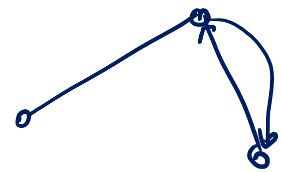
Chapter 6
Graph Algorithms

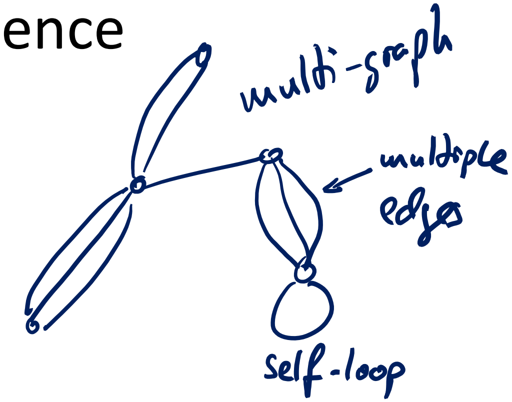
Algorithm Theory
WS 2016/17

Fabian Kuhn

Graphs

Extremely important concept in computer science





Graph $G = (V, E)$

- V: node (or vertex) set
- $E \subseteq V^2$: edge set   edge (u,v)
  - 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
  - Weighted graph: (positive) weight on edges (or nodes)
- (simple) path: sequence $v_0, \ldots, v_k$ of nodes such that
  $$(v_i, v_{i+1}) \in E \text{ for all } i \in \{0, \ldots, k-1\}$$
- …

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):

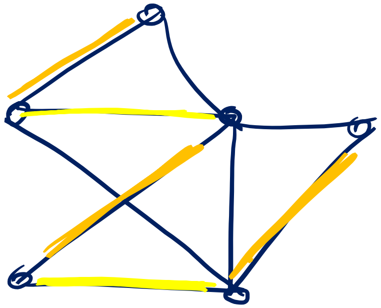
- Compute shortest TSP path/tour in weighted graph

Vertex coloring:

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

Maximum matching:

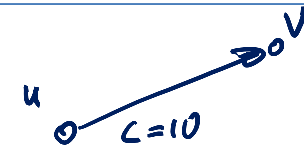
- Matching: set of pair-wise non-adjacent edges
- Goal: maximize the size of the matching



Network Flow

Flow Network:

- Directed simple 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 $t \in V$



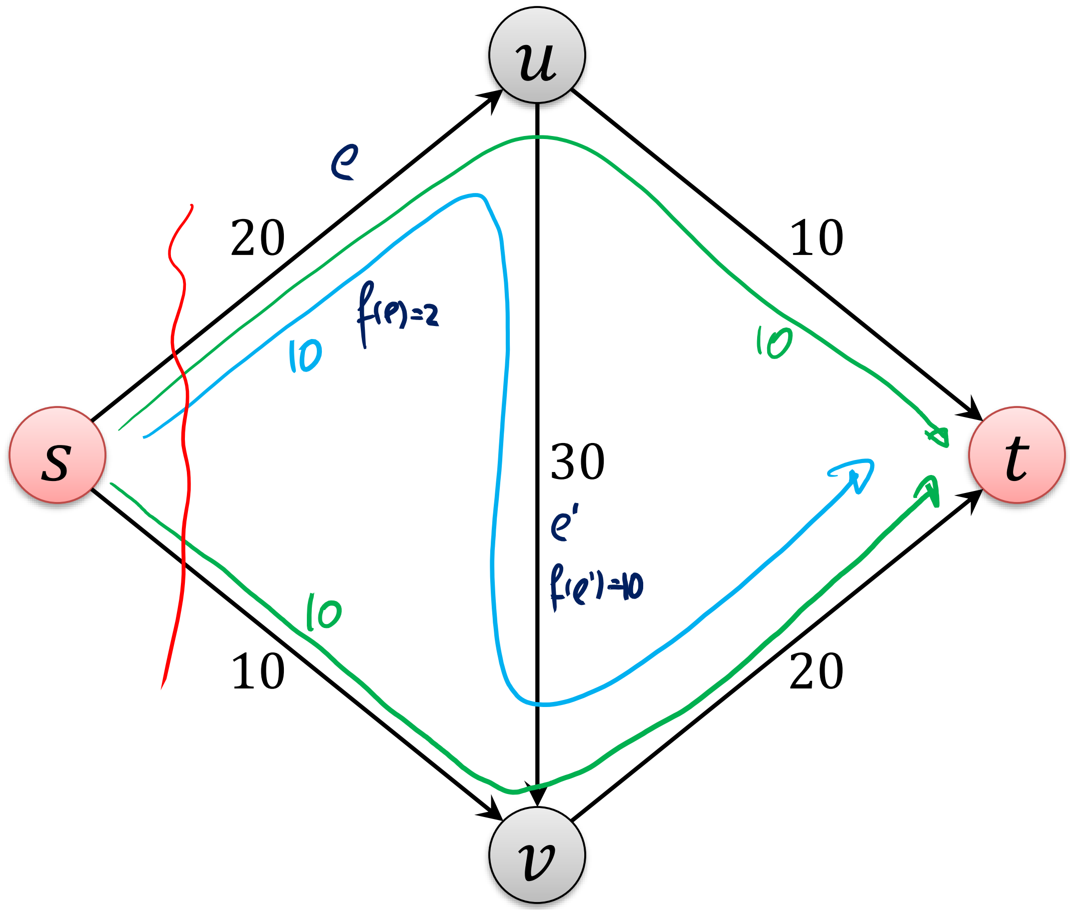
Flow: (informally)

- 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 $\boldsymbol{f}: \boldsymbol{E} \rightarrow \mathbb{R}_{\geq 0}$

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

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:

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

Notation

We define:

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

For a set $S \subseteq V$:

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

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

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

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

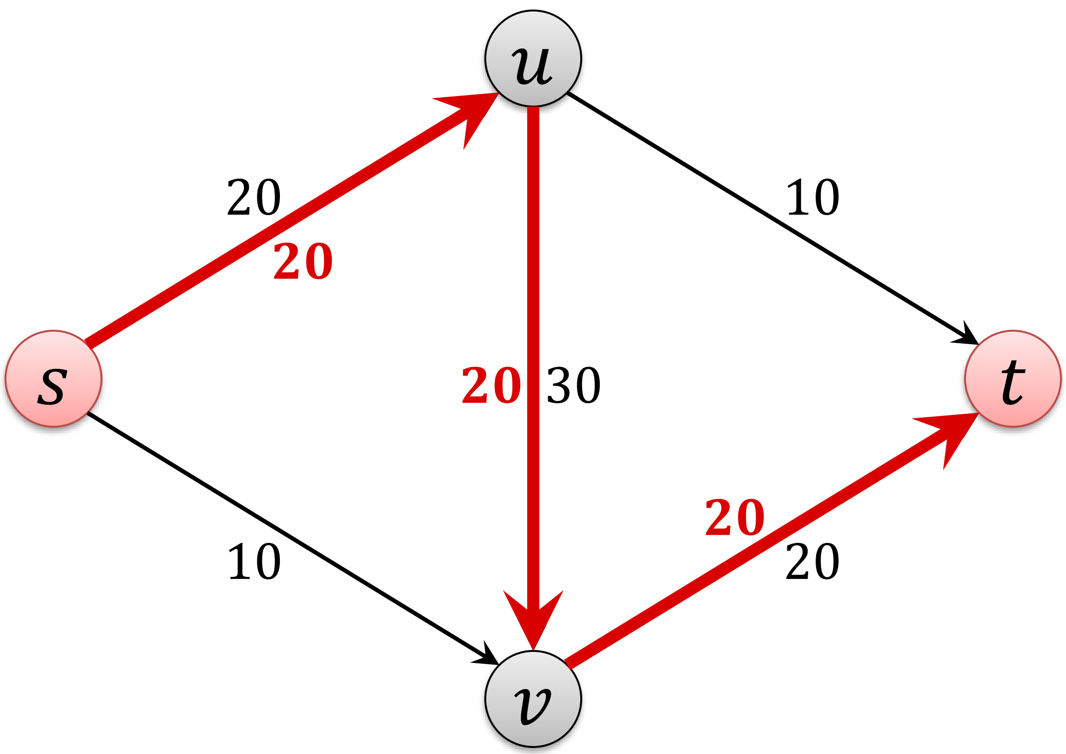
- Classical graph optimization problem
- Many applications (also beyond the obvious ones)
- Requires new algorithmic techniques

Maximum Flow: Greedy?

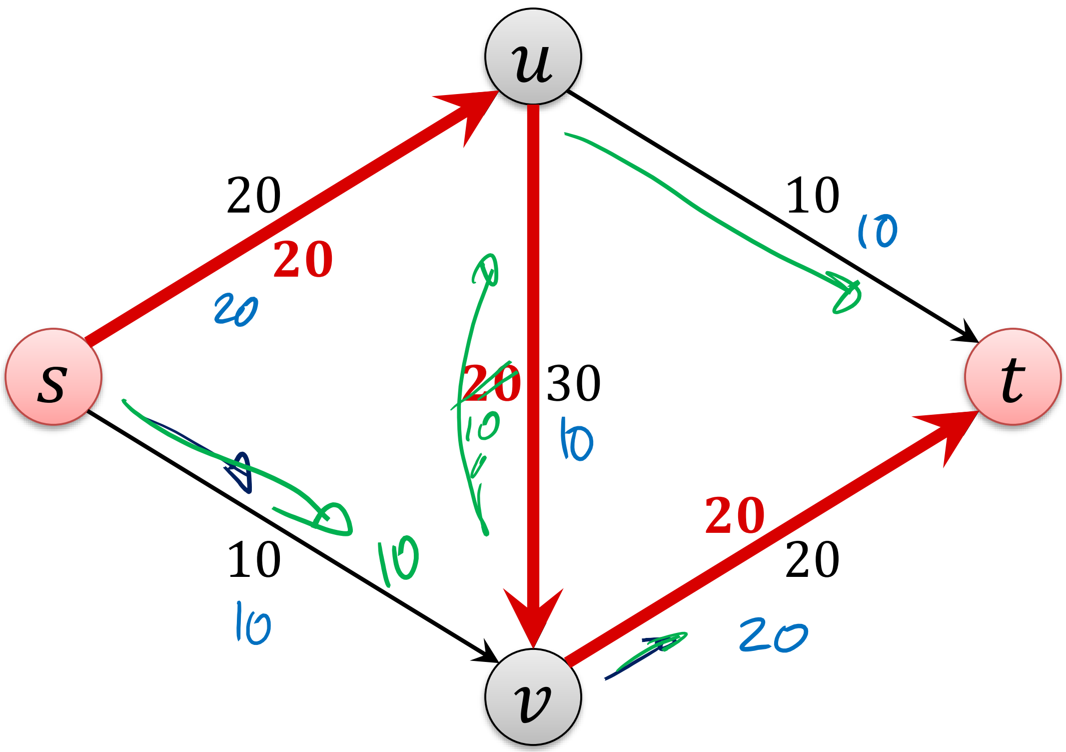
Does greedy work?

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)