Lecture Notes - Week III

Flows

Fernando Dias, Philine Schiewe and Piyalee Pattanaik

January 8, 2024

CHAPTER 1
Flows and Cuts

In **graph theory**, **flow network** is a **directed** graph $G = (V, E)$ where each edge has a **capacity** $u\colon E \to \mathbb{R}_+$ and each edge receives a **flow** $f\colon E \to \mathbb{R}_+$, where the amount of flow allowed in each edge cannot surpass its capacity ($f(e) \leq u(e)$, $e \in E$). Hence, the *excess* of a flow f at $v \in V$:

$$\mathrm{ex}_f(v) := \sum_{e\in\delta^-(v)} f(e) - \sum_{e\in\delta^+(v)} f(e)$$

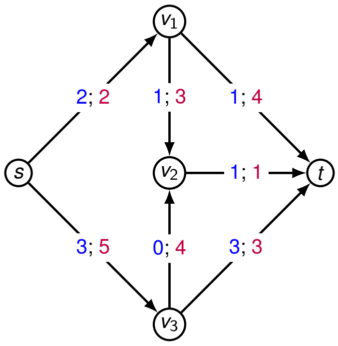
$\delta^-(v) = \{e \in E\colon e = (u, v)\}$ incoming edges

$\delta^+(v) = \{e \in E\colon e = (v, u)\}$ outgoing edges

The flow in this type of graph also have the satisfy **flow conservation** which state that:

Definition 1 *The total net flow entering a node v is zero for **all nodes** in the network except the source s and sink t.*

This can be also expressed based on the vale of flow through through a node. If f satisfies *flow conversation rule* at v, then $\mathrm{ex}_f(v) = 0$. When **all nodes** satisfy flow conservation $\mathrm{ex}_f(v) = 0$ for all $v \in V$, we express such behaviour as *circulation*. Finally, in a path between the source s and the sink t, the *s-t-flow*: $\mathrm{ex}_f(s) \leq 0$, $\mathrm{ex}_f(v) = 0$ for all $v \in V \setminus \{s, t\}$, in which the *value of s-t-flow* can be calculated as $\mathrm{value}(f) = -\mathrm{ex}_f(s) = \mathrm{ex}_f(t)$.



value$(f) = 5$

A **cut** in graph theory corresponds to a **partition** of the nodes in a graph splitting them into **disjoint subsets**. For example, see Figure 1.1.

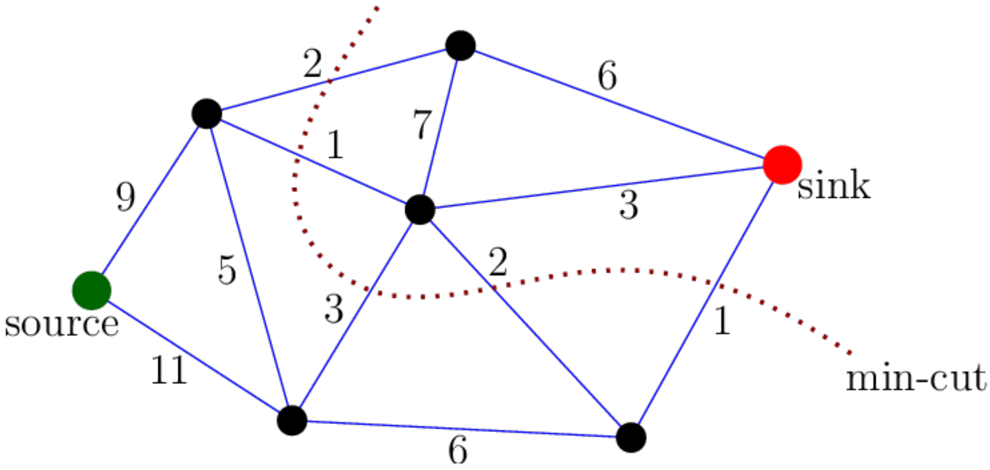


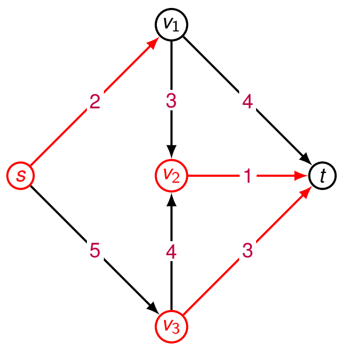
Figure 1.1: Example of a cut in a graph

A specific type of cut is a *s-t-cut* $\delta^+(S)$ where $S \subseteq V$ and $s \in S, t \notin S$. Therefore:

$$\delta^+(S) = \{e = (u, v) \in E \colon u \in S, v \in V \setminus S\}$$

The **capacity** of such cut can be expressed as:

$$u(\delta^+(S)) = \sum_{e \in \delta^+(S)} u(e)$$



capacity $u(\delta^+(\{s, v_2, v_3\})) = 6$

1.1 WEEK DUALITY

Using the definitions of flows and cuts, we can establish the following conclusion:

Lemma 1 *For any $S \subseteq V$ with $s \in S$, $t \notin S$ and any s-t-flow f:*

1. $\text{value}(f) = \sum_{e \in \delta^+(S)} f(e) - \sum_{e \in \delta^-(S)} f(e)$
2. $\text{value}(f) \leq u(\delta^+(S))$

Proof 1 *From the flow conservation for $v \in S \setminus \{s\}$:*

$$\begin{aligned}
\text{value}(f) &= -\text{ex}_f(s) \\
&= \sum_{e \in \delta^+(s)} f(e) - \sum_{e \in \delta^-(s)} f(e) \\
&= \sum_{v \in S} \Big(\sum_{e \in \delta^+(v)} f(e) - \sum_{e \in \delta^-(v)} f(e) \Big) \\
&= \sum_{e \in \delta^+(S)} f(e) - \sum_{e \in \delta^-(S)} f(e)
\end{aligned}$$

This can also expressed as:

$$0 \leq f(e) \leq u(e)$$

CHAPTER 2
Maximum Flows and Minimal Cuts

Once again, the task of find which flow and which cuts a graph can accept is **not challenging**. However, whenever **optimal values** (either minimal or maximal) are required, the configuration of such problems becomes challenging.

First, we state both problems:

Problem 1 *Maximum Flow Problem (MaxFlow) Given a flow network represent as a digraph $G = (v, E)$ with capacities u and unique source and unique sink s and t respectively, such that $s, t \in V$.*
*The goal is to find an s-t-flow of **maximum** value.*

Problem 2 *Minimum Cut Problem (MinCut) Given a flow network represent as a digraph $G = (v, E)$ with capacities u and unique source and unique sink s and t respectively, such that $s, t \in V$.*
*The goal is to find an s-t-cut of **minimum capacity**.*

Although those two problems might seem **unrelated** or even **contradictory**, they can be directly connected via the following lemmas:

Lemma 2 *Let $G = (V, E)$ be a digraph with capacities u and $s, t \in V$. Then*

$$\max\{\text{value}(f) \colon f \ s\text{-}t\text{-flow}\} \leq \min\{u(\delta^+(S)) \colon \delta^+(S) \ s\text{-}t\text{-cut}\}.$$

Lemma 3 *Let $G = (V, E)$ be a digraph with capacities u and $s, t \in V$. Let f be an s-t-flow and $\delta^+(S)$ be an s-t-cut. If*

$$\text{value}(f) = u(\delta^+(S))$$

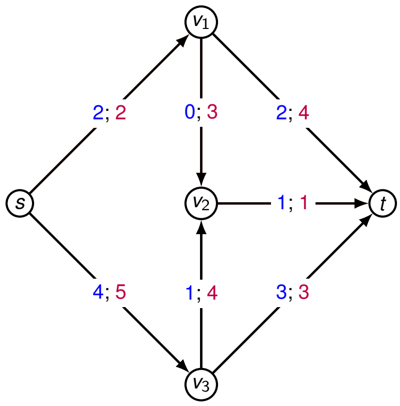
then f is a maximal flow and $\delta^+(S)$ is a minimal cut.

Hence, a single algorithm is enough to solve **both** problems.
Remark: in combinatorics, many problems can be expressed as another. This is a key point for future lectures.

2.1 IDEA FOR FINDING MAXIMAL FLOWS

If there exists non-saturated s-t-path ($f(e) < u(e)$ for all edges), then the flow f can be increased along this path. This means that if the path is not satured, more flow can be put into that path.

However, non-existence of such a path does not guarantee optimality.

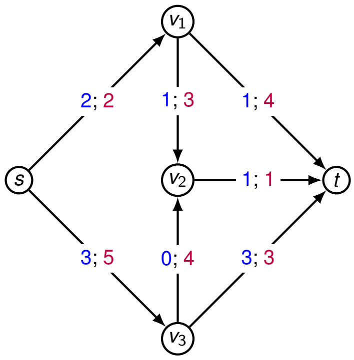


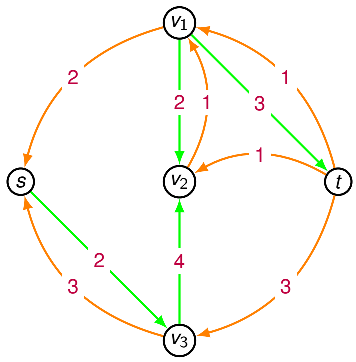
value$(f) = 4$ value$(f) = 5$ value$(f) = 6$
$u(\delta^+(\{s, v_2, v_3\})) = 6$

In this context, we introduced another concept: **residual graphs**. Considering that $G = (V, E)$ is a digraph with capacities u, f be an s-t-flow, a residual graph is the graph $G_f = (V, E_f)$ with $E_f = E_+ \cup E_-$ and capacity u_f:

- *forward edges* $+e \in E_+$:
for $e = (u, v) \in E$ with $f(e) < u(r)$, add $+e = (u, v)$ with *residual capacity* $u_f(+e) = u(e) - f(e)$
- *backward edges* $-e \in E_-$:
for $e = (u, v) \in E$ with $f(e) > 0$, add $-e = (v, u)$ with *residual capacity* $u_f(-e) = f(e)$

Remark: G_f can have parallel edges even if G is simple.



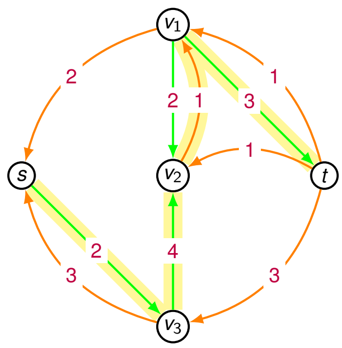


In addition, we can also define f**-augmenting paths**:

Definition 2 *An s-t-path P in G_f is called* augmenting path*. The value:*

$$\blacksquare_f(P) = \min_{a \in E(P)} u_f(a)$$

is called residual capacity *of P.*
Remark: $\blacksquare_f(P) > 0$ *as* $u_f(a) > 0$ *for all* $a \in E_f$*.*



$\blacksquare_f(P) = 1$

With this definition in mind, the following theorem is established.

Theorem 1 *An s-t-flow is optimal if and only if there exists no f-augmenting path.*

Proof idea:

$\Rightarrow$ P f-augmenting path. Construct s-t-flow

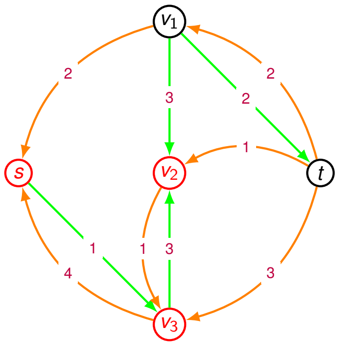
$$\bar{f}(e) = \begin{cases} f(e) + \blacksquare_f(P) & \text{if } +e \in E(P) \\ f(e) - \blacksquare_f(P) & \text{if } -e \in E(P) \\ f(e) & \text{otherwise} \end{cases}$$

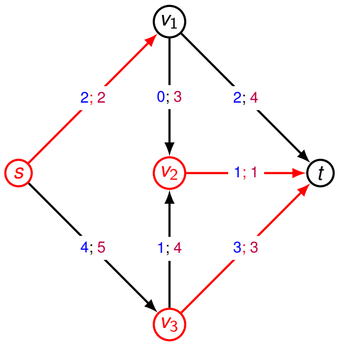
with higher value.

Proof idea:

$\Leftarrow$ There exists no f-augmenting path. Consider s-t-cut $\delta^+(S)$ defined by connected component S of s in G_f. Show that

$$\text{value}(f) = u(\delta^+(S)).$$





With this previous theorem in mind, we can conclude that:

Theorem 2 (Ford and Fulkerson, 1956; Dantzig and Fulkerson, 1956)
In a digraph G with capacities u, the maximum value of an s-t-flow equals the minimum capacity of an s-t-cut.

CHAPTER 3

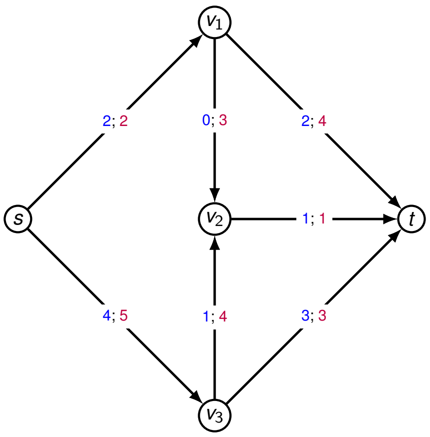
Finding Maximal Flows

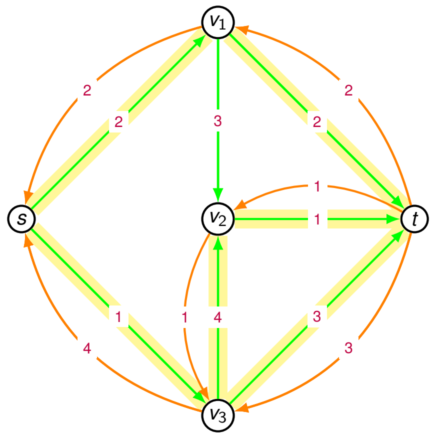
The most common algorithm for maximum flow was first published by L. R. Ford Jr. and D. R. Fulkerson in in 1956. It is commonly known as **Ford-Fulkerson algorithm**. The algorith is as follows:

Algorithm: FORD-FULKERSON ALGORITHM

Input: digraph $G = (V, E)$, capacities $u\colon E \to \mathbb{Z}_+$, $s, t, \in V$
Output: maximal s-t-flow f

```
1 set f(e) = 0 for all e ∈ E
2 while there exists f-augmenting path in G_f do
3     choose f-augmenting path P
4     set ■_f(P) = min_{a∈E(P)} u_f(a)
5     augment f along P by ■_f(P)
6     update G_f
7 return f
```





$\blacksquare_f(P) = 3$
$\blacksquare_f(P) = 2$
$\blacksquare_f(P) = 1$

Analysing the previous algorithms allow us to infer a few details. Lines 1, 4, 5 and 6 can be calculated in **linear time** in terms the number of edges m in a graph. An efficient algorithm to apply in Line 3 is actually **DFS** (Depth-First Search) which is also **linear** in the number of edges m. The *WHILE* loop requires up to $n \cdot U$, where n is the number of nodes and U is $max_{e \in E} u(e)$. The entire algorithm has a runtime proportional to $O(n \cdot m \cdot U)$ (**polynomial**).

Aalto University

Remark: flow f is integer.

An improved version of this algorithm allows for **real values in the capacities**. In this case, for non-integer capacities, $\blacksquare_f$ can be arbitrarily small when P is not chosen carefully , resulting in a runtime $O(n \cdot m^2)$.

The resulting algorithm represent such adaption:

Algorithm: EDMONDS-KARP ALGORITHM

Input: digraph $G = (V, E)$, capacities $u \colon E \to \mathbb{R}_+$, $s, t, \in V$
Output: maximal s-t-flow f

```
set f(e) = 0 for all e ∈ E
while there exists f-augmenting path in G_f do
    choose f-augmenting path P with minimal number of edges
    set ■_f(P) = min_{a∈E(P)} u_f(a)
    augment f along P by ■_f(P)
    update G_f
return f
```

Last but not least, there is also linear programming formulation for this problem. See full model below:

$$\max \quad \sum_{e \in \delta^+(s)} f_e \tag{3.1a}$$

$$\text{s.t.} \quad \sum_{e \in \delta^-(v)} f_e - \sum_{e \in \delta^+(v)} f_e = 0 \qquad v \in V \setminus \{s, t\} \tag{3.1b}$$

$$f_e \leq u(e) \qquad e \in E \tag{3.1c}$$

$$f_e \geq 0 \qquad e \in E \tag{3.1d}$$

The flow conservation flow conversation constraints (3.1b) are part of many LPs and IPs, e.g. for **shortest path**. The coefficient matrix of flow conversation constraints is **node-arc-incidence matrix** and it is **totally unimodular**, i.e., all extreme points are integer.