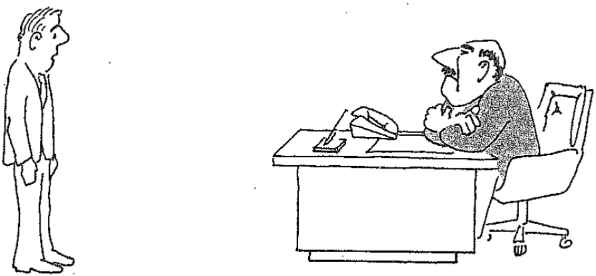
CS 577- Intro to Algorithms

Computational Intractability (Part 3)

Dieter van Melkebeek

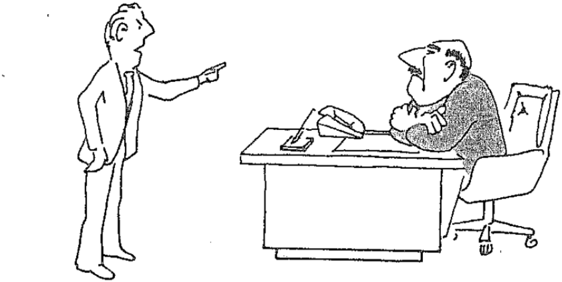
December 1, 2020

Motivation



"I can't find an efficient algorithm, I guess I'm just too dumb."

Motivation



"I can't find an efficient algorithm, because no such algorithm is possible!"

Motivation

"I can't find an efficient algorithm, but neither can all these famous people."

Recap

Recap

- P: decision problems that have polynomial-time algorithms

Recap

- P: decision problems that have polynomial-time algorithms
- NP: decision problems with yes-instances that have polynomial-time verifiable certificates

Recap

- P: decision problems that have polynomial-time algorithms
- NP: decision problems with yes-instances that have polynomial-time verifiable certificates
- Fact: $\mathrm{P} \subseteq \mathrm{NP}$

Recap

- P: decision problems that have polynomial-time algorithms
- NP: decision problems with yes-instances that have polynomial-time verifiable certificates
- Fact: $P \subseteq NP$
- Conjecture: $P \neq NP$

Recap

- P: decision problems that have polynomial-time algorithms
- NP: decision problems with yes-instances that have polynomial-time verifiable certificates
- Fact: $\text{P} \subseteq \text{NP}$
- Conjecture: $\text{P} \neq \text{NP}$
- Definition: B is NP-hard if $(\forall A \in \text{NP})\, A \leq^p B$.

Recap

- P: decision problems that have polynomial-time algorithms
- NP: decision problems with yes-instances that have polynomial-time verifiable certificates
- Fact: P $\subseteq$ NP
- Conjecture: P $\neq$ NP
- Definition: B is NP-hard if $(\forall A \in \text{ NP})\, A \leq^p B$.
- Assume P $\neq$ NP. If B is NP-hard then $B \notin$ P.

Recap

- P: decision problems that have polynomial-time algorithms
- NP: decision problems with yes-instances that have polynomial-time verifiable certificates
- Fact: P $\subseteq$ NP
- Conjecture: P $\neq$ NP
- Definition: B is NP-hard if $(\forall\, A \in \text{ NP})\, A \leq^p B$.
- Assume P $\neq$ NP. If B is NP-hard then $B \notin$ P.
- Theorem: Circuit-SAT is NP-hard.

Establishing NP-Hardness

Establishing NP-Hardness

Strategy

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT
- 3-SAT $\leq^p$ Independent Set

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT
- 3-SAT $\leq^p$ Independent Set

Today's instantiations

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT
- 3-SAT $\leq^p$ Independent Set

Today's instantiations

- Independent Set $\leq^p$ Clique

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT
- 3-SAT $\leq^p$ Independent Set

Today's instantiations

- Independent Set $\leq^p$ Clique
- Independent Set $\leq^p$ Vertex Cover

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT
- 3-SAT $\leq^p$ Independent Set

Today's instantiations

- Independent Set $\leq^p$ Clique
- Independent Set $\leq^p$ Vertex Cover
- 3-SAT $\leq^p$ 3-Coloring

Establishing NP-Hardness

Strategy

To show a new problem C is NP-hard:

- Find a known NP-hard problem B.
- Show that $B \leq^p C$.

Earlier instantiations

- Circuit-SAT $\leq^p$ 3-SAT
- 3-SAT $\leq^p$ Independent Set

Today's instantiations

- Independent Set $\leq^p$ Clique
- Independent Set $\leq^p$ Vertex Cover
- 3-SAT $\leq^p$ 3-Coloring
- 3-SAT $\leq^p$ Subset Sum

Independent Set vs Clique vs Vertex Cover

Independent Set vs Clique vs Vertex Cover

Definitions

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Relationships

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Relationships

- S is independent set in $G \Leftrightarrow S$ is clique in $\overline{G} \doteq (V, \overline{E})$.

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Relationships

- S is independent set in $G \Leftrightarrow S$ is clique in $\overline{G} \doteq (V, \overline{E})$.
- S is independent set in $G \Leftrightarrow \overline{S}$ is vertex cover in G.

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Relationships

- S is independent set in $G \Leftrightarrow S$ is clique in $\overline{G} \doteq (V, \overline{E})$.
- S is independent set in $G \Leftrightarrow \overline{S}$ is vertex cover in G.

Corollary

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Relationships

- S is independent set in $G \Leftrightarrow S$ is clique in $\overline{G} \doteq (V, \overline{E})$.
- S is independent set in $G \Leftrightarrow \overline{S}$ is vertex cover in G.

Corollary

- Independent Set $\leq^p$ Clique

Independent Set vs Clique vs Vertex Cover

Definitions

Fix a graph $G = (V, E)$. A subset $S \subseteq V$ is:

- An independent set if $E \cap S \times S = \emptyset$.
- A clique if $S \times S \subseteq E$.
- A vertex cover if $E \subseteq S \times V$.

Relationships

- S is independent set in $G \Leftrightarrow S$ is clique in $\overline{G} \doteq (V, \overline{E})$.
- S is independent set in $G \Leftrightarrow \overline{S}$ is vertex cover in G.

Corollary

- Independent Set $\leq^p$ Clique
- Independent Set $\leq^p$ Vertex Cover

Satisfiability and Coloring

Satisfiability and Coloring

3-SAT

Satisfiability and Coloring

3-SAT

Input: 3-CNF formula φ

Satisfiability and Coloring

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Satisfiability and Coloring

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

Satisfiability and Coloring

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

3-Coloring

Satisfiability and Coloring

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

3-Coloring

Input: graph $G = (V, E)$

Satisfiability and Coloring

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

3-Coloring

Input: graph $G = (V, E)$

Output: whether G has a 3-coloring, i.e., a mapping $c : V \to [3]$ such that $(\forall\,(u, v) \in E)\, c(u) \neq c(v))$.

3-SAT $\leq^p$ 3-Coloring

3-SAT $\leq^p$ 3-Coloring – variable gadgets

3-SAT $\leq^p$ 3-Coloring – variable gadgets

- Include a color palette: complete graph on vertices $\{\text{red}, \text{green}, \text{blue}\}$

3-SAT $\leq^p$ 3-Coloring – variable gadgets

- Include a color palette: complete graph on vertices $\{\text{red}, \text{green}, \text{blue}\}$
- For each variable x_i, include two new vertices, one labeled x_i and the other $\overline{x_i}$.

3-SAT $\leq^p$ 3-Coloring – variable gadgets

- Include a color palette: complete graph on vertices $\{\text{red}, \text{green}, \text{blue}\}$
- For each variable x_i, include two new vertices, one labeled x_i and the other $\overline{x_i}$.
- Include the edges $(x_i, \overline{x_i})$, (x_i, blue), and $(\overline{x_i}, \text{blue})$.

3-SAT $\leq^p$ 3-Coloring – variable gadgets

- Include a color palette: complete graph on vertices $\{\text{red}, \text{green}, \text{blue}\}$
- For each variable x_i, include two new vertices, one labeled x_i and the other $\overline{x_i}$.
- Include the edges $(x_i, \overline{x_i})$, (x_i, blue), and $(\overline{x_i}, \text{blue})$.
- Bijection between assignments to variables $x_1, \ldots, x_n$ and valid colorings with $\{\text{red}, \text{green}, \text{blue}\}$.

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

- For each 3-clause C_j, include a complete graph on 3 new vertices, each labeled with a unique literal of C_j.

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

- For each 3-clause C_j, include a complete graph on 3 new vertices, each labeled with a unique literal of C_j.
- Include for each new vertex v with label ℓ, a new vertex v'.

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

- For each 3-clause C_j, include a complete graph on 3 new vertices, each labeled with a unique literal of C_j.
- Include for each new vertex v with label ℓ, a new vertex v'.
- Include the edges (v, v'), (v', green), and (v', u), where u denotes the vertex in the variable gadget labeled ℓ.

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

- For each 3-clause C_j, include a complete graph on 3 new vertices, each labeled with a unique literal of C_j.
- Include for each new vertex v with label ℓ, a new vertex v'.
- Include the edges (v, v'), (v', green), and (v', u), where u denotes the vertex in the variable gadget labeled ℓ.
- A valid 3-coloring to the variable gadget can be extended to gadget for clause C_j iff underlying assignment satisfies C_j.

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

- For each 3-clause C_j, include a complete graph on 3 new vertices, each labeled with a unique literal of C_j.
- Include for each new vertex v with label ℓ, a new vertex v'.
- Include the edges (v, v'), (v', green), and (v', u), where u denotes the vertex in the variable gadget labeled ℓ.
- A valid 3-coloring to the variable gadget can be extended to gadget for clause C_j iff underlying assignment satisfies C_j.
- Clauses with less than 3 literals can be handled by repeating a literal in the clause until there are three.

3-SAT $\leq^p$ 3-Coloring – clause gadgets & connections

- For each 3-clause C_j, include a complete graph on 3 new vertices, each labeled with a unique literal of C_j.
- Include for each new vertex v with label ℓ, a new vertex v'.
- Include the edges (v, v'), (v', green), and (v', u), where u denotes the vertex in the variable gadget labeled ℓ.
- A valid 3-coloring to the variable gadget can be extended to gadget for clause C_j iff underlying assignment satisfies C_j.
- Clauses with less than 3 literals can be handled by repeating a literal in the clause until there are three.

Conclusion

φ is satisfiable $\Leftrightarrow$ G is 3-colorable

Satisfiability and Subset Sum

Satisfiability and Subset Sum

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

Satisfiability and Subset Sum

3-SAT

Input: 3-CNF formula φ

E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

Subset Sum

Satisfiability and Subset Sum

3-SAT

Input: 3-CNF formula φ
E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

Subset Sum

Input: $a_1, a_2, \ldots, a_k \in \mathbb{N}$; $t \in \mathbb{N}$

Satisfiability and Subset Sum

3-SAT

Input: 3-CNF formula φ
E.g.: $\varphi = (x_1 \lor \overline{x_2} \lor x_4) \land (\overline{x_1} \lor x_2)$

Output: whether φ has a satisfying assignment.

Subset Sum

Input: $a_1, a_2, \ldots, a_k \in \mathbb{N}$; $t \in \mathbb{N}$

Output: whether there exists $I \subseteq [k]$ such that $\sum_{i \in I} a_i = t$.

Satisfiability and Subset Sum

3-SAT

Input: 3-CNF formula φ
E.g.: $\varphi = (x_1 \vee \overline{x_2} \vee x_4) \wedge (\overline{x_1} \vee x_2)$

Output: whether φ has a satisfying assignment.

Subset Sum

Input: $a_1, a_2, \ldots, a_k \in \mathbb{N}$; $t \in \mathbb{N}$

Output: whether there exists $I \subseteq [k]$ such that $\sum_{i \in I} a_i = t$.

- Subset Sum $\leq^p$ Knapsack

3-SAT $\leq^p$ Subset Sum

3-SAT $\leq^p$ Subset Sum – variable gadgets

- For each variable x_i, include two numbers $a_{2i-1} = a_{2i} = 2^{i-1}$.

3-SAT $\leq^p$ Subset Sum – variable gadgets

- For each variable x_i, include two numbers $a_{2i-1} = a_{2i} = 2^{i-1}$.
- Label a_{2i-1} with x_i, and a_{2i} with $\overline{x_i}$.

3-SAT $\leq^p$ Subset Sum – variable gadgets

- For each variable x_i, include two numbers $a_{2i-1} = a_{2i} = 2^{i-1}$.
- Label a_{2i-1} with x_i, and a_{2i} with $\overline{x_i}$.
- Set $t = \sum_{i=1}^{n} 2^{i-1} = 2^n - 1$.

3-SAT $\leq^p$ Subset Sum – variable gadgets

- For each variable x_i, include two numbers $a_{2i-1} = a_{2i} = 2^{i-1}$.
- Label a_{2i-1} with x_i, and a_{2i} with $\overline{x_i}$.
- Set $t = \sum_{i=1}^{n} 2^{i-1} = 2^n - 1$.
- Bijection between assignments to variables $x_1, \ldots, x_n$ and subsets $I \subseteq [2n]$ such that $\sum_{i \in I} a_i = t$.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.
- For each literal ℓ in C_j, set bits in B_j of the number a_i labeled ℓ to 01.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.
- For each literal ℓ in C_j, set bits in B_j of the number a_i labeled ℓ to 01.
- Set bits in B_j of t to k_j in binary.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.
- For each literal ℓ in C_j, set bits in B_j of the number a_i labeled ℓ to 01.
- Set bits in B_j of t to k_j in binary.
- Create $k_j - 1$ new numbers with bits in B_j set to 01.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.
- For each literal ℓ in C_j, set bits in B_j of the number a_i labeled ℓ to 01.
- Set bits in B_j of t to k_j in binary.
- Create $k_j - 1$ new numbers with bits in B_j set to 01.
- Consider subset $I \subseteq [2n]$ such that $\sum_{i \in I}^{n} a_i = t \bmod 2^n$.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.
- For each literal ℓ in C_j, set bits in B_j of the number a_i labeled ℓ to 01.
- Set bits in B_j of t to k_j in binary.
- Create $k_j - 1$ new numbers with bits in B_j set to 01.
- Consider subset $I \subseteq [2n]$ such that $\sum_{i \in I}^{n} a_i = t \bmod 2^n$. I can be extended with subset of new numbers to I' such that $\sum_{i \in I'} a_i$ equals t in positions B_j iff underlying assignment satisfies C_j.

3-SAT $\leq^p$ Subset Sum – clause gadgets & connections

For each clause C_j with k_j literals:

- Pick bit two new consecutive bit positions B_j.
- For each literal ℓ in C_j, set bits in B_j of the number a_i labeled ℓ to 01.
- Set bits in B_j of t to k_j in binary.
- Create $k_j - 1$ new numbers with bits in B_j set to 01.
- Consider subset $I \subseteq [2n]$ such that $\sum_{i \in I}^{n} a_i = t \bmod 2^n$. I can be extended with subset of new numbers to I' such that $\sum_{i \in I'} a_i$ equals t in positions B_j iff underlying assignment satisfies C_j.

Conclusion

φ is satisfiable $\Leftrightarrow$ there exists I' such that $\sum_{i \in I'} a_i = t$.

Classical NP-Complete Problems

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

  3-D Matching

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

3-D Matching

Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

3-D Matching

Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$

Output: whether there exists $S \subseteq T$ such that each element of $X \cup Y \cup Z$ appears exactly once

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

  ### 3-D Matching

  Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$
  Output: whether there exists $S \subseteq T$ such that each element of $X \cup Y \cup Z$ appears exactly once

- Sequencing: Traveling Salesperson

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

  3-D Matching

  Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$

  Output: whether there exists $S \subseteq T$ such that each element of $X \cup Y \cup Z$ appears exactly once

- Sequencing: Traveling Salesperson

  Hamiltonicity

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

  ### 3-D Matching

  Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$
  Output: whether there exists $S \subseteq T$ such that each element of $X \cup Y \cup Z$ appears exactly once

- Sequencing: Traveling Salesperson

  ### Hamiltonicity

  Input: (di)graph $G = (V, E)$

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

  ### 3-D Matching

  Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$
  Output: whether there exists $S \subseteq T$ such that each element of $X \cup Y \cup Z$ appears exactly once

- Sequencing: Traveling Salesperson

  ### Hamiltonicity

  Input: (di)graph $G = (V, E)$
  Output: whether there exists a (directed) cycle/path that visits every vertex once

Classical NP-Complete Problems

- Constraint satisfaction: Circuit-SAT, CNF-SAT, 3-SAT
- Packing: Independent Set, Clique
- Covering: Vertex Cover
- Partitioning: 3-Colorability

  3-D Matching

  Input: disjoint sets X, Y, Z; $T \subseteq X \times Y \times Z$
  Output: whether there exists $S \subseteq T$ such that each element of $X \cup Y \cup Z$ appears exactly once

- Sequencing: Traveling Salesperson

  Hamiltonicity

  Input: (di)graph $G = (V, E)$
  Output: whether there exists a (directed) cycle/path that visits every vertex once

- Numerical: Subset Sum, Knapsack