Data Structures and Algorithms
(CS210A)

Lecture 36

- A new algorithm design paradigm: Greedy strategy

part III

Continuing Problem from last class

Minimum spanning tree

Problem Description

Input: an undirected graph G=(V,E)

Aim: compute a **spanning tree** (V, E')

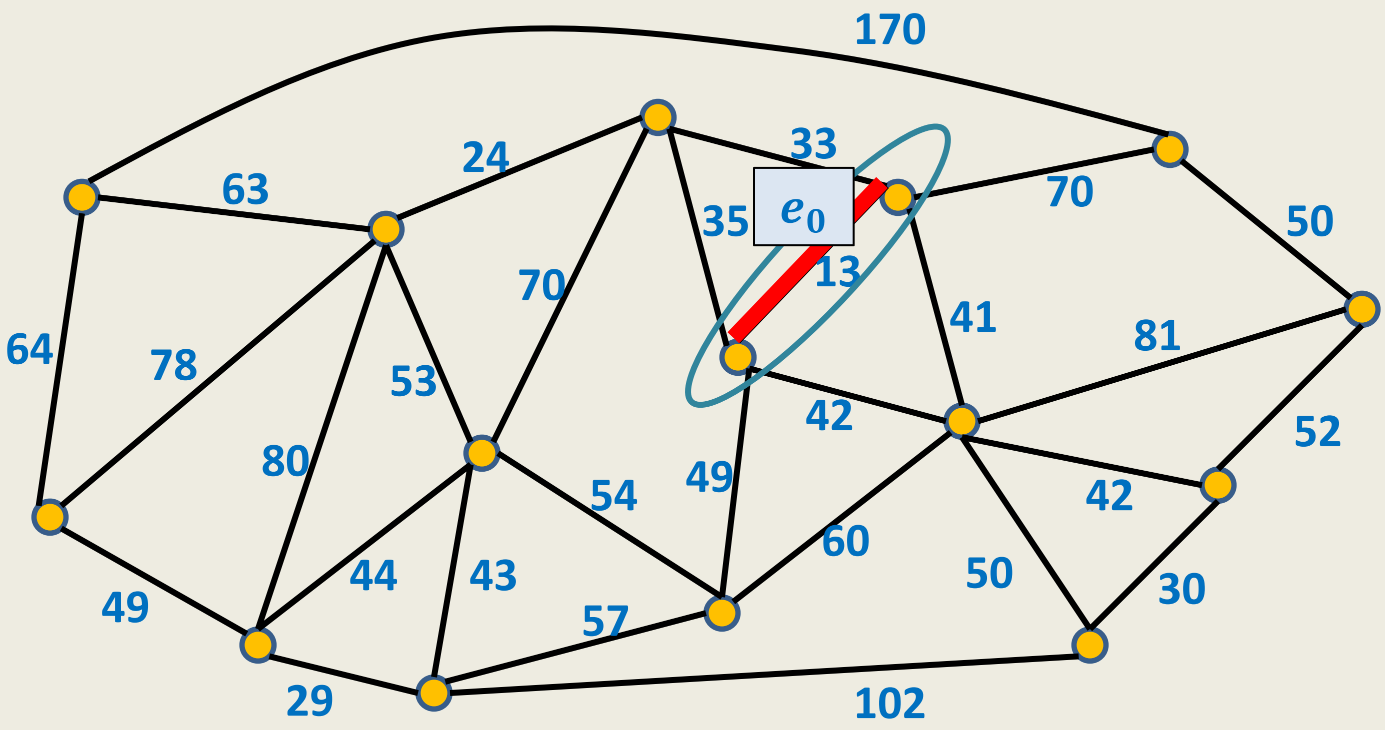
Lemma (proved in last class):

If $e_0 \in E$ is the edge of **least weight** in G,

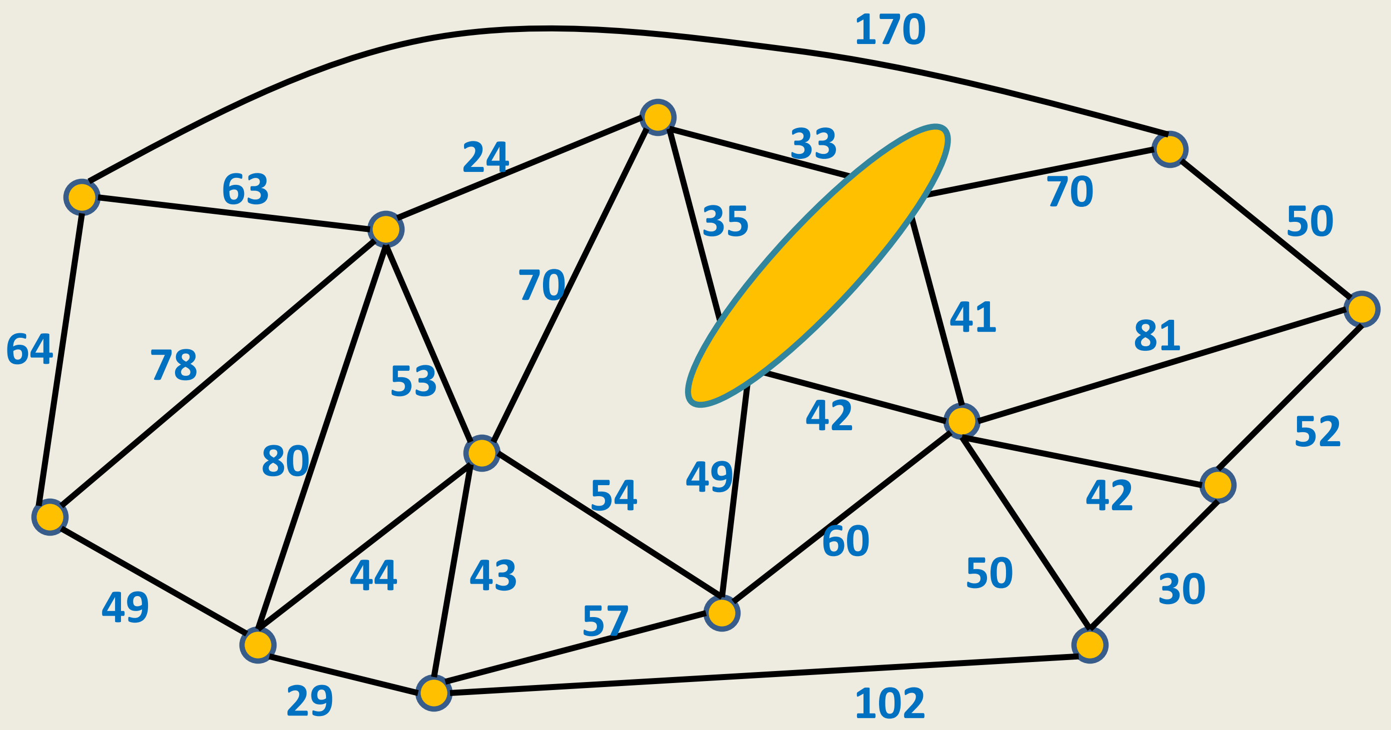
then there is a **MST** T containing e_0.

How to use this **Lemma** to design an algorithm for **MST** ?

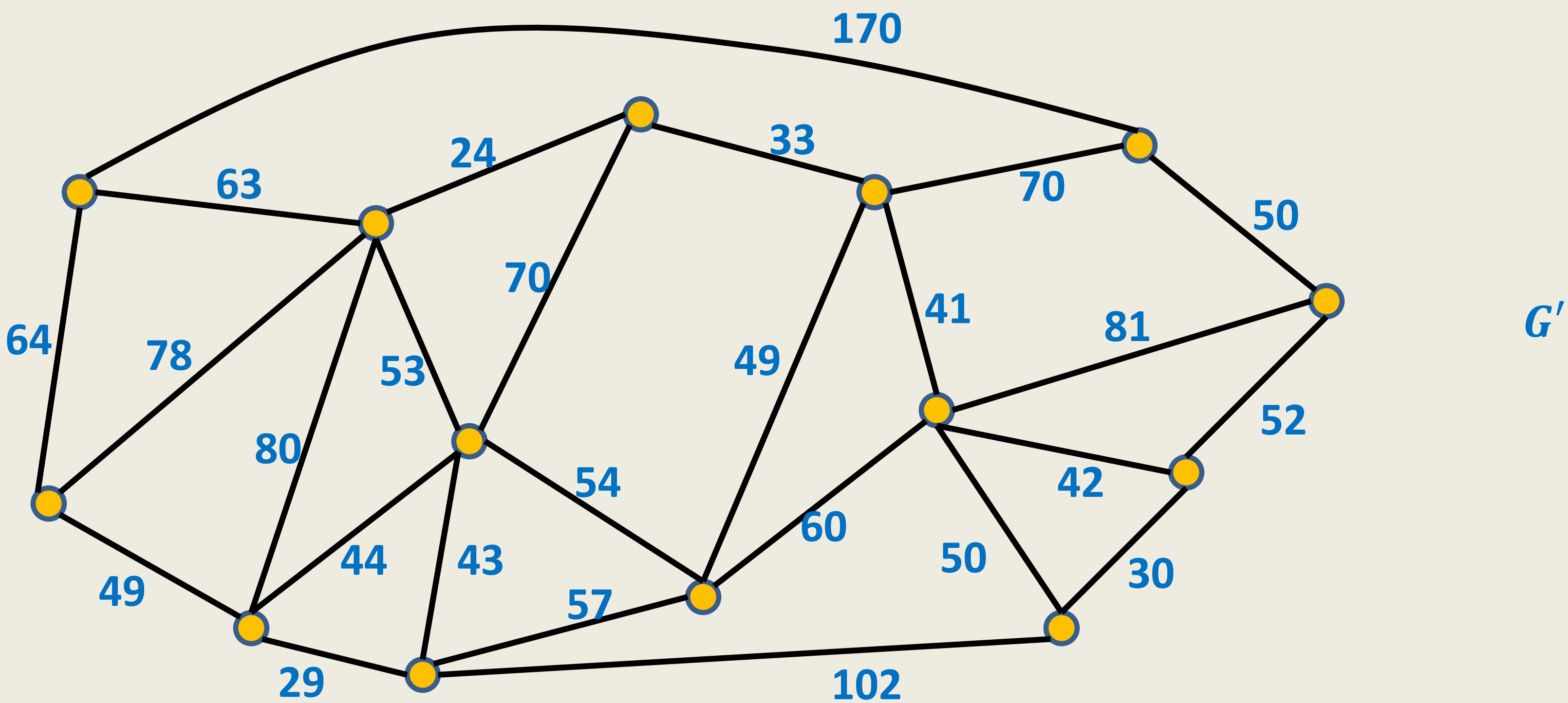
Minimum Spanning Tree (MST)



Minimum Spanning Tree (MST)



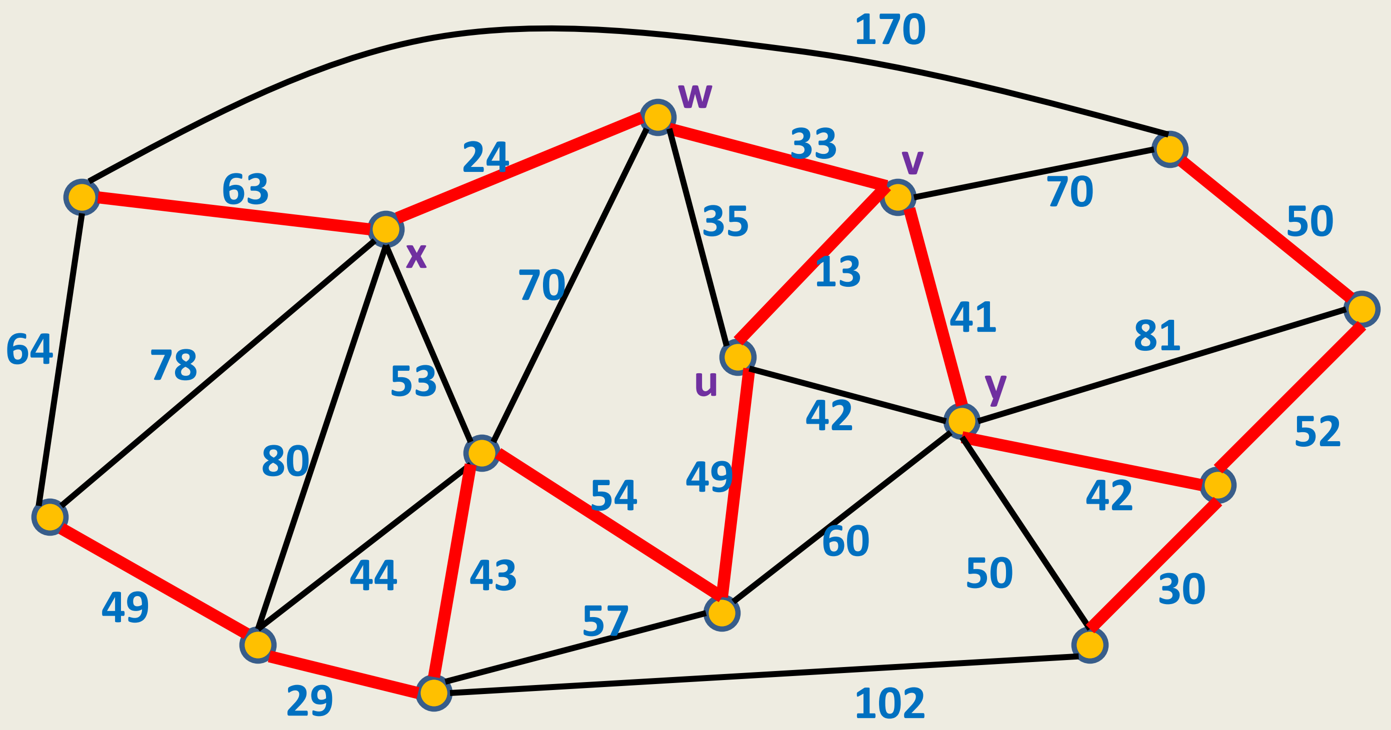
Minimum Spanning Tree (MST)



Theorem:

$\mathbf{w}$(MST($\boldsymbol{G}$)) =

Minimum Spanning Tree (MST)



A useful lesson for design of a graph algorithm

If you have a complicated algorithm for a graph problem, …

- search for **some graph theoretic property**

to design **simpler** and **more efficient** algorithm

Two graph theoretic properties of MST

- **Cut property**
- **Cycle property**

Every algorithm till date is **based on** one of **these properties**!

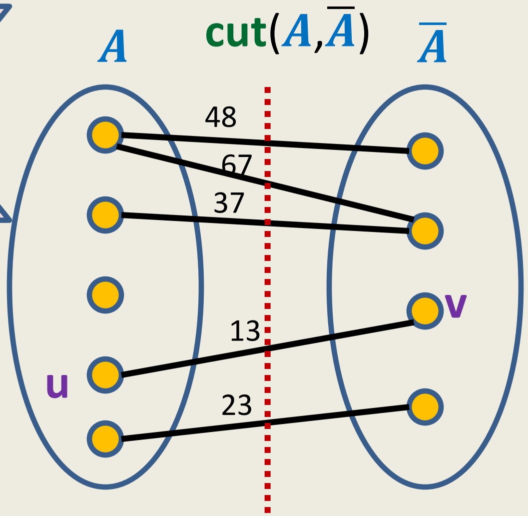
Cut Property

Cut Property

Definition: For any subset $A \subseteq V$

cut(A,$\overline{A}$) = { (u,v) ∊ E |

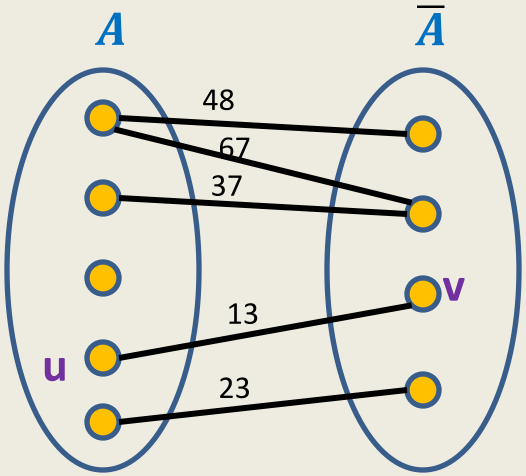
Pursuing **greedy strategy** to minimize weight of MST, what can we say about the edges of cut(A, $\overline{A}$) ?



Cut-property:

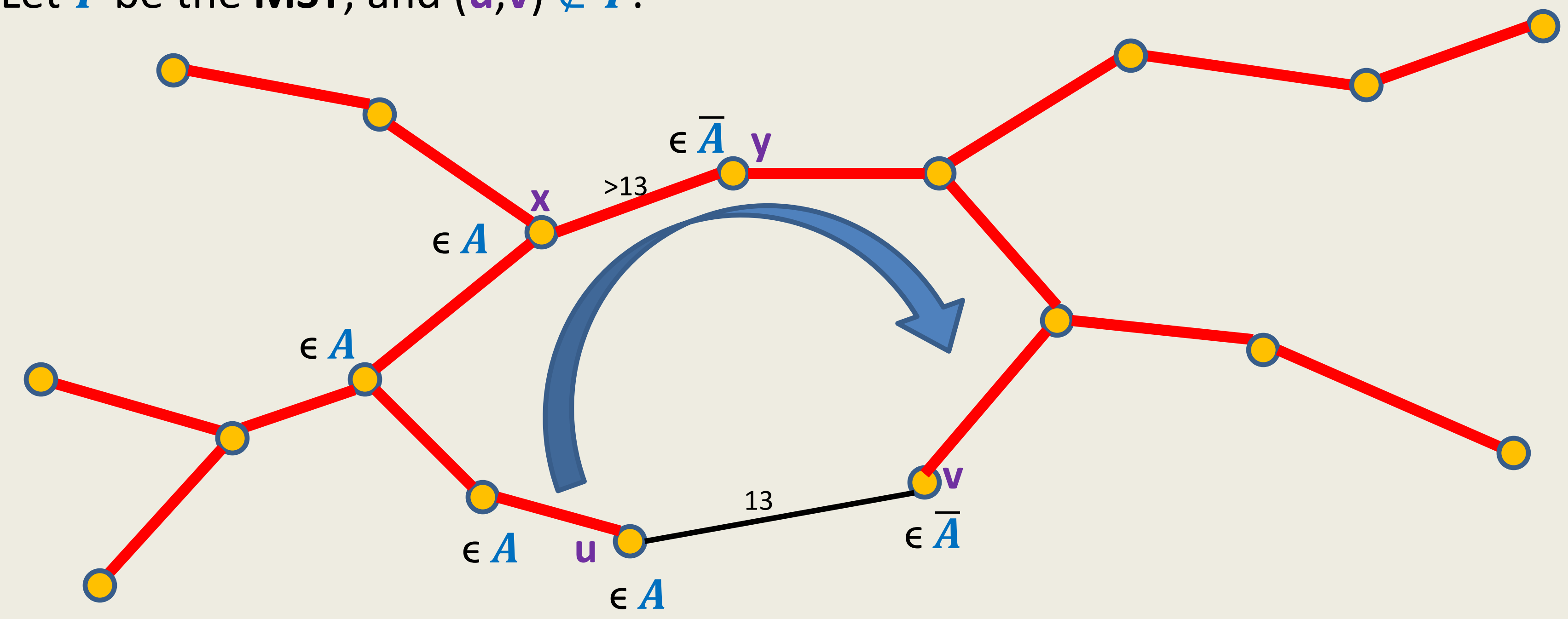
The **least weight edge** of a cut(A,$\overline{A}$) must be in **MST.**

Proof of cut-property



Proof of cut-property

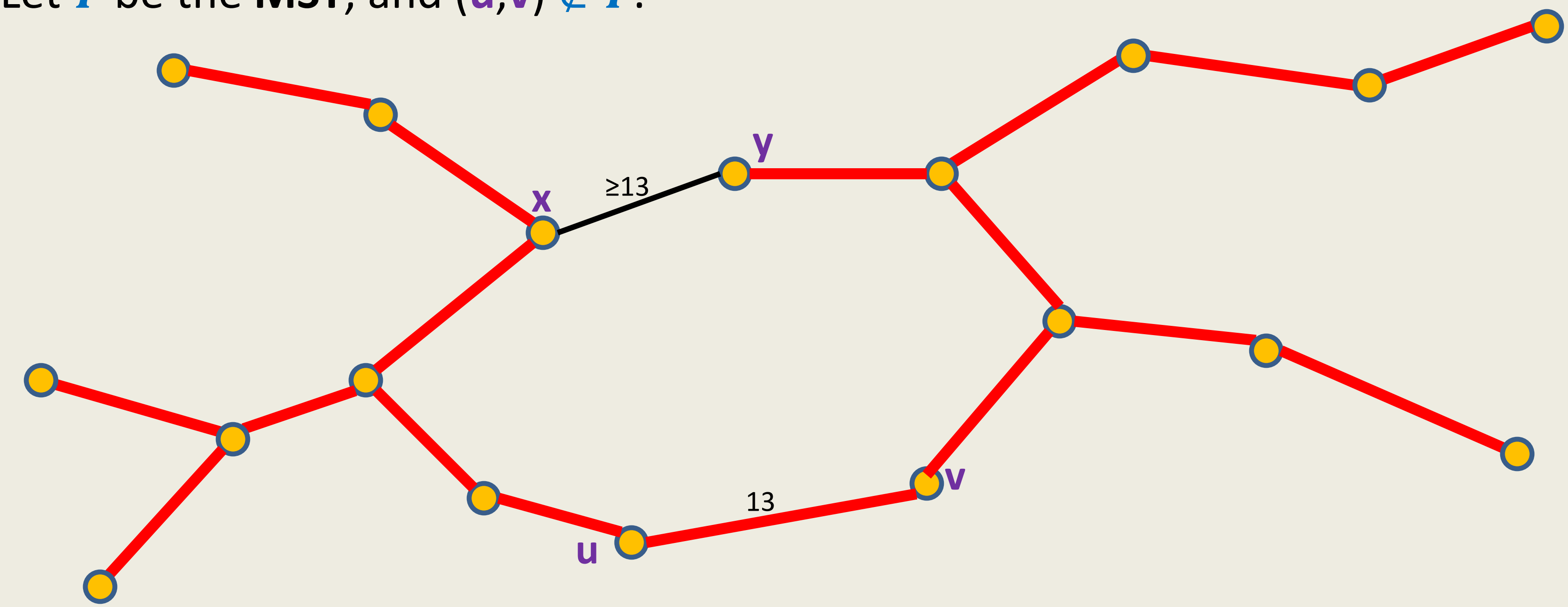
Let T be the **MST**, and (u,v) $\notin T$.



Question: What happens if we **remove** (x,y) from T, and **add** (u,v) to T.

Proof of cut-property

Let T be the **MST**, and (u,v) $\notin T$.



Question: What happens if we **remove** (x,y) from T, and **add** (u,v) to T.

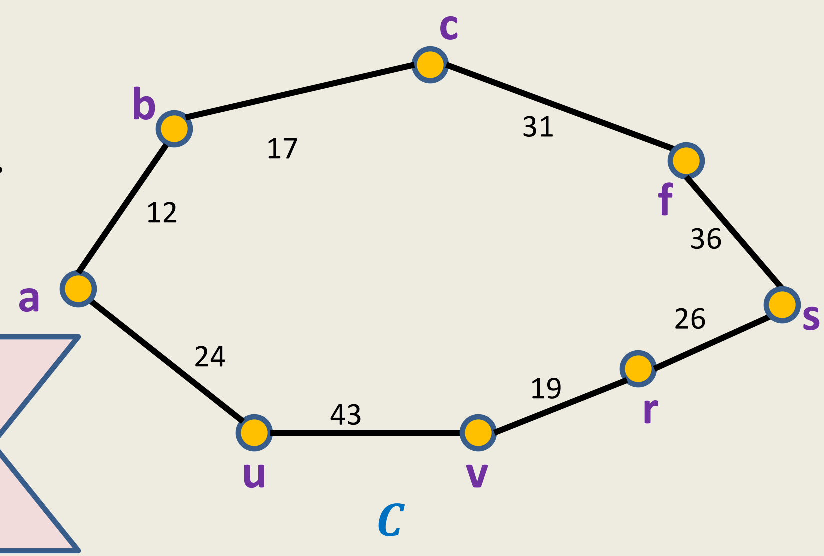
We get a spanning tree T' with weight < **weight**(T)

A contradiction !

Cycle Property

Cycle Property

Let C be any cycle in G.

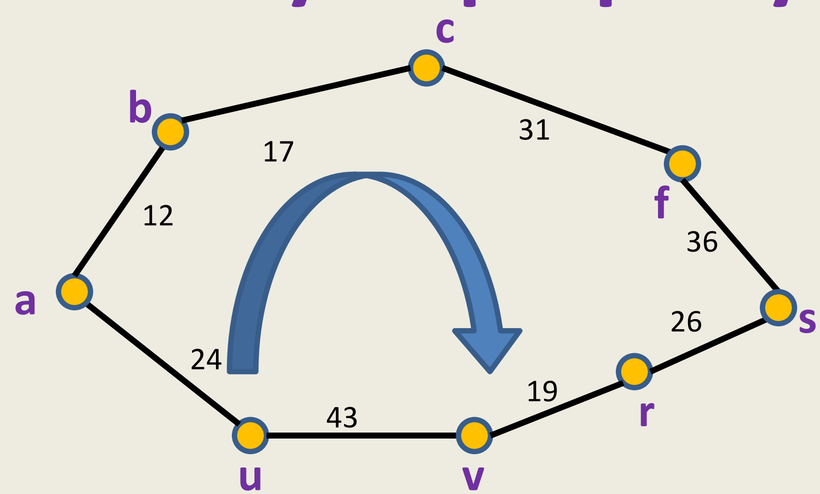


Pursuing **greedy strategy** to minimize weight of MST, what can we say about the edges of cycle C ?

Cycle-property:

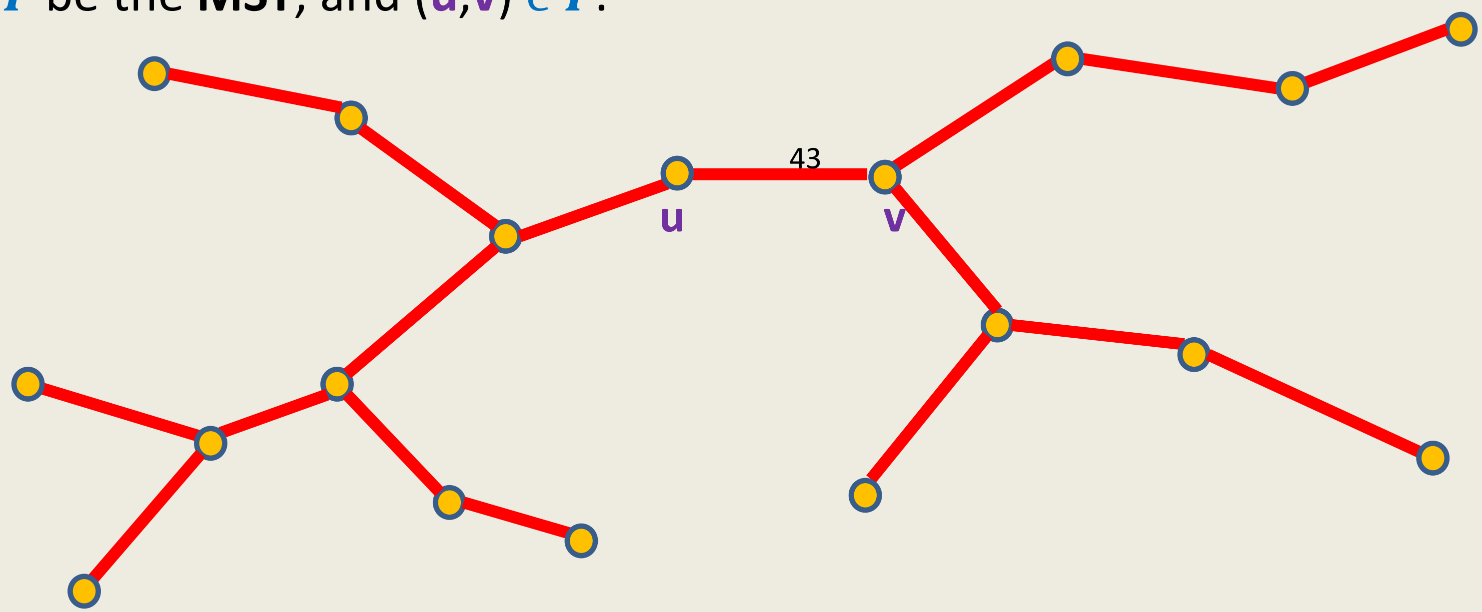
Maximum weight edge of any cycle C **can not** be present in **MST**.

Proof of Cycle property



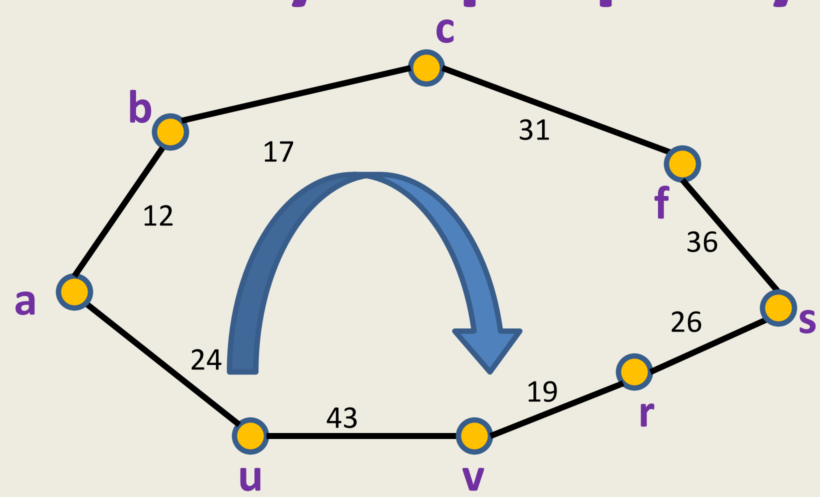
C

Let T be the **MST**, and (**u**,**v**) $\in T$.

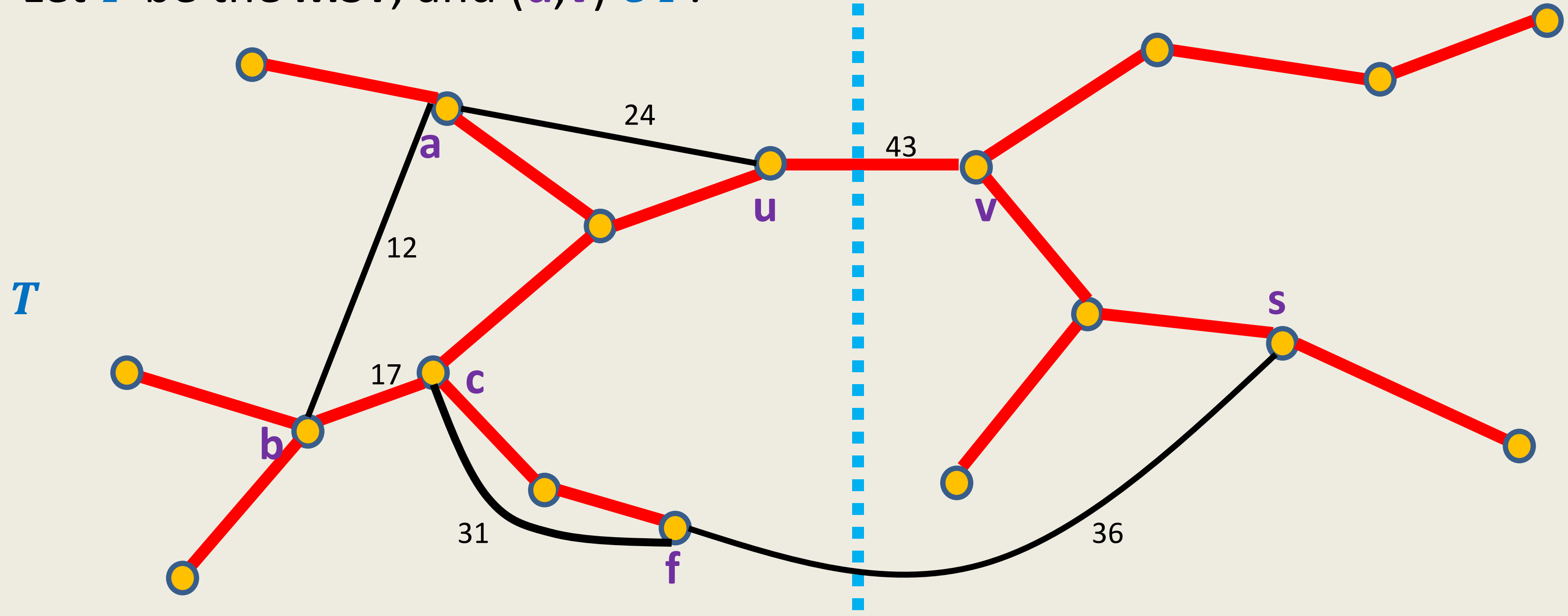


T

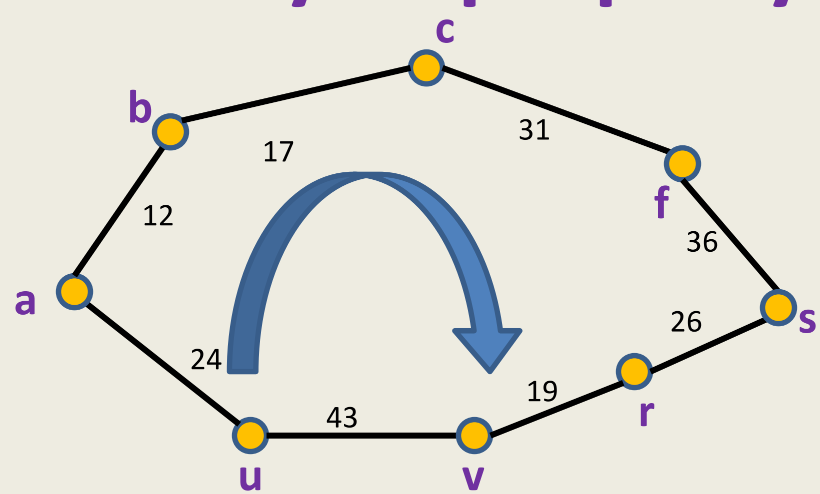
Proof of Cycle property



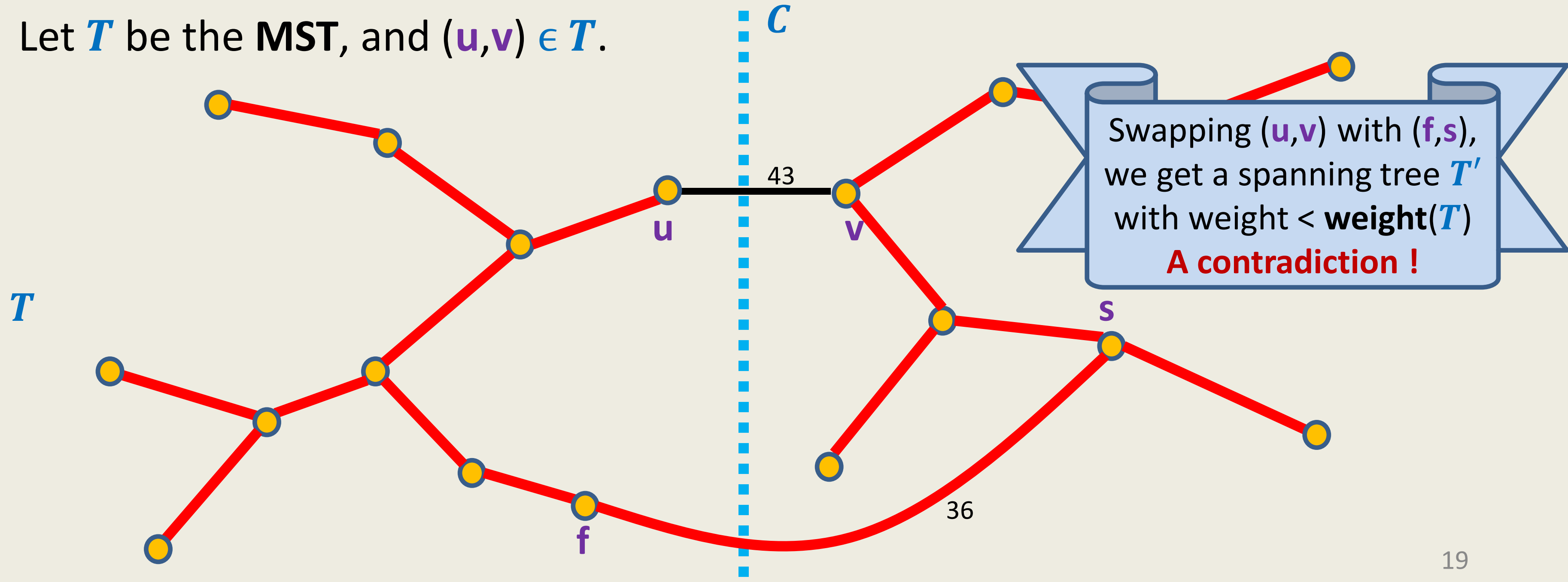
Let $\boldsymbol{T}$ be the **MST**, and (u,v) $\in \boldsymbol{T}$.



Proof of Cycle property

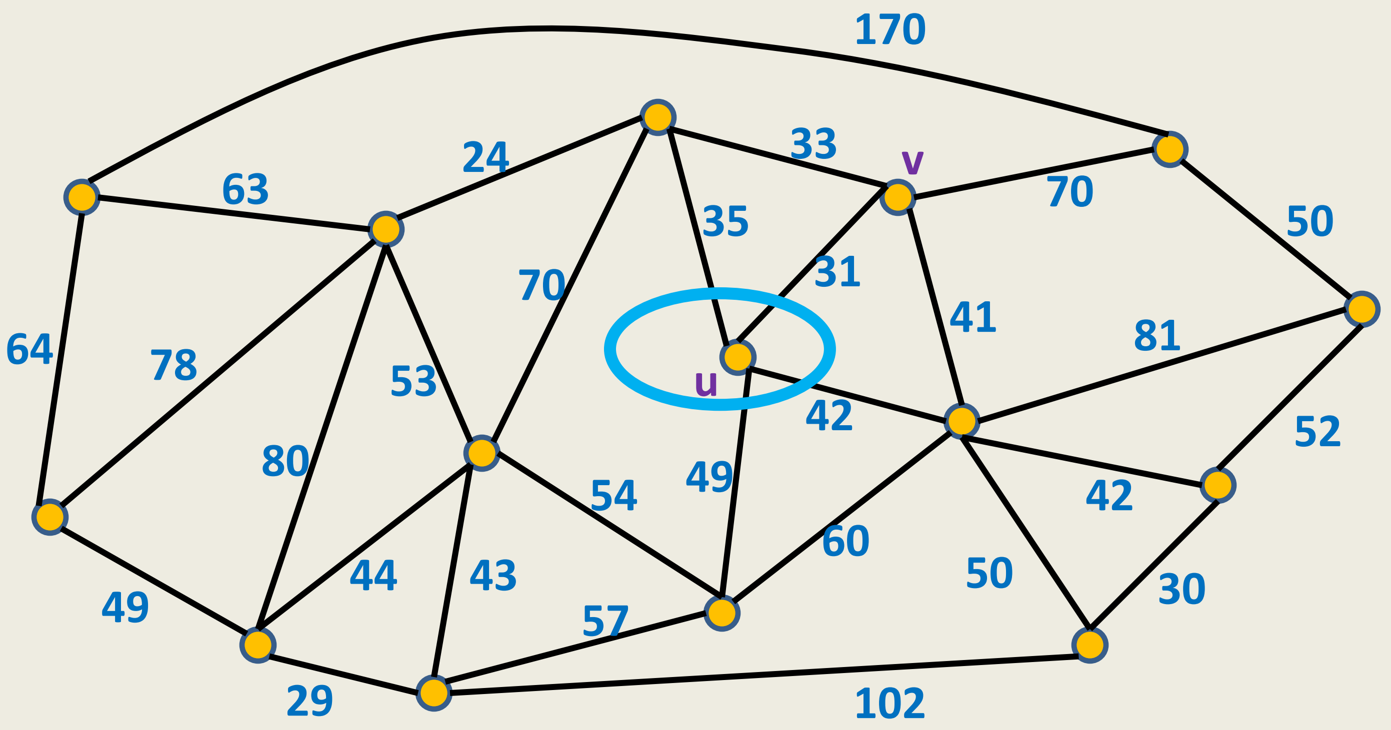


Let T be the **MST**, and (u,v) $\in T$.

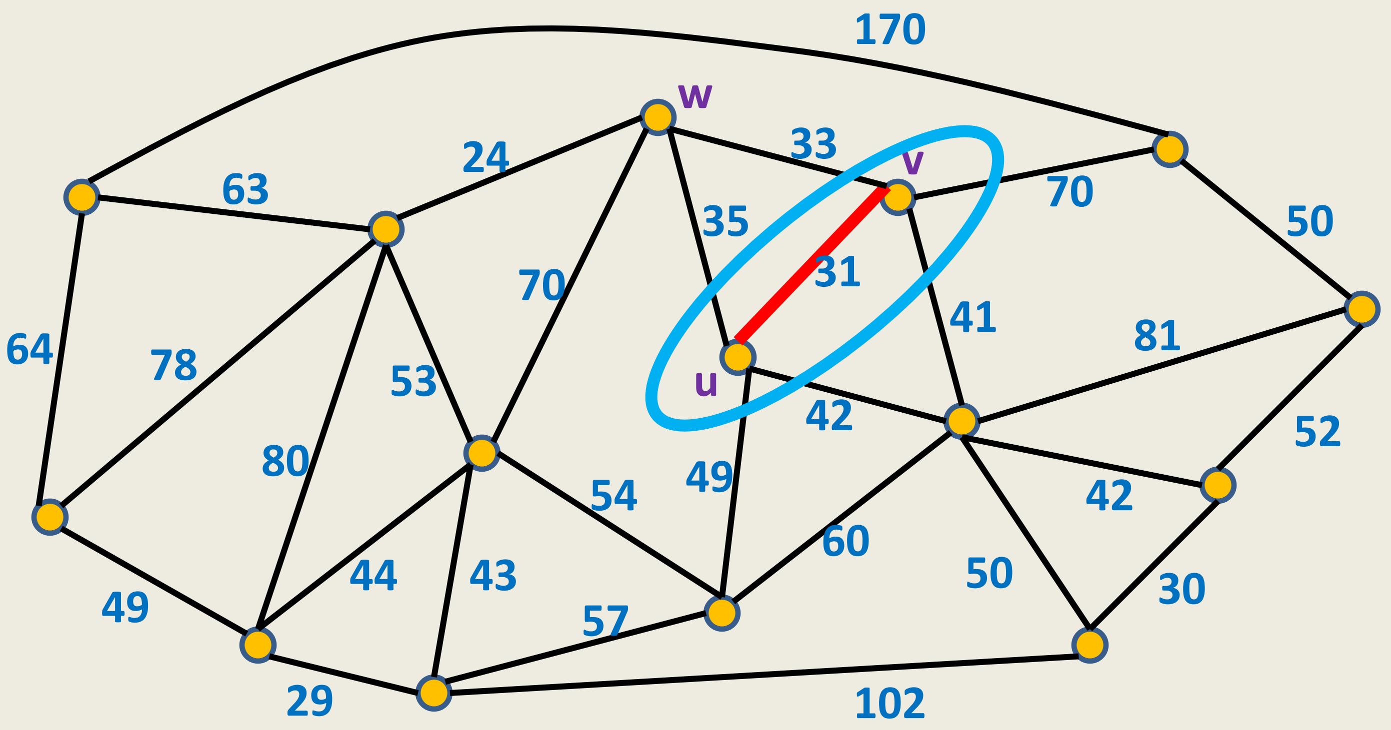


Algorithms based on cut Property

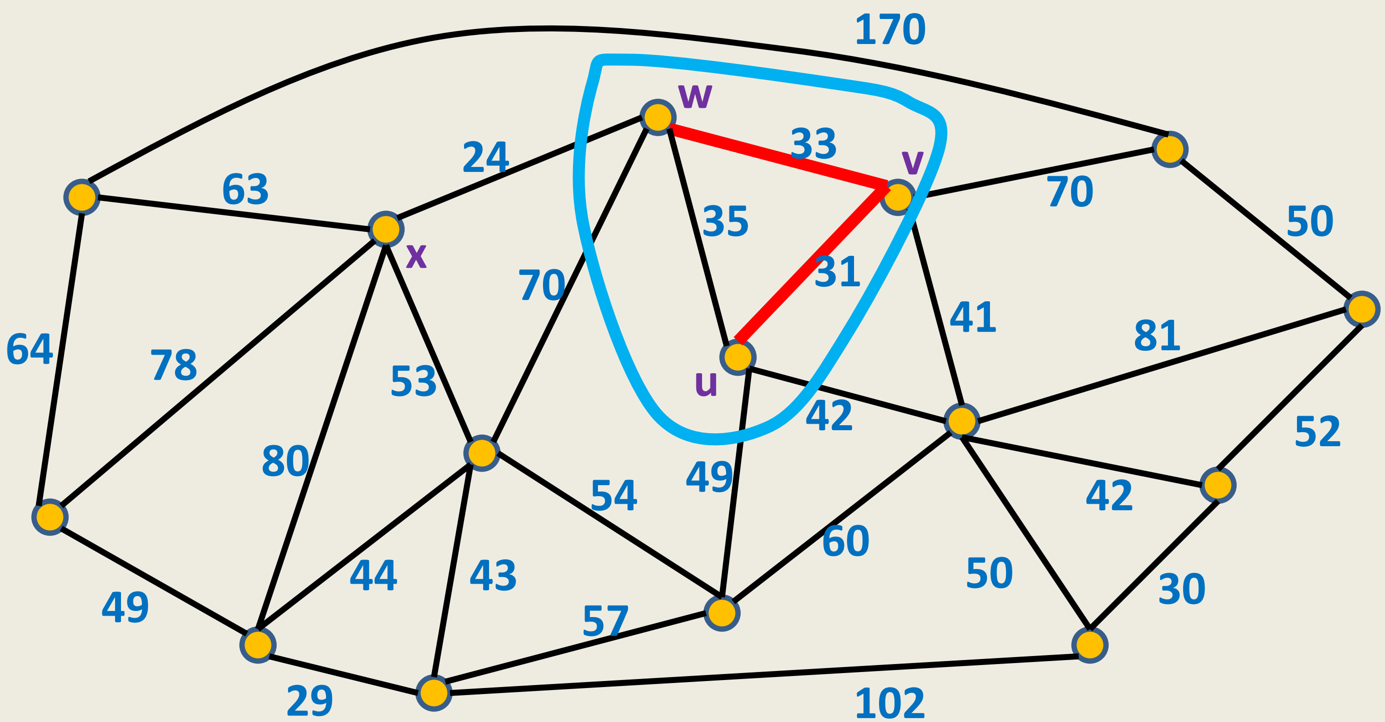
How to use cut property to compute a MST ?



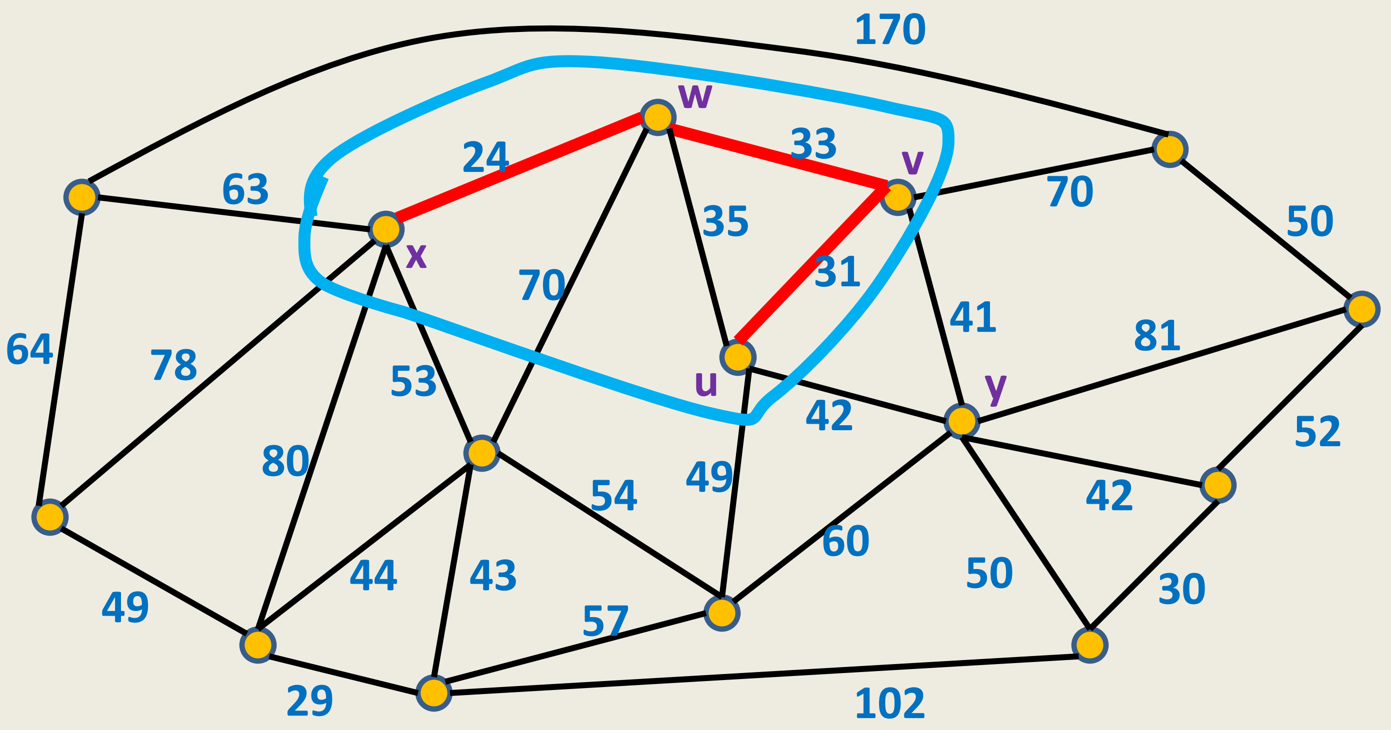
How to use cut property to compute a MST ?



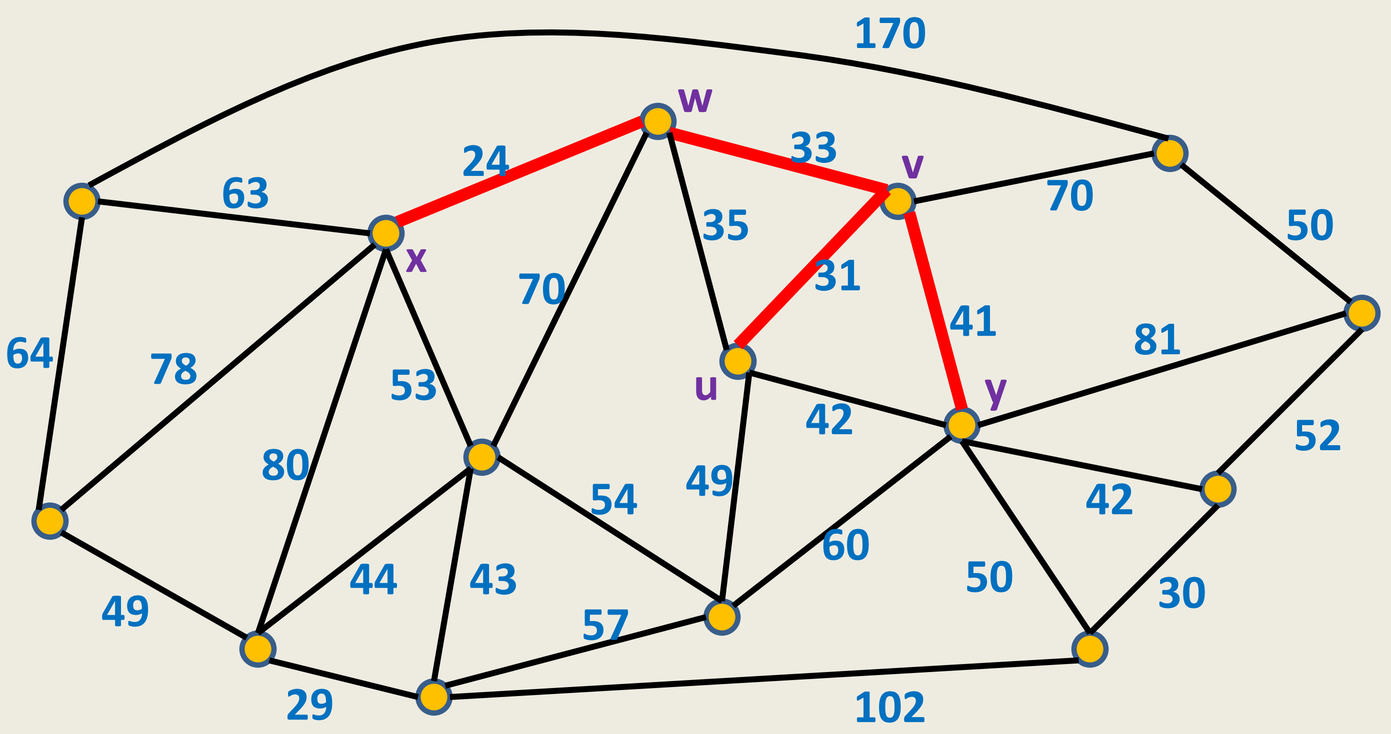
How to use cut property to compute a MST ?



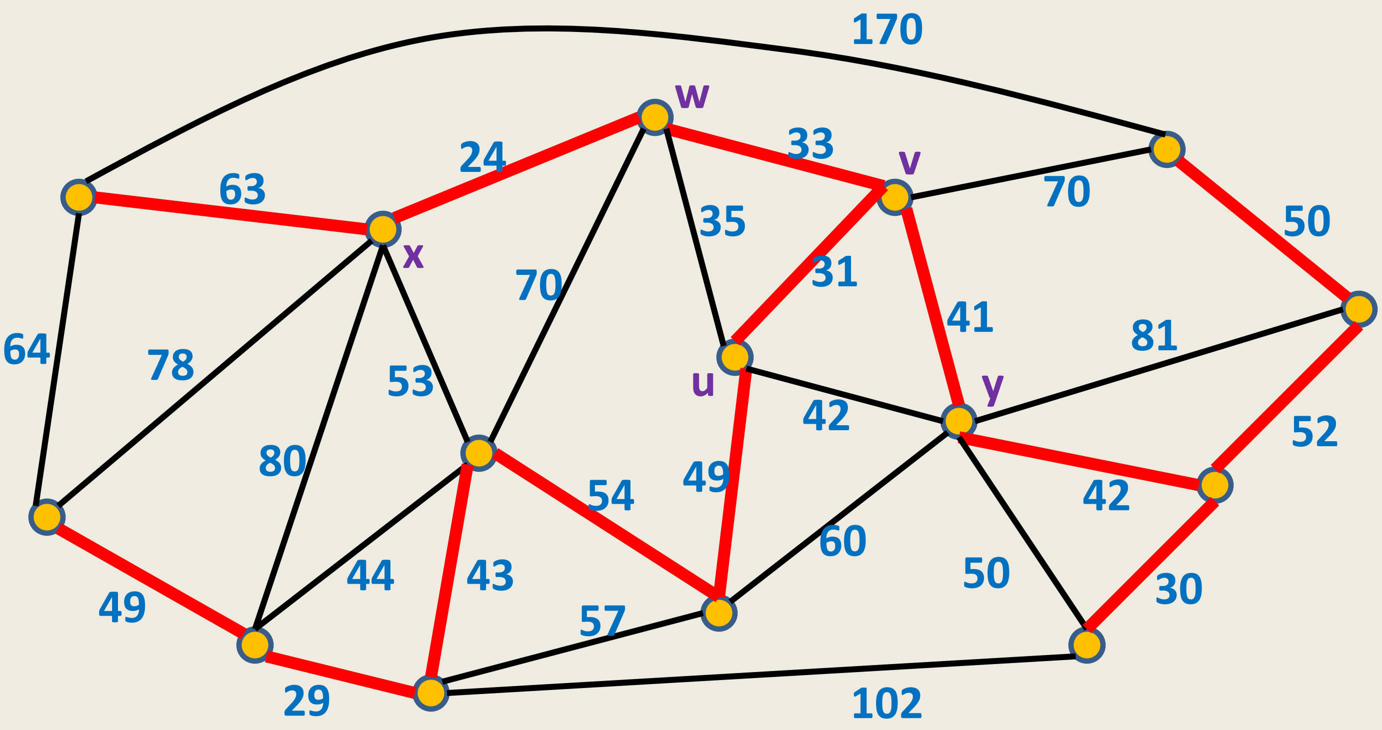
How to use cut property to compute a MST ?



How to use cut property to compute a MST ?



How to use cut property to compute a MST ?



An Algorithm based on cut property

Algorithm (Input: **graph** G =(V,E) with **weights** on edges)

$T \leftarrow \emptyset$;

$A \leftarrow$ {u};

While ($A <> V$ **) do**

{ Compute the least weight edge from **cut**(A,$\bar{A}$);

Let this edge be (**x**,**y**), with **x**$\in A$, **y**$\in \bar{A}$;

$T \leftarrow T \cup$ {(**x**, **y**)};

$A \leftarrow A \cup${**y**};

}

Return T;

Number of iterations of the **While** loop **:** $n - 1$

Time spent in one iteration of While loop: O(m)

➔ **Running time** of the algorithm: O(mn)

Algorithm based on cycle Property

An Algorithm based on cycle property

Description

Algorithm **(**Input: **graph** G =(V,E) with **weights** on edges)

While (E **has any cycle** **) do**

{ Compute any cycle C;

Let (**u**,**v**) be the **maximum weight** edge of the cycle C;

Remove (**u**,**v**) from E;

}

Return E;

Number of iterations of the **While** loop **:** $m - n + 1$

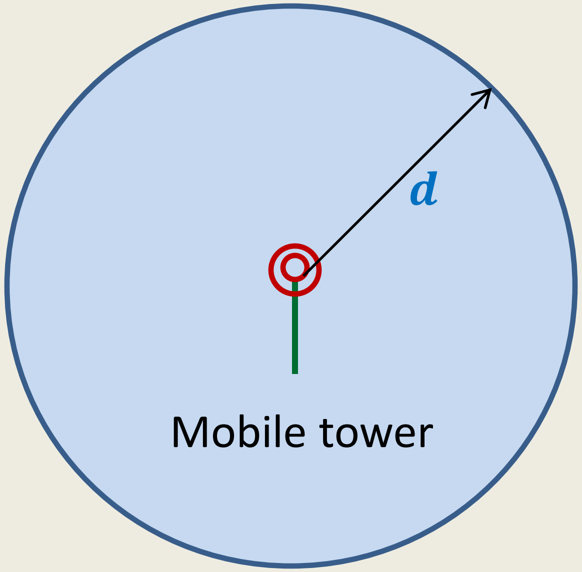
Time spent in one iteration of While loop: **O**(n)

➔ **Running time** of the algorithm: **O**(mn)

Problem 3

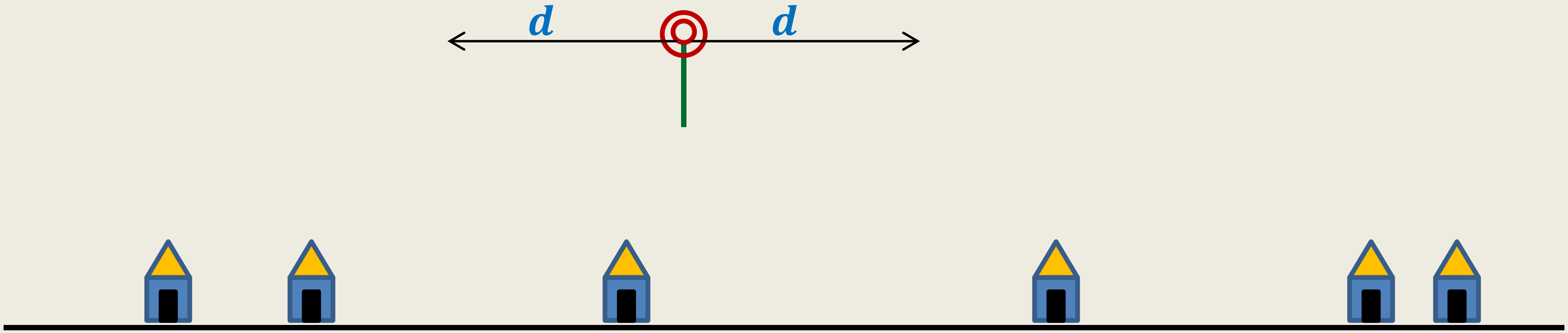
Mobile towers on a road

Mobile towers on a road



A mobile tower can cover any cell phone within radius d.

Mobile towers on a road



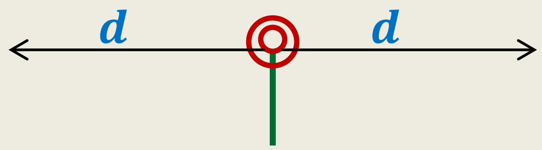
Problem statement:

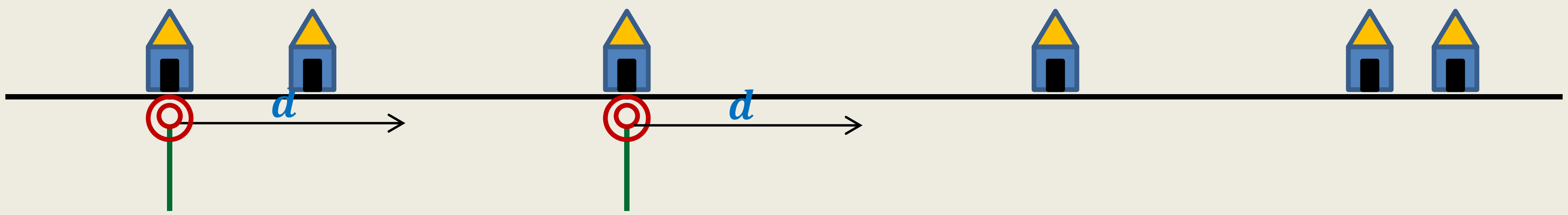
There are n houses located along a road.

We want to place mobile towers such that

- Each house is **covered** by at least one mobile tower.
- The number of mobile towers used is **least** possible.

Mobile towers on a road





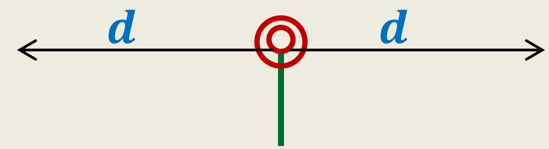
Strategy 1:

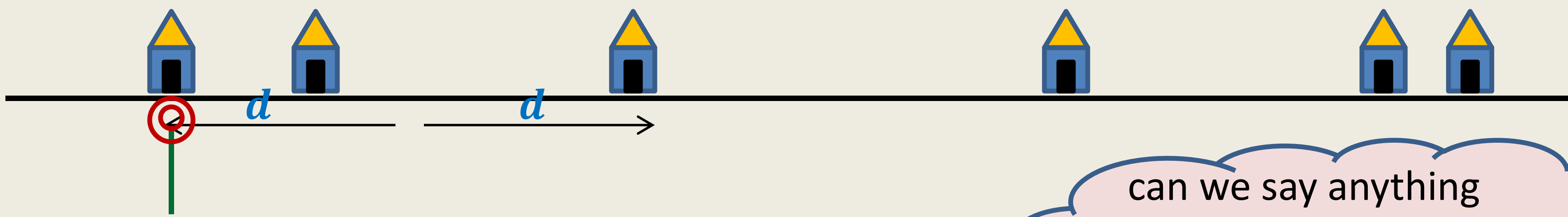
Place tower at first house,

Remove all houses covered by this tower.

Proceed to the next uncovered house …

Mobile towers on a road





can we say anything about the optimal solution ?

Strategy 2:

Place tower at distance d to the right of the first house;

Remove all houses covered by this tower;

Proceed to the next uncovered house along the road…

Lemma: There is an optimal solution for the problem in which the leftmost tower is placed at distance d to the right of the first house

Homework …

Ponder over the following questions before coming for the next class

- Use **cycle property** and/or **cut property** to design a **new algorithm for MST**
- Use some data structure to improve the running time of the algorithms discussed in this class to **O**($m \log n$)