

An alternate characterization of NP

- **Theorem:** A language L is in **NP** iff there exists a polynomial q and a language $R \in \mathbf{P}$ such that

$$L = \{w \mid \exists t \text{ s.t. } |t| \leq q(|w|) \text{ and } \langle w, t \rangle \in R\}$$

- What is this theorem saying? The string t is called a “proof” or a “certificate” that $w \in L$. A polynomial-time decider for R is called a “verifier” or “checker”. The theorem says that **NP** is the class of languages L for which a proof of membership can be efficiently verified.
- In this setting, the question “is **P** equal to **NP**?” is asking: is it easier to verify a proof than to find one?
- Proof of theorem:

– (\Rightarrow) Suppose $L \in \mathbf{NP}$. Then there exists an NTM M_L and a polynomial q such that $L(M_L) = L$ and $T_M(n) \leq q(n)$ for all n . The idea is that for $w \in L$, the certificate will be a sequence corresponding to the non-deterministic choices made by M_L along an accepting computation. Suppose $M_L = (Q, \Sigma, \Gamma, q_0, q_{accept}, q_{reject}, \delta)$, and let $B = \max\{|\delta(q, a)| \mid q \in Q, a \in \Gamma\}$ be the largest number of non-deterministic choices available at any step. Define R to be the language decided by the following TM $M_R =$ “on input $\langle w, t \rangle$...

1. Check that $t = b_1 \# b_2 \# \dots \# b_{q(|w|)}$, where $b_i \in \{1, \dots, B\}$.
2. Simulate M_L on input w . Whenever M_L must make a non-deterministic choice, use the next b_i in the sequence to decide between the $\leq B$ possible choices.
3. If M_L accepts then accept; else reject

The running time of M_R is polynomial in $|w|$: it is essentially equal to $q(|w|)$, with some small additional overhead to carry out the simulation. If $w \in L$ then there exists an accepting computation of M_L on input w . Corresponding to this computation is a sequence b_1, b_2, \dots, b_k with $k \leq q(|w|)$, where the b_i 's correspond to the sequence of non-deterministic choices made by M_L in that computation. In this case M_R accepts $\langle w, b_1 \# b_2 \# \dots \# b_{q(|w|)} \rangle$, for any arbitrary $b_{k+1}, \dots, b_{q(|w|)}$.

On the other hand, if there exists a t of length at most $q(|w|)$ such that M_R accepts $\langle w, t \rangle$, then M_R discovered an accepting computation of M_L on input w , implying $w \in L$. So $w \in L$ iff there exists a t of length at most $q(|w|)$ such that $\langle w, t \rangle \in R$.

- (\Leftarrow) Suppose that for some language L , there exist a polynomial q and a language $R \in \mathbf{P}$ such that

$$L = \{w \mid \exists t \text{ s.t. } |t| \leq q(|w|) \text{ AND } \langle w, t \rangle \in R\}$$

- Since $R \in \mathbf{P}$ there is a Turing machine M_R and a polynomial p such that $L(M_R) = R$ and $T_{M_R}(n) \leq p(n)$. We must show that $L \in \mathbf{NP}$. Here is an NTM that decides L : $M_L =$ “on input w ...

1. Guess a string t of length $\leq q(|w|)$
2. Run M_R on input $\langle w, t \rangle$
3. Accept iff M_R accepts

What is the running time of M_L ? If $n = |w|$ is the input size, then the first step takes time roughly $O(q(n))$, and the second step takes time roughly $p(n + q(n))$ (there is some small overhead that is not being considered here). Since p and q are polynomials, so is $p(n + q(n))$, so M_L runs in polynomial time.

If $w \in L$ then there exists a string t of length at most $q(|w|)$ such that M_R accepts $\langle w, t \rangle$. So there is an accepting computation of M_L on input w : the computation in which it correctly guesses t . On the other hand, if $w \notin L$ then M_R will reject every $\langle w, t \rangle$ with $|t| \leq q(|w|)$, so there is no accepting computation of M_L on input w . Thus $L(M_L) = L$.

P, NP and co-NP

- **Observation:** \mathbf{P} is closed under complement.
- **Proof:** Suppose $L \in \mathbf{P}$. Then there exists a polynomial-time TM M that decides L . If M' is the Turing machine obtained from M by switching its accept and reject states, then M' decides \overline{L} in polynomial time, so $\overline{L} \in \mathbf{P}$.
- **Definition:** $\mathbf{co-NP} = \{\overline{L} \mid L \in \mathbf{NP}\}$ is the class of languages whose complements are in \mathbf{NP} .
- As a corollary to characterization of \mathbf{NP} and the fact that \mathbf{P} is closed under complement, we get: a language L is in $\mathbf{co-NP}$ iff there exists a polynomial q and a language $R \in \mathbf{P}$ such that

$$L = \{w \mid \forall t : \text{if } |t| \leq q(|w|) \text{ then } \langle w, t \rangle \in R\}$$

If \mathbf{NP} is the class of languages whose “yes” instances have efficiently checkable proofs, then $\mathbf{co-NP}$ is the class of languages whose “no” instances have efficiently checkable counterexamples.

- **Theorem:** $\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{co-NP}$.
- **Proof:**
 - ($\mathbf{P} \subseteq \mathbf{NP}$) Determinism is really a special case of non-determinism: a deterministic Turing machine can be viewed as a non-deterministic machine whose transitions are all singleton sets.

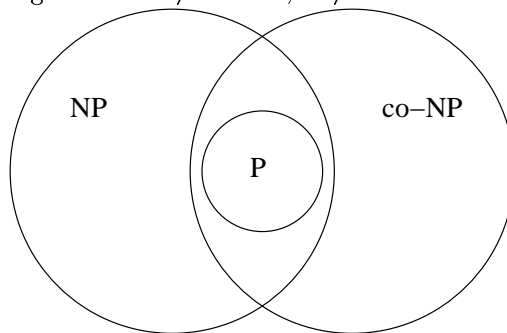
– ($\mathbf{P} \subseteq \mathbf{co-NP}$) Suppose $L \in \mathbf{P}$. As \mathbf{P} is closed under complement, \overline{L} is also in \mathbf{P} . Since $\mathbf{P} \subseteq \mathbf{NP}$ it follows that $\overline{L} \in \mathbf{NP}$. Finally, by definition of $\mathbf{co-NP}$, if $\overline{L} \in \mathbf{NP}$ then $L \in \mathbf{co-NP}$.

- The answers to the following questions are not known:
 1. Is $\mathbf{P} = \mathbf{NP}$? The answer is conjectured to be “no”.
 2. Is $\mathbf{NP} = \mathbf{co-NP}$? The answer is also conjectured to be “no”.
 3. Is $\mathbf{P} = \mathbf{NP} \cap \mathbf{co-NP}$? The conjecture is (you guessed it): “no”.

Possibilities

How we think the world looks

Figure 1: $\mathbf{NP} \neq \mathbf{co-NP}$, $\mathbf{P} \neq \mathbf{NP} \cap \mathbf{co-NP}$



Other possibilities

Figure 2: $\mathbf{NP} \neq \mathbf{co-NP}$, $\mathbf{P} = \mathbf{NP} \cap \mathbf{co-NP}$

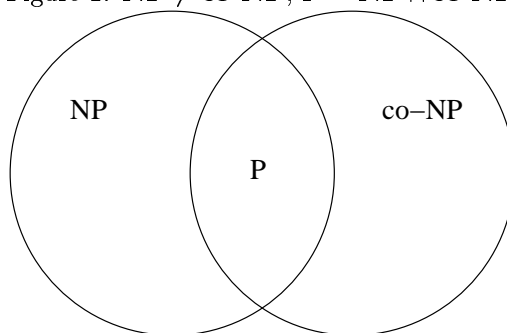
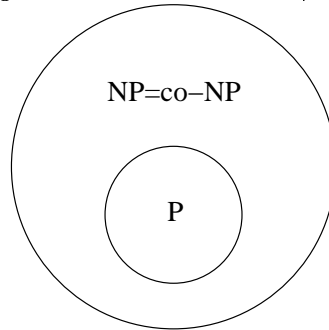


Figure 3: $\mathbf{NP = co-NP}$, $\mathbf{P \neq NP}$ Figure 4: $\mathbf{P = NP = co-NP}$ 