Complexity Theory

More **NP**-Complete Problems
May 23, 2022

iCIS | Software Science
Radboud University

Recap

More **NP**-complete problems
- Graph problems
- Sums of subsets
- Parsing

Extra Topics

Outline

How to prove that a problem is NP-complete?

To show that A is **NP**-complete, take the following steps:

(i) Show that A is in **NP**.

(ii) Pick a decision problem A' of which you know that it is **NP**-complete.

(iii) Prove that $A' \leq_P A$.

Some NP-complete problems (satisfiability)

SAT

Given a formula φ, is φ satisfiable?

CNF

Given a formula φ in conjunctive normal form, is φ satisfiable?

$\leq_3$CNF

Given a formula in at most 3-conjunctive normal form, is it satisfiable?

3CNF

Given a formula in exactly 3-conjunctive normal form, is it satisfiable?

Some NP-complete problems (integers)

ILP

Given an integer linear program, does it have a solution?

Some NP-complete problems (graphs)

Clique:

> Given a graph G and an integer k, does G have a clique with k vertices?

VertexCover

> Given a graph G and an integer k, does G have a vertex cover with k vertices?

3Color

> Given a graph G, does G have a 3-coloring?

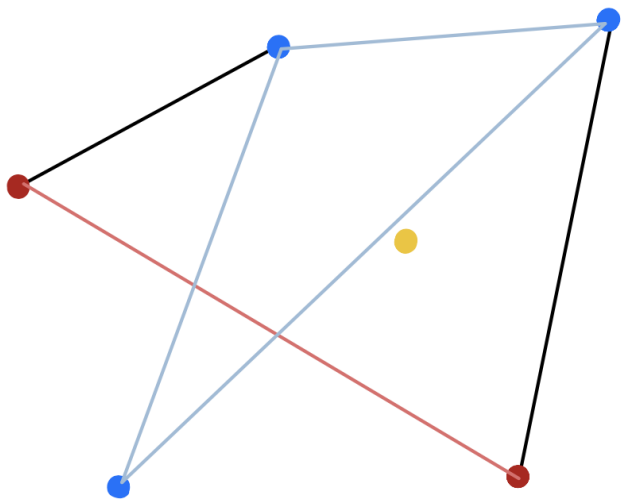
Outline

Covering by 3 cliques

Definition

We say that a graph $G = (V, E)$ can be **covered by 3 cliques** if there are three subsets $V_1 \subseteq V$, $V_2 \subseteq V$, and $V_3 \subseteq V$ such that

- V is covered by V_1, V_2, and V_3: $V = V_1 \cup V_2 \cup V_3$
- The intersections are empty: $V_1 \cap V_2 = V_1 \cap V_3 = V_2 \cap V_3 = \emptyset$
- V_1, V_2, and V_3 are cliques



The NP-completeness

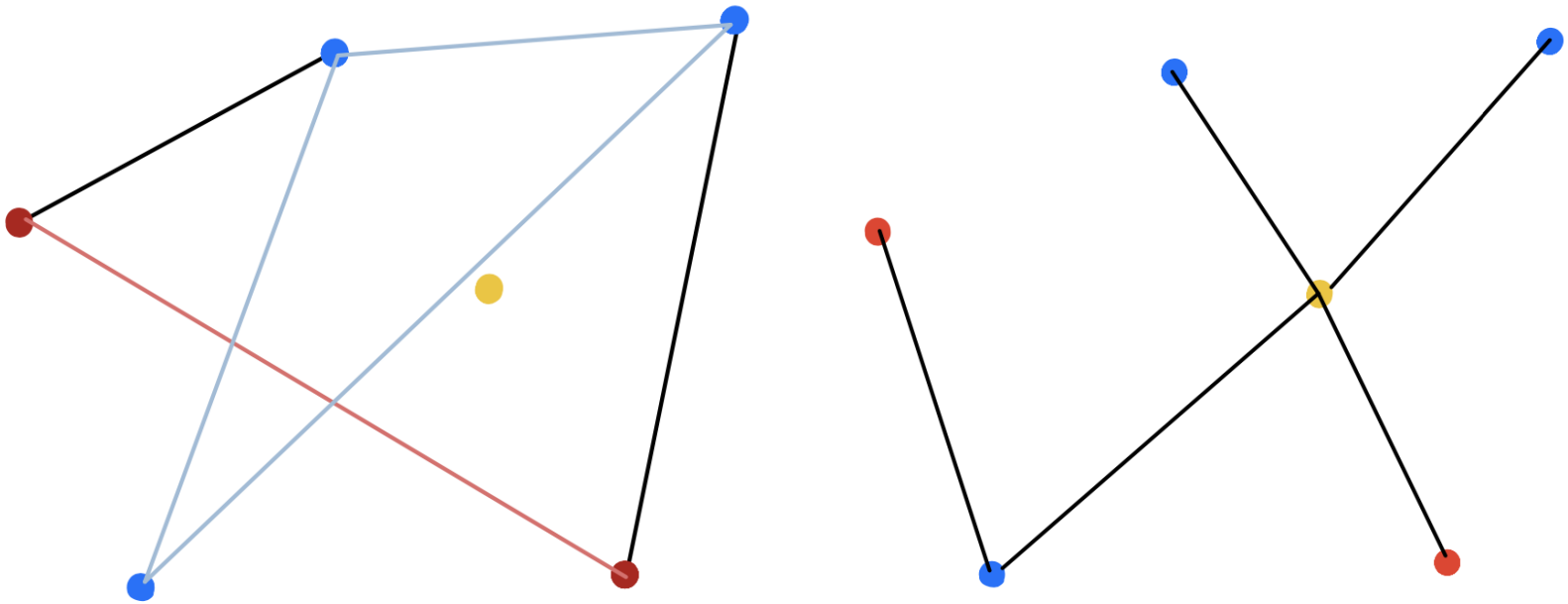
We look at the decision problem Clique-3Cover

Given a graph G, can G be covered with 3 cliques?

Theorem

Clique-3Cover *is* **NP***-complete*

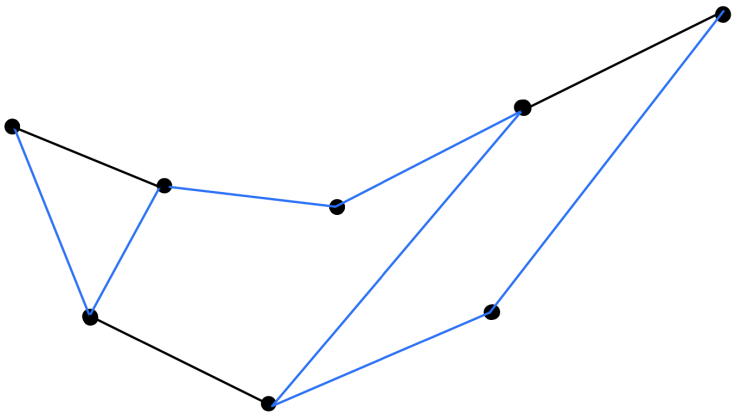
The construction illustrated

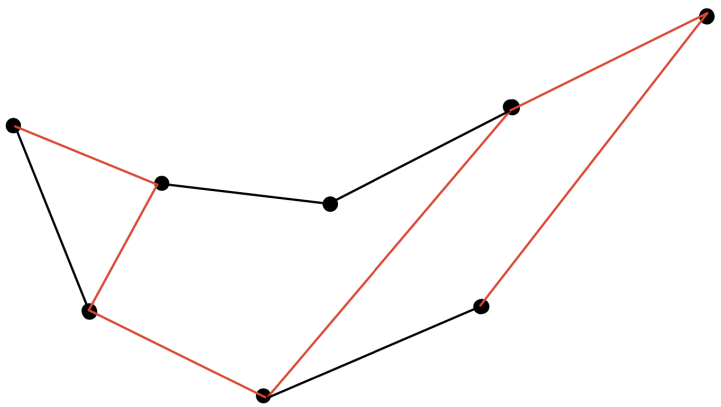


Hamiltonian paths

Definition
Let G be a graph. We say that G has a **Hamiltonian path** if there is a path p in G that crosses every vertex exactly once

Below, the blue path is Hamiltonian while the red is not.





NP-completeness

We look at the decision problem HamPath

> Given a graph G, does G have a Hamiltonian path?

Theorem
HamPath *is* **NP***-complete*

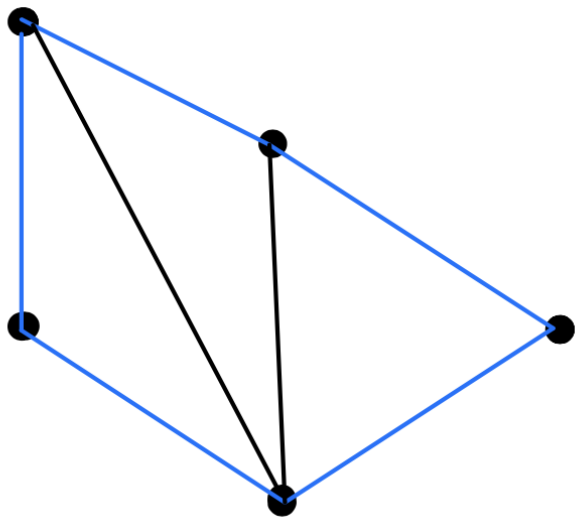
See note!

Hamiltonian cycle

Definition

Let G be a graph. We say that G has a **Hamiltonian cycle** if there is a cycle c in G that crosses every vertex exactly once

A cycle c can be written as $v_1, v_2, \ldots, v_n$ such that $\{v_i, v_{i+1}\} \in E$ and $\{v_n, v_1\} \in E$. For a Hamiltonian cycle: $v_i \neq v_j$ if $i \neq j$ and every vertex occurs in this cycle.



NP-completeness

We look at the decision problem HamCycle

Given a graph G, does G have a Hamiltonian cycle?

Theorem

HamCycle *is* **NP***-complete*

Illustration of the proof

From the construction, we get the following graph

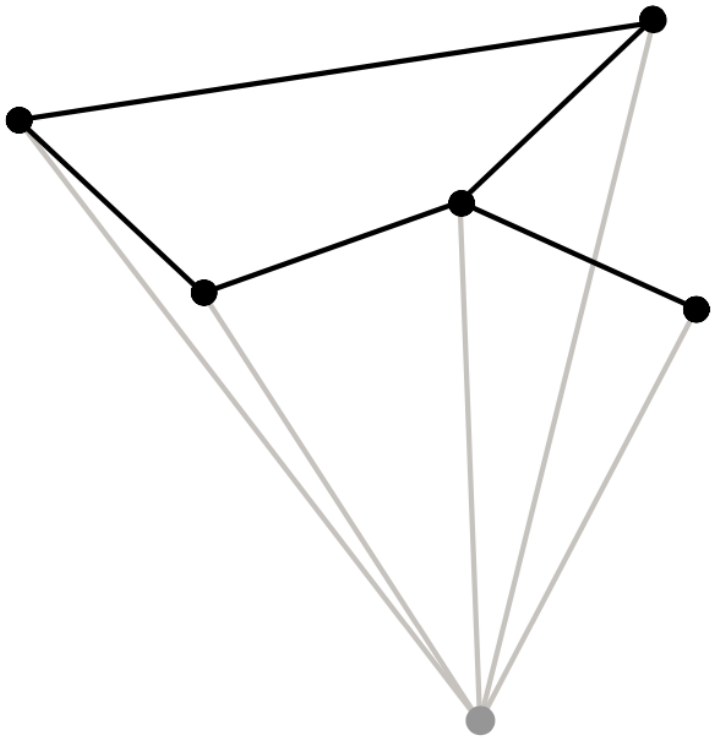
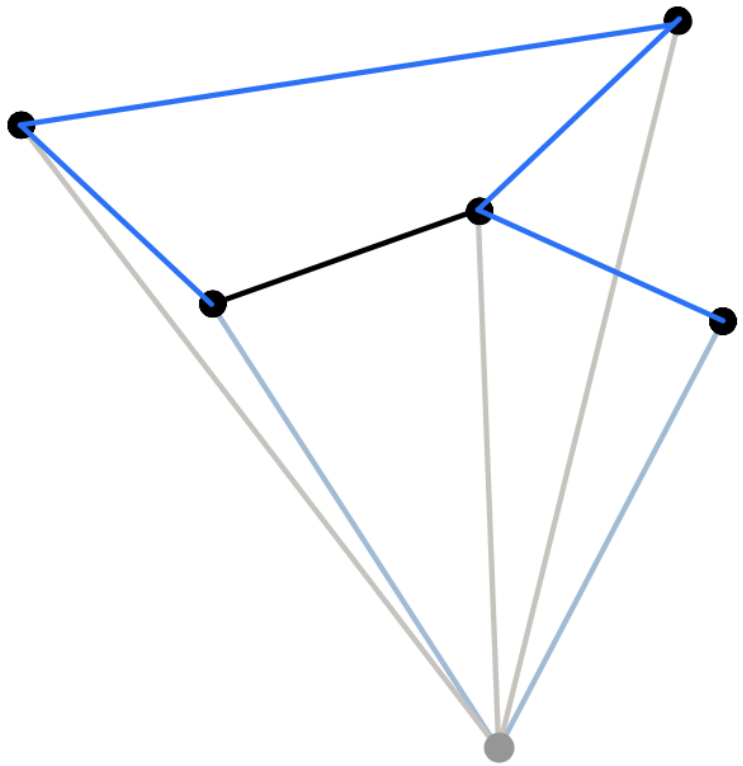


Illustration of the proof

A Hamiltonian cycle in this graph looks as follows



Traveling Salesperson

We look at the decision problem TSP

> Given a **complete** graph G, a function c, and an integer k, is there a cycle in G with cost at most k that crosses every vertex?

Theorem
TSP *is* **NP***-complete.*

Sums of subsets

Suppose, we have

- a finite set S of natural numbers
- an integer k

We are interested in finding subsets $S' \subseteq S$ such that

$$\sum_{x \in S'} x = k$$

Sums of subsets

Suppose, we have

- a finite set S of natural numbers
- an integer k

We are interested in finding subsets $S' \subseteq S$ such that

$$\sum_{x \in S'} x = k$$

Example: take $S = \{1, 4, 6, 9, 12\}$

- There is a subset with sum 14, namely $\{1, 4, 9\}$
- There is no subset with sum 8

NP-completeness

We look at the decision problem SubsSum

> Given a finite set S of natural numbers and an integer k, is there a subset S' of S with sum k?

Theorem

SubsSum *is* **NP***-complete.*

Parsing

Parsing:

> Given a grammar G and a word w, can we derive $S \Rightarrow w$?

Parsing

Parsing:

> Given a grammar G and a word w, can we derive $S \Rightarrow w$?

This is in **P**.

Weighted Parsing

Definition
A **weighted contex free grammar** is a context free grammar in which every production rule is assigned a weight.
The weight of a production is the sum of the individual weights.

Weighted Parsing

Definition
A **weighted contex free grammar** is a context free grammar in which every production rule is assigned a weight.
The weight of a production is the sum of the individual weights.

$$S \overset{3}{\Rightarrow} A \mid B$$

$$A \overset{2}{\Rightarrow} aA$$

$$A \overset{1}{\Rightarrow} aB$$

$$B \overset{0}{\Rightarrow} b$$

Weighted Parsing

Definition
A **weighted contex free grammar** is a context free grammar in which every production rule is assigned a weight.
The weight of a production is the sum of the individual weights.

$$S \overset{3}{\Rightarrow} A \mid B$$

$$A \overset{2}{\Rightarrow} aA$$

$$A \overset{1}{\Rightarrow} aB$$

$$B \overset{0}{\Rightarrow} b$$

The following production has weight 6.

$$S \overset{3}{\Rightarrow} A \overset{2}{\Rightarrow} aA \overset{1}{\Rightarrow} aaB \overset{0}{\Rightarrow} aab$$

NP-completeness

We look at the decision problem WParse

> Given a weight contex-free grammar G, a word w, and an integer k, is there a production of w in G of weight k?

Theorem
WParse *is* **NP***-complete.*

Outline

Some decision problems

VertexCover

> Given a graph G and an integer k, does G have a vertex cover with k vertices?

TSP

> Given a complete graph G, a function c, and an integer k, is there a cycle in G with cost at most k?

Some optimization problems

VertexCover

Given a graph G, find the **minimal** vertex cover of G

TSP

Given a complete graph G and a function c, find a cycle in G with **minimal** cost.

What does NP-completeness mean for these problems?

- Since we know VertexCover and TSP are **NP**-complete, it will be either difficult or impossible to find efficient algorithms that compute minimal vertex covers or minimal cycles.
- But what if we do not go for the **best** solution, but instead for a solution that is **good enough**?

Finding vertex covers

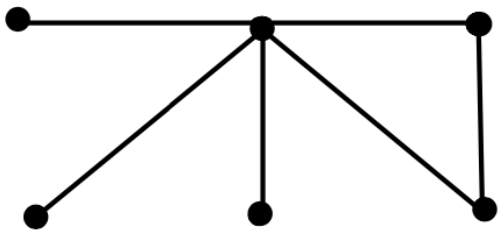
Let $G = (V, E)$ be a graph.
To compute a vertex cover C of G, we take the following steps

1. Let $C := \emptyset$ and $E' := E$
2. While E' is not empty
   2.1. Take any edge $\{u, v\}$ from E'
   2.2. Take $C := C \cup \{u, v\}$
   2.3. Remove every edge from E' that touches either u or v.

Example

E' is black

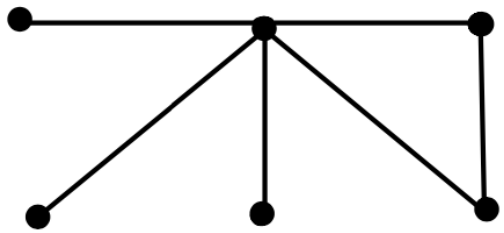
C is blue



Example

E' is black

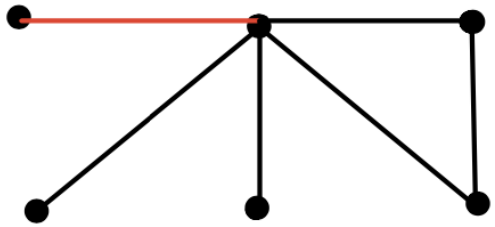
C is blue



We select

E' is black

C is blue



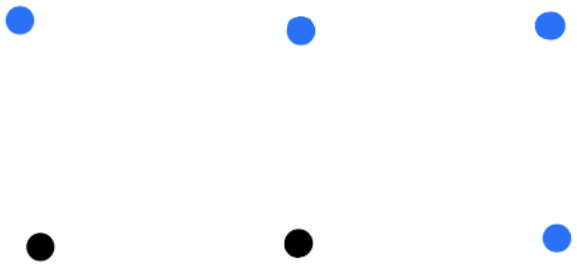
Example

Example

We select

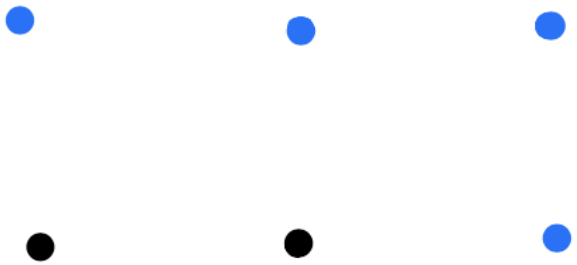
It is not minimal!

The algorithm gave a vertex cover of size 4, namely the following.

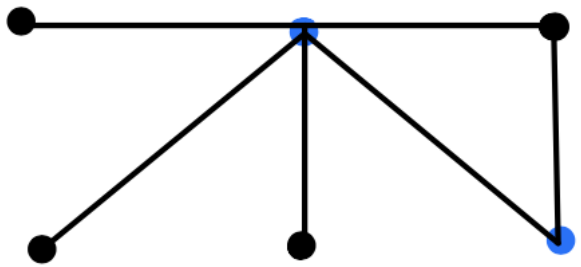


It is not minimal!

The algorithm gave a vertex cover of size 4, namely the following.



But the minimal vertex cover has size 2



But it is a decent approximation

Theorem
The size of the vertex cover computed by the algorithm, is at most twice as big as the size of the minimal vertex cover.

So: while the algorithm does not give the best solution, it gives a solution that is “good enough” within reasonable time.

More approximation algorithms

There are approximation algorithms for

- The traveling salesperson problem
- The knapsack problem
- Finding colorings of graphs

And there are more complexity classes: PSPACE

Definition
A decision problem is in **PSPACE** if there is a Turing machine that decides it using only a polynomial amount of space.

Definition
A decision problem is in **NPSPACE** if there is a nondeterministic Turing machine that decides it using only a polynomial amount of space.

Theorem (Savitch's theorem)
PSPACE $=$ **NPSPACE**.

And there are more complexity classes: PSPACE

Definition
A decision problem is in **PSPACE** if there is a Turing machine that decides it using only a polynomial amount of space.

Definition
A decision problem is in **NPSPACE** if there is a nondeterministic Turing machine that decides it using only a polynomial amount of space.

Theorem (Savitch's theorem)
PSPACE $=$ **NPSPACE**.

Quite more theory can be developed. See the master course "Complexity Theory" (in the master MFoCS).

iCIS | Software Science
Radboud University