Although we do not know how to separate $P$ from $NP$, it is a standard hypothesis of computational complexity theory that the two classes differ.

**Hypothesis 1.** $P \neq NP$

Another common hypothesis is that $NP$ and $coNP$ differ.

We note $P = NP \Rightarrow NP = coNP$ (since $P$ is closed under complement of languages). This could also be written as $NP \neq coNP \Rightarrow P \neq NP$. Thus the hypothesis that $NP \neq coNP$ is a stronger hypothesis than $P \neq NP$.

In general we may hypothesize that the entire polynomial time hierarchy is a strict hierarchy.

**Hypothesis 2** (generalized). For all $k$:

1. $\Sigma_{k+1}^p \neq \Sigma_k^p$
2. $\Sigma_k^p \neq \Pi_k^p$

![Polynomial-time hierarchy](image)

The second hypothesis at least as strong as the first:

**Proposition 3.** $\Sigma_{k+1}^p = \Sigma_k^p \Rightarrow \Sigma_k^p = \Pi_k^p$ (equivalently $\Sigma_k^p \neq \Pi_k^p \Rightarrow \Sigma_{k+1}^p \neq \Sigma_k^p$)

**Proof.** We will show that $\Pi_k^p \subseteq \Sigma_k^p$. Consider the following

$$\Pi_k^p \subseteq \Sigma_{k+1}^p \overset{\text{by asmp.}}{=} \Sigma_k^p$$

By complements, we also have $\Sigma_k^p \subseteq \Pi_k^p$.

We next show that if the hypothesis $\Sigma_k^p \neq \Pi_k^p$ fails, then the entire polynomial time hierarchy collapses to the $k$th level.
Theorem 4. If $\Sigma^p_k = \Pi^p_k$ for some $k$ then $PH = \Sigma^p_k$

Proof. By induction, we will prove that $\Sigma^p_{k+i+1} = \Sigma^p_{k+i}(= \Pi^p_k)$. The base case is done in the same way as the inductive step.

Let $L \in \Sigma^p_{k+i+1}$. Then there exists some polynomial $p$ and $R \in P$ such that

$$x \in L \Leftrightarrow \exists^p y_1 \forall^p y_2 \exists^p y_3 \cdot Q^p y_{k+i+1}(x, y_1, y_2, \ldots, y_{k+i+1}) \in R$$

Now, define $L' \in \Pi^p_{k+i}$ by

$$\langle x, y \rangle \in L' \Leftrightarrow \forall^p y_2 \exists^p y_3 \cdots Q^p y_{k+i}(x, y_1, y_2, \ldots, y_{k+i}) \in R$$

By induction $L' \in \Pi^p_k = \Sigma^p_k$ so there is a polynomial $p$ and $R \in P$ such that

$$\langle x, y_1 \rangle \in L' \Leftrightarrow \exists^p z_1 \forall^p z_2 \cdots Q^p z_k(x, y_1, z_1, z_2, \ldots, z_k) \in R$$

Then

$$x \in L \Leftrightarrow \exists^p y_1 \exists^p z_1 \forall^p z_2 \cdots Q^p z_k(x, y_1, z_1, z_2, \ldots, z_k) \in R$$

This concludes the proof that $L \in \Sigma^p_k$. \qed

Corollary 5. If $PH$ has a complete problem then $PH$ collapses (i.e. $PH = \Sigma^p_k$ for some $k$)

Proof. Assume $L$ is complete for $PH$. Then $L \in \Sigma^p_k$ for some $k$. $\Sigma^p_k$ is closed under reduction and since any $L' \in PH$ reduces to $L$, $PH \subseteq \Sigma^p_k$. \qed

Corollary 6. If $PH = \text{PSPACE}$ then $PH$ collapses

Proof. $\text{PSPACE}$ has complete problems. \qed

Theorem 7 (Karp-Lipton). If $NP \subseteq P/poly$ then $PH = \Sigma^p_2 = \Pi^p_2$

Proof. Assume $NP \subseteq P/Poly$. We will show $\Pi^p_2 \subseteq \Sigma^p_2$. Let $L \in \Pi^p_2$ then there is some polynomial $p$ and $R \in P$ such that

$$x \in L \Leftrightarrow \forall^p y \exists^p z(x, y, z) \in R$$

Define $L' \in NP$ by $\langle x, y \rangle \in L' \Leftrightarrow \exists^p z(x, y, z) \in R$ and define $L'_{\text{pre}} \in NP$ by $\langle x, y, z_{\text{pre}} \rangle \in L'_{\text{pre}} \Leftrightarrow \exists^p z$ such that $\langle x, y, z \rangle \in R$ and $z_{\text{pre}}$ is a prefix of $z$. By assumption we have polynomial-sized circuits ($L'_{\text{pre}} \in P/poly$). Using circuits for $L'_{\text{pre}}$ we can find a witness $z$, given $\langle x, y \rangle$, in polynomial time, if it exists.

Converting this to a multi-output circuit $C$, we obtain that $C(\langle x, y \rangle)$ outputs a witness $z$ if it exists.

Now, we can write $x \in L \Leftrightarrow \exists^p C \forall^p y(x, y, C(\langle x, y \rangle)) \in R$. Thus $L \in \Sigma^p_2$. \qed

Theorem 8 (Meyer). If $EXP \subseteq P/poly$ then $EXP = \Sigma^p_2 = \Pi^p_2$. 

2
Proof. Let $L \in EXP$ and let $M$ be a 1-tape Turing machine deciding $L$ in time $2^{nk}$. Consider the computation tableau for $M$, and assume each position $(i, t)$ encodes as a string $z_{i,t}$ the content of cell $i$ at time $t$, whether the tape head is scanning cell $i$ at time $t$, and if so, stores the internal state of the machine. Now consider the following “tableau” language for $M$.

$$L_M = \{(x, i, t, z) \mid \text{on input } x \text{ we have } z_{i,t} = z \text{ for } M\}$$

By simulating $M$ we have $L_M \in EXP \subseteq P/poly$. Using polynomial size circuits for $L_M$ we can construct a polynomial-size multi-output circuit $C$ such that $C((x, i, t)) = z$. Now we write $x \in L \iff \exists \mathbb{P} C \forall i, t : \text{“}C((x, i, t))\text{” follow from } C((x, i-t, t-1)), C((x, i, t-1)) \text{ and } C((x, i+1, t-1))$ and $C((x, 1, 2^{nk}))$ is accepting”.

**Theorem 9.** $BPP \subseteq P/poly$

*Proof.* The proof technique here is known as the probabilistic method, which is used for showing existence of an object by deriving that the probability of finding the object is strictly greater than 0 (which yields the existence).

We shall assume our alphabet is $\Sigma = \{0, 1\}^*$. Let $L \in BPP$. By success amplification there is a probabilistic polynomial time Turing machine $M$ that computes $L$ with error $< \frac{1}{2^n}$.

Fix $x \in \{0, 1\}^n$. Taking the probability over the for internal coin flips, $Pr[M\text{ errs on } x] < \frac{1}{2^n}$. By a “union bound” we have the following

$$Pr[M\text{ errs on some } x \in \{0, 1\}^n] \leq \Sigma_{x \in \{0,1\}^n} Pr[M\text{ errs on } x] < 2^n \frac{1}{2^n} = 1$$

Thus, there are some internal coin-flips where $M$ makes no error on any $x \in \{0, 1\}^n$. Convert $M$ into a circuit and hardwire the “lucky” coin-flips (mentioned right above).

**Theorem 10.** $BPP \subseteq \Sigma_2^p \cap \Pi_2^p$
Proof. We will prove $BPP \subseteq \Sigma^p_2$. Suppose $L \in BPP$ and let us recall the definition of this:

$L \in BPP \iff \exists \text{polynomial } p, L' \in P$

$$x \in L : \Pr[\langle x, y \rangle \in L'] \geq \frac{2}{3}$$

$$x \not\in L : \Pr[\langle x, y \rangle \not\in L'] \geq \frac{2}{3}$$

for $y \in \{0,1\}^{p(|x|)}$.

Recall the details of success amplification: If we use $m$ coin-flips to get error $< \frac{1}{3}$ then by running $O(k)$ independent trials, i.e., by using $O(km)$ coin-flips, we can get error $< \frac{1}{2^k}$.

Then there exists a Turing machine $M$ using $m = m(n)$ random bits to get error $< \frac{1}{2^m}$. That is, we have $L' \in P$ such that

$$x \in L : \Pr[\langle x, y \rangle \in L'] \geq 1 - \frac{1}{2^m}$$

$$x \not\in L : \Pr[\langle x, y \rangle \in L'] < \frac{1}{2^m}$$

where $y \in \{0,1\}^m$. For a given $x$, let $S_x = \{y \in \{0,1\}^m | \langle x, y \rangle \in L'\}$:

**Figure 3:** (†) [on the left] and (‡) [on the right]

$$x \in L : |S_x| \geq (1 - \frac{1}{2^m})2^m \quad (\ast)$$

$$x \not\in L : |S_x| < \frac{1}{2^m}2^m \quad (\ast\ast)$$

The idea now is that by a few random shifts one can cover the entire probability space in (†) but not in (‡): 

Notation: $S \subseteq \{0,1\}^m, y \in \{0,1\}^m$; Shift $S$ by $y \oplus S := \{y \oplus x | x \in S_x\}$.

**Claim 1.**

$$x \in L \iff \exists u_1, u_2, \ldots, u_m \in \{0,1\}^m : \forall r \in \{0,1\}^m : \bigvee_{i=1}^m \langle x, r \oplus u_i \rangle \in L'$$

Note: The above expression is a $\Sigma^p_2$ statement as needed for the conclusion.
This may also be rewritten as

\[ \exists u_1, u_2, \ldots, u_m \in \{0, 1\}^m \forall r \in \{0, 1\}^m \exists i : r \oplus u_i \in S_x \]

and further (note \( r \oplus u_i \in S_x \iff r \in u_i \oplus S_x \))

\[ \exists u_1, u_2, \ldots, u_m \in \{0, 1\}^m : \bigcup_{i=1}^{m} u_i \oplus S_x = \{0, 1\}^m \]

**Proof of claim.** \( x \not\in L \): Let \( u_1, u_2, \ldots, u_m \) be arbitrary.

\[ |\bigcup_{i} u_i \oplus S_x| \leq \sum_{i=1}^{m} |u_i \oplus S_x| \leq \frac{m}{2m} 2^m < 2^m \]

Hence for all choices for \( u_1, \ldots, u_m \) we have that \( \bigcup_{i=1}^{m} u_i \oplus S_x \neq \{0, 1\}^m \).

\( x \in L \): Pick \( u_1, u_2, \ldots, u_m \) at random. Fix \( r \in \{0, 1\}^m \) and consider the rewrites

\[
\Pr[r \not\in \bigcup_{i=1}^{m} u_i \oplus S_x] = \Pr[\forall i : r \not\in u_i \oplus S_x] = \Pr[\forall i : u_i \not\in r \oplus S_x] \\
= \prod_{i=1}^{m} \Pr[u_i \not\in r \oplus S_x] \leq \left( \frac{1}{2m} \right)^m < \frac{1}{2} \quad (1)
\]

Taking a union bound over all \( r \), we get that \( \Pr[r \not\in \bigcup_{i=1}^{m} u_i \oplus S_x \text{ for some } x] < 1 \). Hence concluding by the probabilistic method there exists some choice for \( u_1, \ldots, u_m \) such that \( \bigcup_{i=1}^{m} u_i \oplus S_x = \{0, 1\}^m \).

\( \triangle \)

\( \Box \)