SAPIENZA
UNIVERSITÀ DI ROMA

"Sapienza" University of Rome
Faculty of Information Engineering,
Informatics and Statistics
Department of Computer Science

# Optimization

Lecture notes integrated with the book "Combinatorial Optimization",
W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, A. Schrijver

*Author*
Simone Bianco

29 febbraio 2024

# Indice

# Information and Contacts

Personal notes and summaries collected as part of the *Optimization* course offered by the degree in Computer Science of the University of Rome "La Sapienza".

Further information and notes can be found at the following link:
**https://github.com/Exyss/university-notes**. Anyone can feel free to report inaccuracies, improvements or requests through the Issue system provided by GitHub itself or by contacting the author privately:


- Email: **bianco.simone@outlook.it**
- LinkedIn: **Simone Bianco**


The notes are constantly being updated, so please check if the changes have already been made in the most recent version.

**Suggested prerequisites:**

Preventive learning of material related to the *Algorithms 2* and *Calculus 1* courses is recommended

**Licence:**

# 1

# Graph flows

## 1.1 Networks and flows

**Definition 1: Graph**

A **graph** is a mathematical structure $G = (V, E)$, where $V$ is the set of **vertices** in $G$ and $E \subseteq V \times V$ is the set of **edges** that link two vertices in $G$.

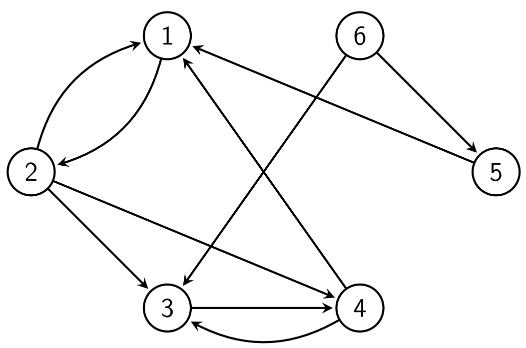
We will assume that each graph is *simple*, meaning that there are no multiple edges between the same nodes and no loops (that being an edge from a vertex to itself).

From now on, we will assume that $|V(G)| = n$ and $|E(G)| = m$.

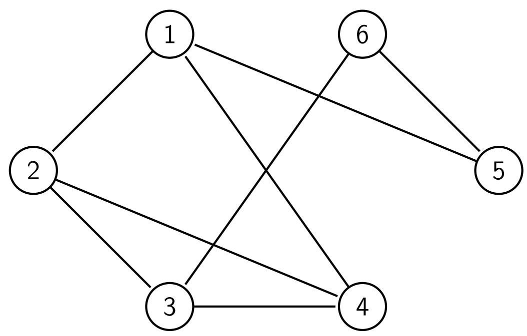
A graph can be **directed** or **undirected**. In a directed graph we consider the edges $(u, v) \in E(G)$ and $(v, u) \in E(G)$ as two different edges, while in an undirected graph they represent the same edge.

**Example:**

**Directed graph**




**Undirected graph**

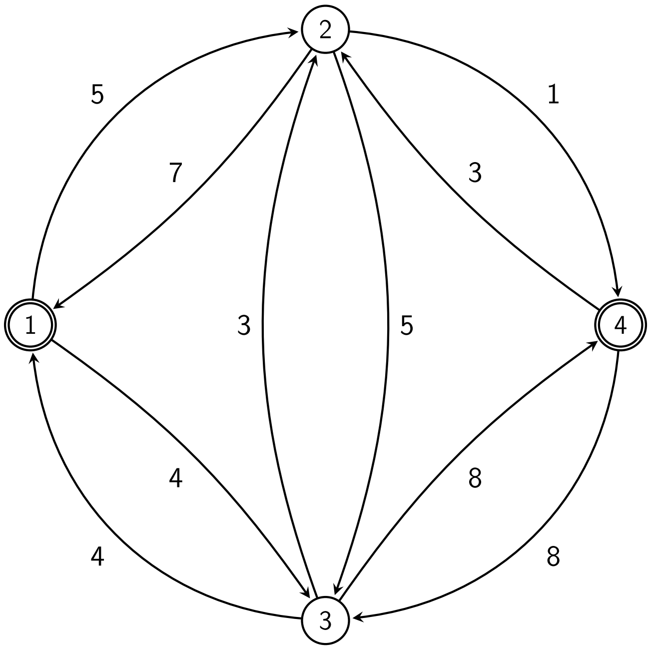


**Definition 2: Network**

A **network** is a mathematical structure $N = (G, s, t, c)$ where:

- $G = (V, E)$ is a directed graph
- $s$ and $t$ are two vertices of $G$ ($s, t \in V(G)$), respectively called the **source** and the **sink**
- $c : E(G) \to \mathbb{R}^+$ is a weight function on the edges called **capacity**
- $(u, v) \in E(G) \implies (v, u) \in E(G)$

**Example:**




*The numbers on the edges represent the capacities of the edges, while the nodes 1 and 4 are chosen respectively as the source and the sink of the network*

Essentially, a network effectively describes a *water system* made up of *pipes* (the edges) that can transport a maximum amount of fluid inside them (the capacity of the edges). In fact, the last property of a network defines the idea of a bi-directional *flow of water* inside the pipes. In particular, we note that each pipe can have a different capacity based on the direction of the flow.

**Definition 3: Flow**

Given a network $N = (G, s, t, c)$, a **flow** is a weight function $f : E(G) \to \mathbb{R}$ on the edges defined by the following properties:

- **Skew-symmetric**: $\forall (u, v) \in E(G) \quad f(u, v) = -f(v, u)$, meaning that the incoming flow in an edge is the inverse of the outgoing flow in the corresponding opposite edge
- **Capacity bounded**: $\forall (u, v) \in E(G) \quad f(u, v) \leq c(u, v)$, meaning that the flow can't be greater than the supported capacity
- **Conservation of flow**: $\forall v \in V(G) - \{s, t\}$ it holds that

$$\sum_{\substack{(u,v)\in E(G): \\ f(u,v)>0}} f(u, v) = \sum_{\substack{(v,w)\in E(G): \\ f(v,w)>0}} f(v, w)$$

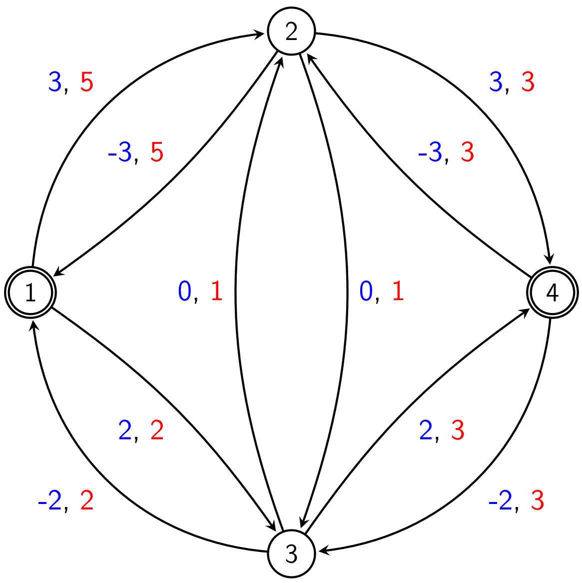
  meaning that the incoming flow in $v$ is the same as the outgoing flow from $v$

**Definition 4: Flow value**

Given a network $N = (G, s, t, c)$ and a flow $f$ on $G$, we define the **value of** $f$, noted by $\mathrm{Val}(f)$, as the sum of the flow outgoing from the source $s$ or the sum of the flow incoming to the sink $t$:

$$\mathrm{Val}(f) := \sum_{(s,u)\in E(G)} f(s, u) = \sum_{(v,t)\in E(G)} f(v, t)$$

**Example:**




*For each edge, the numbers in blue and red respectively represent its flow and its capacity. The flow value of the given flow is 5.*

**Observaation 1: Nullification of flow for middle edges**

Given a network $N = (G, s, t, c)$ and a flow $f$ defined on $G$, for each vertex different from $s$ and $t$ it holds that:

$$\sum_{(u,v)\in E(G)} f(u,v) = 0$$

*Proof.*

Due to the conservation of flow and the skew-symmetric properties, for all nodes $v \neq s, t$ we know that:

$$\sum_{\substack{(u,v)\in E(G):\\ f(u,v)>0}} f(u,v) = \sum_{\substack{(v,w)\in E(G):\\ f(v,w)>0}} f(v,w) = \sum_{\substack{(v,w)\in E(G):\\ f(v,w)>0}} -f(w,v)$$

which implies that:

$$\sum_{\substack{(u,v)\in E(G):\\ f(u,v)>0}} f(u,v) = -\sum_{\substack{(v,w)\in E(G):\\ f(v,w)>0}} f(w,v) \implies$$

$$\sum_{\substack{(u,v)\in E(G):\\ f(u,v)>0}} f(u,v) + \sum_{\substack{(v,w)\in E(G):\\ f(v,w)>0}} f(w,v) = 0$$

Again, by the skew-symmetric property we get that:

$$\sum_{\substack{(u,v)\in E(G):\\ f(u,v)>0}} f(u,v) + \sum_{\substack{(v,w)\in E(G):\\ f(v,w)>0}} f(w,v) = 0 \implies$$

$$\sum_{\substack{(u,v)\in E(G):\\ f(u,v)>0}} f(u,v) + \sum_{\substack{(w,v)\in E(G):\\ f(w,v)<0}} f(w,v) = 0$$

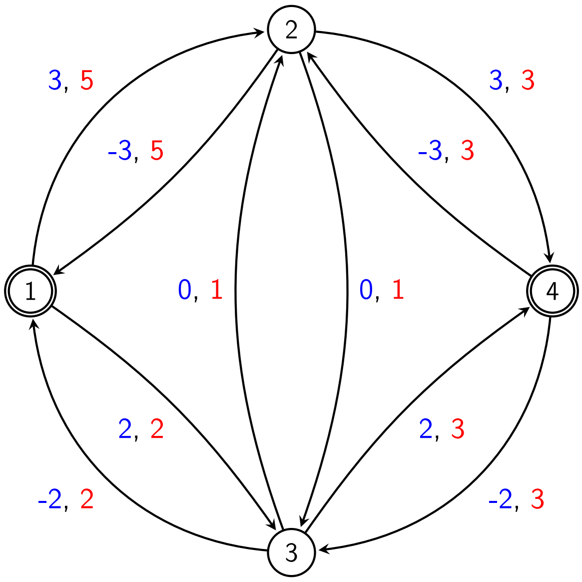
Then, by adding each edge incoming in $v$ that has no flow we conclude that:

$$\sum_{\substack{(u,v)\in E(G):\\ f(u,v)>0}} f(u,v) + \sum_{\substack{(w,v)\in E(G):\\ f(w,v)<0}} f(w,v) + \sum_{\substack{(x,v)\in E(G):\\ f(x,v)=0}} f(x,v) = 0 \implies \sum_{(u,v)\in E(G)} f(u,v) = 0$$

□

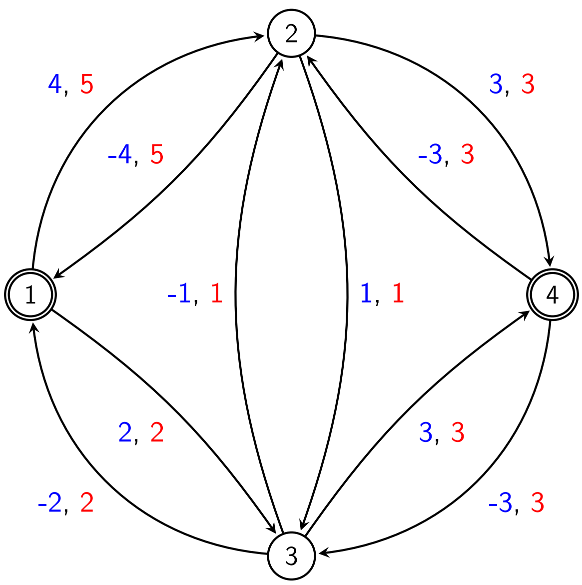
## 1.2 Residual graphs, flow increase and $s - t$ cuts

Consider the network shown in the last example of the previous section.




We notice that some *pipes* aren't completely *"filled up"*, meaning that their capacity could support a bigger flow. In particular, due to conservation of flow, not all pipes can be filled to the maximum capacity. In fact, we know that flow outgoing from the source must still be equal to the incoming flow of the sink.

Thus, we can increase the flow value by 1 only by using the remaining capacities in the path $1 \to 2 \to 3 \to 4$




*The flow value of the new flow is 6.*

**Definition 5: Residual capacity**

Given a network $N = (G, s, t, c)$ and a flow $f$ on $G$, the **residual capacity** is a weight function $r : E(G) \to \mathbb{R}^+$ defined as:

$$r(u, v) = c(u, v) - f(u, v)$$

**Observaation 2**

Given a network $N = (G, s, t, c)$ and a flow $f$ on $G$, we note that:

$$r(u, v) + r(v, u) = c(u, v) + c(v, u)$$

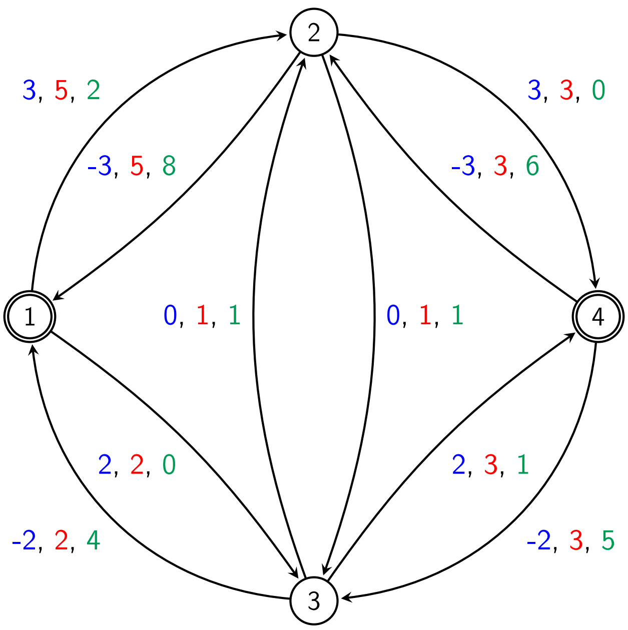
**Definition 6: Residual graph**

Given a network $N = (G, s, t, c)$ and a flow $f$ on $G$, we define $R \subseteq G$ as the **residual graph** of $G$ if:

$$(u, v) \in E(R) \iff r(u, v) > 0$$

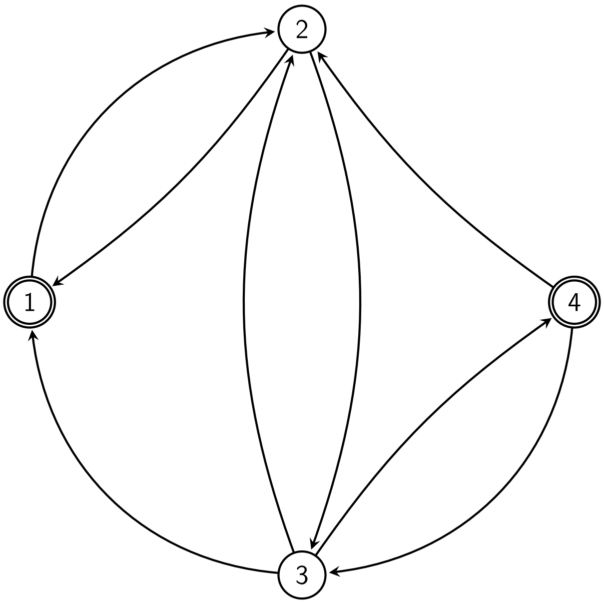
**Example:**

Consider the first graph previously shown with flow value 5. We now add the residual capacities obtained through that flow.




*For each edge, the number in green represents its residual capacity*

Thus, the residual graph is the following:




**Proposition 1: Flow-augmenting path**

Given a network $N=(G,s,t,c)$ and a flow $f$ on $G$, let $R \subseteq G$ be the residual graph of $G$ on $f$ and let $P$ be a direct path $s \to t$ in $R$.

Given $\alpha := \min\limits_{(u,v)\in E(P)} r(u,v)$, we define $f' : E(G) \to R$ as:

$$f'(u,v) = \begin{cases} f(u,v)+\alpha & \text{if } (u,v) \in E(P) \\ f(u,v)-\alpha & \text{if } (v,u) \in E(P) \\ f(u,v) & \text{otherwise} \end{cases}$$

The function $f'$ is a flow for $N$ such that $\mathrm{Val}(f') = \mathrm{Val}(f) + \alpha$.

*Proof.*

Suppose that $(u,v) \in E(P)$:

- By using the skew-symmetric property of $f$, we get that:

  $$f'(u,v) = f(u,v) + \alpha \implies -f'(u,v) = -f(u,v) - \alpha = f(v,u) - \alpha$$

  Also, since $(u,v) \in E(P)$, for $(v,u)$ we get that

  $$f'(v,u) = f(v,u) - \alpha = -f'(u,v)$$

- By assumption, we know that $f'(u,v) = f(u,v) + \alpha$. Thus, by definition of $\alpha$ we get that:

  $$\alpha \le r(u,v) = c(u,v) - f(u,v) \implies$$
  $$f'(u,v) = f(u,v) + \alpha \le f(u,v) + c(u,v) - f(u,v) = c(u,v)$$

If $(v,u) \in E(P)$, we can get the same result by repeating the same steps. Otherwise, the flow remains unchanged, meaning that the property is already satisfied. Thus, we conclude that $f'$ is *skew-symmetric* and *capacity bounded*.

Consider now a vertex $x \in V(G) - \{s,t\}$. If $x$ is not in the path $P$, the flow on each of its incoming and outgoing edges remains unchanged. Otherwise, if $x$ is in the path $P$, we get that:

$$\sum_{\substack{(u,x)\in E(G): \\ f'(u,x)>0}} f'(u,x) = \sum_{\substack{(u,x)\in E(G): \\ f'(u,x)>0}} (f(u,x)+\alpha)$$

$$\sum_{\substack{(x,v)\in E(G): \\ f'(x,v)>0}} f'(x,v) = \sum_{\substack{(x,v)\in E(G): \\ f'(x,v)>0}} (f(x,v)+\alpha)$$

$\square$

**Definition 7: $s-t$ cut**

Given a network $N = (G,s,t,c)$ and a flow $f$ on $G$, we define a **$\mathbf{s-t}$ cut of** $G$ as a subset $\mathcal{U} \subseteq V(G)$ that makes a partition on $V(G)$ such that $s \in \mathcal{U}, t \notin \mathcal{U}$.

Additionally, the vertices inside of $\mathcal{U}$ U are called the *s-part* of the cut, while the vertices outside of $\mathcal{U}$ are called the *t-part* of the cut.

**Observaation 3: Flow of an $s-t$ cut**

Given a network $N = (G,s,t,c)$, a flow $f$ on $G$ and an $s-t$ cut $\mathcal{U} \subseteq G$, we have that:

$$\text{Val}(f) = \sum_{\substack{(u,v)\in E(G): \\ u\in\mathcal{U}, v\notin\mathcal{U}}} f(u,v)$$

*Proof.*

Consider the total sum of the flows of all the vertices in $\mathcal{U}$:

$$\sum_{x\in\mathcal{U}} \sum_{(u,v)\in E(G)} f(u,v)$$

By definition, we know that $s \in \mathcal{U}$. We can separate the flows outgoing from $s$ from the rest of the flows:

$$\sum_{x\in\mathcal{U}} \sum_{(u,v)\in E(G)} f(u,v) = \sum_{(s,w)\in E(G)} f(s,w) + \sum_{x\in\mathcal{U}-\{s\}} \sum_{(u,v)\in E(G)} f(u,v)$$

Due to the Nullification of flow for middle edges, for all vertices $u, v \neq s$ we know that the total flow is qual to 0, implying that:

$$\sum_{(s,w)\in E(G)} f(s,w) + \sum_{x\in\mathcal{U}-\{s\}} \sum_{(u,v)\in E(G)} f(u,v) = \sum_{(s,w)\in E(G)} f(s,w) + 0 = \text{Val}(f)$$

We now consider again the initial total sum. We notice that:

$$\sum_{x\in\mathcal{U}}\sum_{(u,v)\in E(G)} f(u,v) = \sum_{\substack{(u,v)\in E(G):\\ u\in\mathcal{U}}} f(u,v)$$

Additionally, for any $(u,v)$ if $u,v\in\mathcal{U}$ then $f(u,v)$ cancels out with $f(v,u)$, implying that:

$$\sum_{\substack{(u,v)\in E(G):\\ u\in\mathcal{U}}} f(u,v) = \sum_{\substack{(u,v)\in E(G):\\ u\in\mathcal{U},v\notin\mathcal{U}}} f(u,v)$$

By combining the shown equalities, we conclude the proof.

□

**Definition 8: Capacity of an $s-t$ cut**

Given a network $N=(G,s,t,c)$, a flow $f$ on $G$ and an $s-t$ cut $\mathcal{U}\subseteq G$, we define the **capacity of an s − t cut on** $G$, noted by $c(V(G)\backslash\mathcal{U})$, as the sum of the capacities of the edges outgoing from *s-part* to the *t-part*.

$$c(V(G)\backslash\mathcal{U}) := \sum_{\substack{(u,v)\in E(G):\\ x\in\mathcal{U},u\notin\mathcal{U}}} c(u,v)$$

**Lemma 1**

Given a network $N=(G,s,t,c)$, the maximum value of a flow on $G$ at most the minimal capacity of an $s-t$ cut on $G$:

$$\max_{f\,:\,\text{flow on }G}(\mathrm{Val}(f)) \leq \min_{\mathcal{U}\,:\,s-t\text{ cut on }G}(c(V(G)\backslash\mathcal{U}))$$

*Proof.*

Let $f$ be the flow on $G$ that maximizes $\mathrm{Val}(f)$ and let $\mathcal{U}\subseteq G$ be the $s-t$ cut on $G$ that minimizes capacity. By the Flow of an $s-t$ cut and by the *capacity bounded* property of $f$, we get that:

$$\mathrm{Val}(f) = \sum_{\substack{(u,v)\in E(G):\\ u\in\mathcal{U},v\notin\mathcal{U}}} f(u,v) \leq \sum_{\substack{(u,v)\in E(G):\\ u\in\mathcal{U},v\notin\mathcal{U}}} c(u,v) = c(V(G)\backslash\mathcal{U})$$

□

## 1.3 The Ford-Fulkerson algorithm

**Algorithm 1: The Ford-Fulkerson algorithm**

Given a network $N = (G, s, t, c)$, we define the following algorithm:

```
function FordFulkerson(G)
    Start with the trivial flow f with all 0s
    while True do
        Compute the residual graph R ⊆ G on flow f
        Find a path P in R from s → t
        if P does not exist then
            Return f
        else
            Increase f through the value α obtained by P
        end if
    end while
end function
```

We note that the $\mathsf{FordFulkerson}(G)$ terminates only if the augment value eventually becomes 0, implying that there is no flow-augmenting path to be used. Thus, if the capacities defined by $c$ are all integers, the algorithm always terminates.

Moreover, the flow-augmenting path of each iteration of $\mathsf{FordFulkerson}(G)$ can be found with a simple BFS search, requiring $O(n+m)$, which is also the cost needed for computing the residual graph of each iteration. Thus, the computational complexity of the algorithm is $O(k(n+m))$, where $k$ is the maximum number of iterations of the while loop.

It's easy to notice that this value $k$ depends too much on the shape of the graph $G$. However, due to the *capacity bounded* property, in the worst case we have that $k$ equals the maximum flow value.

**Lemma 2**

Given a network $N = (G, s, t, c)$, if the algorithm $\mathsf{FordFulkerson}(G)$ terminates then it returns a flow $f$ such that there exists an $s - t$ cut $\mathcal{U} \subseteq G$ for which we have that $\text{Val}(f) = c(V(G) \setminus \mathcal{U})$

*Proof.*

Let $R$ be the residual graph of $G$ on $f = \mathsf{FordFulkerson}(G)$ and let $r$ be the residual capacity function obtained through $f$. We define $\mathcal{U} \subseteq G$ as:

$$\mathcal{U} = \{x \in V(G) \mid \exists s \to x \text{ in } G'\}$$

Since the algorithm terminates when there is no path from $s \to t$, we know that $t \notin \mathcal{U}$, implying that $\mathcal{U}$ is an $s-t$ cut of $G$. Thus, by the Flow of an $s-t$ cut, we have that:

$$\mathrm{Val}(f) = \sum_{\substack{(x,v)\in E(G):\\ x\in\mathcal{U}, v\notin\mathcal{U}}} f(x,u)$$

By way of contradiction, we suppose that $\exists (x,v) \in E(G') \subseteq E(G)$ such that $x \in \mathcal{U}, v \notin \mathcal{U}$. By definition of $\mathcal{U}$, we know that $s \to x$ in $G'$, so by adding $(x,v)$ to the path we get that $s \to x \to y$, which contradicts $y \notin \mathcal{U}$.

Thus, such edge can't exists, implying that $\forall (x,v) \in E(G)$ such that $x \in \mathcal{U}, v \notin \mathcal{U}$ it holds that $(x,v) \notin E(G')$, which by definition of residue graph implies that:

$$\forall (x,v) \in E(G) \text{ s.t. } x \in \mathcal{U}, v \notin \mathcal{U} \;\; f(x,v) = c(x,v)$$

concluding that:

$$\mathrm{Val}(f) = \sum_{\substack{(x,v)\in E(G):\\ x\in\mathcal{U}, v\notin\mathcal{U}}} f(x,u) = \sum_{\substack{(x,v)\in E(G):\\ x\in\mathcal{U}, v\notin\mathcal{U}}} c(x,u) = c(V(G)\backslash\mathcal{U})$$

□

### 1.3.1 The min flow, max flow theorem

**Theorem 1: Min flow, max flow theorem**

Given a network $N = (G, s, t, c)$, the maximum value of a flow on $G$ at most the minimal capacity of an $s-t$ cut on $G$:

$$\max_{f\,:\,\text{flow on } G}(\mathrm{Val}(f)) = \min_{\mathcal{U}\,:\,s-t \text{ cut on } G}(c(V(G)\backslash\mathcal{U}))$$

*Proof.*

By the Lemma 1, we already know that:

$$\max_{f\,:\,\text{flow on } G}(\mathrm{Val}(f)) \leq \min_{\mathcal{U}\,:\,s-t \text{ cut on } G}(c(V(G)\backslash\mathcal{U}))$$

We now consider $f' = \mathsf{FordFulkerson}(G)$. By the Lemma 2, we know that there exists an $s-t$ cut $\mathcal{U}' \subseteq G$ such that $\mathrm{Val}(f') = c(V(G)\backslash\mathcal{U}')$.

Thus, it's easy to conclude that:

$$\max_{f\,:\,\text{flow on } G}(\mathrm{Val}(f)) \geq \mathrm{Val}(f') = c(V(G)\backslash\mathcal{U}) \geq \min_{\mathcal{U}\,:\,s-t \text{ cut on } G}(c(V(G)\backslash\mathcal{U}))$$

□

**Corollary 1: Optimality of Ford-Fulkerson**

The Ford-Fulkerson algorithm returns a flow with **maximum value**