# 22.2.5
# Intractability

# **P** versus **NP**

**Proposition 22.6.**

**P** $\subseteq$ **NP**.

For a problem in **P** no need for a certificate!

Proof.

Consider problem $X \in$ **P** with algorithm $A$. Need to demonstrate that $X$ has an efficient certifier:

1. Certifier $C$ on input $s, t$, runs $A(s)$ and returns the answer.
2. $C$ runs in polynomial time.
3. If $s \in X$, then for every $t$, $C(s, t) =$ "yes".
4. If $s \notin X$, then for every $t$, $C(s, t) =$ "no". □

# P versus NP

**Proposition 22.6.**

**P** $\subseteq$ **NP**.

For a problem in **P** no need for a certificate!

**Proof.**

Consider problem $X \in$ **P** with algorithm $A$. Need to demonstrate that $X$ has an efficient certifier:

1. Certifier $C$ on input $s, t$, runs $A(s)$ and returns the answer.
2. $C$ runs in polynomial time.
3. If $s \in X$, then for every $t$, $C(s, t) =$ "yes".
4. If $s \notin X$, then for every $t$, $C(s, t) =$ "no". ☐

# Exponential Time

**Definition 22.7.**

**Exponential Time** (denoted **EXP**) is the collection of all problems that have an algorithm which on input $s$ runs in exponential time, i.e., $O(2^{\mathrm{poly}(|s|)})$.

Example: $O(2^n)$, $O(2^{n \log n})$, $O(2^{n^3})$, ...

# Exponential Time

**Definition 22.7.**

**Exponential Time** (denoted **EXP**) is the collection of all problems that have an algorithm which on input $s$ runs in exponential time, i.e., $O(2^{\mathrm{poly}(|s|)})$.

Example: $O(2^n)$, $O(2^{n \log n})$, $O(2^{n^3})$, ...

# NP versus EXP

**Proposition 22.8.**

**NP** $\subseteq$ **EXP**.

Proof.

Let $X \in$ **NP** with certifier $C$. Need to design an exponential time algorithm for $X$.

1. For every $t$, with $|t| \leq p(|s|)$ run $C(s,t)$; answer "yes" if any one of these calls returns "yes".
2. The above algorithm correctly solves $X$ (exercise).
3. Algorithm runs in $O(q(|s| + |p(s)|)2^{p(|s|)})$, where $q$ is the running time of $C$. □

# Examples

1. **SAT**: try all possible truth assignment to variables.
2. **Independent Set**: try all possible subsets of vertices.
3. **Vertex Cover**: try all possible subsets of vertices.

# Is **NP** efficiently solvable?

We know $\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{EXP}$.

# Is **NP** efficiently solvable?

We know **P** $\subseteq$ **NP** $\subseteq$ **EXP**.

## Big Question

Is there are problem in **NP** that does not belong to **P**? Is **P** $=$ **NP**?

# If $\mathbf{P} = \mathbf{NP}\ldots$

Or: If pigs could fly then life would be sweet.

1. Many important optimization problems can be solved efficiently.
2. The RSA cryptosystem can be broken.
3. No security on the web.
4. No e-commerce ...
5. Creativity can be automated! Proofs for mathematical statement can be found by computers automatically (if short ones exist).

# If $\mathbf{P} = \mathbf{NP}\ldots$

Or: If pigs could fly then life would be sweet.

1. Many important optimization problems can be solved efficiently.
2. The $\mathrm{RSA}$ cryptosystem can be broken.
3. No security on the web.
4. No e-commerce ...
5. Creativity can be automated! Proofs for mathematical statement can be found by computers automatically (if short ones exist).

# If $\mathbf{P} = \mathbf{NP} \ldots$

Or: If pigs could fly then life would be sweet.

1. Many important optimization problems can be solved efficiently.
2. The $\mathrm{RSA}$ cryptosystem can be broken.
3. No security on the web.
4. No e-commerce . . .
5. Creativity can be automated! Proofs for mathematical statement can be found by computers automatically (if short ones exist).

# If $\mathbf{P} = \mathbf{NP}$ . . .

Or: If pigs could fly then life would be sweet.

1. Many important optimization problems can be solved efficiently.
2. The RSA cryptosystem can be broken.
3. No security on the web.
4. No e-commerce . . .
5. Creativity can be automated! Proofs for mathematical statement can be found by computers automatically (if short ones exist).

# If $\mathbf{P} = \mathbf{NP} \ldots$

Or: If pigs could fly then life would be sweet.

1. Many important optimization problems can be solved efficiently.
2. The RSA cryptosystem can be broken.
3. No security on the web.
4. No e-commerce . . .
5. Creativity can be automated! Proofs for mathematical statement can be found by computers automatically (if short ones exist).

# **P** versus **NP**

## Status

Relationship between **P** and **NP** remains one of the most important open problems in mathematics/computer science.

Consensus: Most people feel/believe $\boldsymbol{P} \neq \boldsymbol{NP}$.

Resolving **P** versus **NP** is a Clay Millennium Prize Problem. You can win a million dollars in addition to a Turing award and major fame!

# Review question: If **P** = **NP** this implies that...

(A) **Vertex Cover** can be solved in polynomial time.

(B) **P** = **EXP**.

(C) **EXP** $\subseteq$ **P**.

(D) All of the above.

# THE END

...

# (for now)