



Chapter 5

Graph Algorithms

Algorithm Theory
WS 2014/15

Fabian Kuhn

Graphs

Extremely important concept in computer science

Graph $G = (V, E)$

- V : **node** (or **vertex**) set
- $E \subseteq V^2$: **edge** set
 - Simple graph: no self-loops, no multiple edges
 - Undirected graph: we often think of edges as sets of size 2 (e.g., $\{u, v\}$)
 - Directed graph: edges are sometimes also called arcs (u, v)
 - Weighted graph: (positive) weight on edges (or nodes)
- (simple) path: sequence v_0, \dots, v_k of nodes such that

$$\underline{(v_i, v_{i+1})} \in E \text{ for all } i \in \{0, \dots, k - 1\}$$
- ...



Many real-world problems can be formulated as optimization problems on graphs

Graph Optimization: Examples

Minimum spanning tree (MST):

- Compute min. weight spanning tree of a weighted undir. Graph

Shortest paths:

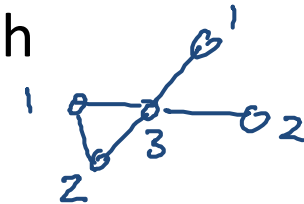
- Compute (length) of shortest paths (single source, all pairs, ...)

Traveling salesperson (TSP):

- Compute shortest TSP path/tour in weighted graph

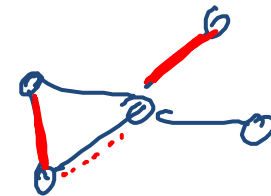
Vertex coloring:

- Color the nodes such that neighbors get different colors
- Goal: minimize the number of colors



Maximum matching:

- Matching: set of pair-wise non-adjacent edges
- Goal: maximize the size of the matching



Network Flow

Flow Network:

- Directed graph $G = (V, E)$, $E \subseteq V^2$
- Each (directed) edge e has a capacity $c_e \geq 0$
 - Amount of flow (traffic) that the edge can carry
- A single **source** node $s \in V$ and a single **sink** node $t \in V$

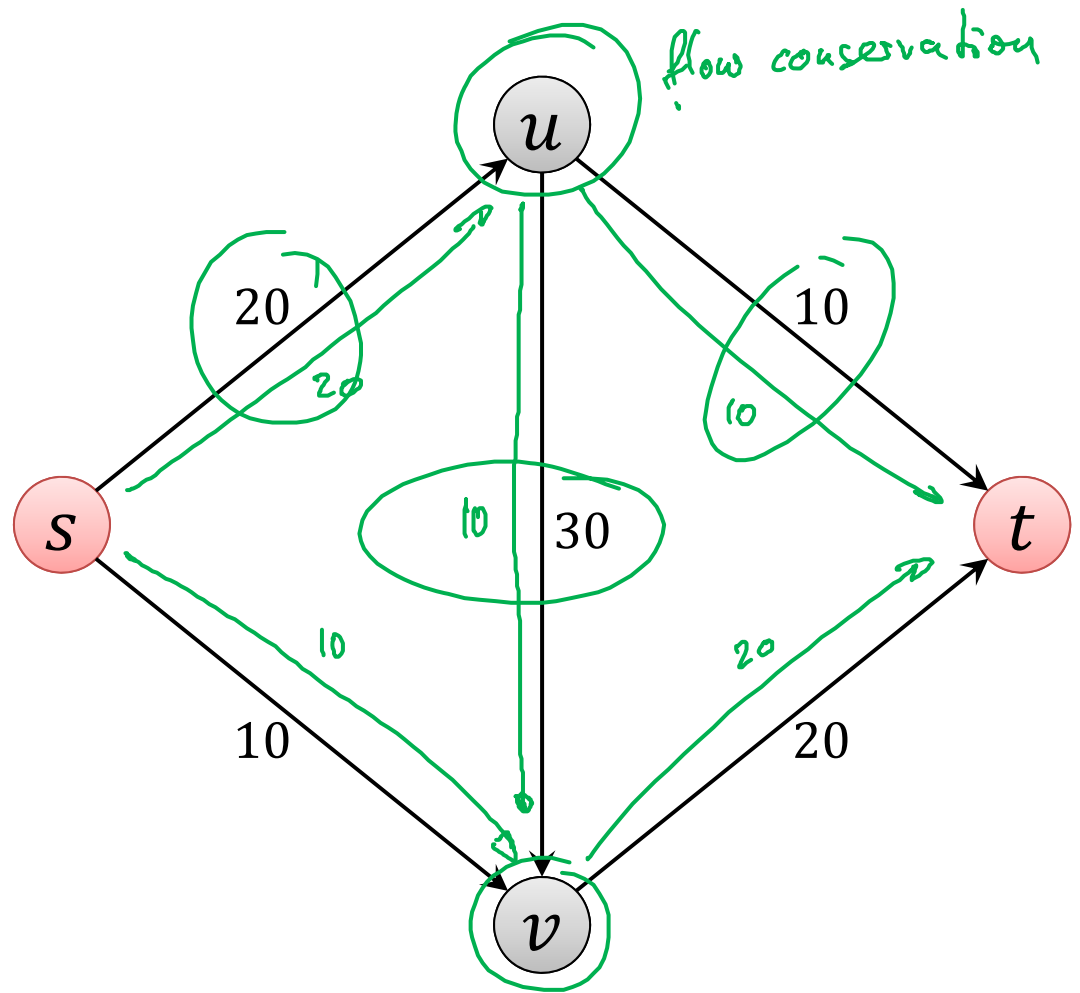
Flow: (informally)

- Traffic from s to t such that each edge carries at most its capacity

Examples:

- Highway system: edges are highways, flow is the traffic
- Computer network: edges are network links that can carry packets, nodes are switches
- Fluid network: edges are pipes that carry liquid

Example: Flow Network



Network Flow: Definition

Flow: function $f: E \rightarrow \mathbb{R}_{\geq 0}$

- $f(e)$ is the amount of flow carried by edge e $f(e) \geq 0$

Capacity Constraints:

- For each edge $e \in E$, $f(e) \leq c_e$

Flow Conservation:

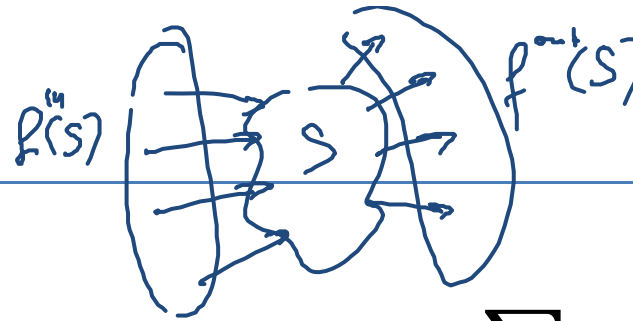
- For each node $v \in V \setminus \{s, t\}$,

$$\sum_{e \text{ into } v} f(e) = \sum_{e \text{ out of } v} f(e)$$

Flow Value:

$$|f| := \sum_{e \text{ out of } s} f((s, u)) = \sum_{e \text{ into } t} f((v, t))$$

Notation



We define:

$$\underline{f^{\text{in}}(v)} := \sum_{e \text{ into } v} f(e), \quad \underline{f^{\text{out}}(v)} := \sum_{e \text{ out of } v} f(e)$$

For a set $S \subseteq V$:

$$\underline{f^{\text{in}}(S)} := \sum_{e \text{ into } S} f(e), \quad \underline{f^{\text{out}}(S)} := \sum_{e \text{ out of } S} f(e)$$

Flow conservation: $\forall v \in V \setminus \{s, t\}: \underline{f^{\text{in}}(v)} = \underline{f^{\text{out}}(v)}$

Flow value: $\underline{|f|} = \underline{f^{\text{out}}(s)} = \underline{f^{\text{in}}(t)}$

For simplicity: Assume that all capacities are positive integers

The Maximum-Flow Problem

Maximum Flow:

Given a flow network, find a flow of maximum possible value

- Classical graph optimization problem
- Many applications (also beyond the obvious ones)
- Requires new algorithmic techniques

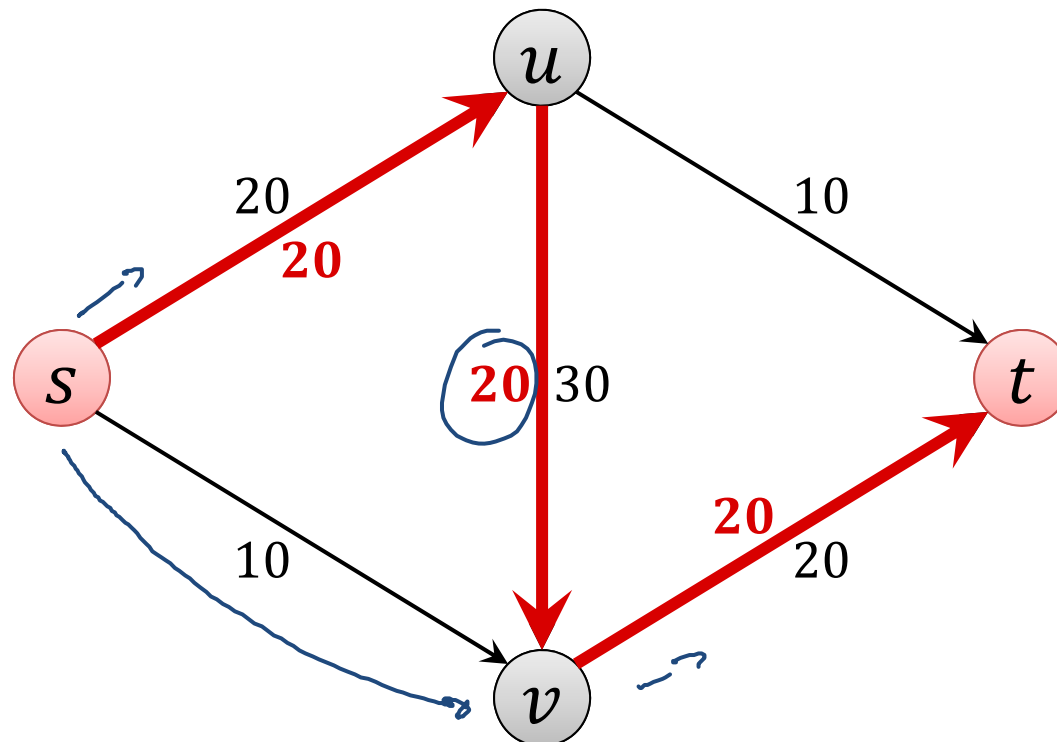
goal of lecture:
general alg. design / analysis techniques

Maximum Flow: Greedy?

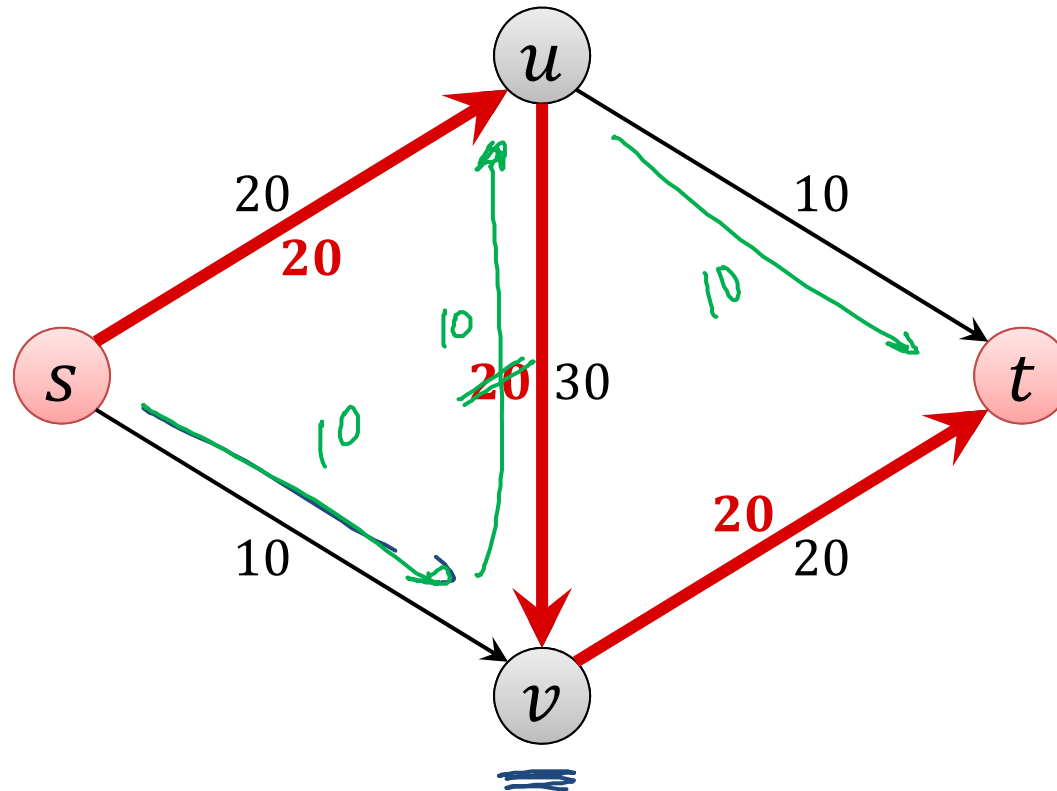
Does greedy work?

A natural greedy algorithm:

- As long as possible, find an s - t -path with free capacity and add as much flow as possible to the path



Improving the Greedy Solution



- Try to push 10 units of flow on edge (s, v)
- Too much incoming flow at v : reduce flow on edge (u, v)
- Add that flow on edge (u, t)

Residual Graph

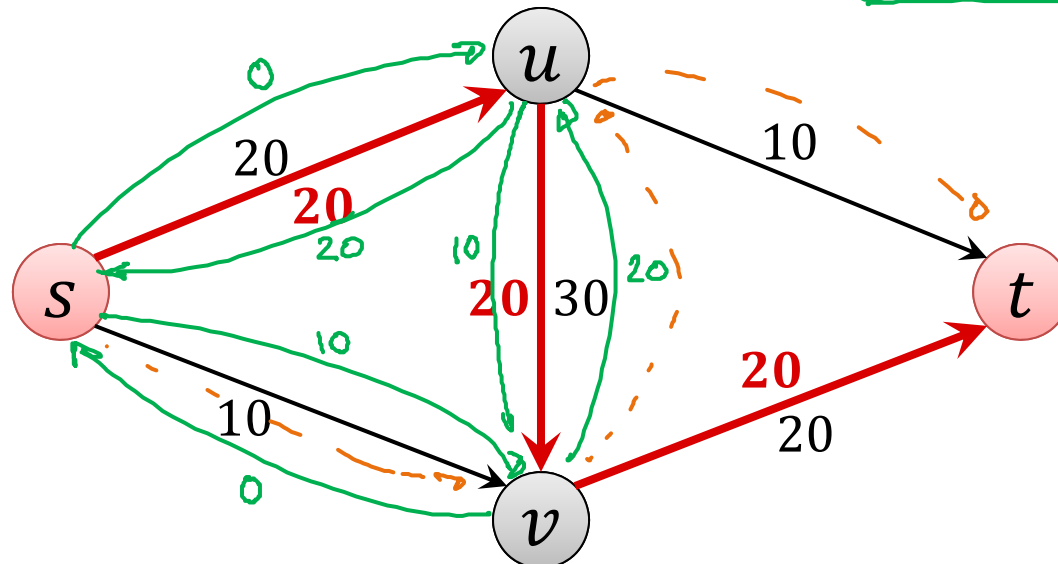
flow f : flow function satisfying
 $\forall e: 0 \leq f(e) \leq c_e, \forall v \neq s, t: f^{in}(v) = f^{out}(v)$



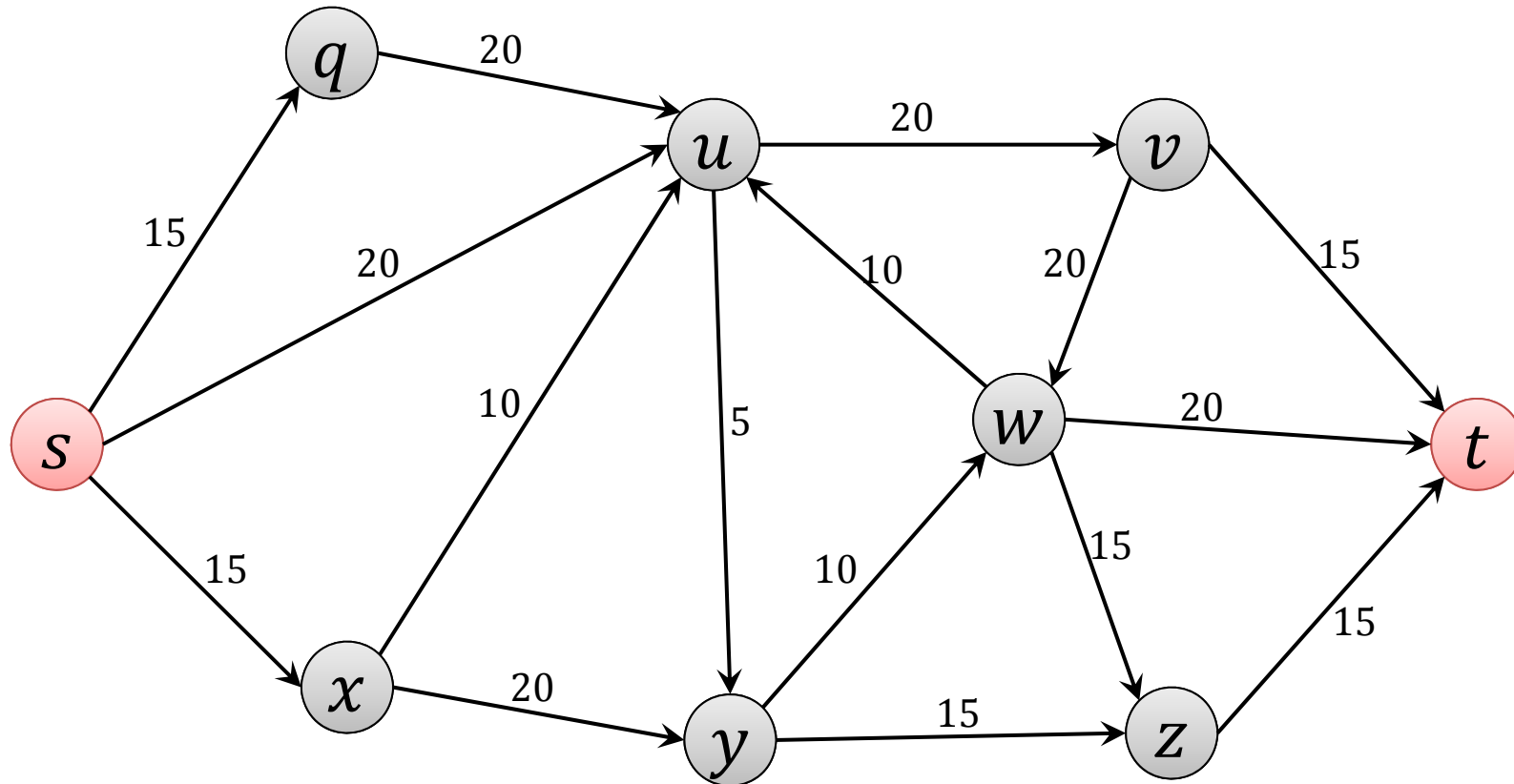
Given a flow network $G = (V, E)$ with capacities c_e (for $e \in E$)

For a flow f on G , define **directed graph** $G_f = (V_f, E_f)$ as follows:

- Node set $V_f = V$
- For each edge $e = (u, v)$ in E , there are two edges in E_f :
 - forward edge $e = (u, v)$ with residual capacity $c_e - f(e)$
 - backward edge $e' = (v, u)$ with residual capacity $f(e)$



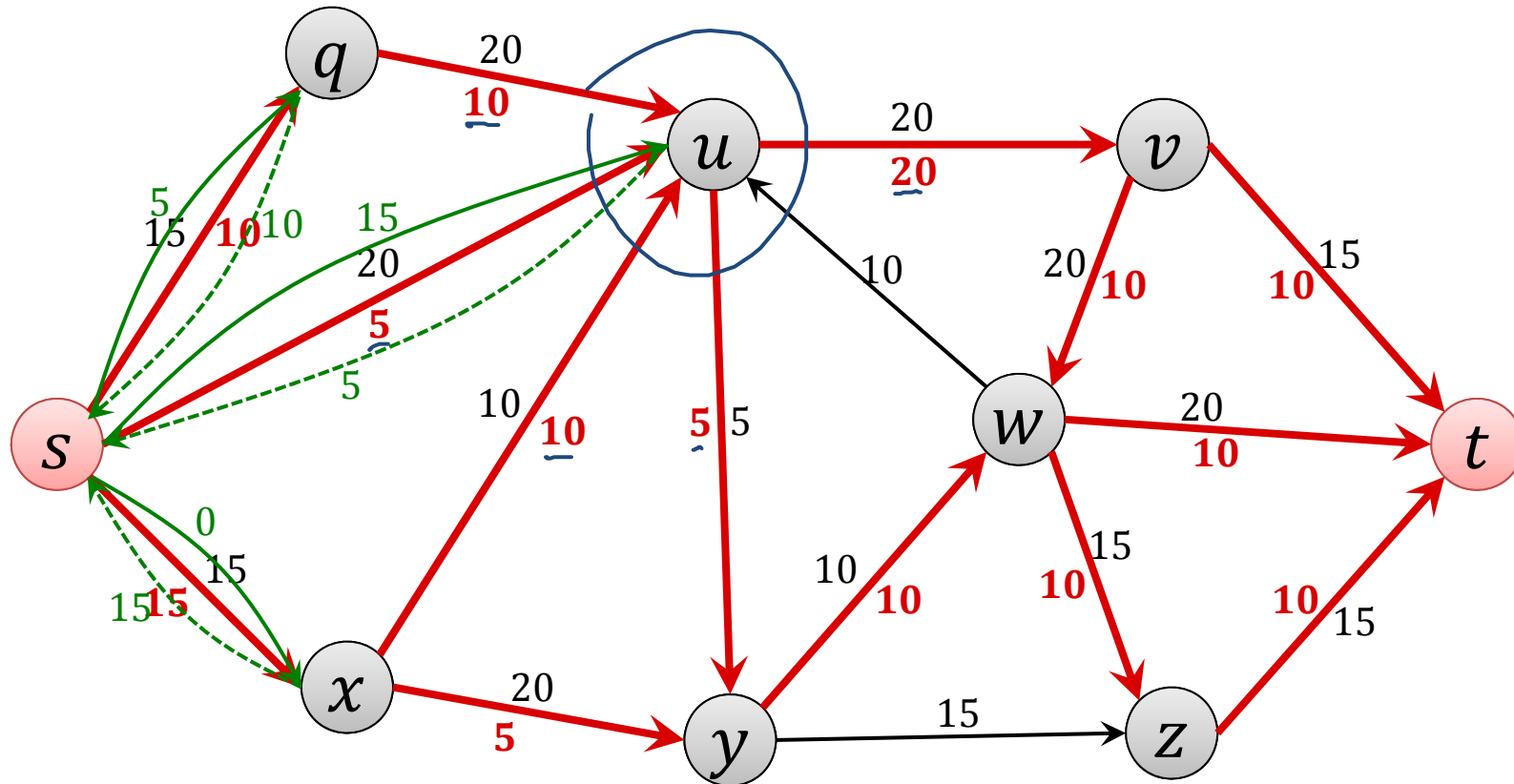
Residual Graph: Example



Residual Graph: Example

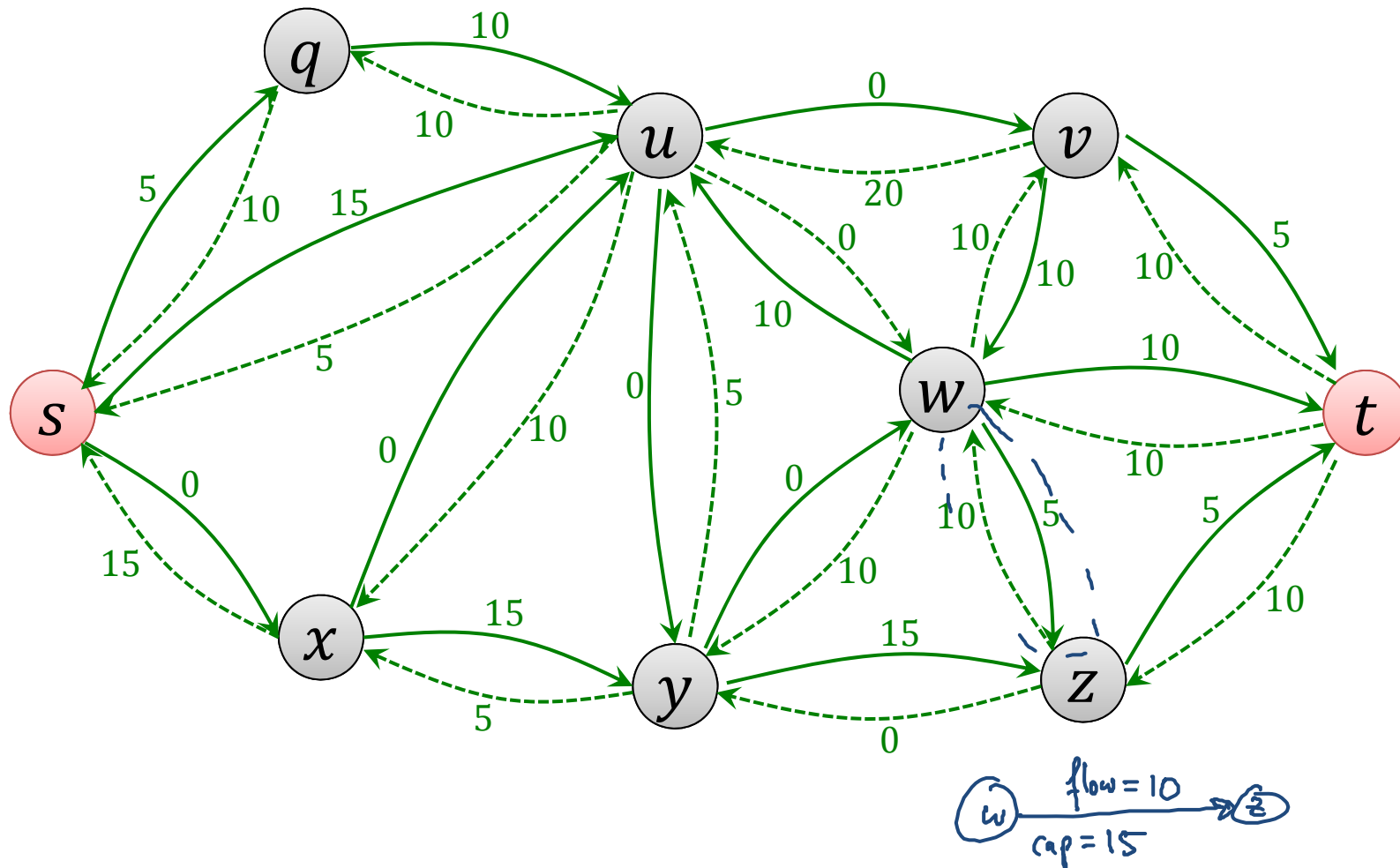
Flow f

flow conservation



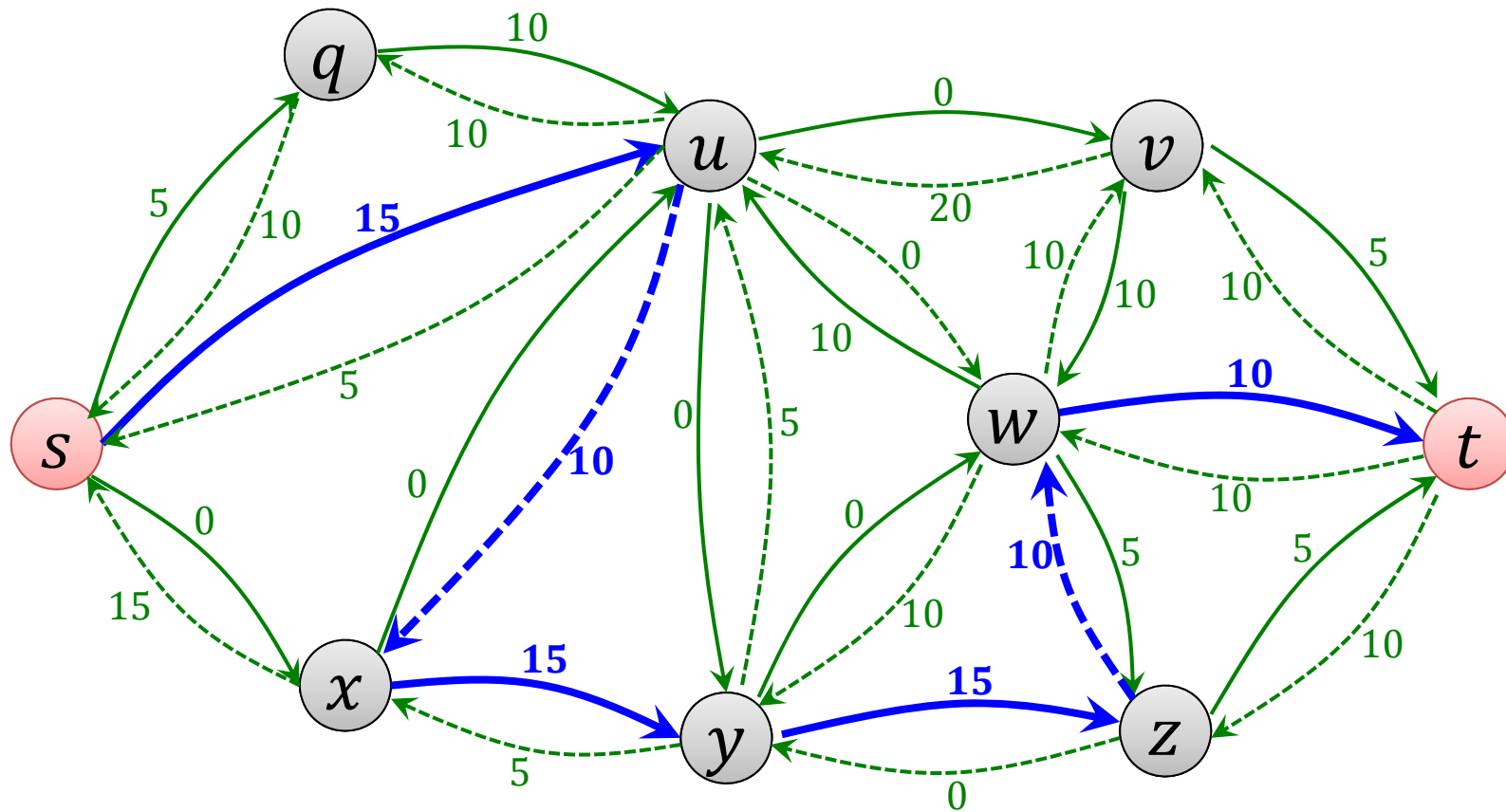
Residual Graph: Example

Residual Graph G_f



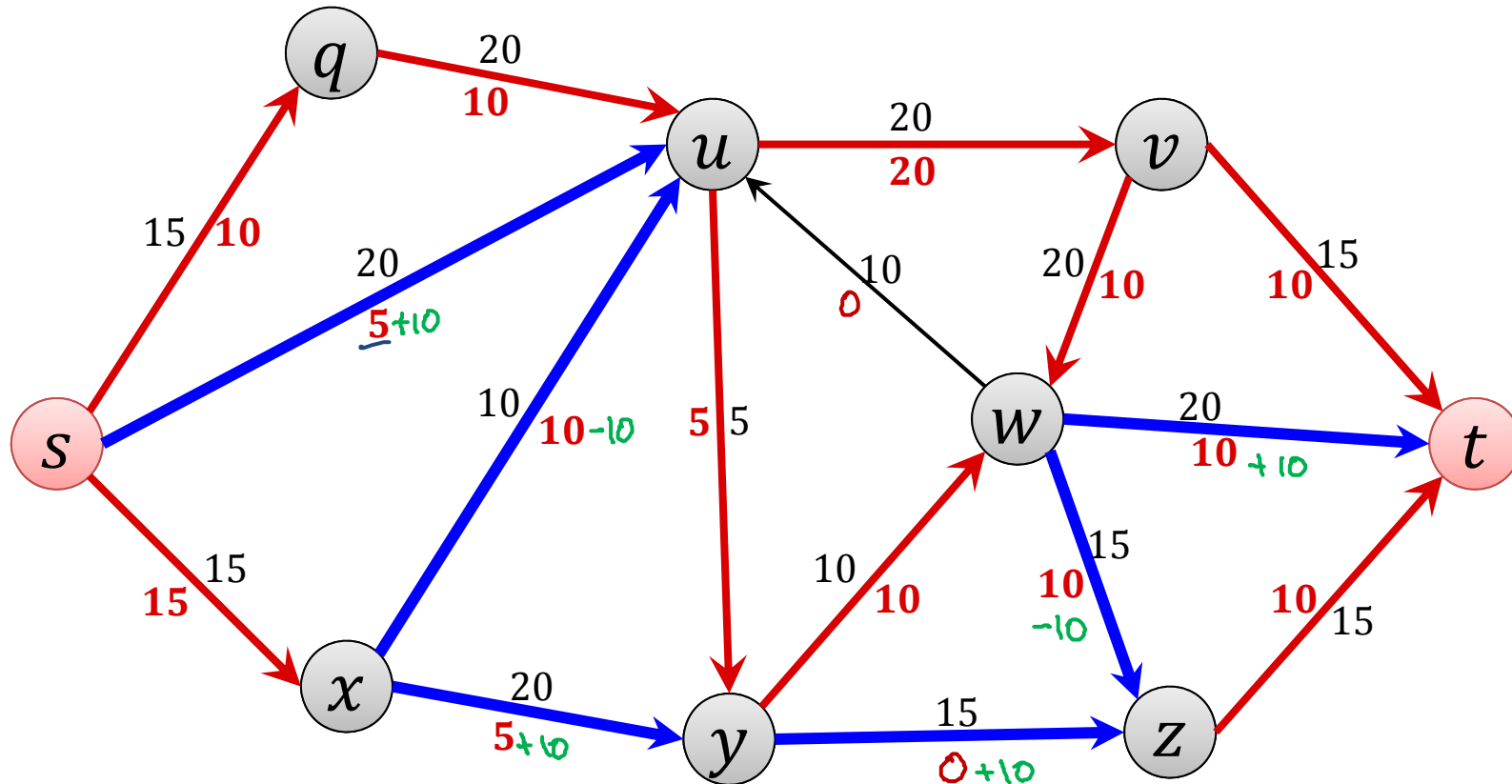
Augmenting Path

Residual Graph G_f



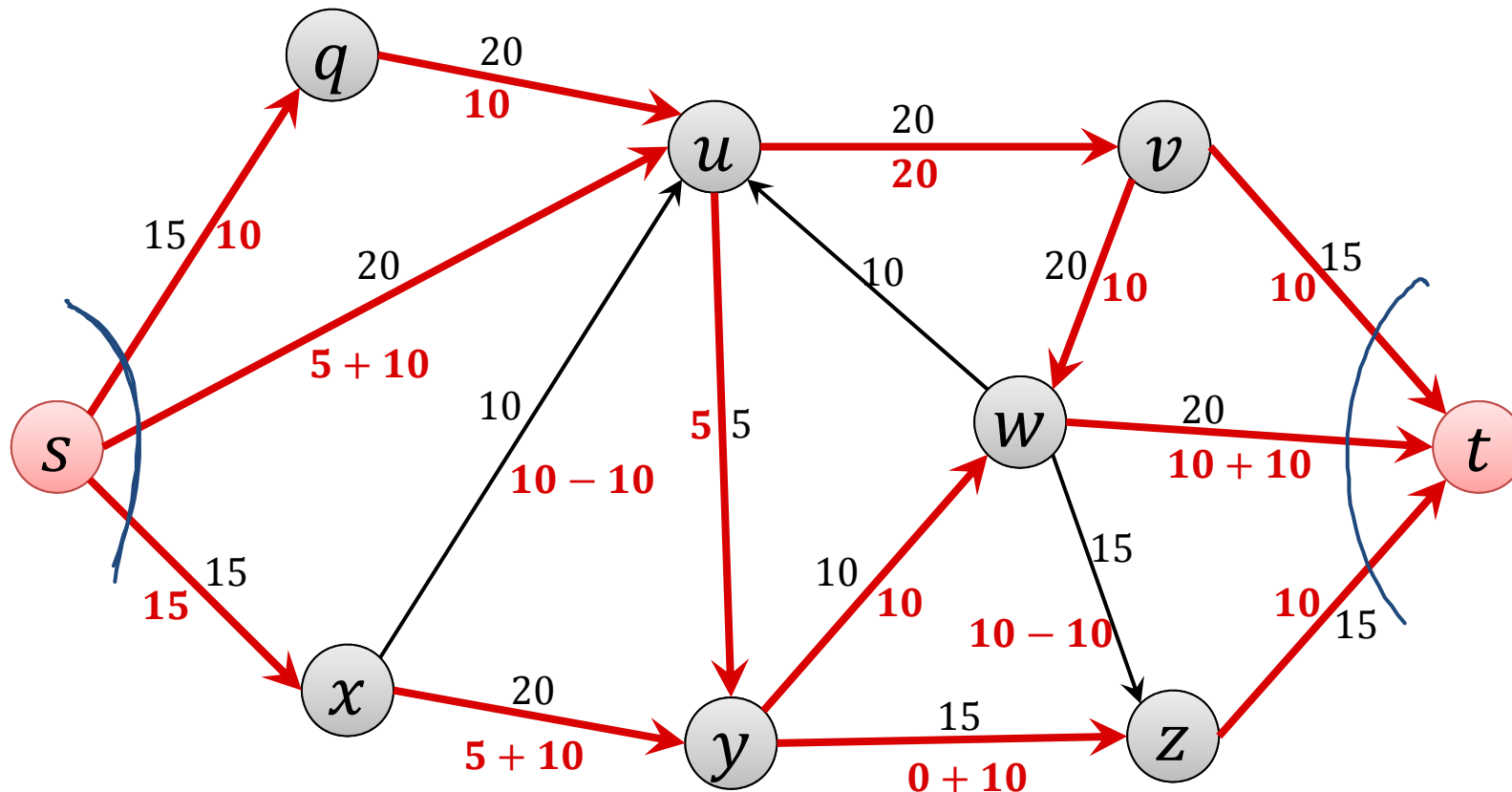
Augmenting Path

Augmenting Path



Augmenting Path

New Flow



Augmenting Path

Definition:

An **augmenting path** P is a (simple) s - t -path on the **residual graph** G_f on which each edge has residual capacity > 0 .

bottleneck (P, f) : minimum residual capacity on any edge of the augmenting path P

Augment flow f to get flow f' :

- For every forward edge (u, v) on P :

$$\underline{f'((u, v))} := \underline{f((u, v))} + \underline{\text{bottleneck}(P, f)}$$

- For every backward edge (u, v) on P :

$$\underline{f'((v, u))} := \underline{f((v, u))} - \underline{\text{bottleneck}(P, f)}$$

Augmented Flow

Lemma: Given a flow f and an augmenting path \underline{P} , the resulting augmented flow f' is legal and its value is

$$\underline{|f'|} = \underline{|f|} + \underline{\text{bottleneck}(P, f)}.$$

Proof:

f' is legal: $\forall e \in E: 0 \leq f'(e) \leq c_e$ (I)
 $\forall v \in V \setminus \{s, t\}: f'^{\text{in}}(v) = f'^{\text{out}}(v)$ (II)

(I) forward edge $e=(u,v)$ on P



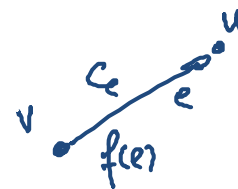
residual cap:

$$c_e - f(e) \geq \text{bottleneck}(P, f)$$

$$0 \leq f'(e) = f(e) + \text{bottleneck}(P, f) \leq c_e$$

↑
because $f(e) \geq 0$

backward edge (u,v) on P



res. cap. of $(u,v) = f(e)$

$$f(e) \geq \text{bottleneck}(P, f)$$

$$c_e \geq f(e) \geq f'(e) = f(e) - \text{bottleneck}(P, f) \geq 0$$

(I) holds

Augmented Flow

Lemma: Given a flow f and an augmenting path P , the resulting augmented flow f' is legal and its value is

$$|f'| = |f| + \text{bottleneck}(P, f).$$

Proof:

flow conservation

