CSE408
Maximum Flow

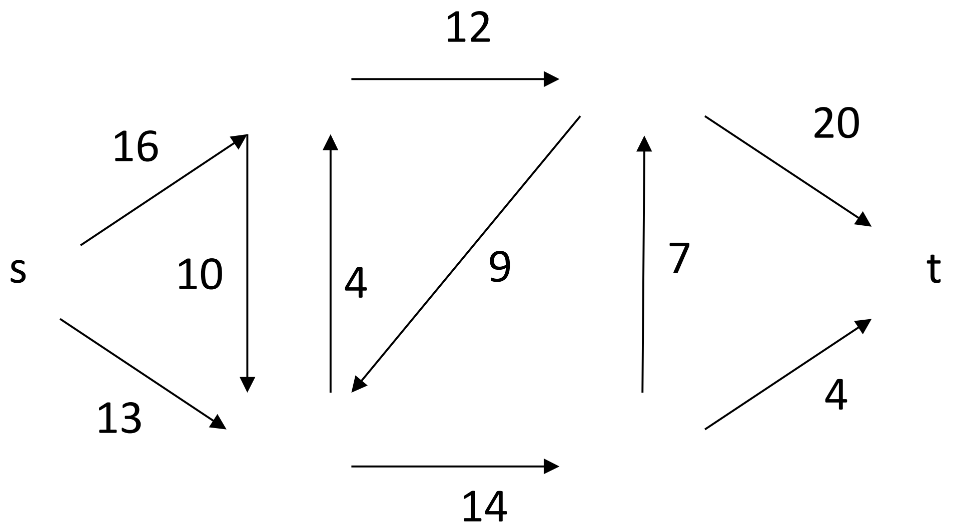
Lecture #30

Maximum Flow

- Maximum Flow Problem
- The Ford-Fulkerson method
- Maximum bipartite matching

Flow networks:

- A flow network $G=(V,E)$: a directed graph, where each edge $(u,v) \in E$ has a nonnegative capacity $c(u,v)>=0$.
- If $(u,v) \notin E$, we assume that $c(u,v)=0$.
- two distinct vertices :**a source s and a sink t.**

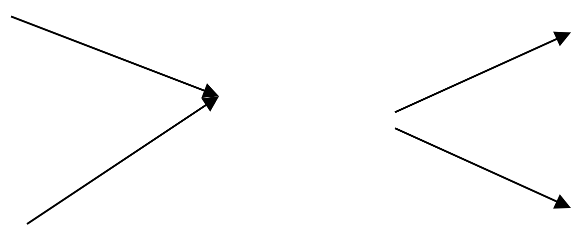


Flow:

- G=(V,E): a flow network with capacity function c.
- s-- the source and t-- the sink.
- A flow in G: a real-valued function f:V*V → R satisfying the following two properties:
- Capacity constraint: For all u,v $\in$V,

  we require $f(u,v) \leq c(u,v)$.
- Flow conservation: For all u $\in$V-{s,t}, we require

$$\sum_{e.in.v} f(e) = \sum_{e.out.v} f(e)$$

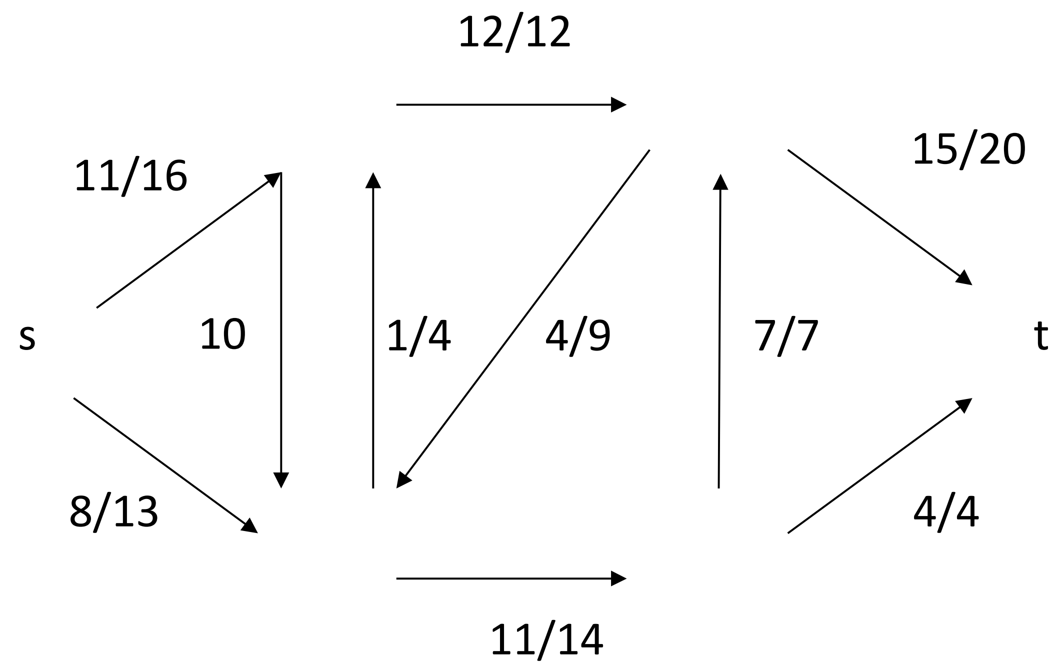


Net flow and value of a flow f:

- The quantity f (u,v) is called the net flow from vertex u to vertex v.
- The value of a flow is defined as

$$|f| = \sum_{v \in V} f(s,v)$$

  - The total flow from source to any other vertices.
  - The same as the total flow from any vertices to **the sink.**



A flow f in G with value $|f| = 19$

Maximum-flow problem:

- Given a flow network G with source s and sink t
- Find a flow of maximum value from s to t.

- How to solve it efficiently?

The Ford-Fulkerson method:

- This section presents the Ford-Fulkerson method for solving the maximum-flow problem.We call it a "method" rather than an "algorithm" because it encompasses several implementations with different running times.The Ford-Fulkerson method depends on three important ideas that transcend the method and are relevant to many flow algorithms and problems: residual networks,augmenting paths,and cuts. These ideas are essential to the important max-flow min-cut theorem,which characterizes the value of maximum flow in terms of cuts of the flow network.

Continue:

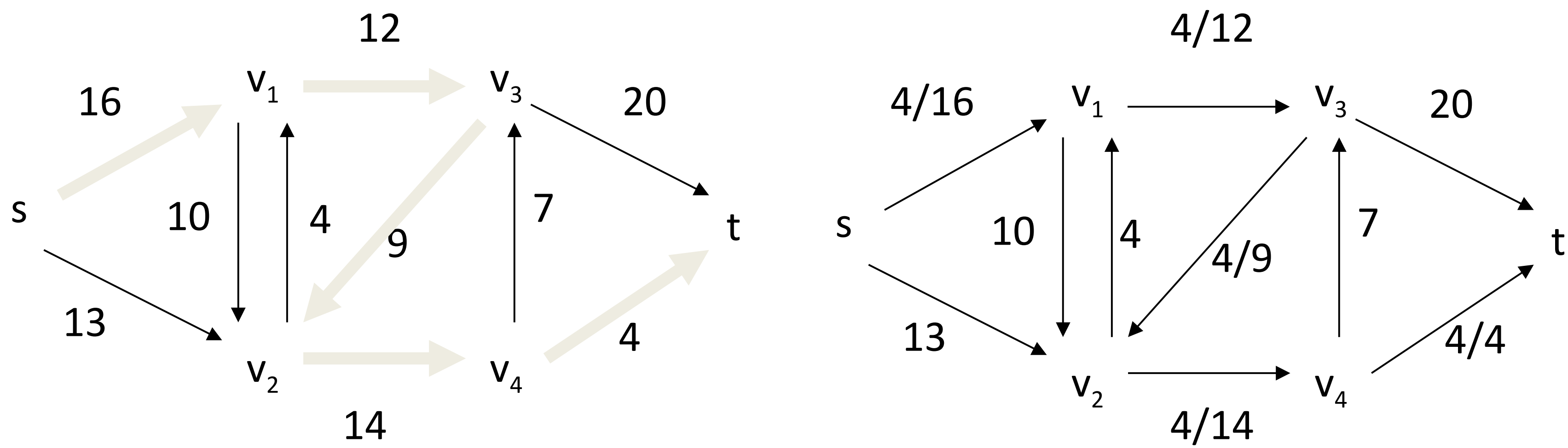
- FORD-FULKERSON-METHOD(G,s,t)
- initialize flow *f* to *0*
- while there exists an *augmenting* path *p*
- do *augment* flow *f* along *p*
- return *f*

Residual networks:

- Given a flow network and a flow, the **residual network** consists of edges that can admit more net flow.
- $G=(V,E)$ --a flow network with source s and sink t
- f: a flow in G.
- The amount of additional net flow from u to v before exceeding the capacity $c(u,v)$ is the residual capacity of (u,v), given by: $c_f(u,v)=c(u,v)-f(u,v)$

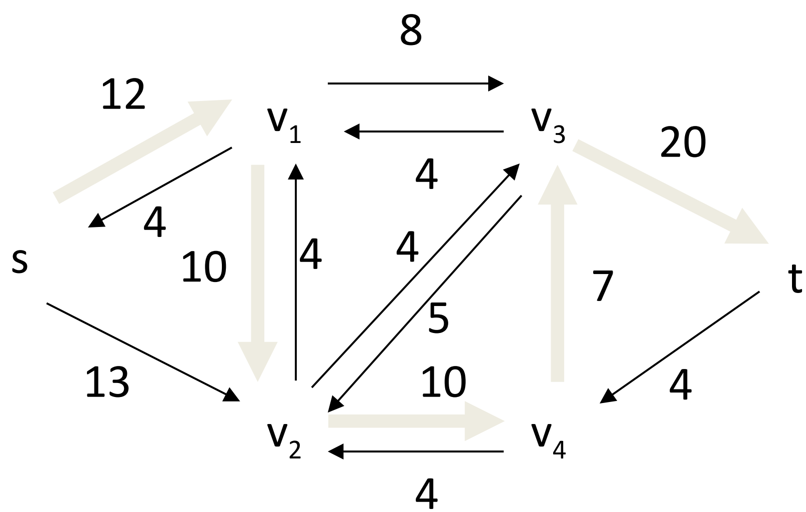
  in the other direction: $c_f(v, u)=c(v, u)+f(u, v)$.

Example of residual network



(a)

Example of Residual network (continued)



(b)

Fact 1:

- Let G=(V,E) be a flow network with source s and sink t, and let f be a flow in G.
- Let G_f be the residual network of G induced by f,and let f' be a flow in G_f.Then, the flow sum f+f' is a flow in G with value

$$|f + f'| = |f| + |f'|$$

- f+f': the flow in the same direction will be added.

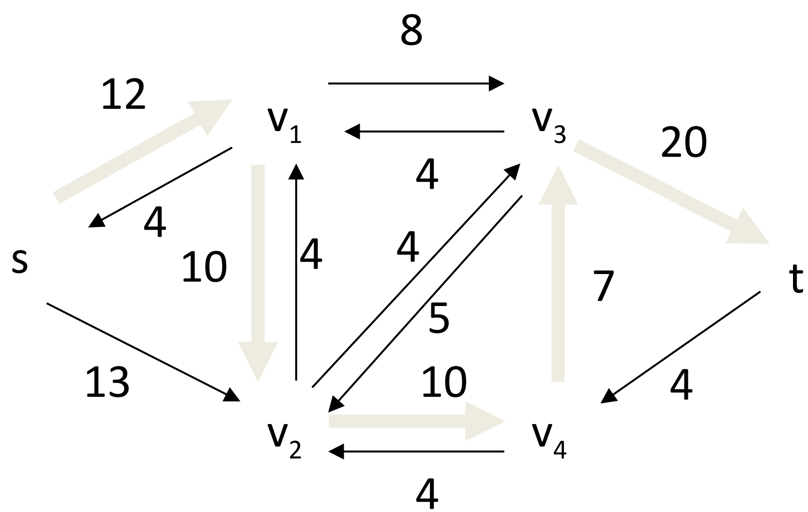
  the flow in different directions will be cnacelled.

Augmenting paths:

- Given a flow network $G=(V,E)$ and a flow f, an augmenting path is a simple path from s to t in the residual network G_f.
- Residual capacity of p : the maximum amount of net flow that we can ship along the edges of an augmenting path p, i.e., $c_f(p)=\min\{c_f(u,v):(u,v) \text{ is on } p\}$.

The residual capacity is 1.

Example of an augment path (bold edges)



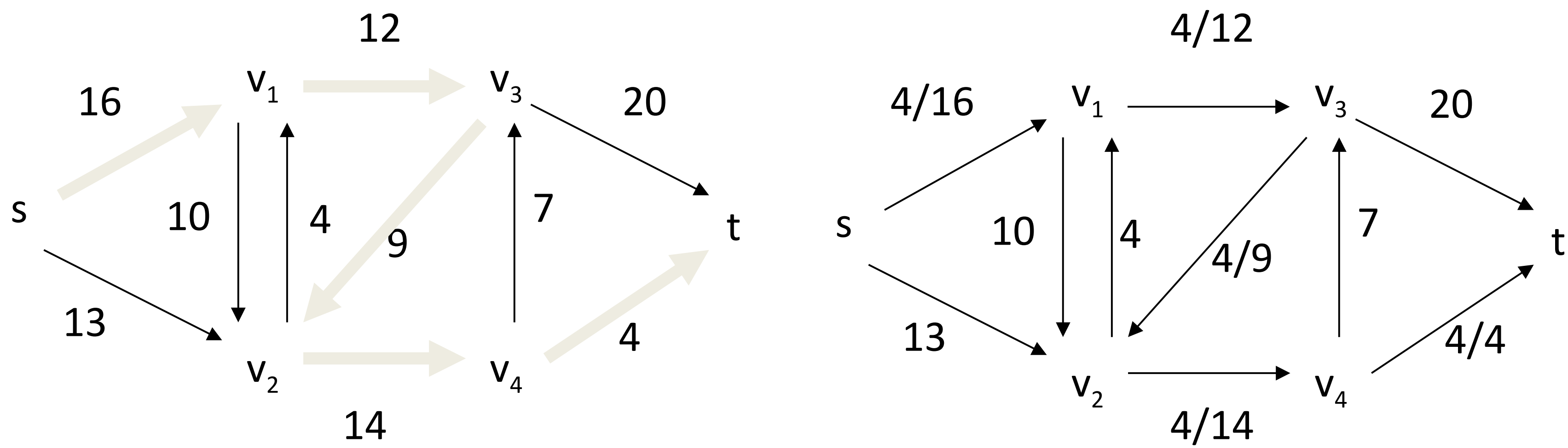
(b)

The basic Ford-Fulkerson algorithm:

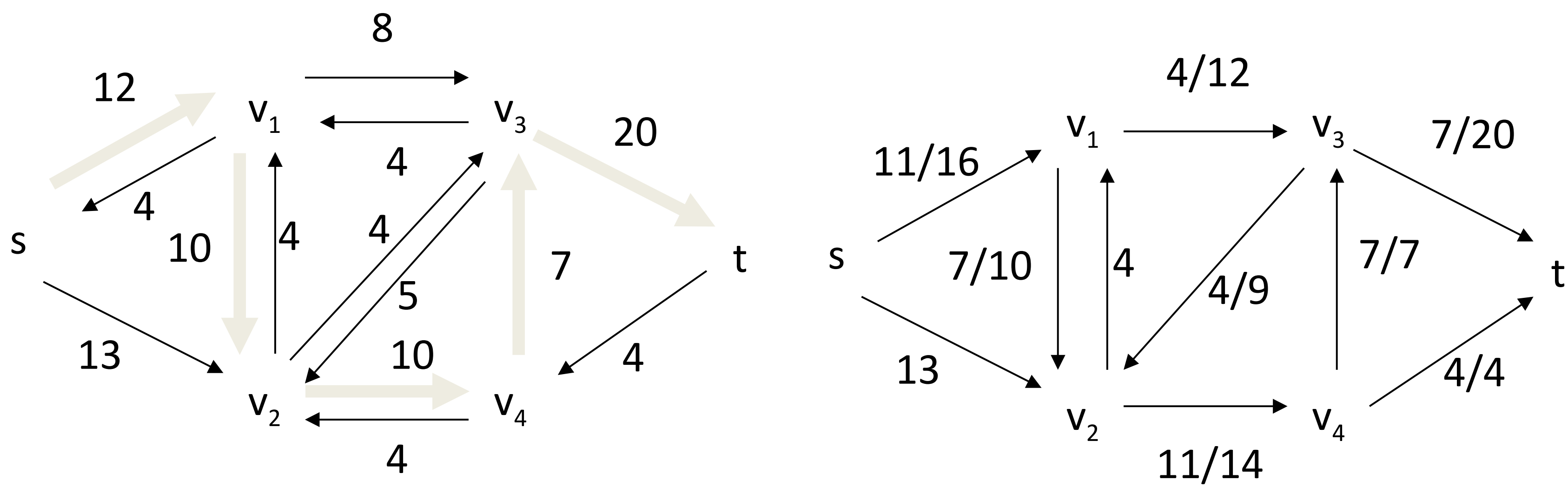
- FORD-FULKERSON(G,s,t)
- for each edge (u,v) $\in E[G]$
- do f[u,v] $\leftarrow 0$
- f[v,u] $\leftarrow 0$
- while there exists a path p from s to t in the residual network G_f
- do $c_f(p) \leftarrow \min\{c_f(u,v): (u,v) \text{ is in } p\}$
- for each edge (u,v) in p
- do $f[u,v] \leftarrow f[u,v]+c_f(p)$
-

Example:

- The execution of the basic Ford-Fulkerson algorithm.
- (a)-(d) Successive iterations of the while loop: The left side of each part shows the residual network G_f from line 4 with a shaded augmenting path p.The right side of each part shows the new flow f that results from adding f_p to f.The residual network in (a) is the input network G.(e) The residual network at the last while loop test.It has no augmenting paths,and the flow f shown in (d) is therefore a maximum flow.

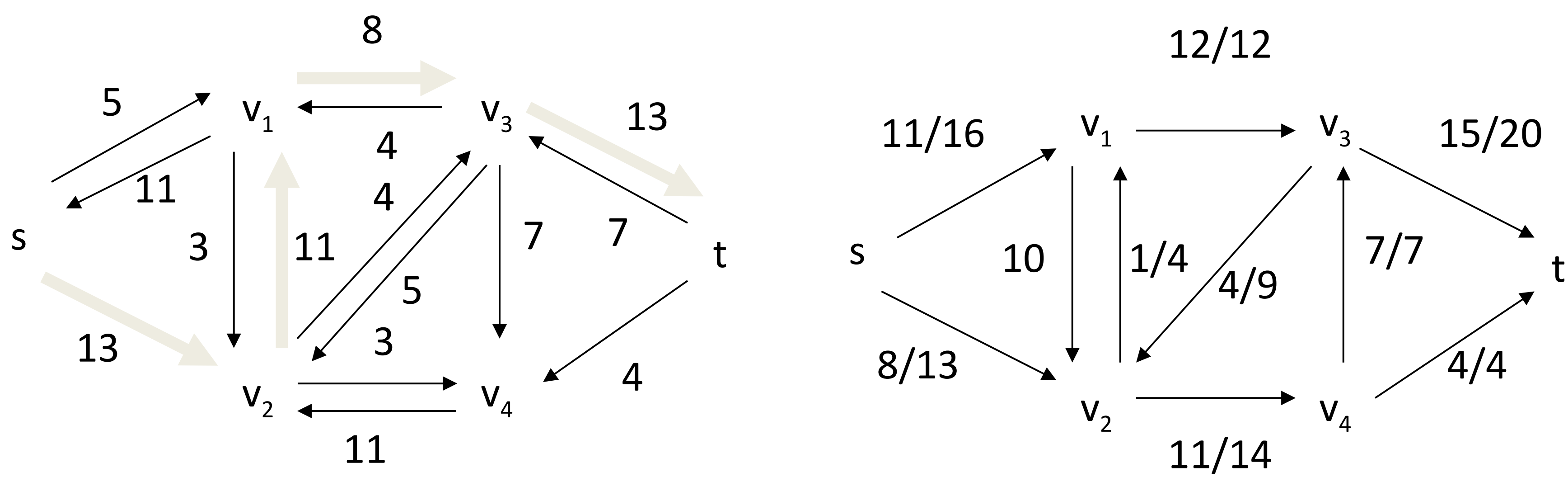


(a)



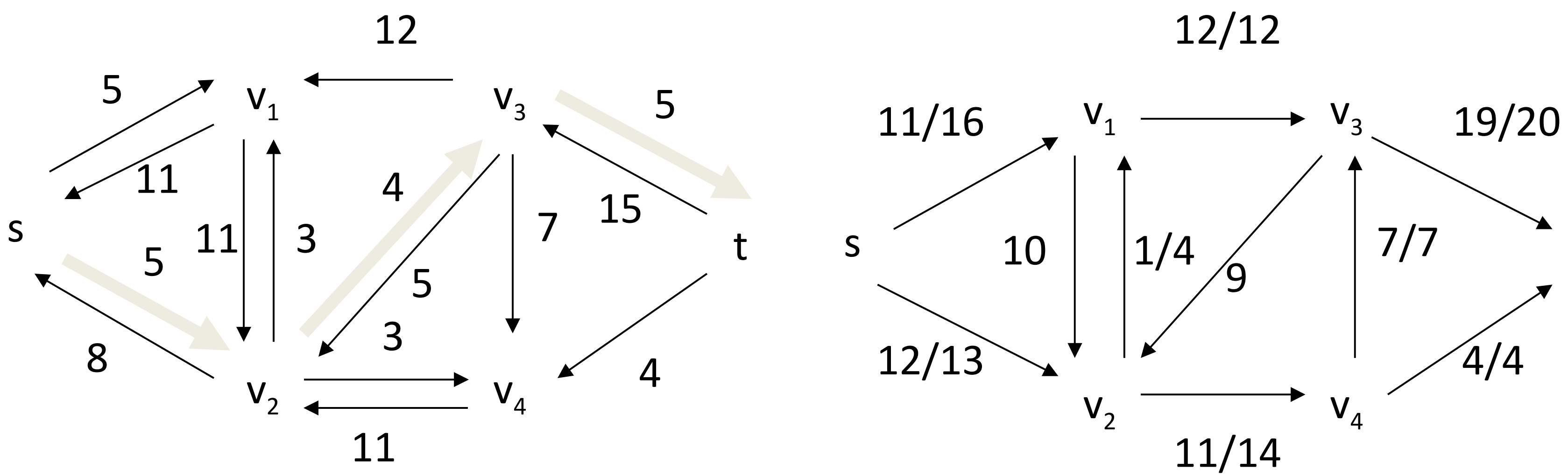
8
12
v_1
v_3
20
4
4
s
10
4
4
7
t
5
13
10
4
v_2
v_4
4
4/12
v_1
v_3
7/20
11/16
s
7/10
4
4/9
7/7
t
13
4/4
v_2
v_4
11/14

(b)

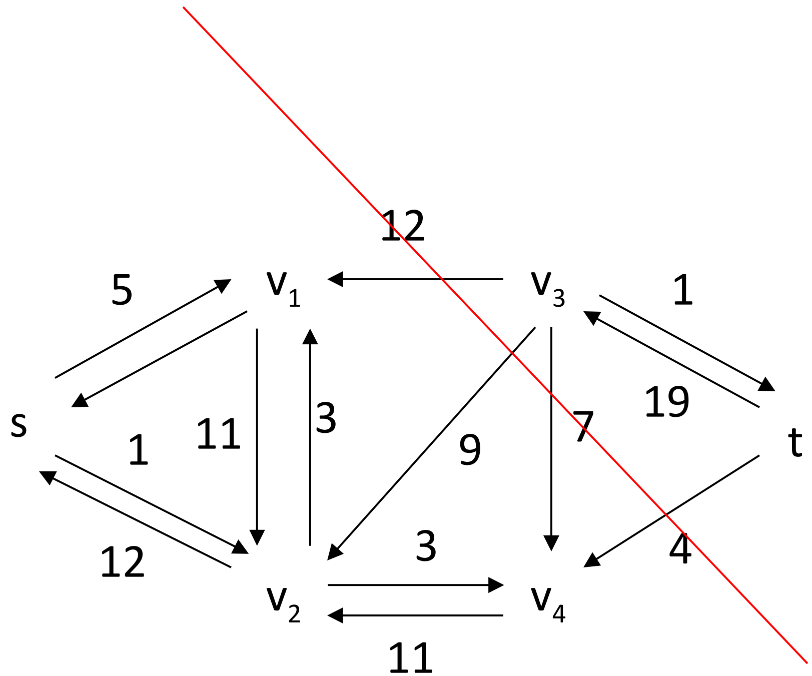


8
5
v1
v3
13
4
11
4
s
3
11
7
7
t
5
13
3
v2
v4
4
11
12/12
11/16
v1
v3
15/20
s
10
1/4
4/9
7/7
t
8/13
4/4
v2
v4
11/14

(c)


12
5
v1
v3
5
11
4
s
11
3
7
15
t
5
5
8
3
v2
v4
4
11
12/12
11/16
v1
v3
19/20
s
10
1/4
9
7/7
t
12/13
4/4
v2
v4
11/14

(d)



(e)

Time complexity:

- If each c(e) is an *integer*, then time complexity is O(|E|f*), where f* is the maximum flow.
- Reason: each time the flow is increased by at least one.
- This might not be a polynomial time algorithm since f* can be represented by log (f*) bits. So, the input size might be log(f*).

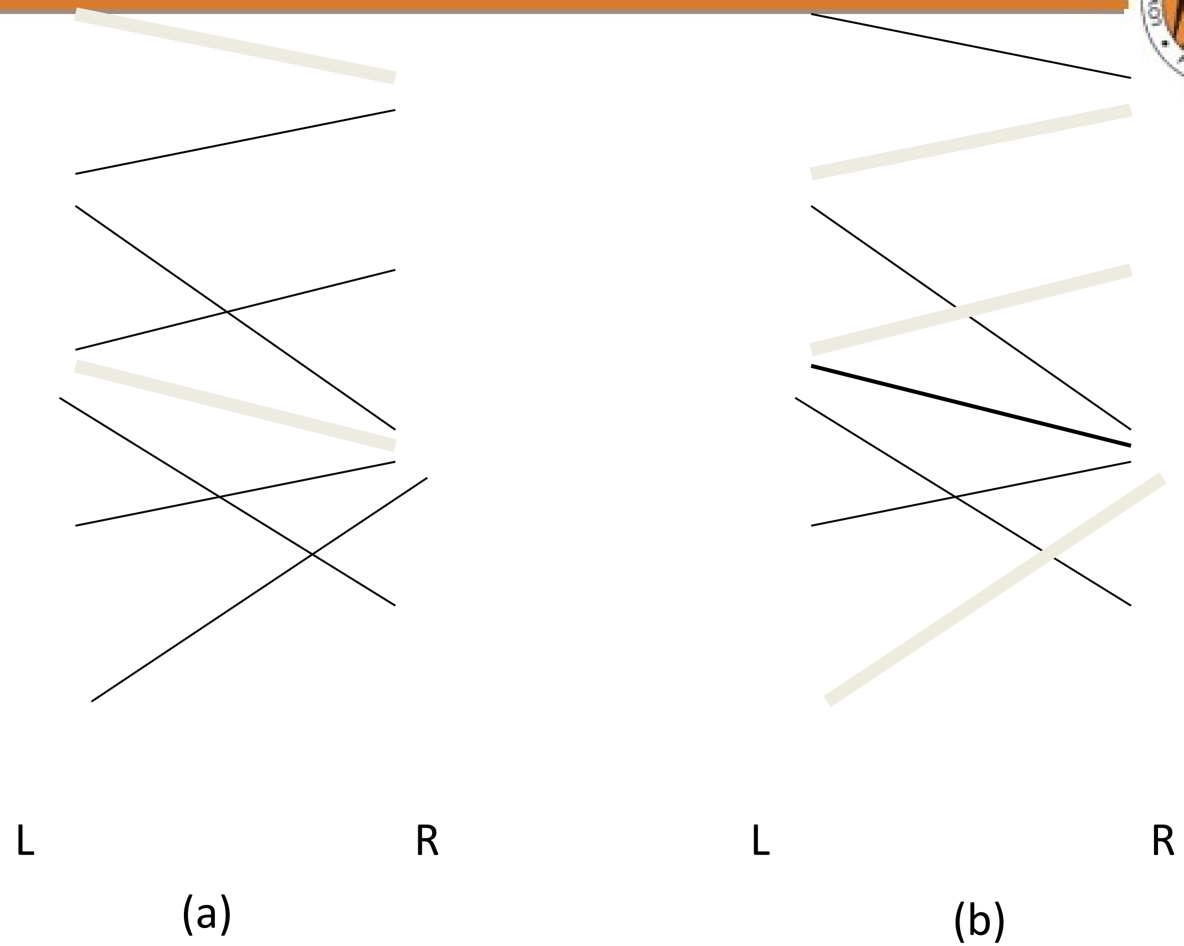
The Edmonds-Karp algorithm

- Find the augmenting path using breadth-first search.
- Breadth-first search gives the shortest path for graphs (Assuming the length of each edge is 1.)
- Time complexity of Edmonds-Karp algorithm is $O(VE^2)$.
- The proof is very hard and is not required here.

Maximum bipartite matching:

- Bipartite graph: a graph (V, E), where $V=L\cup R$, $L\cap R$=empty, and for every $(u, v)\in E$, $u \in L$ and $v \in R$.
- Given an undirected graph G=(V,E), a matching is a subset of edges $M\subseteq E$ such that for all vertices $v\in V$,at most one edge of M is incident on v.We say that a vertex $v \in V$ is matched by matching M if some edge in M is incident on v;otherwise, v is unmatched. A maximum matching is a matching of maximum cardinality,that is, a matching M such that for any matching M', we have .

$$|M| \geq |M'|$$

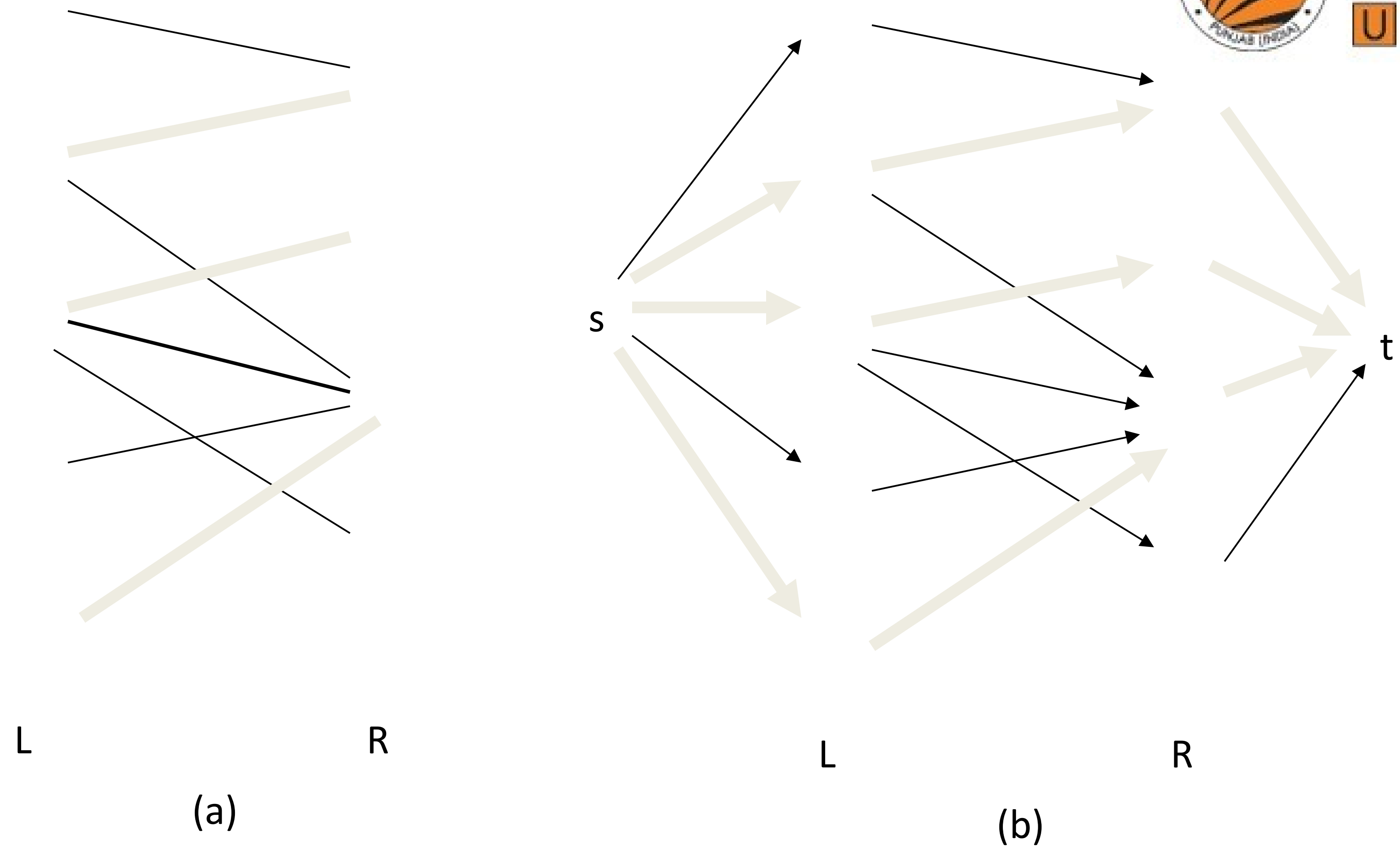


A bipartite graph G=(V,E) with vertex partition V=L∪R.(a)A matching with cardinality 2.(b) A maximum matching with cardinality 3.

Finding a maximum bipartite matching:

- We define the corresponding flow network $G'=(V',E')$ for the bipartite graph G as follows. Let the source s and sink t be new vertices not in V, and let $V'=V\cup\{s,t\}$.If the vertex partition of G is $V=L\cup R$, the directed edges of G' are given by $E'=\{(s,u):u\in L\}\cup\{(u,v):u\in L, v\in R, \text{and } (u,v)\in E\}\cup\{(v,t):v\in R\}$.Finally, we assign unit capacity to each edge in E'.

- We will show that a matching in G corresponds directly to a flow in G's corresponding flow network G'. We say that a flow f on a flow network $G=(V,E)$ is integer-valued if $f(u,v)$ is an integer for all $(u,v)\in V*V$.

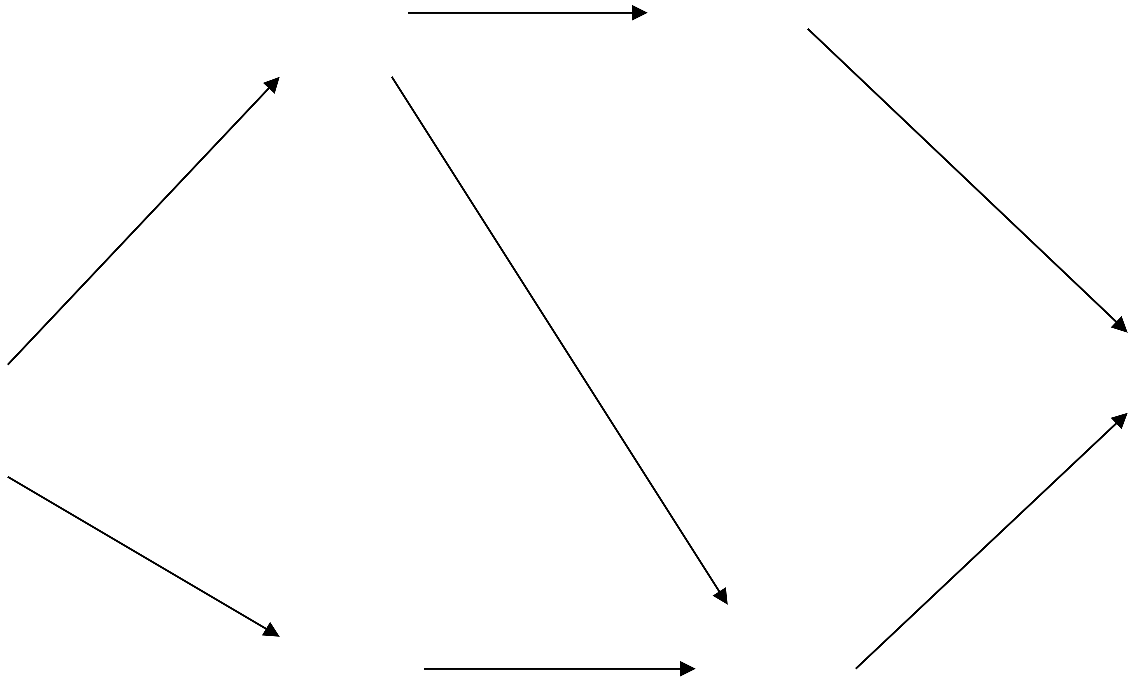


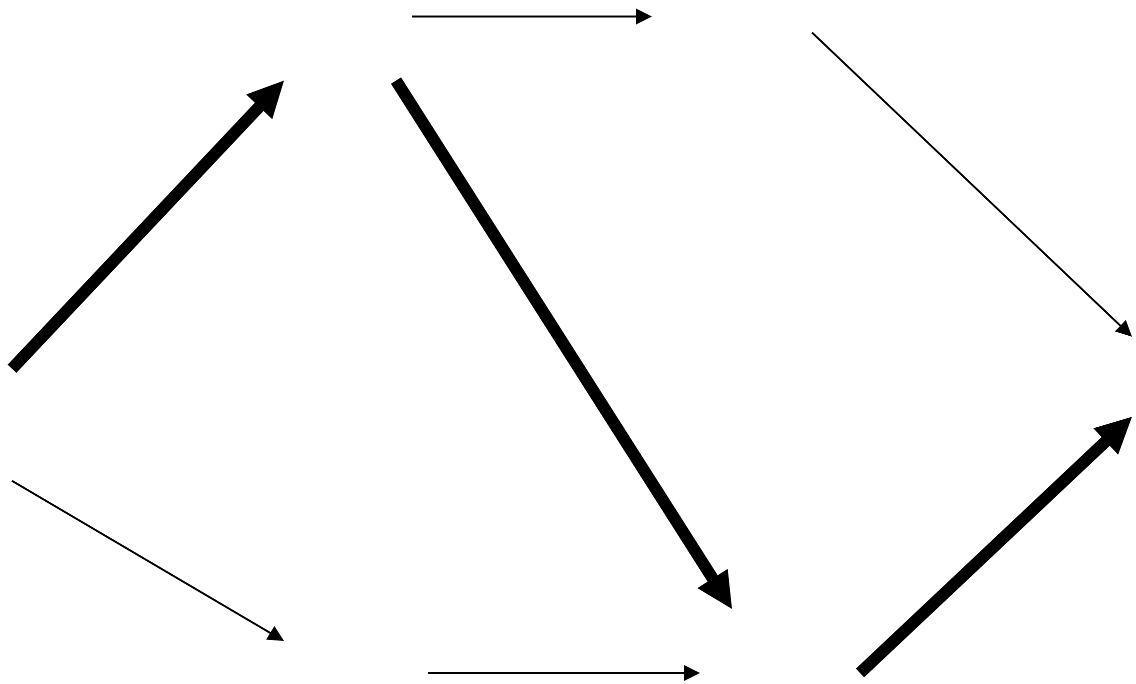
(a)The bipartite graph G=(V,E) with vertex partition V=L∪R. A maximum matching is shown by shaded edges.(b) The corresponding flow network.Each edge has unit capacity.Shaded edges have a flow of 1,and all other edges carry no flow.

Continue:

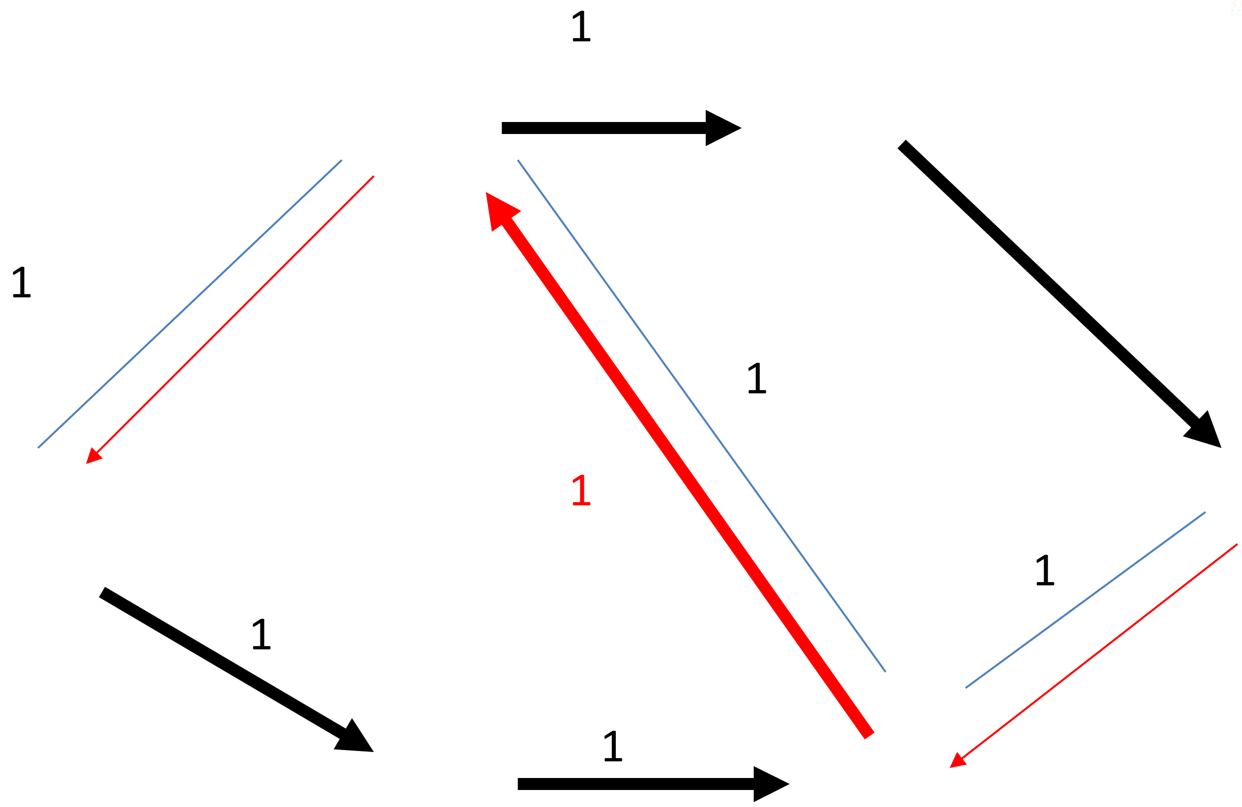
- Lemma .
- Let G=(V,E) be a bipartite graph with vertex partition V=L∪R,and let G’=(V’,E’) be its corresponding flow network.If M is a matching in G, then there is an integer-valued flow f in G’ with value $|M| = |f|$.Conversely, if f is an integer-valued flow in G’,then there is a matching M in G with cardinality $|M| = |f|$.
- Reason: The edges incident to s and t ensures this.
  - Each node in the first column has in-degree 1
  - Each node in the second column has out-degree 1.
  - So each node in the bipartite graph can be involved once in the flow.

Example:





Aug. path:



Residual network. Red edges are new edges in the residual network.
The new aug. path is bold. Green edges are old aug. path. old flow=1.

Thank You !!!