♥ P vs. NP ♥

the biggest open question in computer science

– an invitation –

37th Chaos Communication Congress

Questions are very much welcome! Please interrupt me mid-sentence.

Ingo Blechschmidt

The landscape of complexity classes

Def. An algorithm A **runs in polynomial time** if and only if there is some polynomial p such that, for every input I

$$\text{number of steps for computing } A(I) \quad \leq \quad p(|I|),$$

where $|I|$ is the length of an encoding of I in bits.

The landscape of complexity classes

Def. An algorithm A **runs in polynomial time** if and only if there is some polynomial p such that, for every input I

$$\text{number of steps for computing } A(I) \quad \leq \quad p(|I|),$$

where $|I|$ is the length of an encoding of I in bits.

Def. A problem is **in P** if and only if there is is a decision algorithm which **runs in polynomial time**.

Ex. Primality testing, node reachability, …

The landscape of complexity classes

Def. An algorithm A **runs in polynomial time** if and only if there is some polynomial p such that, for every input I

$$\text{number of steps for computing } A(I) \quad \leq \quad p(|I|),$$

where $|I|$ is the length of an encoding of I in bits.

Def. A problem is **in P** if and only if there is is a decision algorithm which **runs in polynomial time**.

Ex. Primality testing, node reachability, …

Def. A problem is **in NP** if and only if there is an algorithm which verifies **wannabe certificates for a positive answer** in polynomial time.

Ex. 3SAT, Sudoku, TSP, graph coloring, proof search, …

The landscape of complexity classes

Def. An algorithm A **runs in polynomial time** if and only if there is some polynomial p such that, for every input I

$$\text{number of steps for computing } A(I) \quad \leq \quad p(|I|),$$

where $|I|$ is the length of an encoding of I in bits.

Def. A problem is **in P** if and only if there is is a decision algorithm which **runs in polynomial time**.

Ex. Primality testing, node reachability, …

Def. A problem is **in NP** if and only if there is an algorithm which verifies **wannabe certificates for a positive answer** in polynomial time.

Ex. 3SAT, Sudoku, TSP, graph coloring, proof search, …

Prop. Every P-problem is also in NP: $P \subseteq NP$.

Further complexity classes

$$\begin{array}{ccccccc} P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXP \\ & & \cup\!\!\mid & & \cup\!\!\mid & & \\ & & \text{NP-C} & & \text{PSPACE-C} & & \end{array}$$

A problem T is in …

- **P** iff there is a **polynomial-time** decision algorithm.
- **NP** iff there is a polynomial-time algorithm which verifies wannabe certificates for a positive answer.

Further complexity classes

$$\begin{array}{ccccccc} P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXP \\ & & \cup\!\!\shortmid & & \cup\!\!\shortmid & & \\ & & NP\text{-}C & & PSPACE\text{-}C & & \end{array}$$

A problem T is in …

- **P** iff there is a **polynomial-time** decision algorithm.
- **NP** iff there is a polynomial-time algorithm which verifies wannabe certificates for a positive answer.
- **NP-C** iff it is in NP and if every NP-problem is **reducible** to T in polynomial time.

Further complexity classes

$$\begin{array}{ccccccc} P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXP \\ & & \cup\!\!\mid & & \cup\!\!\mid & & \\ & & NP\text{-}C & & PSPACE\text{-}C & & \end{array}$$

A problem T is in …

- **P** iff there is a **polynomial-time** decision algorithm.
- **NP** iff there is a polynomial-time algorithm which verifies wannabe certificates for a positive answer.
- **NP-C** iff it is in NP and if every NP-problem is **reducible** to T in polynomial time.
- **PSPACE** iff there is a **polynomial-space** decision algorithm.

Further complexity classes

$$\begin{array}{ccccccc} P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXP \\ & & \cup\!| & & \cup\!| & & \\ & & NP\text{-}C & & PSPACE\text{-}C & & \end{array}$$

A problem T is in …

- **P** iff there is a **polynomial-time** decision algorithm.
- **NP** iff there is a polynomial-time algorithm which verifies wannabe certificates for a positive answer.
- **NP-C** iff it is in NP and if every NP-problem is **reducible** to T in polynomial time.
- **PSPACE** iff there is a **polynomial-space** decision algorithm.
- **PSPACE-C** iff it is in PSPACE and if every PSPACE-problem is reducible to T in polynomial time.

Further complexity classes

$$
\begin{array}{ccccccc}
\mathrm{P} & \subseteq & \mathrm{NP} & \subseteq & \mathrm{PSPACE} & \subseteq & \mathrm{EXP} \\
 & & \cup\!\!\mid & & \cup\!\!\mid & & \\
 & & \text{NP-C} & & \text{PSPACE-C} & &
\end{array}
$$

A problem T is in …

- **P** iff there is a **polynomial-time** decision algorithm.
- **NP** iff there is a polynomial-time algorithm which verifies wannabe certificates for a positive answer.
- **NP-C** iff it is in NP and if every NP-problem is **reducible** to T in polynomial time.
- **PSPACE** iff there is a **polynomial-space** decision algorithm.
- **PSPACE-C** iff it is in PSPACE and if every PSPACE-problem is reducible to T in polynomial time.
- **EXP** iff there is an **exponential-time** decision algorithm.

Further complexity classes

$$\begin{array}{ccccccc} P & \subseteq & NP & \subseteq & PSPACE & \subseteq & EXP \\ & & \cup\!\! \mid & & \cup\!\! \mid & & \\ & & NP\text{-}C & & PSPACE\text{-}C & & \end{array}$$

A problem T is in …

- **P** iff there is a **polynomial-time** decision algorithm.
- **NP** iff there is a polynomial-time algorithm which verifies wannabe certificates for a positive answer.
- **NP-C** iff it is in NP and if every NP-problem is **reducible** to T in polynomial time.
- **PSPACE** iff there is a **polynomial-space** decision algorithm.
- **PSPACE-C** iff it is in PSPACE and if every PSPACE-problem is reducible to T in polynomial time.
- **EXP** iff there is an **exponential-time** decision algorithm.

P ≠ EXP, hence P ≠ NP or NP ≠ PSPACE or PSPACE ≠ EXP.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

Def. A problem is in P^B iff there is a polynomial-time decision B-algorithm.

Def. A problem is in NP^B iff there is a polynomial-time B-algorithm which verifies wannabe certificates for a positive answer.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

Def. A problem is in P^B iff there is a polynomial-time decision B-algorithm.

Def. A problem is in NP^B iff there is a polynomial-time B-algorithm which verifies wannabe certificates for a positive answer.

Prop. $P^B \subseteq NP^B \subseteq PSPACE^B$.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

Def. A problem is in P^B iff there is a polynomial-time decision B-algorithm.

Def. A problem is in NP^B iff there is a polynomial-time B-algorithm which verifies wannabe certificates for a positive answer.

Prop. $P^B \subseteq NP^B \subseteq PSPACE^B$.
Prop. If B is in NP-C, then $NP \subseteq P^B$.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

Def. A problem is in P^B iff there is a polynomial-time decision B-algorithm.

Def. A problem is in NP^B iff there is a polynomial-time B-algorithm which verifies wannabe certificates for a positive answer.

Prop. $\mathrm{P}^B \subseteq \mathrm{NP}^B \subseteq \mathrm{PSPACE}^B$.
Prop. If B is in NP-C, then $\mathrm{NP} \subseteq \mathrm{P}^B$.

Thm. For some B, $\mathrm{P}^B = \mathrm{NP}^B$; and for some B, $\mathrm{P}^B \neq \mathrm{NP}^B$.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

Def. A problem is in P^B iff there is a polynomial-time decision B-algorithm.

Def. A problem is in NP^B iff there is a polynomial-time B-algorithm which verifies wannabe certificates for a positive answer.

Prop. $\mathrm{P}^B \subseteq \mathrm{NP}^B \subseteq \mathrm{PSPACE}^B$.
Prop. If B is in NP-C, then $\mathrm{NP} \subseteq \mathrm{P}^B$.

Thm. For some B, $\mathrm{P}^B = \mathrm{NP}^B$; and for some B, $\mathrm{P}^B \neq \mathrm{NP}^B$.
Proof, first part. Pick for B some problem in PSPACE-C. Then $\mathrm{PSPACE} \subseteq P^B \subseteq \mathrm{NP}^B \subseteq \mathrm{PSPACE}^B \subseteq \mathrm{PSPACE}$.

The relativization barrier

Let B be a problem. A ***B*-algorithm** is an algorithm which has access to an **oracle for *B***.

Def. A problem is in P^B iff there is a polynomial-time decision B-algorithm.

Def. A problem is in NP^B iff there is a polynomial-time B-algorithm which verifies wannabe certificates for a positive answer.

Prop. $P^B \subseteq NP^B \subseteq PSPACE^B$.
Prop. If B is in NP-C, then $NP \subseteq P^B$.

Thm. For some B, $P^B = NP^B$; and for some B, $P^B \neq NP^B$.
Proof, first part. Pick for B some problem in PSPACE-C. Then $PSPACE \subseteq P^B \subseteq NP^B \subseteq PSPACE^B \subseteq PSPACE$.
Proof, second part. Pick for B a zero/one **random oracle**. Then the problem "do n consecutive ones occur in the first 2^n drawings of B?" is in NP^B but not in P^B.