

CSC373 Worksheet 5 Solution

August 11, 2020

1. *Proof.* Assume that a flow network $G = (V, E)$ violates the assumption that the network contains a path $s \rightsquigarrow v \rightsquigarrow t$ for all vertices $v \in V$. Let u be a vertex for which there is no path $s \rightsquigarrow u \rightsquigarrow t$.

I must show such that there is no flow at vertex u . That is, there exists a maximum flow f in G such that $f(u, v) = f(v, u) = 0$ for all vertices $v \in V$.

Assume for the sake of contradiction that there is some vertex u with flow f . That is, there exists some vertices $v \in V$ such that $f(u, v) > 0$ or $f(v, u) > 0$.

I see that three cases follows, and I will prove each separately.

1. **Cases 1:** $f(u, v) = 0$ and $f(v, u) > 0$

Here, assume that $f(u, v) = 0$ for all $v \in V$ and $f(v, u) > 0$ for some $v \in V$.

Then, we can write $\sum_{v \in V} f(u, v) = 0$ and $\sum_{v \in V} f(v, u) > 0$

But this violates the flow conservation property (i.e $\sum_{v \in V} f(u, v) = \sum_{v \in V} f(v, u)$)

Thus, by proof by contradiction, $f(u, v) = 0$ and $f(v, u) = 0$ for all $v \in V$ and all $u \in V$ with no path $s \rightsquigarrow u \rightsquigarrow t$.

2. **Cases 2:** $f(u, v) > 0$ and $f(v, u) = 0$

Here, assume that $f(u, v) > 0$ for some $v \in V$ and $f(v, u) = 0$ for all $v \in V$.

Then, by similar work as case 1, the same result follows.

3. Cases 3: $f(u, v) > 0$ and $f(v, u) > 0$

Here, assume that $f(u, v) > 0$ and $f(v, u) > 0$ for some $v \in V$.

Since $s \rightsquigarrow v \rightsquigarrow t$ and u is connected by some vertices v , we can write $s \rightsquigarrow u \rightsquigarrow t$.

Then, this violates the fact in header that the vertex u has no path $s \rightsquigarrow u \rightsquigarrow t$.

Thus, by proof by contradiction, $f(u, v) = 0$ and $f(v, u) = 0$ for all $v \in V$ and all $u \in V$ with no path $s \rightsquigarrow u \rightsquigarrow t$.

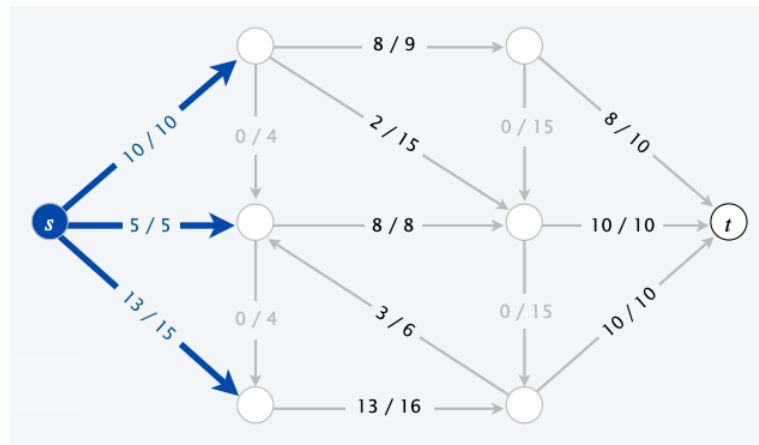
□

Notes

• Maximum Flow:

- Finds a flow of maximum value ^[1]

Example



Here, the maximum flow is $10 + 5 + 13 = 28$

• Flow Network:

- $G = (V, E)$ is a directed graph in which each edge $(u, v) \in E$ has a nonnegative capacity $c(u, v) \geq 0$.
- Two vertices must exist: **source** s and **sink** t
- **path** from source s to vertex v to sink t is represented by $s \rightsquigarrow v \rightsquigarrow t$



- **Capacity:**

- Is a non-negative function $f : V \times V \rightarrow \mathbb{R}_{\geq 0}$
- Has **capacity constraint** where for all $u, v \in V$ $0 \leq f(u, v) \leq c(u, v)$
 - * Means flow cannot be above capacity constraint

- **Flow:**

- Is a real valued function $f : V \times V \rightarrow \mathbb{R}$ in G
- Satisfies **capacity constraint** (i.e for all $u, v \in V$, $0 \leq f(u, v) \leq c(u, v)$)
- Satisfies **flow conservation**

For all $u \in V - \{s, t\}$, we require

$$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v) \quad (1)$$

* Means flow into vertex u is the same as flow going out of vertex u . ^[1]

* $\sum_{v \in V} f(u, v)$ means flow out of vertex u

* $\sum_{v \in V} f(v, u)$ means flow into vertex u

* $v \in V$ in $\sum_{v \in V} f(u, v)$ means all vertices that are an edge away from vertex u

Example:



References

- 1) Princeton University, Network Flow 1, link
2. I need to formulate the problem of determining whether both of professor Adam's two children can go to the same school as maximum-flow problem.

The problem statement tells us the following:

1. There is 1 supersource (location of home)
2. There is 1 sink (location of school)
3. There are two sources (s_1 as child 1, s_2 as child 2)
4. Edge (u, v) has capacity of 0 or more (0 representing unavailable sidewalk, 1 for sidewalk with capacity of 1, 2 for street with capacity of 2 and so on)
5. Each vertex represents corner of intersection, and two children can have their paths crossing here.
6. Has flow of 2, 1 or 0 (1 is where one of the two children walking on the road. 0 is none.)

Here we are to find whether children must go on to a vertex and out to the same edge with the flow of 2, or determine whether there is only edge to school with capacity of 1 or less.

If none, then both children can safely go to school.



Notes:

- **Cross at a Corner**

- Means to walk across the street at a corner of the intersection.



- **Multiple Sources and Sinks**

- Has edges (s, s_i) where $i = 1 \dots n$ and (t_j, t) where $j = 1 \dots n$ with capacity of ∞

Example:

Lucky Puck Company having a set of m factories $\{s_1, s_2, \dots, s_m\}$, and a set of n warehouses and n warehouses $\{t_1, t_2, \dots, t_n\}$



3. I need to show how to transform a flow network $G = (V, E)$ with vertex capacities into an equivalent flow network $G' = (V', E')$ without vertex capacities.

For each vertex capacities, change as follows.



After transformation, there will be m more edges and vertices, where m represents the number of vertex capacities in G .

Notes:

- **Vertex Capacities**

- Each vertex v has limit $l(v)$ on how much flow can pass through v

4. I need to show how to convert the problem of finding a flow f that obeys the constraints into the problem of finding a maximum flow in a single source, single-sink flow network

The steps are as follows:

- Combine all sources s_i into a single source s
- Combine all sinks t_j into a single sink t
- Connect source s to each adjacent vertex v with edge weight $\sum_i f(s_i, v) = p_i$
 - The total edge weight from s should be $\sum_i p_i$
- Connect each adjacent vertex v of t to t with edge weight $\sum_j f(v, t_j) = q_j$
 - The total edge weight to t should be $\sum_j q_j$
- Find a simple path from s to t with the maximum amount of total flow



Correct Solution:

I need to show how to convert the problem of finding a flow f that obeys the constraints into the problem of finding a maximum flow in a single source, single-sink flow network

The steps are as follows:

- Combine all sources s_i into a single source s
- Combine all sinks t_j into a single sink t
- Connect source s to each adjacent vertex v with edge weight $\sum_i f(s_i, v) = p_i$
 - The total edge weight from s should be $\sum_i p_i$
- Connect each adjacent vertex v of t to t with edge weight $\sum_j f(v, t_j) = q_j$
 - The total edge weight to t should be $\sum_j q_j$
- Find a simple path from s to t with the maximum amount of total flow

Example



Notes:

- **Ford-Fulkerson Method**
 - Is a greedy algorithm that solves the maximum-flow problem
 - * Determines maximum flow from start vertex to sink vertex in a graph
 - Called method (not algorithm) because several different implementations with different running time is used

FORD-FULKERSON-METHOD(G, s, t)

- 1 initialize flow f to 0
- 2 **while** there exists an augmenting path p in the residual network G_f
- 3 augment flow f along p
- 4 **return** f

• Residual Network

- Indicates how much more flow is allowed in each edge in the network graph ^[1]
- Consists of edges with capacities that represents how we can change the flow on edges of G .
- Provides roadmap for adding flow to the original flow network



Steps

- 1) $Flow = Capacity$: Opposite arrow



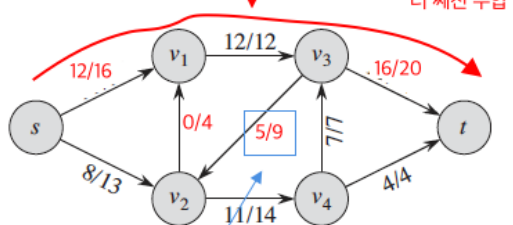
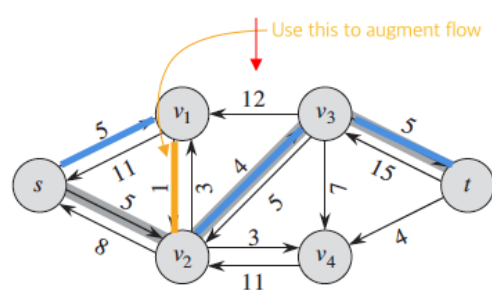
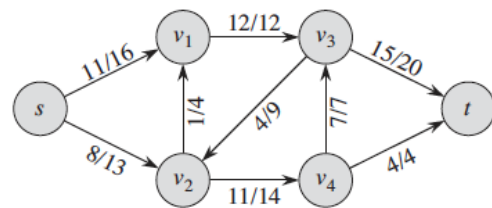
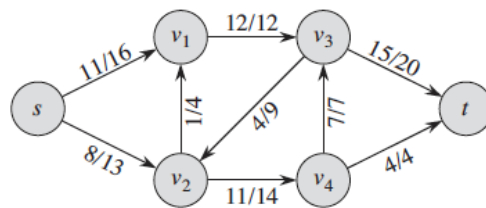
- 2) $Flow < Capacity$:

- $Flow$: Opposite Arrow
- $Capacity - Flow$: Current Arrow



• Augmenting Path

- Is a path from source S to sink T where you can increase the amount of flow
- Is a path that doesn't contain cycle (simple path) [2]



A good augmentation

An augmentation but not good
(decrease 가 필요한 곳에 쓰이면 더 좋았을 걸)

- Edge (u, v) of an augmented path can be increased by upto $c_f(u, v)$ without violating the capacity constraint

• Augmentation

- 한국어로 '불필요한 수압 decrease 해서 앞으로 가는 수압 더 쉐게 만들기'
- Is symbolized by $f \uparrow f'$

- * f is a flow in G
- * f' is a flow in the residual network G_f

References

- 1) Hacker Earth, Maximum Flow, link
 - 2) Stack Overflow, What Exactly Is Augmentation Path, link
5. The augmented flow satisfies flow conservation, but not capacity constraint.

Proof. Let $G = (V, E)$ be a flow network with sources s and sink t . Let f, f' be a flow in G . Let (u, v) be an edge in E where $u \in V - \{s, t\}$ and $v \in V$. We note that if $(u, v) \in E$, then $(v, u) \notin E$ and $f(v, u) = 0$. Thus, we can re-write the definition of flow augmentation (equation (26.4)) as

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) & [\text{If } (u, v) \in E] \\ 0 & [\text{Otherwise}] \end{cases} \quad (1)$$

which implies that the value of the augmentation of flow $f \uparrow f'$ on edge (u, v) is the sum of flow $f(u, v)$ and $f'(u, v)$ in G .

I need to show if the augmented flow of f and $f' \in G$ and satisfy the flow conservation property but not capacity constraint.

I will do so in parts

- **Part 1: Proving that $f \uparrow f'$ satisfies the flow conservation property**

Here I prove that the augmented flow satisfies flow conservation. That is,

$$\sum_{v \in V} f \uparrow f'(u, v) = \sum_{v \in V} f \uparrow f'(v, u) \quad (2)$$

.

And indeed we have,

$$\sum_{v \in V} f \uparrow f'(u, v) = \sum_{v \in V} f(u, v) + f'(u, v) \quad [\text{By augmentation def.}] \quad (3)$$

$$= \sum_{v \in V} f(u, v) + \sum_{v \in V} f'(u, v) \quad (4)$$

$$= \sum_{v \in V} f(v, u) + \sum_{v \in V} f'(v, u) \quad [\text{By flow conserv. of } f \text{ and } f'] \quad (5)$$

$$= \sum_{v \in V} f(v, u) + f'(v, u) \quad (6)$$

$$= f \uparrow f'(v, u) \quad (7)$$

• **Part 2: Disproving that $f \uparrow f'$ satisfies the capacity constraint**

Here, I need to disprove that the augmented flow satisfies capacity constraint. That is,

$$(f \uparrow f')(u, v) > c(u, v) \quad (8)$$

Let $f(u, v) = f'(u, v) = 10$ and $c(u, v) = 8$.

Then, we can write $(f \uparrow f')(s, t) = 20$ and $c(u, v) = 8$.

Thus, we can conclude the augmentation of flow doesn't satisfy capacity constraint.

□

Notes:

- I need clarification from professor about the meaning of $f' \in G$. Is f' a flow from flow network or residual network?
- I feel I am struggling because I am jumping to solution without understanding the problem
- I feel constructing a predicate logic would have helped to better understand this problem
- Noticed that a solution in University of Texas really elaborated on $f \uparrow f'(u, v)$ before moving onto strategizing and constructing a solution

capacity constraint property.

First, we prove that $f \uparrow f'$ satisfies the flow conservation property. We note that if edge $(u, v) \in E$, then $(v, u) \notin E$ and $f(v, u) = 0$. Thus, we can rewrite the definition of flow augmentation (equation (26.4)), when applied to two flows, as

$$f \uparrow f'(u, v) = \begin{cases} f(u, v) + f'(u, v), & \text{if } (u, v) \in E \\ 0, & \text{otherwise.} \end{cases}$$

The definition implies that the new flow on each edge is simply the sum of the two flows on that edge. We now prove that in $f \uparrow f'$, the net incoming flow for each vertex equals the net outgoing flow. Let $u \notin \{s, t\}$ be any vertex of G . We have

- Noticed that a solution in University of Texas made quick sketches before laying the outline of proof
- **Flow Network (cont'd)** [Important!]
 - Flow network requires that
 - 1) $G = (V, E)$ is a directed graph
 - 2) each edge $(u, v) \in E$ has a non-negative capacity $c(u, v) \geq 0$
 - 3) If E contains an edge (u, v) , then there is no edge (v, u) in the reverse direction (no anti-parallel edge)
- **Augmentation (cont'd)**
 - Flow value can be increased by

$$c_f(p) = \min_{(u,v) \in p} c_f(u, v) \quad (9)$$

Example:

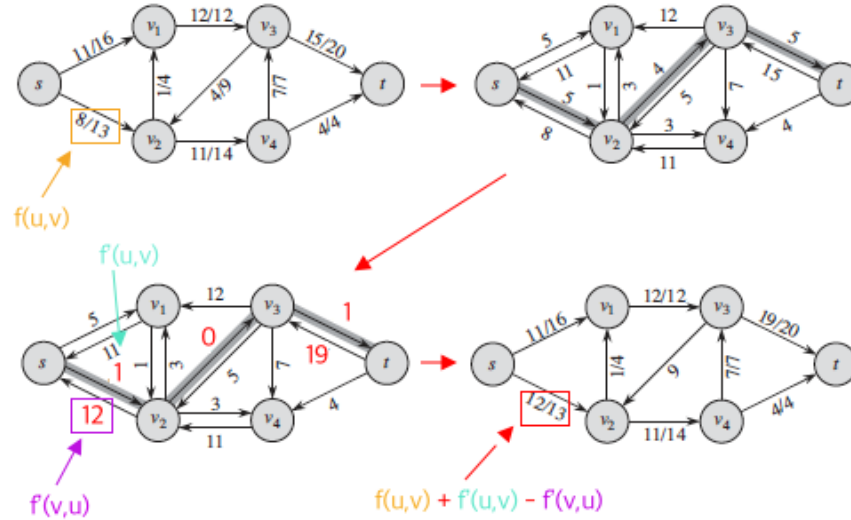


In this example, augmentation by 2.

- Augmentation of flow f by f' or $f \uparrow f'$ is a function $V \times V \rightarrow \mathbb{R}$ is defined by

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - f'(v, u) & [\text{If } (u, v) \in E] \\ 0 & [\text{Otherwise}] \end{cases} \quad (10)$$

- Augmentation of flow $f \uparrow f'$ is the sum of flow on edge (u, v) in both flow network G and residual network G'



[NOTE!!] I really need to ask professor about this. I can't seem to understand using simple example why $f \uparrow f'(u, v) = f(u, v) + f'(u, v) - f'(v, u)$.

• **Proof of flow conservation for $f \uparrow f'$ when $f \in G$ and $f' \in G_f$**

Let $G = (V, E)$ be a flow network with sources s and sink t . Let f be a flow in G . Let G_f be a residual network of G induced by f and let f' be a flow in G_f . Let (u, v) be an edge in E where $u \in V - \{s, t\}$ and $v \in V$. We note that if $(u, v) \in E$, then $(v, u) \notin E$ and $f(v, u) = 0$. Thus, the definition

$$(f \uparrow f')(u, v) = \begin{cases} f(u, v) + f'(u, v) - f'(v, u) & [\text{If } (u, v) \in E] \\ 0 & [\text{Otherwise}] \end{cases} \quad (1)$$

implies that the augmented flow $f \uparrow f'(u, v)$ on edge (u, v) is the sum of flow $f(u, v)$ in flow network G and flow $f'(u, v)$ minus its antiparallel flow $-f'(v, u)$ in residual flow network G' .

We now prove that the augmented flow satisfies flow conservation. That is,

$$\sum_{v \in V} f \uparrow f'(u, v) = \sum_{v \in V} f \uparrow f'(v, u) \quad (2)$$

And indeed we have

$$\sum_{v \in V} f \uparrow f'(u, v) = \sum_{v \in V} f(u, v) + f'(u, v) - f'(v, u) \quad [\text{By augmentation def.}] \quad (3)$$

$$= \sum_{v \in V} f(u, v) + \sum_{v \in V} f'(u, v) - \sum_{v \in V} f'(v, u) \quad (4)$$

$$= \sum_{v \in V} f(v, u) + \sum_{v \in V} f'(v, u) - \sum_{v \in V} f'(u, v) \quad [\text{By flow conserv. of } f \text{ and } f'] \quad (5)$$

$$= \sum_{v \in V} f(v, u) + f'(v, u) - f'(u, v) \quad (6)$$

$$= \sum_{v \in V} f \uparrow f'(v, u) \quad (7)$$

– Flow in residual network also obey flow conservation

• **Proof of capacity constraint for $f \uparrow f'$ when $f \in G$ and $f' \in G_f$**

Predicate Logic: $\forall f \in G, \forall f' \in G_f, \forall (u, v) \in E$ where $u, v \in V, 0 \leq (f \uparrow f')(u, v) \wedge (f \uparrow f')(u, v) \leq c(u, v)$

Let $G = (V, E)$ be a flow network with sources s and sink t . Let f be a flow in G . Let G_f be a residual network of G induced by f and let f' be a flow in G_f . Let (u, v) be an edge in E where $u, v \in V$.

I need to prove that $f \uparrow f'$ satisfies capacity constraint. That is, $0 \leq (f \uparrow f')(u, v) \wedge (f \uparrow f')(u, v) \leq c(u, v)$.

I see there are two parts. I will prove each parts separately.

1. **Part 1** ($0 \leq (f \uparrow f')(u, v)$)

Here, I need to show $0 \leq (f \uparrow f')(u, v)$. That is, $0 \leq f(u, v) + f'(u, v) - f'(v, u)$.

And indeed we have,

$$(f \uparrow f')(u, v) = f(u, v) + f'(u, v) - f'(v, u) \quad (8)$$

$$\geq f(u, v) + f'(u, v) - c_f(v, u) \quad [\text{Since } f'(v, u) \leq c_f(v, u)] \quad (9)$$

$$= f(u, v) + f'(u, v) - f(u, v) \quad [\text{By def. of residual capacity}] \quad (10)$$

$$= f'(u, v) \quad (11)$$

$$\geq 0 \quad [\text{By cap. const. of } f' \text{ in } G_f] \quad (12)$$

$$(13)$$

– $c_f(v, u) = f(u, v)$ is allowed

2. **Part 2** $((f \uparrow f')(u, v) \leq c(u, v))$

Here, I need to show $(f \uparrow f')(u, v) \leq c(u, v)$. That is, $f(u, v) + f'(u, v) - f'(v, u) \leq c(u, v)$.

And indeed we have,

$$(f \uparrow f')(u, v) = f(u, v) + f'(u, v) - f'(v, u) \quad (14)$$

$$\leq f(u, v) + f'(u, v) \quad [\text{Since } f'(u, v) \geq 0 \text{ by cap. cons. of } f'] \quad (15)$$

$$= f(u, v) + c_f(u, v) \quad [\text{Since } f'(u, v) \leq c_f(u, v)] \quad (16)$$

$$= f(u, v) + (c(u, v) - f(u, v)) \quad [\text{By def of res. capacity}] \quad (17)$$

$$= c(u, v) \quad (18)$$

$$(19)$$

References

1) University of Teaxs, CSE 5311 Homework 5 Solution, link

6. Rough Works:

Let $G = (V, E)$ be an undirected graph.

I need to show how to determine the edge connectivity of G by running a maximum-flow algorithm.

- Convert undirected graph to directed graph

First, I need to convert the undirected graph G to a directed graph.

I do so by assigning G' as a directed graph and transforming each edges in G to two directed edges (u, v) and (v, u) .

- Setup directed graph as a flow network

Second, I need to setup graph G' as a flow network.

I do so by assigning each edge in G' with capacity of 1 and flow of 1, and assigning f' as max-flow in G' found using maximum-flow algorithm.

- Find the minimum of max-flow in G'

Third, I need to find the edge connectivity of the directed graph G' . That is, the minimum number of edges that must be removed to disconnect G' . In other words, the minimum number of edges required to remove a vertex from graph G' .

And I do so by claiming that the edge connectivity of G' or G is $\min_{u \neq v} f'(u, v)$. That is, the minimum of max-flows over $u \in V$ in graph G' .

Therefore, we can find edge over all vertices u with $|V| - 1$ running time.

- Prove correctness of Algorithm

Suppose k is the edge connectivity of the graph and S is the set of k edges such that removal of S will disconnect the graph into 2 non-empty subgraphs G_1 and G_2 . WLOG assume the node $u \in G_1$. Let w be a node in G_2 . Since $u \neq w$, the value $f^*(u, w)$ will be computed by the algorithm. By the min-cut theorem $f^*(u, w)$ equals the min cut size between the pair (u, w) , which is at most k since S disconnects u and w . Therefore, we have

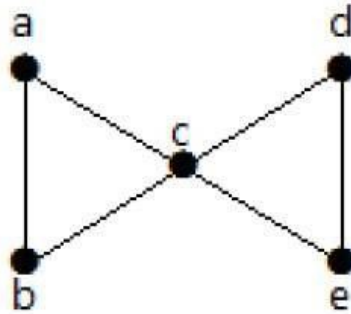
$$c^* \leq f^*(u, w) \leq k \quad (20)$$

But c^* cannot be smaller than k since that would imply a cut set of size smaller than k , contradicting the fact that k is the edge connectivity. Therefore $c^* = k$ and the algorithm returns the edge connectivity of the graph correctly.

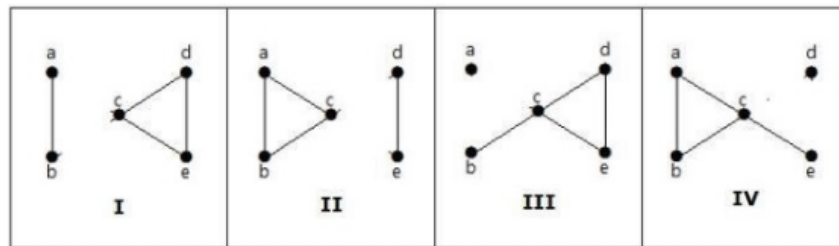
Notes

- I feel that this maximum-flow algorithm maximizes flow value in a vertex with edges and antiparallel edges.
- I wonder what does this maximum-flow algorithm do.
- I am wondering how I can
- **Maximum Flow:** Is a vertex in flow network with the maximum value of flow
- **Edge Connectivity:** Is the minimum number k of edges that must be removed to disconnect the graph. That is, the **number of edges in a smallest cut set of G** [2]

Example:

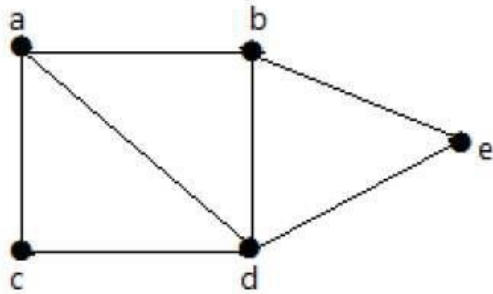


Here are the four ways to disconnect the graph by removing two edges –

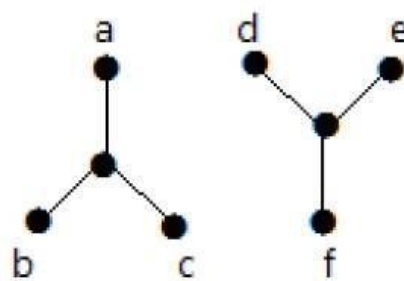


- **Connected:** A graph is said to be connected if there is a path between every pair of vertex

Example:



Connected



Not Connected

- **Cut:**
 - Is denoted (S, T)
 - Is a partition of V into S and $T = V - S$ such that $s \in S$ and $t \in T$

References

- 1) National Taiwan University, Voluntary Exercise 3, [link](#)
- 2) TutorialsPoint, Graph Theory - Connectivity, [link](#)