Lecture III - Flows

[1] Department of Mathematics and Systems Analysis,
Systems Analysis Laboratory, Aalto University, Finland

January 22, 2024

Aalto University

Previously on..

- Shortest Path: Dijkstra;
- Minimum Spanning Tree: Prim and Kruskal

PREVIOUSLY ON...

Flow

Aalto University

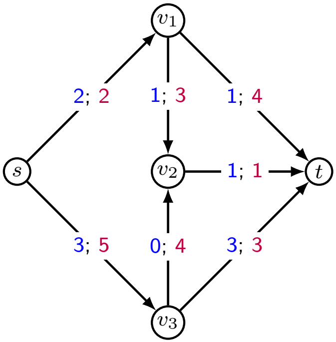
Combinatorial Optimization

- $G = (V, E)$ digraph with *capacities* $u \colon E \to \mathbb{R}_+$
- *flow* $f \colon E \to \mathbb{R}_+$ with $f(e) \leq u(e)$, $e \in E$
- *excess* of a flow f at $v \in V$:

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

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

- f satisfies *flow conversation rule* at v if $\text{ex}_f(v) = 0$
- *circulation*: $\text{ex}_f(v) = 0$ for all $v \in V$
- *s-t-flow*: $\text{ex}_f(s) \leq 0$, $\text{ex}_f(v) = 0$ for all $v \in V \setminus \{s, t\}$
- *value of s-t-flow*: $\text{value}(f) = -\text{ex}_f(s) = \text{ex}_f(t)$



$\text{value}(f) = 5$

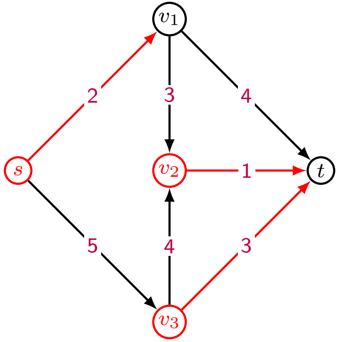
Cut

- $G = (V, E)$ digraph with *capacities* $u\colon E \to \mathbb{R}_+$
- *s-t-cut* $\delta^+(S)$: for $S \subseteq V$ with $s \in S, t \notin S$

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

- *capacity of an s-t-cut*:

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



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

Aalto University

Combinatorial Optimization

Weak duality

Lemma

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. flow conservation for $v \in S \setminus \{s\}$:

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

2. use $0 \leq f(e) \leq u(e)$

□

Aalto University

Combinatorial Optimization

Maximum Flow Problem (MaxFlow)

Instance: digraph $G = (v, E)$, capacities u, $s, t \in V$
Task: Find an s-t-flow of maximum value.

Minimum Cut Problem (MinCut)

Instance: digraph $G = (v, E)$, capacities u, $s, t \in V$
Task: Find an s-t-cut of minimum capacity.

Relationship between MaxFlow and MinCut

Lemma

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

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

Lemma

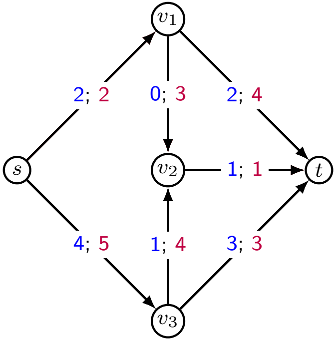
Let $G = (V, E)$ be a digraph with capacities u and $s, t \in V$. Let f be an s-t-flow and $\delta^+(S)$ be an s-t-cut. If

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

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

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.
- Non-existence of such a path does not guarantee optimality.



value$(f) = 4$ value$(f) = 5$

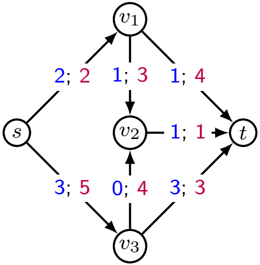
value$(f) = 6$

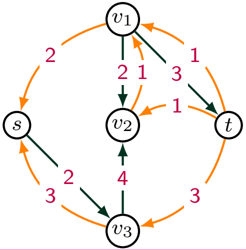
$u(\delta^+(\{s, v_2, v_3\})) = 6$

Residual Graph

- $G = (V, E)$ a digraph with capacities u, f be an s-t-flow
- residual 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* $u_f(-e) = f(e)$

❦ G_f can have parallel edges even if G is simple





f-augmenting paths

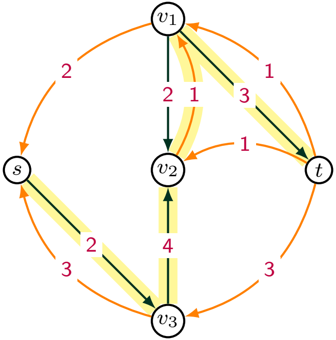
Definition

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

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

is called *residual capacity* of P.

☙ $\Delta_f(P) > 0$ as $u_f(a) > 0$ for all $a \in E_f$



$\Delta_f(P) = 1$

Augmenting path theorem

Aalto University

Combinatorial Optimization

Theorem

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

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

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

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

Augmenting path theorem

MaxFlow-MinCut theorem

Aalto University

Combinatorial Optimization

Theorem (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.

Ford-Fulkerson Algorithm

Aalto University

Combinatorial Optimization

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

set $f(e) = 0$ for all $e \in E$

while *there exists f-augmenting path in G_f* **do**

  choose f-augmenting path P

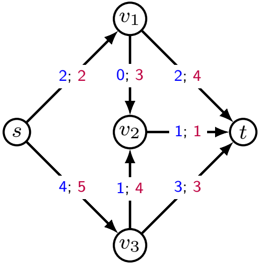
  set $\Delta_f(P) = \min_{a \in E(P)} u_f(a)$

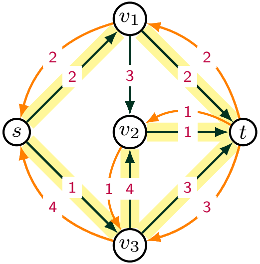
  augment f along P by $\Delta_f(P)$

  update G_f

return f

Ford-Fulkerson Algorithm





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

Aalto University

Combinatorial Optimization

Ford-Fulkerson Algorithm – Analysis

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 $e \in E$
2 **while** *there exists* f*-augmenting path in* G_f **do**
3 choose f-augmenting path P
4 set $\Delta_f(P) = \min_{a \in E(P)} u_f(a)$
5 augment f along P by $\Delta_f(P)$
6 update G_f
7 **return** f

- set $U = \max_{e \in E} u(e)$
- line 1, 5, 6: $O(m)$
- line 3: DFS $O(m)$
- line 4: $O(m)$, $\Delta_f(P) \in \mathbb{Z}_+$
- iterations while loop in line 2: $O(n \cdot U)$ (value$(f) \leq n \cdot U$)

$\Rightarrow$ $O(n \cdot m \cdot U)$

❦ flow f is integer

Edmonds-Karp Algorithm

Aalto University

Combinatorial Optimization

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 Δ_f(P) = min_{a∈E(P)} u_f(a)
    augment f along P by Δ_f(P)
    update G_f
return f
```

- for non-integer capacities, Δ_f can be arbitrarily small when P is not chosen carefully
- total runtime $O(n \cdot m^2)$

Linear programming formulation

$$
\begin{aligned}
\max \quad & \sum_{e \in \delta^+(s)} f_e \\
\text{s.t.} \quad & \sum_{e \in \delta^-(v)} f_e - \sum_{e \in \delta^+(v)} f_e = 0 \qquad v \in V \setminus \{s,t\} \\
& f_e \leq u(e) \qquad e \in E \\
& f_e \geq 0 \qquad e \in E
\end{aligned}
$$

- flow conversation constraints are part of many LPs and IPs, e.g. for TSP
- coefficient matrix of flow conversation constraints is node-arc-incidence matrix
- coefficient matrix is *totally unimodular*, i.e., all extreme points are integer

$\Rightarrow$ you can find integer solutions by linear programming


thank you