
Stockholm University

# Tutorial 2

TA: Anna Lindeberg

# Quiz!

At least one answer is correct for each question.

# Quiz!

At least one answer is correct for each question.

## Question 1

What is true for every problem in $P$?

(a) they can be solved by a deterministic algorithm in polynomial time
(b) they are decision problems
(c) they are optimization problems
(d) they can be solved by a non-deterministic algorithm in polynomial time
(e) they are verifiable in polynomial time

# Answers

## Question 1

What is true for every problem in $P$?

(a) **they can be solved by a deterministic algorithm in polynomial time**

(b) **they are decision problems**

(c) they are optimization problems

(d) **they can be solved by a non-deterministic algorithm in polynomial time**

(e) **they are verifiable in polynomial time**

# Quiz!

## Question 2

What is true for every problem in $NP$?

(a) they can be solved by a deterministic algorithm in polynomial time
(b) they are decision problems
(c) they are optimization problems
(d) they can be solved by a non-deterministic algorithm in polynomial time
(e) they are verifiable in polynomial time

# Answers

## Question 2

What is true for every problem in $NP$?

(a) **???** they can be solved by a deterministic algorithm in polynomial time **???** **(we don't know!)**

(b) **they are decision problems**

(c) they are optimization problems

(d) **they can be solved by a non-deterministic algorithm in polynomial time**

(e) **they are verifiable in polynomial time**

# Quiz!

## Question 3

What is $P$ respectively $NP$ short for?

(a) **P**roblematic respectively **N**on-**P**roblematic
(b) **P**olynomial respectively **N**on-deterministic **P**olynomial
(c) **P**olynomial respectively **N**on-**P**olynomial
(d) Easy-**P**easy respectively **N**ot Easy-**P**easy

# Answers

## Question 3

What is $P$ respectively $NP$ short for?

(a) **P**roblematic respectively **N**on-**P**roblematic

(b) **Polynomial respectively *N*on-deterministic *P*olynomial**

(c) **P**olynomial respectively **N**on-**P**olynomial

(d) Easy-**P**easy respectively **N**ot Easy-**P**easy

# Quiz!

## Question 4

How do we prove that a decision problem lies in the class $NP$?

(a) Provide a deterministic algorithm that solves the problem in polynomial time

(b) Provide a deterministic algorithm that verifies if a certificate[1] is correct

(c) Reduce the problem (in polynomial time) to another decision problem we already know lies in $NP$

(d) Reduce (in polynomial time) a decision problem we already know lies in $NP$ to the problem in question

(e) Provide a non-deterministic algorithm that solves the problem in polynomial time

[1] i.e. YES-instance

# Answers

## Question 4

How can we prove that a decision problem lies in the class $NP$?

(a) **Provide a deterministic algorithm that solves the problem in polynomial time**

(b) **Provide a deterministic algorithm that verifies if a certificate is correct**

(c) Reduce the problem to another decision problem we already know lies in $NP$

(d) Reduce a decision problem we already know lies in $NP$ to the problem in question

(e) **Provide a non-deterministic algorithm that solves the problem in polynomial time**

# Quiz!

## Question 5

What are we absolutely certain of?

(a) Every problem in $P$ lies in $NP$

(b) There is a problem in $NP$ that is not in $P$

(c) Every problem in $NP$ lies in $P$

(d) Every $NP$-hard problem is $NP$-complete

(e) Every $NP$-complete problem is $NP$-hard

(f) Every problem in $NP$ is $NP$-complete

# Answers

## Question 5

What are we absolutely certain of?

(a) **Every problem in $P$ lies in $NP$**

(b) There is a problem in $NP$ that is not in $P$

(c) Every problem in $NP$ lies in $P$

(d) Every $NP$-hard problem is $NP$-complete

(e) **Every $NP$-complete problem is $NP$-hard**

(f) Every problem in $NP$ is $NP$-complete

# Quiz!

## Question 6

What is true for every $NP$-hard problem?

(a) They lie in $NP$ and are $NP$-complete
(b) They are optimization problem
(c) They are decision problems
(d) They are at least as difficult to solve as any other $NP$-hard problem
(e) They can be solved by a non-deterministic algorithm in polynomial time

# Answers

## Question 6

What is true for every $NP$-hard problem?

(a) They lie in $NP$ and are $NP$-complete

(b) They are optimization problem

(c) They are decision problems

(d) **They are at least as difficult to solve as any other $NP$-hard problem**

(e) They can be solved by a non-deterministic algorithm in polynomial time

Short arguments for why:

| Problems | Why correct? | Problems | Why incorrect? |
|---|---|---|---|
| 1(a), 1(b), 2(b), 2(d), 2(e) | Definition of $P$ resp. $NP$ | 1(c), 2(c) | Definition of $P$ resp. $NP$ |
| 1(d), 1(e) | $P \subseteq NP$ | 3(a),3(d) | Nonsense |
| 3(b) | Just correct | 3(c) | Just incorrect (common mistake) |
| 4(a), 5(a) | $P \subseteq NP$ | 4(c), 4(d) | You would need a reduction in both directions. Just one is not enough! |

Short arguments for why:

| Problems | Why correct? | Problems | Why incorrect? |
|---|---|---|---|
| 4(b), 4(e) | Definition of $P$ resp. $NP$ | 5(b), 5(c), 5(f) | We don't know wether $P = NP$ or $P \subsetneq NP$ |
| 5(e) | Definition of $NP$-complete | 5(d), 6(a) | E.g. optimization problems may be $NP$-hard, but definitely not in $NP$ |
| 6(d) | Reductions | 6(b), 6(c) | $NP$-hard can be either optimization or decision problems |
| | | 6(e) | Counterex: halting problem |

# Highlights

- $P$ and $NP$ contains decision problems

# Highlights

- $P$ and $NP$ contains decision problems
- $P \subseteq NP$ is known, but $P = NP$ or $P \subsetneq NP$ is open

# Highlights

- $P$ and $NP$ contains decision problems
- $P \subseteq NP$ is known, but $P = NP$ or $P \subsetneq NP$ is open
- $NP$-complete $\subsetneq$ $NP$-hard, e.g. optimization problems can be $NP$-hard

# Highlights

- $P$ and $NP$ contains decision problems
- $P \subseteq NP$ is known, but $P = NP$ or $P \subsetneq NP$ is open
- $NP$-complete $\subsetneq NP$-hard, e.g. optimization problems can be $NP$-hard
- Show in $P$ by providing polynomial-time algorithm that **solves** the problem

# Highlights

- $P$ and $NP$ contains decision problems
- $P \subseteq NP$ is known, but $P = NP$ or $P \subsetneq NP$ is open
- $NP$-complete $\subsetneq NP$-hard, e.g. optimization problems can be $NP$-hard
- Show in $P$ by providing polynomial-time algorithm that **solves** the problem
- Show in $NP$ by providing polynomial-time algorithm that **verifies** the problem

# Highlights

- $P$ and $NP$ contains decision problems
- $P \subseteq NP$ is known, but $P = NP$ or $P \subsetneq NP$ is open
- $NP$-complete $\subsetneq NP$-hard, e.g. optimization problems can be $NP$-hard
- Show in $P$ by providing polynomial-time algorithm that **solves** the problem
- Show in $NP$ by providing polynomial-time algorithm that **verifies** the problem
- $A$ is a $NP$-hard problem if we can reduce a known $NP$-hard problem $B$ **TO** $A$

# Highlights

- $P$ and $NP$ contains decision problems
- $P \subseteq NP$ is known, but $P = NP$ or $P \subsetneq NP$ is open
- $NP$-complete $\subsetneq NP$-hard, e.g. optimization problems can be $NP$-hard
- Show in $P$ by providing polynomial-time algorithm that **solves** the problem
- Show in $NP$ by providing polynomial-time algorithm that **verifies** the problem
- $A$ is a $NP$-hard problem if we can reduce a known $NP$-hard problem $B$ **TO** $A$
- $A$ is a $NP$-complete problem if it is in $NP$ and it is $NP$-hard

# Minimum spanning tree

Consider the following problem
**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$
**Output:** The weight of a minimum spanning tree

# Minimum spanning tree

Consider the following problem
**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$
**Output:** The weight of a minimum spanning tree

Is it a decision problem?

# Minimum spanning tree

Consider the following problem
**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$
**Output:** The weight of a minimum spanning tree

Is it a decision problem? No – what is one way to rephrase it as a decision problem?

# Minimum spanning tree

Consider the following problem
**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$
**Output:** The weight of a minimum spanning tree

Is it a decision problem? No – what is one way to rephrase it as a decision problem?

**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$, and $k \in \mathbb{N}$
**Output:** True if there is a minimum spanning tree of weight $\leq k$, otherwise False

# Minimum spanning tree

Consider the following problem
**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$
**Output:** The weight of a minimum spanning tree

Is it a decision problem? No – what is one way to rephrase it as a decision problem?

**Input:** A graph $G = (V, E)$ with edge-weights $\sigma : E \to \mathbb{N}$, and $k \in \mathbb{N}$
**Output:** True if there is a minimum spanning tree of weight $\leq k$, otherwise False