Complexity Theory

Complexity Classes
May 9, 2022

iCIS | Software Science
Radboud University

Decision Problems

Complexity Classes

NP-hardness and **NP**-completeness

Outline

Decision Problems

Complexity Classes

NP-hardness and **NP**-completeness

Basics on Decision Problems

- Decision problem: a yes/no-problem. So, does a given input satisfy a certain property?
- Decision problems represent **subsets** of $\{0, 1\}^*$.

Basics on Decision Problems

- Decision problem: a yes/no-problem. So, does a given input satisfy a certain property?
- Decision problems represent **subsets** of $\{0, 1\}^*$.
- Examples of decision problems:
  (i) Given $n \in \mathbb{N}$, is n prime?
  (ii) Given a graph G, does G have a Hamiltonian path?
  (iii) Given a graph G, does G have an Euler path?
  (iv) Given a formula φ, is φ satisfiable?

Recall

- Hamiltonian path: visits every **node** exactly once
- Euler path: visits every **edge** exactly once

Encodings

- More complicated data types (graphs, formulas) need to be encoded.
- So: represent the set of graphs/formulas as subsets of $\{0, 1\}^*$
- Precisely defining such an encoding is “high effort, little gain”.
- We will leave the encodings implicit.

Recall: Operations on Languages

Recall from Languages and Automata:

- $A \cap B = \{x \in A \mid x \in B\}$
- $x \in A \cup B$ if and only if $x \in A$ or $x \in B$
- $\overline{A} = \{w \in \{0,1\}^* \mid w \notin A\}$
- $x \in AB$ if we can write $x = vw$ with $v \in A$ and $w \in B$
- $x \in A^*$ if we can write $x = w_1 \dots w_n$ with $w_i \in A$ for all i

Note: we work with subsets of $\{0,1\}^*$.

Outline

Decision Problems

Complexity Classes

NP-hardness and **NP**-completeness

Polynomial Decision Problems

Definition

Let X be a decision problem. An algorithm A **decides** X if for all $w \in \{0, 1\}^*$, we have $w \in X$ if and only if $A(w) = 1$.

Polynomial Decision Problems

Definition

Let X be a decision problem. An algorithm A **decides** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if $A(w) = 1$.

Definition

An algorithm A **runs in polynomial time** if for its time complexity we have $T(n) = \mathcal{O}(n^k)$ for some k

Polynomial Decision Problems

Definition

Let X be a decision problem. An algorithm A **decides** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if $A(w) = 1$.

Definition

An algorithm A **runs in polynomial time** if for its time complexity we have $T(n) = \mathcal{O}(n^k)$ for some k

Definition

A decision problem X is said to be **polynomial** if there is an algorithm that runs in polynomial time and that decides X.
If this is the case, we write $X \in \mathbf{P}$.

Polynomial Decision Problems

Definition
Let X be a decision problem. An algorithm A **decides** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if $A(w) = 1$.

Definition
An algorithm A **runs in polynomial time** if for its time complexity we have $T(n) = \mathcal{O}(n^k)$ for some k

Definition
A decision problem X is said to be **polynomial** if there is an algorithm that runs in polynomial time and that decides X.
If this is the case, we write $X \in \mathbf{P}$.

Some Decision Problems

Are the following decision problems in **P**?

- Given $n \in \mathbb{N}$, is n even?

Some Decision Problems

Are the following decision problems in **P**?

- Given $n \in \mathbb{N}$, is n even?
- Given a formula φ, does φ contain a negation?

Some Decision Problems

Are the following decision problems in **P**?

- Given $n \in \mathbb{N}$, is n even?
- Given a formula φ, does φ contain a negation?
- Given a graph G and nodes x and y, is there a path from x to y?
  Hint: think of what you learned in Algorithms and Data Structures!

Some Decision Problems

Are the following decision problems in **P**?

- Given $n \in \mathbb{N}$, is n even?
- Given a formula φ, does φ contain a negation?
- Given a graph G and nodes x and y, is there a path from x to y?
  Hint: think of what you learned in Algorithms and Data Structures!
- Given a formula φ, is φ in conjunctive normal form?
  A formula is in conjunctive normal form if it is a conjunction of disjunctions of possibly negated atoms
  Examples: $(x \vee \neg y) \wedge (x \vee y)$, $\neg x \vee \neg y$
  Counterexamples: $(x \wedge y) \to z$, $(x \wedge y) \vee (x \wedge \neg y)$

Some Decision Problems

Are the following decision problems in **P**?

- Given $n \in \mathbb{N}$, is n even?
- Given a formula φ, does φ contain a negation?
- Given a graph G and nodes x and y, is there a path from x to y?
  Hint: think of what you learned in Algorithms and Data Structures!
- Given a formula φ, is φ in conjunctive normal form?
  A formula is in conjunctive normal form if it is a conjunction of disjunctions of possibly negated atoms
  Examples: $(x \vee \neg y) \wedge (x \vee y)$, $\neg x \vee \neg y$
  Counterexamples: $(x \wedge y) \rightarrow z$, $(x \wedge y) \vee (x \wedge \neg y)$

Closure Operations

Theorem

- *If* $A \in \mathbf{P}$, *then* $\overline{A} \in \mathbf{P}$ *(complement)*
- *If* $A \in \mathbf{P}$ *and* $B \in \mathbf{P}$, *then* $A \cup B \in \mathbf{P}$ *(union)*
- *If* $A \in \mathbf{P}$ *and* $B \in \mathbf{P}$, *then* $A \cap B \in \mathbf{P}$ *(intersection)*
- *If* $A \in \mathbf{P}$ *and* $B \in \mathbf{P}$, *then* $A \cdot B \in \mathbf{P}$ *(concatenation)*

A short remark on encodings

Definition
Suppose, we have two encodings $e_1, e_2 : I \to \{0,1\}^*$.
We say that e_1 and e_2 are **polynomially related** if we have polynomial functions $f, g : I \to I$ such that for all $i \in I$ we have

$$f(e_1(i)) = e_2(i)$$

$$g(e_2(i)) = e_1(i)$$

A short remark on encodings

Definition
Suppose, we have two encodings $e_1, e_2 : I \to \{0,1\}^*$.
We say that e_1 and e_2 are **polynomially related** if we have polynomial functions $f, g : I \to I$ such that for all $i \in I$ we have

$$f(e_1(i)) = e_2(i)$$

$$g(e_2(i)) = e_1(i)$$

Note: if e_1 and e_2 are polynomial related, then we can decide Q in polynomial time with encoding e_1 if and only if we can decide Q in polynomial time with encoding e_2.

Nondeterministic Polynomial Decision Problems

Definition
An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ **verifies** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if there is a $y \in \{0,1\}^*$ such that $A(w, y) = 1$. This y is called **the certificate**.

Nondeterministic Polynomial Decision Problems

Definition

An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ **verifies** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if there is a $y \in \{0,1\}^*$ such that $A(w, y) = 1$. This y is called **the certificate**.

Definition

An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ verifies X **in nondeterministic polynomial time** if A verifies X and if for its time complexity we have $T(n) = \mathcal{O}(n^k)$ for some k.

Nondeterministic Polynomial Decision Problems

Definition
An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ **verifies** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if there is a $y \in \{0,1\}^*$ such that $A(w, y) = 1$. This y is called **the certificate**.

Definition
An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ verifies X **in nondeterministic polynomial time** if A verifies X and if for its time complexity we have $T(n) = \mathcal{O}(n^k)$ for some k.

Definition
We say that $X \in$ **NP** if there is an algorithm A that verifies X in nondeterministic polynomial time.

Nondeterministic Polynomial Decision Problems

Definition
An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ **verifies** X if for all $w \in \{0,1\}^*$, we have $w \in X$ if and only if there is a $y \in \{0,1\}^*$ such that $A(w,y) = 1$. This y is called **the certificate**.

Definition
An algorithm $A : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$ verifies X **in nondeterministic polynomial time** if A verifies X and if for its time complexity we have $T(n) = \mathcal{O}(n^k)$ for some k.

Definition
We say that $X \in$ **NP** if there is an algorithm A that verifies X in nondeterministic polynomial time.

Examples

Are the following problems in **NP**?

- Given $n \in \mathbb{N}$, is n a composite number?

Examples

Are the following problems in **NP**?

- Given $n \in \mathbb{N}$, is n a composite number?
- Given a formula φ, is φ satisfiable?

Examples

Are the following problems in **NP**?

- Given $n \in \mathbb{N}$, is n a composite number?
- Given a formula φ, is φ satisfiable?
- Given a graph G, does G have a Hamiltonian path?

Examples

Are the following problems in **NP**?

- Given $n \in \mathbb{N}$, is n a composite number?
- Given a formula φ, is φ satisfiable?
- Given a graph G, does G have a Hamiltonian path?
- Suppose, we have n items with weight w_i and value v_i. Can we pick items in such a way that the sum of values is at least V and the sum of the weights is at most W? (*Knapsack problem*)

Examples

Are the following problems in **NP**?

- Given $n \in \mathbb{N}$, is n a composite number?
- Given a formula φ, is φ satisfiable?
- Given a graph G, does G have a Hamiltonian path?
- Suppose, we have n items with weight w_i and value v_i. Can we pick items in such a way that the sum of values is at least V and the sum of the weights is at most W? (*Knapsack problem*)
- Given an $n^2 \times n^2$ Sudoku, does it have a solution?

Examples

Are the following problems in **NP**?

- Given $n \in \mathbb{N}$, is n a composite number?
- Given a formula φ, is φ satisfiable?
- Given a graph G, does G have a Hamiltonian path?
- Suppose, we have n items with weight w_i and value v_i. Can we pick items in such a way that the sum of values is at least V and the sum of the weights is at most W? (*Knapsack problem*)
- Given an $n^2 \times n^2$ Sudoku, does it have a solution?

Closure Operations

Theorem

- *If* $A \in \mathbf{NP}$ *and* $B \in \mathbf{NP}$*, then* $A \cup B \in \mathbf{NP}$ *(union)*
- *If* $A \in \mathbf{NP}$ *and* $B \in \mathbf{NP}$*, then* $A \cap B \in \mathbf{NP}$ *(intersection)*
- *If* $A \in \mathbf{NP}$ *and* $B \in \mathbf{NP}$*, then* $A \cdot B \in \mathbf{NP}$ *(concatenation)*

Open problem: if $A \in \mathbf{NP}$, is $\overline{A} \in \mathbf{NP}$?

Intermezzo: non-deterministic Turing Machines

Using Turing machines, we can define **P** and **NP** as follows:

- $X \in$ **P** if there is a **deterministic** Turing machine running in polynomial time that decides X
- $X \in$ **NP** if there is a **nondeterministic** Turing machine running in polynomial time that decides X

This is why we call **NP** nondeterministic polynomial.

P versus NP

- Is **P** a subset of **NP**?

P versus NP

- Is **P** a subset of **NP**?
  Yes!

P versus NP

- Is **P** a subset of **NP**?
  Yes!
- Are **P** and **NP** equal?

P versus NP

- Is **P** a subset of **NP**?
  Yes!
- Are **P** and **NP** equal?
  That is an open problem

Famous problem in computer science: do we have $\mathbf{P} = \mathbf{NP}$?

Outline

Decision Problems

Complexity Classes

NP-hardness and **NP**-completeness

Reducibility

Definition
We say that A **(polynomially) reduces to** B if we have a function $f : \{0,1\}^* \to \{0,1\}^*$ such that

- $f \in \mathcal{O}(n^k)$ for some k;
- for all $w \in \{0,1\}^*$ we have $w \in A$ if and only if $f(w) \in B$.

If this is the case, we write $A \leq_P B$.

Properties of Reduction

Theorem

- *For each decision problem A, we have $A \leq_P A$*
- *If $A \leq_P B$ and $B \leq_P C$, then we also have $A \leq_P C$*
- *If $A \leq_P B$ and B is in* **P**, *then $A \in$* **P**
- *If $A \leq_P B$ and B is in* **NP**, *then $A \in$* **NP**

Properties of Reduction

Theorem

- *For each decision problem A, we have $A \leq_P A$*
- *If $A \leq_P B$ and $B \leq_P C$, then we also have $A \leq_P C$*
- *If $A \leq_P B$ and B is in* **P***, then $A \in$* **P**
- *If $A \leq_P B$ and B is in* **NP***, then $A \in$* **NP**

NP-hard and NP-complete

Definition
A decision problem B is said to be **NP-hard** if for all decision problems $A \in \mathbf{NP}$ we have $A \leq_P B$.

NP-hard and NP-complete

Definition
A decision problem B is said to be **NP-hard** if for all decision problems $A \in$ **NP** we have $A \leq_P B$.

Definition
A decision problem B is said to be **NP-complete** if $B \in$ **NP** and if B is **NP**-hard.

NP-hard and NP-complete

Definition

A decision problem B is said to be **NP-hard** if for all decision problems $A \in$ **NP** we have $A \leq_P B$.

Definition

A decision problem B is said to be **NP-complete** if $B \in$ **NP** and if B is **NP**-hard.

Theorem

If A reduces to B, A is **NP***-hard, then B is* **NP***-hard*

NP-hard and NP-complete

Definition

A decision problem B is said to be **NP-hard** if for all decision problems $A \in \mathbf{NP}$ we have $A \leq_P B$.

Definition

A decision problem B is said to be **NP-complete** if $B \in \mathbf{NP}$ and if B is **NP**-hard.

Theorem

If A reduces to B, A is **NP***-hard, then B is* **NP***-hard*

Proof.

Suppose, $A \leq_P B$ and A is **NP**-hard.
Let $X \in \mathbf{NP}$. To show: $X \leq_P B$.
Since A is **NP**-hard: $X \leq_P A$.
Since we also have $A \leq_P B$, we get $X \leq_P B$. □

NP-hard and NP-complete

Definition
A decision problem B is said to be **NP-hard** if for all decision problems $A \in \mathbf{NP}$ we have $A \leq_P B$.

Definition
A decision problem B is said to be **NP-complete** if $B \in \mathbf{NP}$ and if B is **NP**-hard.

Theorem
If A reduces to B, A is **NP***-hard, then B is* **NP***-hard*

Proof.
Suppose, $A \leq_P B$ and A is **NP**-hard.
Let $X \in \mathbf{NP}$. To show: $X \leq_P B$.
Since A is **NP**-hard: $X \leq_P A$.
Since we also have $A \leq_P B$, we get $X \leq_P B$. □

SAT

The **Boolean satisfiability problem**, also known as SAT is the following problem

> Given a formula φ, is φ satisfiable? So: is there an assignment of atoms to truth values under which φ is true?

We already saw: SAT is in **NP**.
In fact: SAT is **NP**-complete.

Variations of SAT

Definition
A formula is in **disjunctive normal form normal form** if it is written as disjunction of conjunctions of possibly negated atoms.

Variations of SAT

Definition
A formula is in **disjunctive normal form normal form** if it is written as disjunction of conjunctions of possibly negated atoms.

Examples:

- $(A \wedge \neg B \wedge C) \vee (A \wedge C) \vee (A \wedge B \wedge \neg B)$
- $A \wedge \neg B \wedge C$

Not in DNF:

- $(A \vee \neg B \wedge C) \vee (A \wedge C)$
- $A \wedge \neg(B \vee C) \wedge C$

Variations of SAT

Definition
A formula is in **disjunctive normal form normal form** if it is written as disjunction of conjunctions of possibly negated atoms.

Examples:

- $(A \wedge \neg B \wedge C) \vee (A \wedge C) \vee (A \wedge B \wedge \neg B)$
- $A \wedge \neg B \wedge C$

Not in DNF:

- $(A \vee \neg B \wedge C) \vee (A \wedge C)$
- $A \wedge \neg(B \vee C) \wedge C$

SAT for formulas in DNF is in **P**.

Variations of SAT

Definition
A formula is in **conjunctive normal form** if it is written as conjunction of disjunctions of possibly negated atoms.

Variations of SAT

Definition
A formula is in **conjunctive normal form** if it is written as conjunction of disjunctions of possibly negated atoms.

Examples:

- $(A \vee \neg B \vee C) \wedge (A \vee C) \wedge (A \vee B \vee \neg B)$
- $A \vee \neg B \vee C$

Not in CNF:

- $(A \vee \neg B \wedge C) \wedge (A \vee C)$
- $A \vee \neg(B \wedge C) \vee C$

Variations of SAT

Definition
A formula is in **conjunctive normal form** if it is written as conjunction of disjunctions of possibly negated atoms.

Examples:

- $(A \lor \neg B \lor C) \land (A \lor C) \land (A \lor B \lor \neg B)$
- $A \lor \neg B \lor C$

Not in CNF:

- $(A \lor \neg B \land C) \land (A \lor C)$
- $A \lor \neg(B \land C) \lor C$

SAT for formulas in CNF is **NP**-complete.
We call this problem: CNF.

Putting formulas in CNF

Note: every formula ϕ is equivalent to a formula ψ in CNF.
One way to determine ψ:

- Remove (bi)implications (use $A \to B \equiv \neg A \vee B$)
- Put negations next to literals (use $\neg(A \wedge B) \equiv \neg A \vee \neg B$ and $\neg(A \vee B) \equiv \neg A \wedge \neg B$)
- Put in CNF using $(A \wedge B) \vee C \equiv (A \vee C) \wedge (B \vee C)$