We define the class of sets which truth-table reduce to a set in \( \text{NP} \) as

\[
\Theta^P_2 = P^\text{NP}_{tt} = \{ A \mid \exists B \in \text{NP}, A \leq^P_{tt} B \}.
\]

This may seem like just YACC, but it turns out to be a natural class with another nice characterization: \( P^{\text{NP}[\log n]} \), or \( P \) with an \( \text{NP} \) oracle to which only \( O(\log n) \) queries are allowed. Clearly, \( \text{NP} \subseteq \Theta^P_2 \subseteq \Delta^P_2 = P^{\text{NP}} \).

Let’s first look at the structure of a truth-table reduction to a set \( B \). We can decompose the computation into two polynomial time phases:

1. \( P_1(x) \), which takes the input \( x \) and outputs a list of strings \( y_1, y_2, \ldots, y_k \), where both the lengths of the \( y_i \) and the number of strings \( k \) is bounded by a polynomial in the length of \( x \).
2. \( P_2(x, \chi_B(y_1)\chi_B(y_2) \cdots \chi_B(y_k)) \), which takes \( x \) and the answers to all the queries \( y_i \in B \) and determines whether \( x \in A \). (Note: we have concatenated the answers to the queries into a single string.)

Since all queries are made to \( B \) simultaneously, or in parallel, \( \Theta^P_2 \) is frequently denoted \( P^{\text{NP}}_\parallel \).

**Claim 1:** \( P^{\text{NP}[\log n]} \subseteq P^{\text{NP}}_\parallel \).

**proof** This direction is relatively easy. Let \( A \in P^B \), for \( B \in \text{NP} \). Suppose that at most \( c \log n \) queries to \( B \) are ever made, if \( y_1 \) is the first query, then there are two possible second queries, \( y_1^0 \) and \( y_1^1 \), depending on whether the answer to the first question \( y_1 \in B \) was "no" or "yes". Similarly, there are at most four possible third queries, depending on the answers to both the first and second questions. The depth of this process is at most \( c \log n \), so there are at most \( 2^{c \log n} = n^c \) queries that could ever be made. To phrase this as a \( tt \)-reduction to \( B \), \( P_1 \) could derive all these possible strings. With the answers to all of them, \( P_2 \) would recreate what the original process for \( A \) would have done. \( \square \)

**Claim 2:** \( P^{\text{NP}}_\parallel \subseteq P^{\text{NP}[\log n]} \).

**proof** Here we use the wonderful “mind-changing” technique from Wagner (and Hemaspaandra?). Suppose that \( A \in P^{B}_\parallel \) for a \( B \in \text{NP} \). We must argue that \( A \in P^{B'[\log n]}_\parallel \) for some \( B' \in \text{NP} \). Let \( P_1(x) = (y_1, y_2, \ldots, y_k) \) and \( P_2(x, z) \) be the two halves of the \( P^{B}_\parallel \) computation.

We say that a string \( z \), \( |z| = k \), is valid if

\[
\text{the } i\text{th bit of } z \text{ is } 1 \Rightarrow y_i \in B.
\]
Note that if $B \in NP$, then to test whether any $z$ is valid can also be done in $NP$. This is a subtle and important point.

Now the clever set incorporating the mind-changing business is as follows (where $x \prec y$ means that the string $y$ can be obtained by changing some zeroes in the string $x$ to ones):

$$B' = \{ \langle x, j \rangle \mid j \leq k_x \text{ and } \exists z_1, z_2, \ldots, z_j \text{ such that}\}
\begin{align*}
&\text{each } z_i \text{ is valid of length } k_x, \text{ and } \\
&z_1 \prec z_2 \prec \cdots \prec z_j, \text{ and } \\
&\forall i \ (1 \leq i < j) \ P_2(x, z_i) \neq P_2(x, z_{i+1})\}\}$$

The main thing to notice about $B'$ is that $B \in NP \Rightarrow B' \in NP$.

Define the “mind-change” function as

$$mc_{B'}(x) = \max\{j \mid \langle x, j \rangle \in B'\}.$$ 

Using binary search, $mc_{B'}(x)$ can be computed with at most $O(\log n)$ queries to $B'$. Note also that knowing $P_2(x, 0^k)$ and the parity of $mc_{B'}(x)$ determines whether $P_2$ accepts $x$ on the correct query answers:

$$x \in A \iff P_2(x, 0^k) \oplus \text{odd}(mc_{B'}(x)).$$

Therefore, $A \in P^{B'[\log n]} \subseteq P^{NP[\log n]}.$ $\square$