Algorithms and Data Structures

C7. Graphs: Outlook

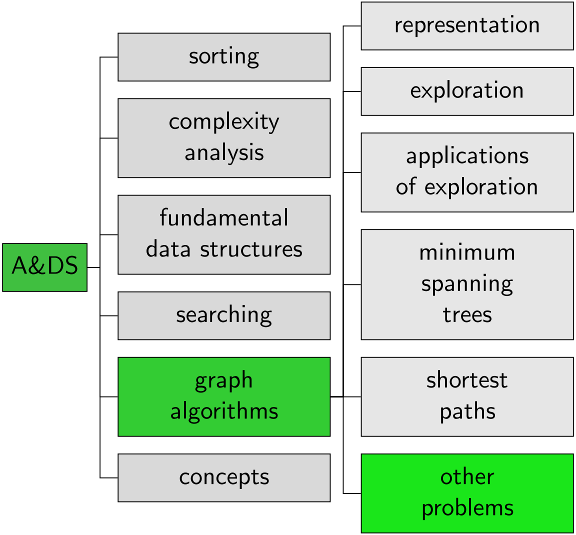
Gabriele Röger and Patrick Schnider

University of Basel

May 22, 2025

Other Graph Problems

Content of the Course



Crash Course Complexity Theory

- Decision problems: Seeking a Yes/No answer
  Given weighted graph, vertices s, t and number K.
  Is there a path from s to t that costs at most K?
- Search problem: Seeking an actual solution
  Given weighted graph and vertices s, t.
  Find a shortest path from s to t.

Crash Course Complexity Theory

We distinguish different classes of problems:

- P: decision problems that can be solved with a polynomial-time algorithm (in $O(p)$ for some polynomial p).

Crash Course Complexity Theory

We distinguish different classes of problems:

- P: decision problems that can be solved with a polynomial-time algorithm (in $O(p)$ for some polynomial p).
- NP: decision problems, where the yes instances have proofs that can be verified in polynomial time.
  Proof: e.g. specific path of cost $\leq$ K

Crash Course Complexity Theory

We distinguish different classes of problems:

- P: decision problems that can be solved with a polynomial-time algorithm (in $O(p)$ for some polynomial p).
- NP: decision problems, where the yes instances have proofs that can be verified in polynomial time.
  Proof: e.g. specific path of cost $\leq$ K
- $P \neq NP$? We do not know.

Crash Course Complexity Theory

We distinguish different classes of problems:

- P: decision problems that can be solved with a polynomial-time algorithm (in $O(p)$ for some polynomial p).
- NP: decision problems, where the yes instances have proofs that can be verified in polynomial time.
  Proof: e.g. specific path of cost $\leq$ K
- P $\neq$ NP? We do not know.
- NP-hard problems: Problems that are at least as hard as the hardest problems in NP.
  $\rightarrow$ no polynomial-time algorithms known.

Crash Course Complexity Theory

We distinguish different classes of problems:

- P: decision problems that can be solved with a polynomial-time algorithm (in $O(p)$ for some polynomial p).
- NP: decision problems, where the yes instances have proofs that can be verified in polynomial time.
  Proof: e.g. specific path of cost $\leq$ K
- P $\neq$ NP? We do not know.
- NP-hard problems: Problems that are at least as hard as the hardest problems in NP.
  $\rightarrow$ no polynomial-time algorithms known.
- NP-complete decision problems: NP-hard & in NP

Crash Course Complexity Theory

We distinguish different classes of problems:

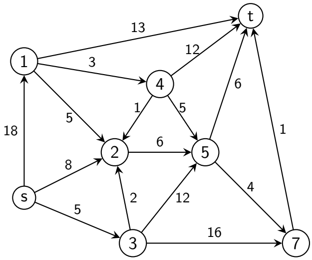
- P: decision problems that can be solved with a polynomial-time algorithm (in $O(p)$ for some polynomial p).
- NP: decision problems, where the yes instances have proofs that can be verified in polynomial time.
  Proof: e.g. specific path of cost $\leq$ K
- P $\neq$ NP? We do not know.
- NP-hard problems: Problems that are at least as hard as the hardest problems in NP.
  $\rightarrow$ no polynomial-time algorithms known.
- NP-complete decision problems: NP-hard & in NP
- NP-equivalent search problems: corresponding decision problem NP-complete.

Flows in Graphs I

Definition (Flow Network)

A flow network $N = (G, s, t, k)$ is given by

- a directed graph $G = (V, E)$,
- a source $s \in V$,
- a sink $t \in V$, and
- a capacity function $k : E \to \mathbb{R}_+^\infty$.



Flows in Graphs II

Definition (Flow)

An s-t flow f assigns every edge a value from $\mathbb{R}_{\geq 0}$, where

- the value does not exceed the capacity of the edge::

$$f(e) \leq k(e) \text{ for all } e \in E$$

Flows in Graphs II

Definition (Flow)

An s-t flow f assigns every edge a value from $\mathbb{R}_{\geq 0}$, where

- the value does not exceed the capacity of the edge::
$$f(e) \leq k(e) \text{ for all } e \in E$$
- for all vertices except for the source and the sink the incoming flow matches the outgoing flow:
$$\sum_{\substack{(u,w)\in E\\ w=v}} f((u,w)) = \sum_{\substack{(u,w)\in E\\ u=v}} f((u,w)) \text{ for all } v \in V \setminus \{s,t\}$$

Flows in Graphs II

Definition (Flow)

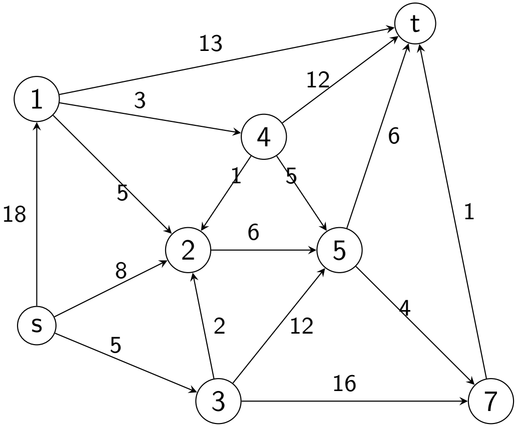
An s-t flow f assigns every edge a value from $\mathbb{R}_{\geq 0}$, where

- the value does not exceed the capacity of the edge::
$$f(e) \leq k(e) \text{ for all } e \in E$$
- for all vertices except for the source and the sink
the incoming flow matches the outgoing flow:
$$\sum_{\substack{(u,w)\in E\\ w=v}} f((u,w)) = \sum_{\substack{(u,w)\in E\\ u=v}} f((u,w)) \text{ for all } v \in V \setminus \{s,t\}$$

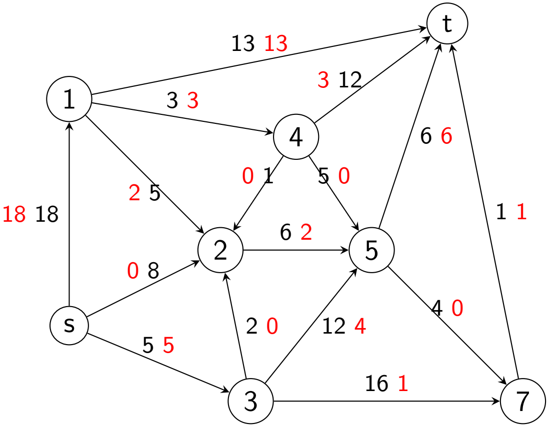
The value of the flow is the net flow into the sink:

$$|f| = \sum_{\substack{(u,w)\in E\\ w=t}} f((u,w)) - \sum_{\substack{(u,w)\in E\\ u=t}} f((u,w))$$

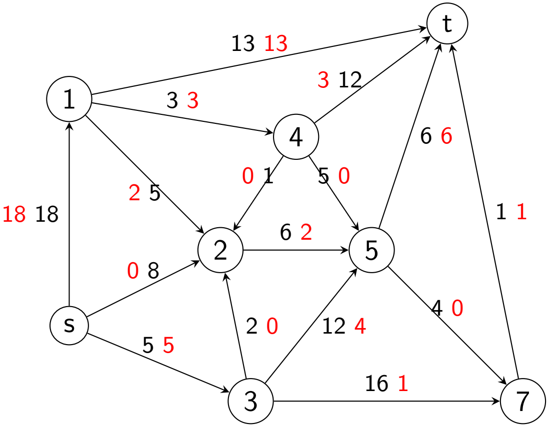
Example



Example

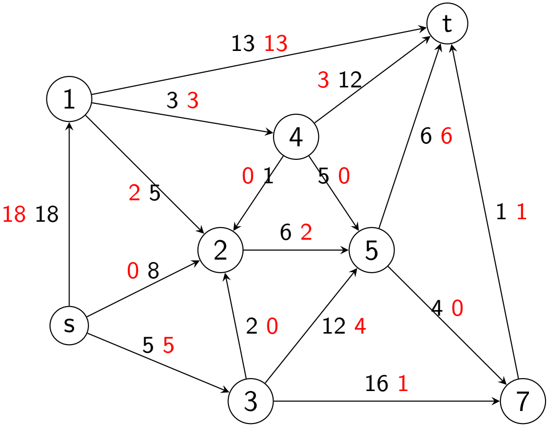


Example



How hard is it to find a maximal flow?

Example



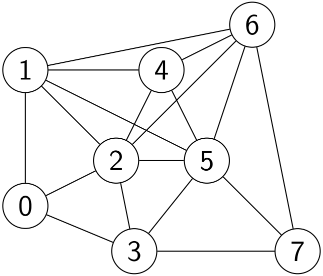
How hard is it to find a maximal flow?

For example with the Edmonds-Karp algorithm in $O(|E|^2|V|)$

Cliques

Definition (Clique)

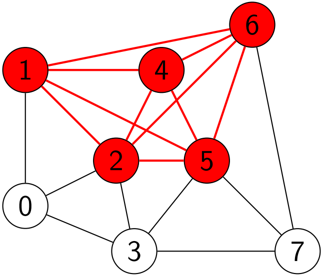
A clique in an undirected graph (V, E) is a subset $C \subseteq V$ of the vertices such that each pair of distinct vertices in C is connected by an edge: for $u, v \in C$ with $u \neq v$ it holds that $\{u, v\} \in E$.



Cliques

Definition (Clique)

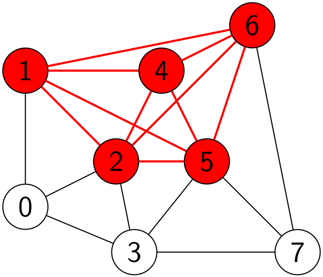
A clique in an undirected graph (V, E) is a subset $C \subseteq V$ of the vertices such that each pair of distinct vertices in C is connected by an edge: for $u, v \in C$ with $u \neq v$ it holds that $\{u, v\} \in E$.



Cliques

Definition (Clique)

A clique in an undirected graph (V, E) is a subset $C \subseteq V$ of the vertices such that each pair of distinct vertices in C is connected by an edge: for $u, v \in C$ with $u \neq v$ it holds that $\{u, v\} \in E$.

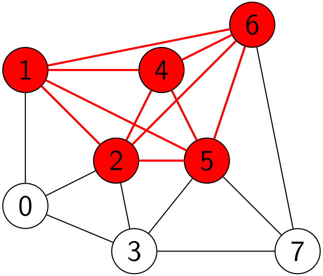


How hard is it, to determine a largest clique in a graph?

Cliques

Definition (Clique)

A clique in an undirected graph (V, E) is a subset $C \subseteq V$ of the vertices such that each pair of distinct vertices in C is connected by an edge: for $u, v \in C$ with $u \neq v$ it holds that $\{u, v\} \in E$.



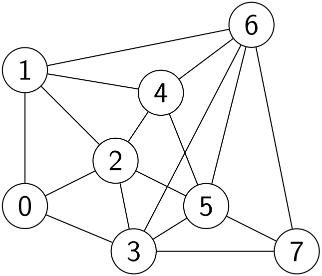
How hard is it, to determine a largest clique in a graph?

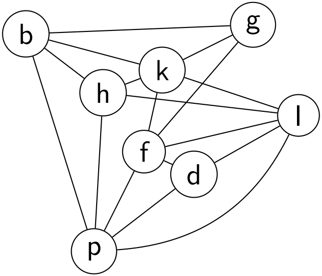
NP-equivalent

Graph Isomorphism

Definition (Graph Isomorphism)

Two graphs are isomorphic, if they are identical up to renaming vertices.

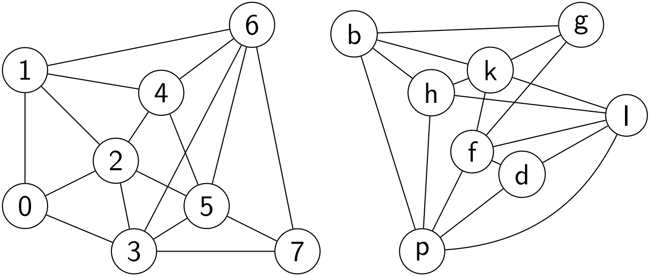




Graph Isomorphism

Definition (Graph Isomorphism)

Two graphs are isomorphic, if they are identical up to renaming vertices.

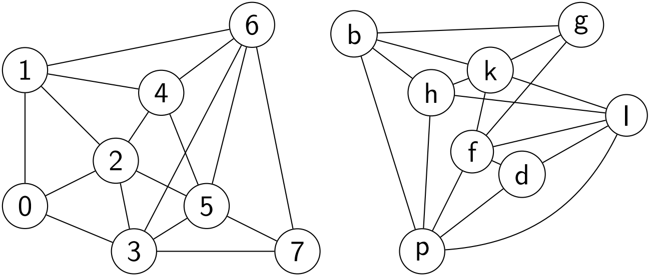


How hard is it to decide whether two given graphs are isomorphic?

Graph Isomorphism

Definition (Graph Isomorphism)

Two graphs are isomorphic, if they are identical up to renaming vertices.



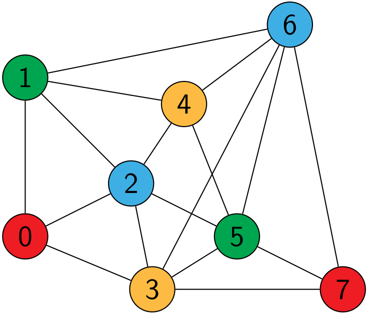
How hard is it to decide whether two given graphs are isomorphic?

In NP, but unknown whether in P and/or NP-complete

Graph Coloring

Definition (k-Colorability)

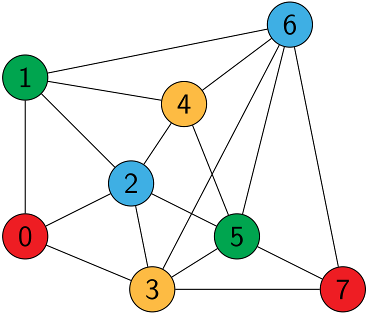
An undirected graph $G = (V, E)$ is k-colorable ($k \in \mathbb{N}$), if there is a coloring $f : V \to \{1, \ldots, k\}$ with $f(v) \neq f(w)$ for all $\{v, w\} \in E$.



Graph Coloring

Definition (k-Colorability)

An undirected graph $G = (V, E)$ is k-colorable ($k \in \mathbb{N}$), if there is a coloring $f : V \to \{1, \ldots, k\}$ with $f(v) \neq f(w)$ for all $\{v, w\} \in E$.

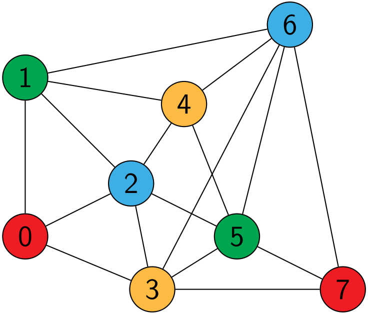


How hard is it to decide whether a give graph is k-colorable?

Graph Coloring

Definition (k-Colorability)

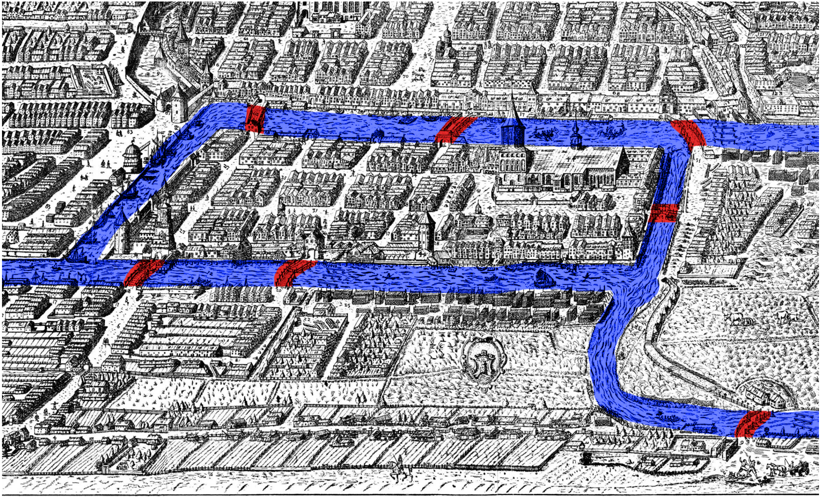
An undirected graph $G = (V, E)$ is k-colorable ($k \in \mathbb{N}$), if there is a coloring $f : V \to \{1, \ldots, k\}$ with $f(v) \neq f(w)$ for all $\{v, w\} \in E$.



How hard is it to decide whether a give graph is k-colorable?

NP-complete

Seven Bridges of Königsberg

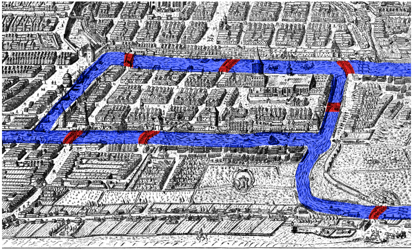


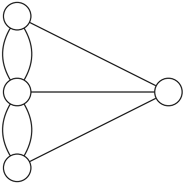
Is there a walk through the city crossing each bridge exactly once?

Eulerian Trail

Definition (Eulerian Trail)

An Eulerian trail is a path that uses every edge exactly once.

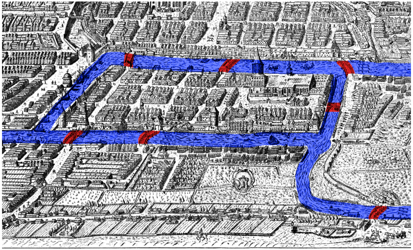


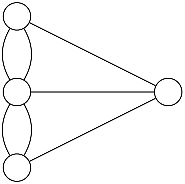


Eulerian Trail

Definition (Eulerian Trail)

An Eulerian trail is a path that uses every edge exactly once.



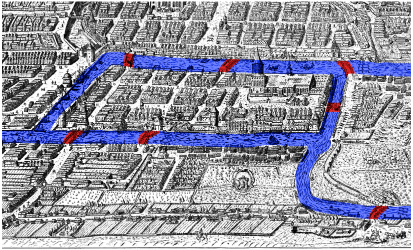


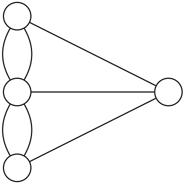
How hard is it to decide whether a graph has an Eulerian trail?

Eulerian Trail

Definition (Eulerian Trail)

An Eulerian trail is a path that uses every edge exactly once.





How hard is it to decide whether a graph has an Eulerian trail?

Has Eulerian trail iff exactly zero or two vertices have odd degree, and all of its vertices with nonzero degree belong to a single connected component.

Quiz

Quiz

kahoot.it