithm

Problem: Laying Telephone Wire



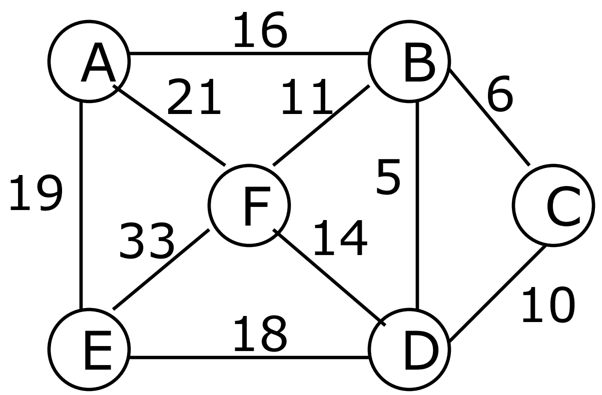
Wiring: Better Approach



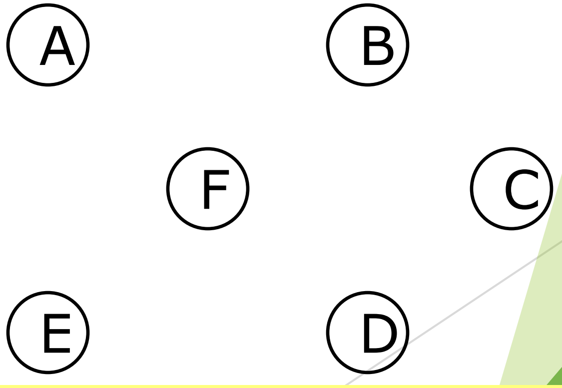
Minimize the total length of wire connecting the customers

Minimum Spanning Tree

- Definition
  - A minimum-weight tree in a weighted graph which contains all of the graph's vertices.
- Called
  - MST, shortest spanning tree, SST



Undirected graph

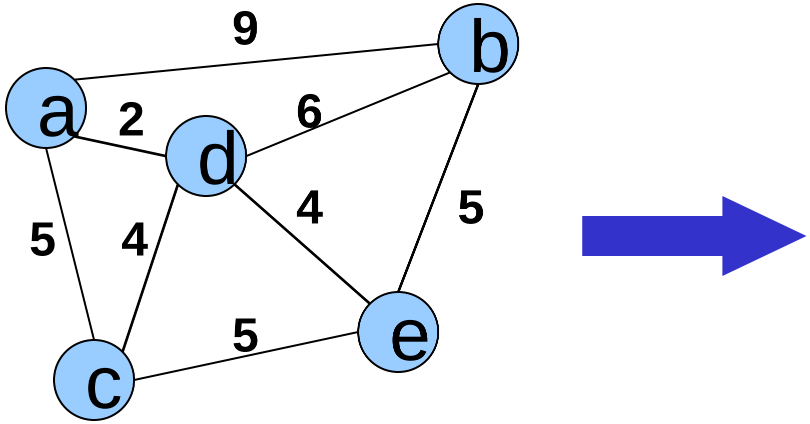


A minimum-cost spanning tree

Minimum Spanning Tree

- **A minimum spanning tree is a subgraph of an undirected weighted graph G, such that**
  - it is a tree (i.e., it is acyclic)
  - it covers all the vertices V
  - contains |V| - 1 edges
- the total cost associated with tree edges is the minimum among all possible spanning trees

How Can We Generate a MST?



Minimum Spanning Tree Algorithm

Kruskal's algorithm

Prim algorithm

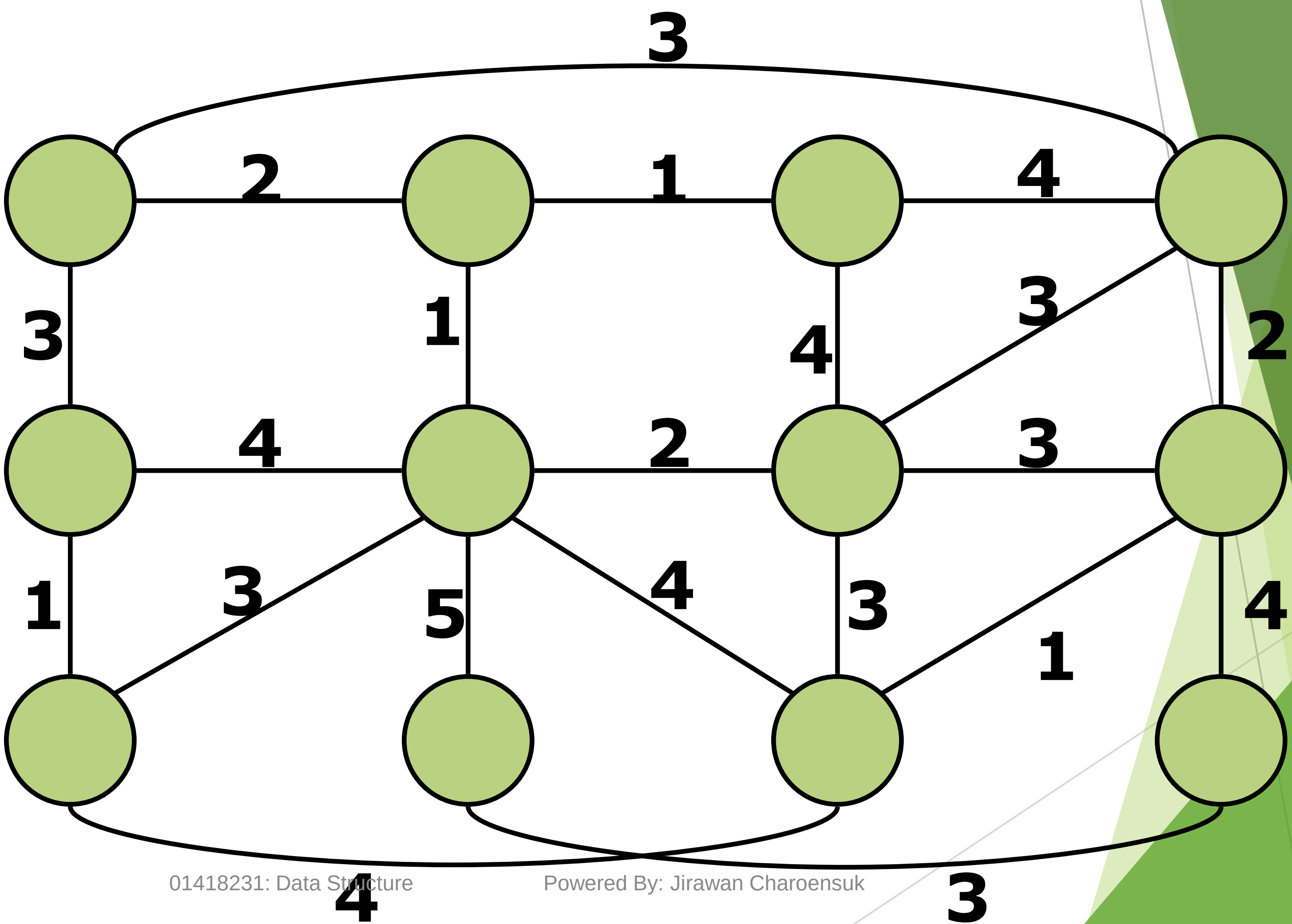
Dijkstra’s algorithm

Kruskal's algorithm

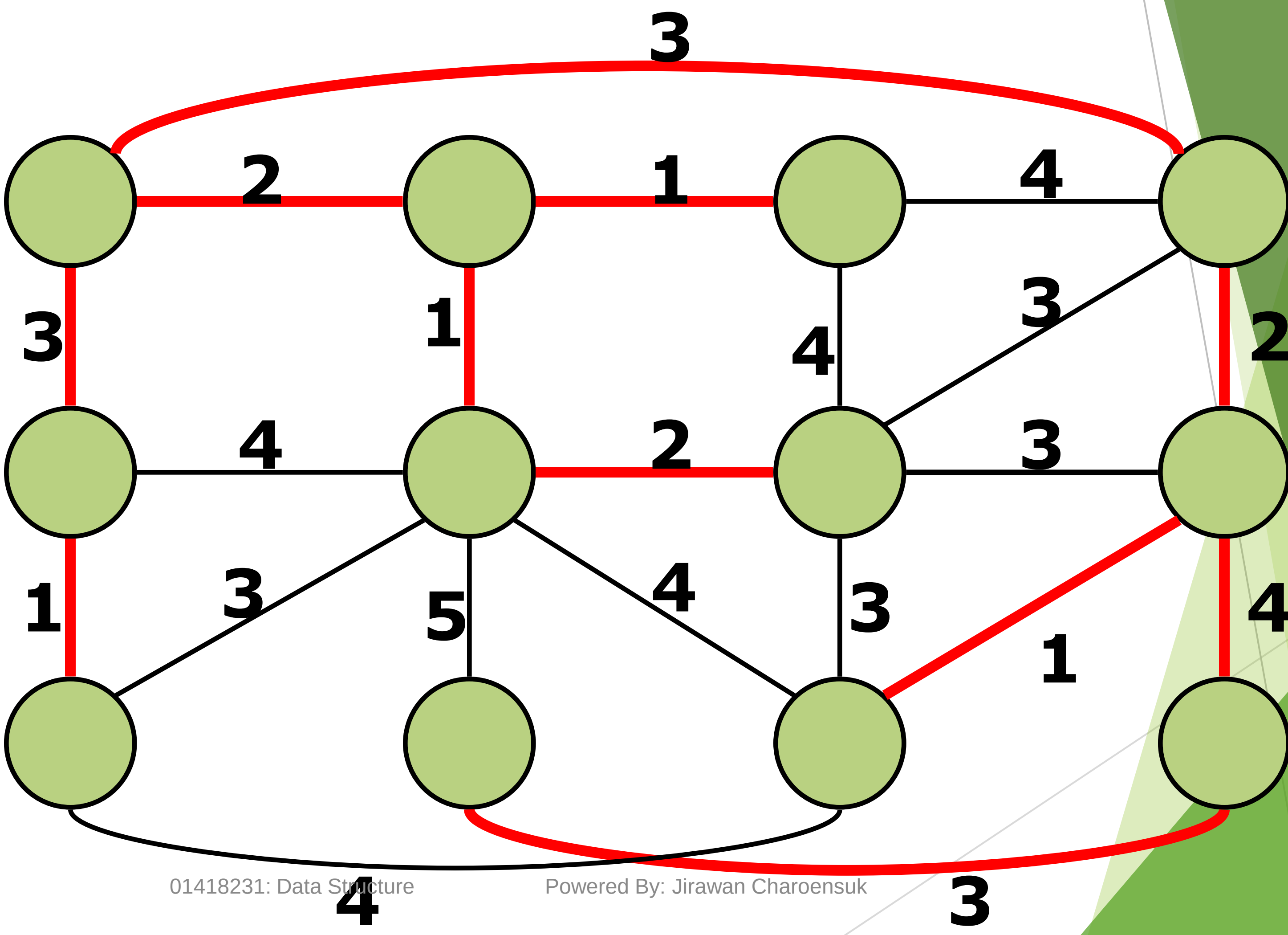
Kruskal's algorithm

- **Step 1**
  - Find the cheapest edge in the graph
    - (if there is more than one, pick one at random).
- **Step 2**
  - Find the cheapest unmarked edge in the graph that doesn't close a colored or red circuit
- **Step 3**
  - Repeat Step 2 until you reach out to every vertex of the graph
  - Or you have N-1 colored edges, where N is the number of Vertices

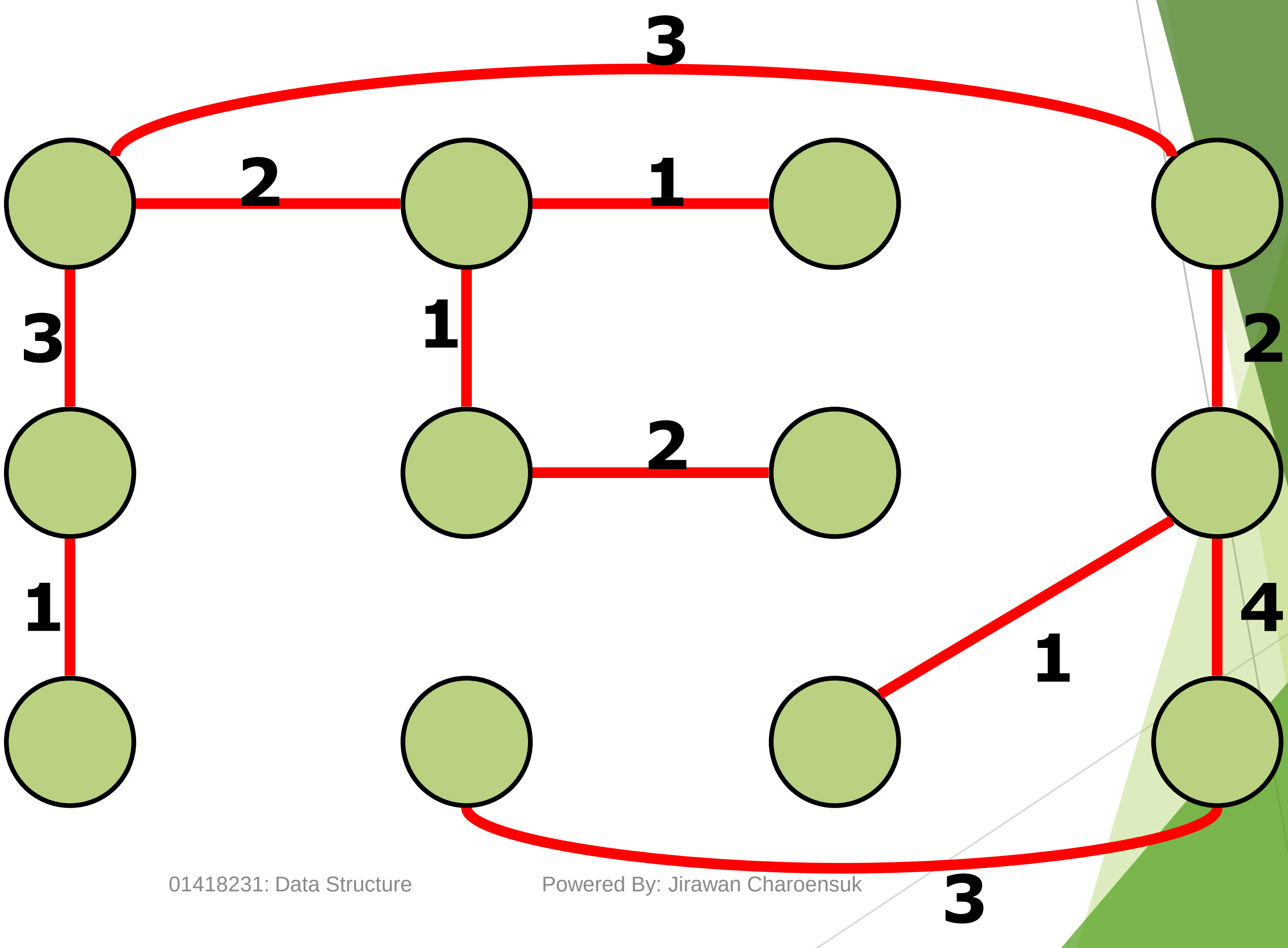
Kruskal's Algorithm



Kruskal’s Algorithm



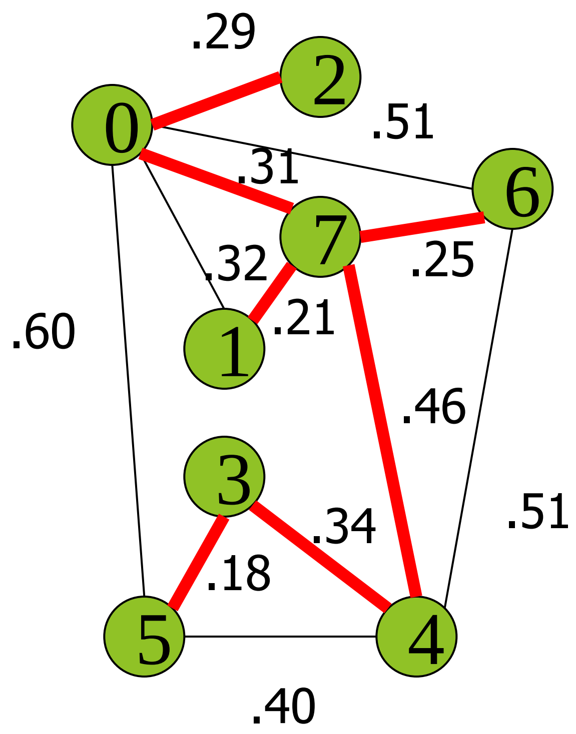
Kruskal’s Algorithm



Example 2- Kruskal's Algorithm

- **Rearrange weight (Min -> Max)**

3-5 = .18
1-7 = .21
6-7 = .25
0-2 = .29
0-7 = .31
0-1 = .32
4-3 = .34
4-5 = .40
4-7 = .46
0-6 = .51
4-6 = .51
0-5 = .60

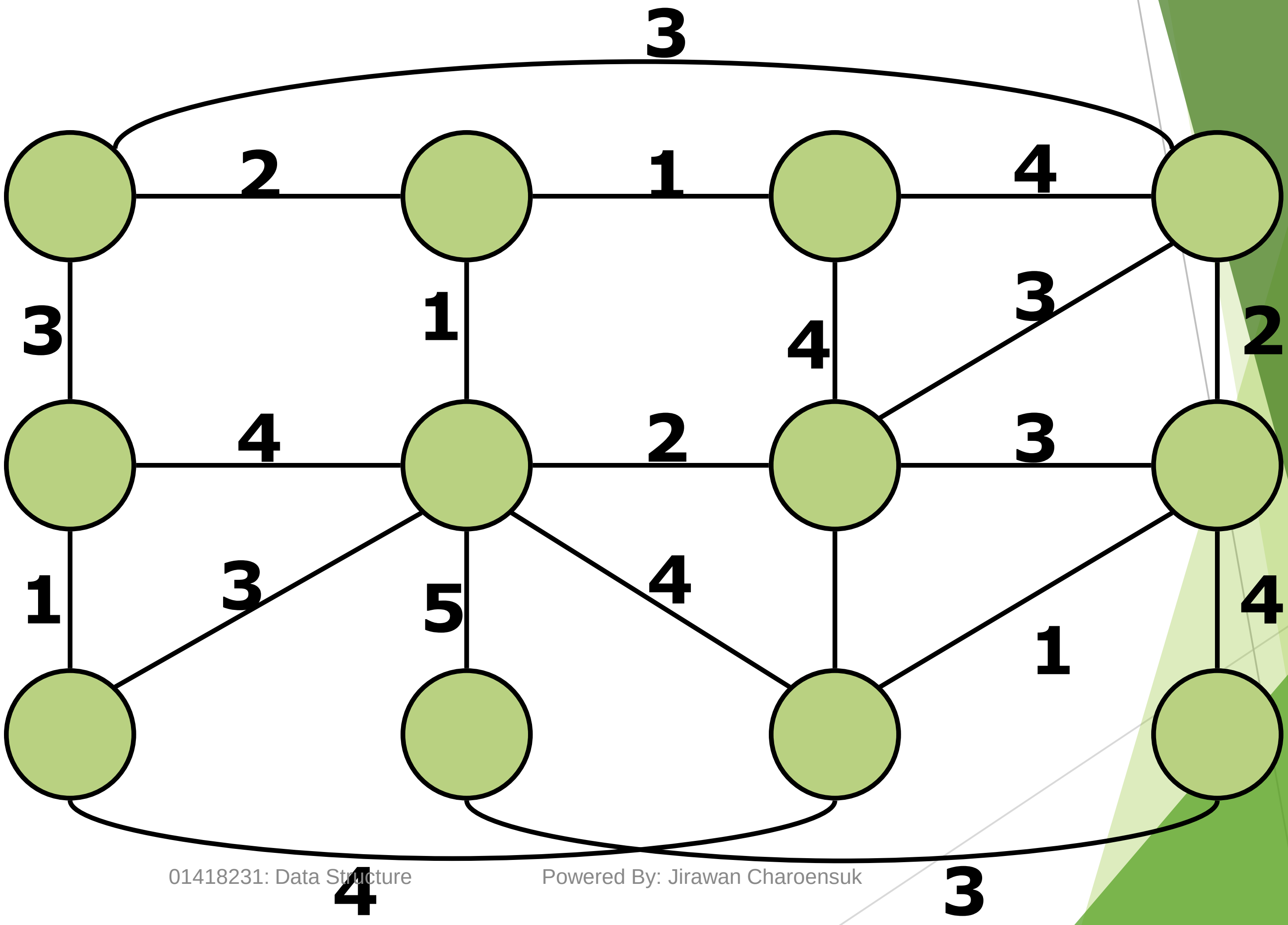


Prim algorithm

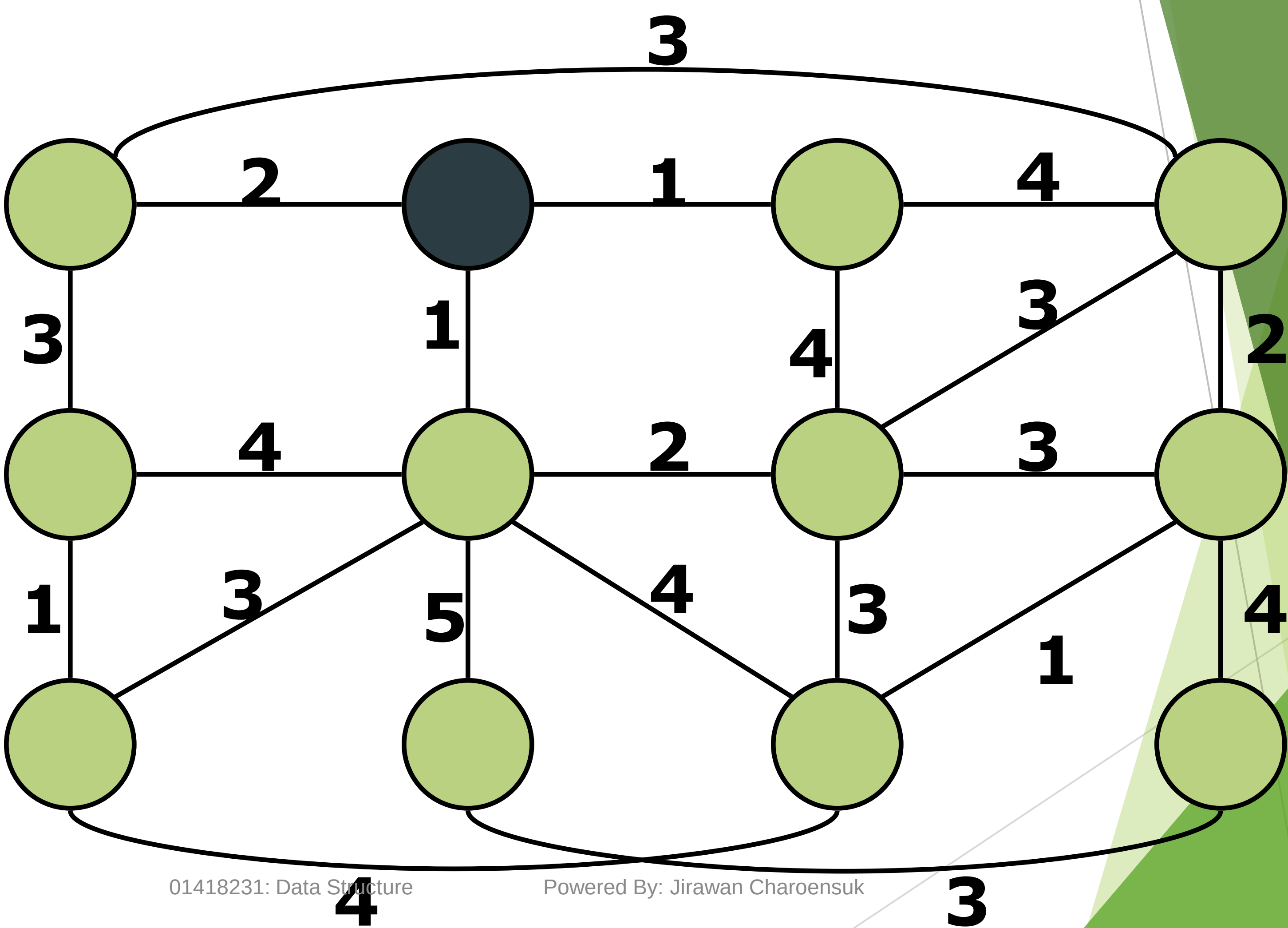
Prim's Algorithm

- **Step 0**
  - Pick any vertex as a starting vertex. (Call it S)
- **Step 1**
  - Find the nearest neighbor of S (call it P1)
  - Mark both P1 and the edge SP1
- **Step 2**
  - Find the nearest cheapest uncolored neighbor to the red subgraph (i.e., the closest vertex to any red vertex).
  - Mark it and the edge connecting the vertex
- **Step 3**
  - Repeat Step 2 until all vertices are marked red

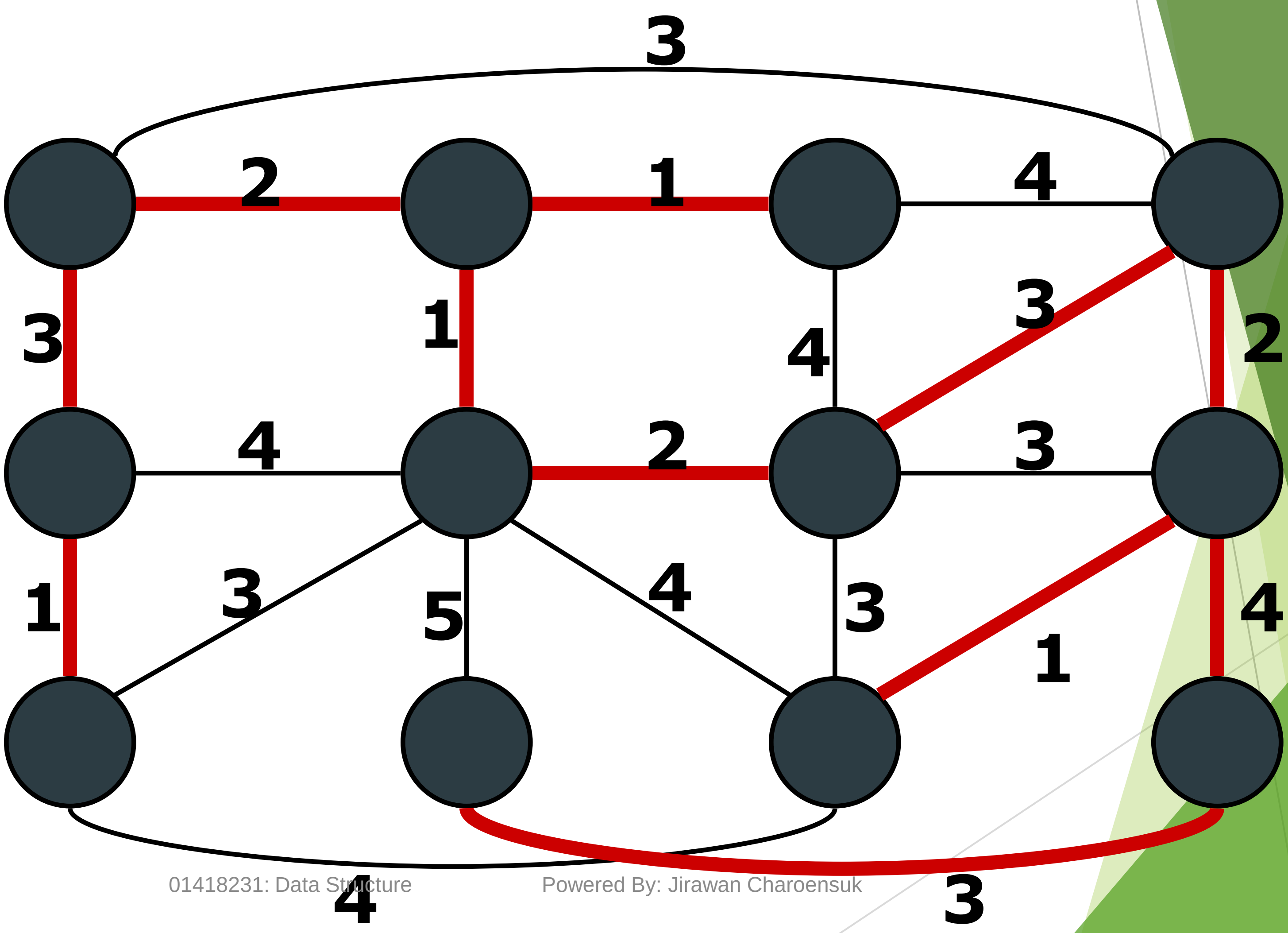
Prim's Algorithm



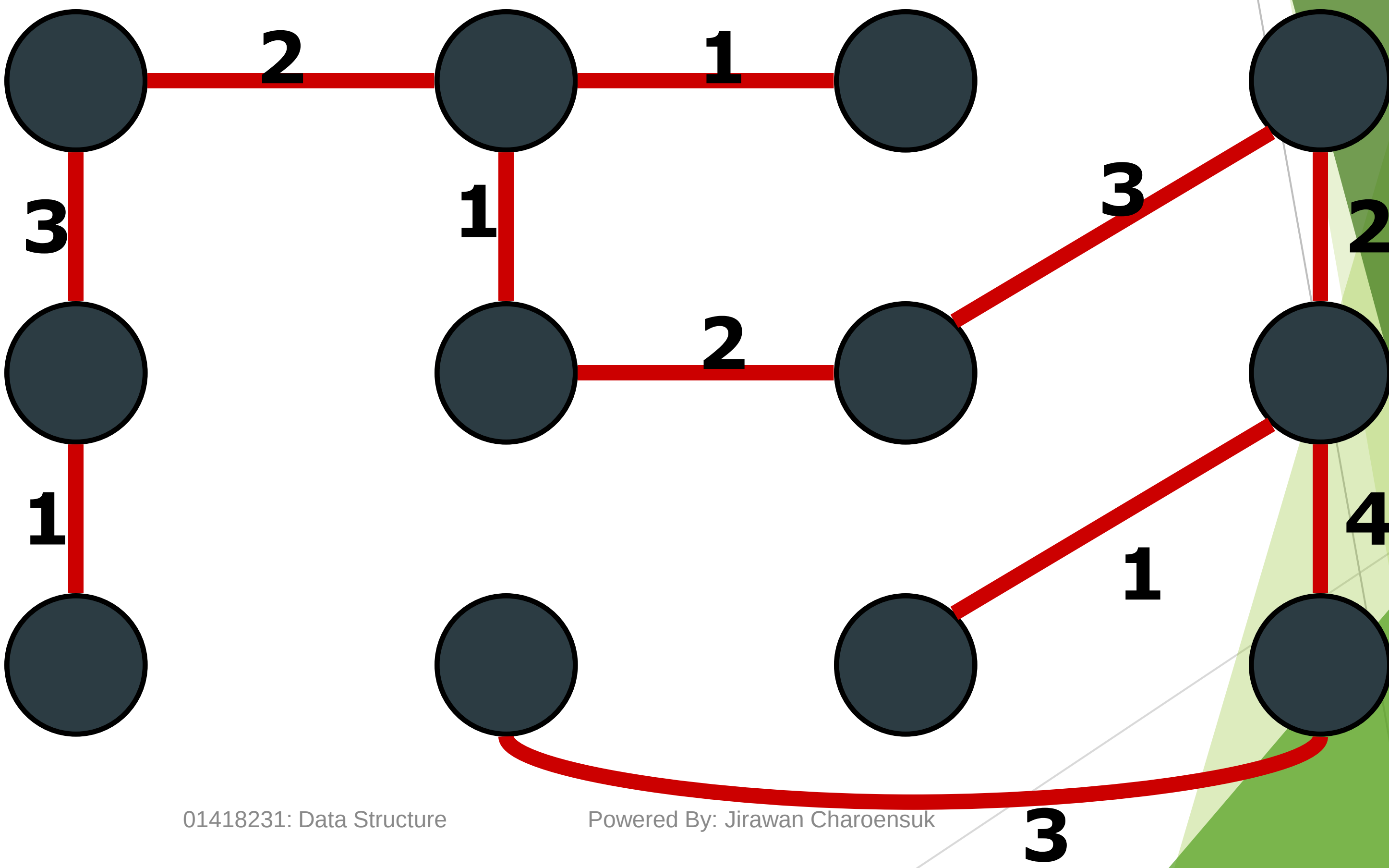
Prim’s Algorithm



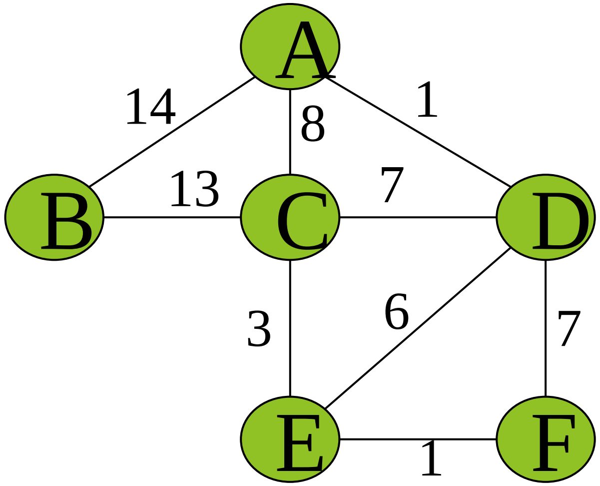
Prim's Algorithm

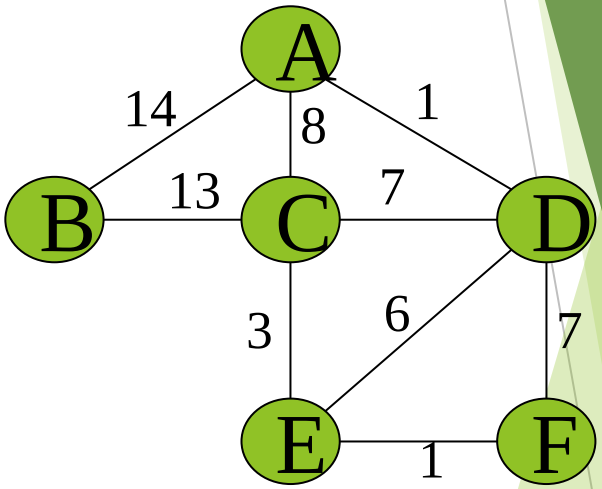


Prim's Algorithm



Example 2 - Prim’s algorithm





Dijkstra’s algorithm

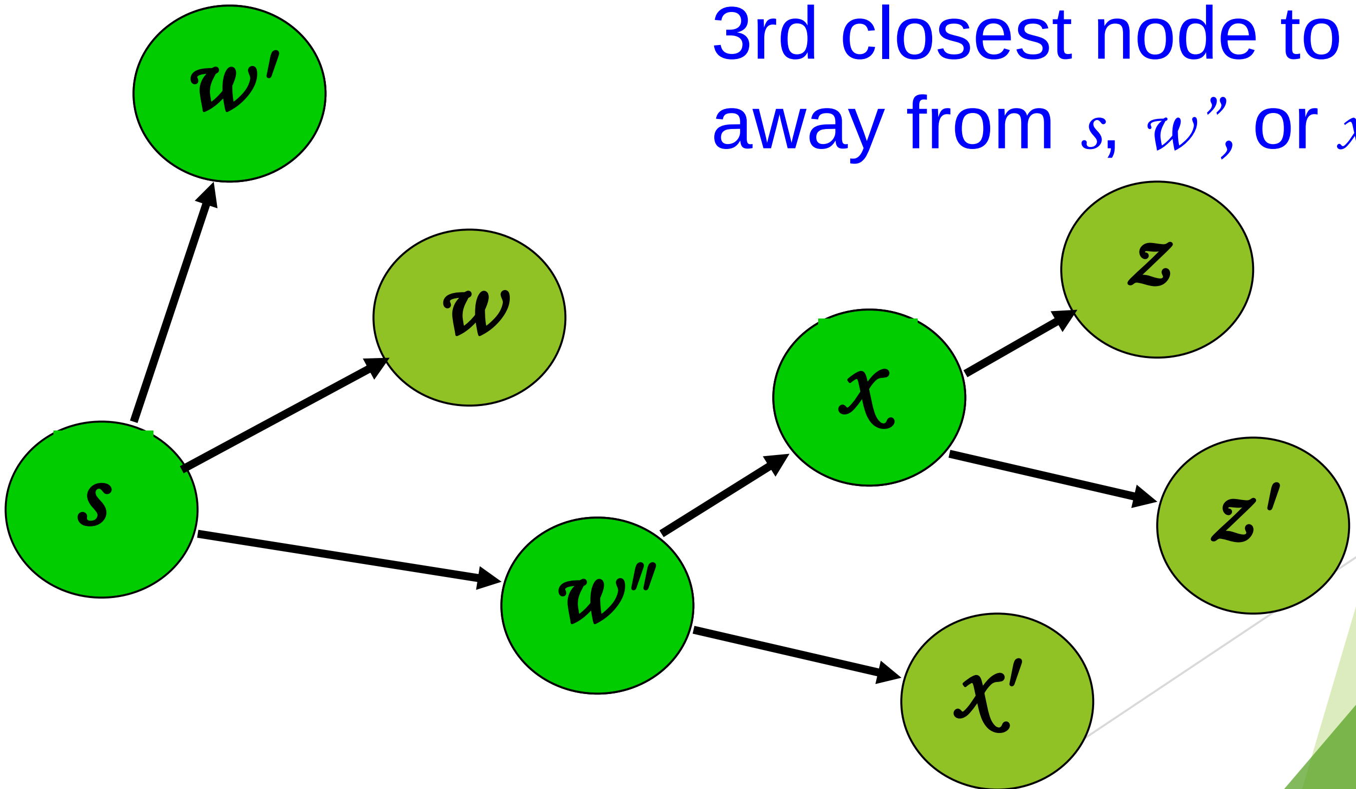
Dijkstra Algorithm: Finding shortest paths in order

Find shortest paths from source s to all other destinations

Closest node to s is 1 hop away

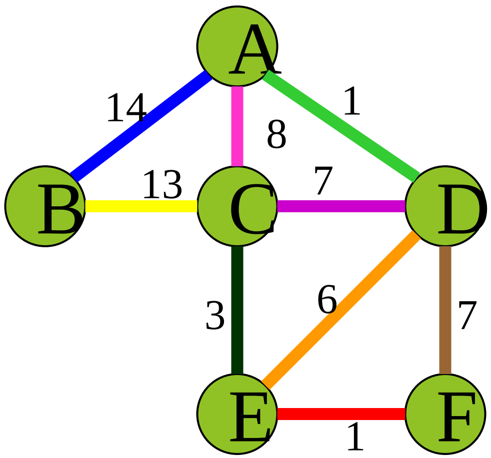
2nd closest node to s is 1 hop away from s or w''

3rd closest node to s is 1 hop away from s, w'', or x



Dijkstra’s algorithm

- Shortest path Start at 1

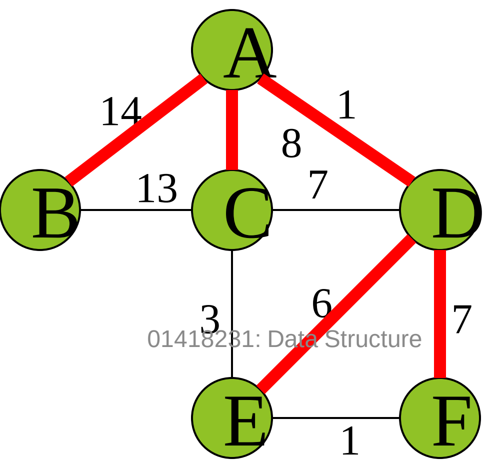


AB14 AC8 AD1

AD1	AC8	AB14	1

DC8 DE7 DF8

DE7	AC8	DC8	DF8	AB14	2



3

EF8 EC10

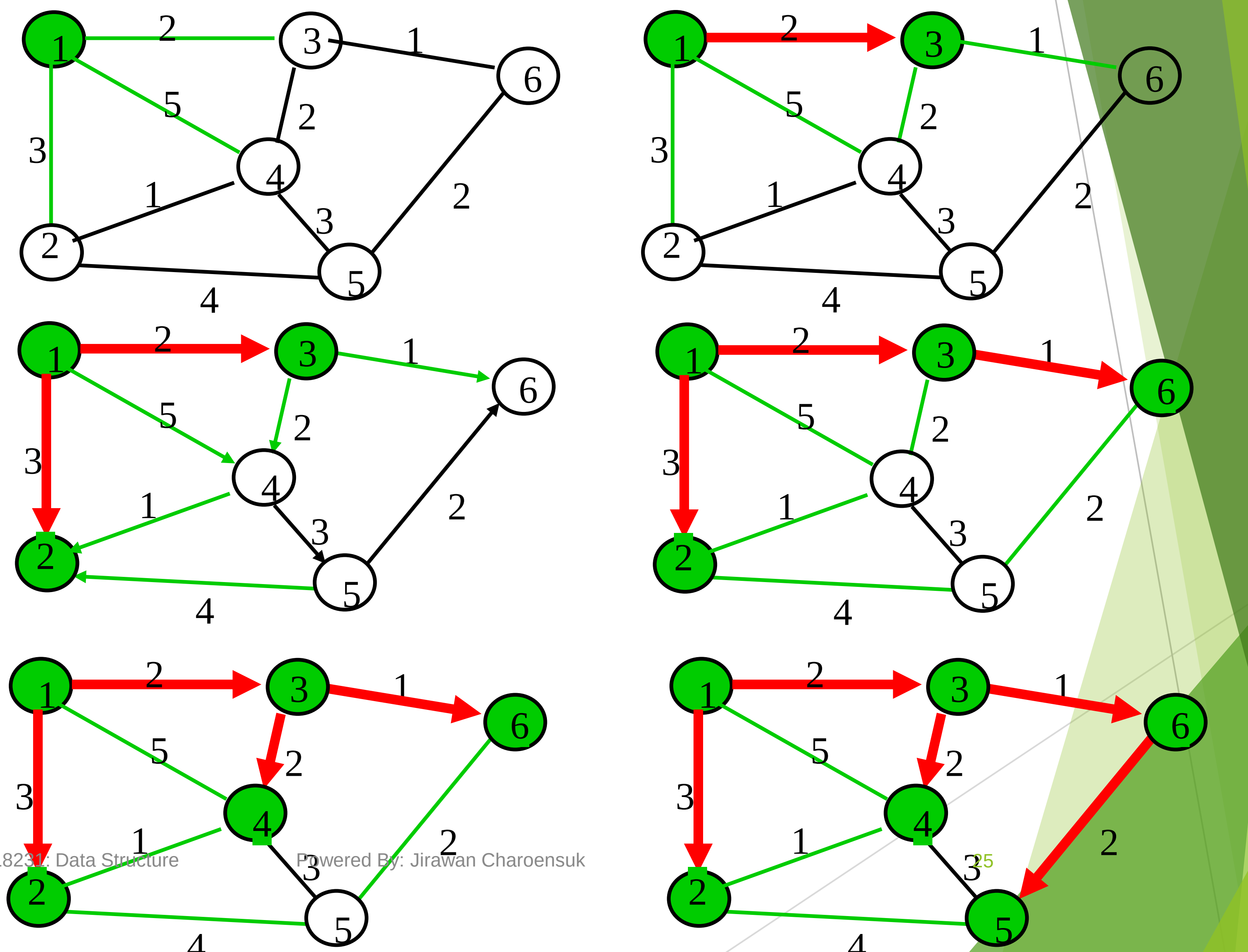
AC8	DC8	DF8	EF8	EC10	AB14

CB21

4

DC8	DF8	EF8	EC10	AB14	CB21

Example 2 : Dijkstra's Algorithm



website

- Kruskal
  - http://students.ceid.upatras.gr/~papagel/project/kruskal.htm
- Prim
  - http://students.ceid.upatras.gr/~papagel/project/prim.htm
- Dijkstra
  - http://students.ceid.upatras.gr/~papagel/project/kef5_7_1.htm

Question