Generation or Listing

Given a hypergraph $\mathcal{H}$ represented by a membership oracle $\Omega$, generate sequentially, without repetitions, all hyperedges of $\mathcal{H}$. 

$\Omega$ $\Omega$ $\Omega$ $\Omega$ $\Omega$

$H_1 \uparrow$ $H_2 \uparrow$ $H_3 \uparrow$ $H_N \uparrow$

$t_1$ $t_2$ $t_3$ $\ldots$ $\ldots$ $\ldots$ $\ldots$ $t_{N+1}$ STOP
Given a hypergraph $\mathcal{H}$ represented by a membership oracle $\Omega$, generate sequentially, without repetitions, all hyperedges of $\mathcal{H}$.

Polynomial Delay: $t_k \leq poly(|\Omega|)$ for all $k = 1, \ldots, N + 1$. 
Given a hypergraph $\mathcal{H}$ represented by a membership oracle $\Omega$, generate sequentially, without repetitions, all hyperedges of $\mathcal{H}$.

Polynomial Delay: \[ t_k \leq \text{poly}(|\Omega|) \quad \text{for all } k = 1, \ldots, N + 1. \]

Incrementally Polynomial: \[ t_k \leq \text{poly}(|\Omega|, k - 1) \quad \text{for all } k = 1, \ldots, N + 1. \]
**Generation or Listing**

Given a hypergraph $\mathcal{H}$ represented by a membership oracle $\Omega$, generate sequentially, without repetitions, all hyperedges of $\mathcal{H}$.

$$
\begin{align*}
\Omega & \implies t_1 \\
\uparrow & \implies \cdots \\
\uparrow & \implies t_2 \\
\uparrow & \implies \cdots \\
\uparrow & \implies t_3 \\
\uparrow & \implies \cdots \\
\uparrow & \implies \cdots \\
\uparrow & \implies \cdots \\
\uparrow & \implies \cdots \\
\uparrow & \implies t_{N+1} \\
\cdots & \implies \text{STOP}
\end{align*}
$$

**Polynomial Delay:**

$$
t_k \leq \text{poly}(|\Omega|) \quad \text{for all } k = 1, \ldots, N + 1.
$$

**Incrementally Polynomial:**

$$
t_k \leq \text{poly}(|\Omega|, k - 1) \quad \text{for all } k = 1, \ldots, N + 1.
$$

**Polynomial Total Time:**

$$
\sum_{k=1}^{N+1} t_k \leq \text{poly}(|\Omega|, N).
$$
Remarks

\[ \text{NEXT}(G, H, \Omega) \]
Given a hypergraph \( H \) represented by a membership oracle \( \Omega \), and an explicitly listed subfamily \( G \subseteq H \), \textbf{decide} whether \( G = H \) or not, and \textbf{if not, find} a new set \( H \in H \setminus G \).

\textbf{Theorem.} The hypergraph \( H \) can be generated in \textbf{incremental polynomial time} using the oracle \( \Omega \) if and only if the above decision problem \( \text{NEXT}(G, H, \Omega) \) can be solved in \textbf{polynomial time} for every subfamily \( G \subseteq H \).
NEXT(\(G, H, \Omega\))

Given a hypergraph \(H\) represented by a membership oracle \(\Omega\), and an explicitly listed subfamily \(G \subseteq H\), decide whether \(G = H\) or not, and if not, find a new set \(H \in H \setminus G\).

Theorem. The hypergraph \(H\) can be generated in incremental polynomial time using the oracle \(\Omega\) if and only if the above decision problem \(\text{NEXT}(G, H, \Omega)\) can be solved in polynomial time for every subfamily \(G \subseteq H\).

- In principle, it could be possible to have \(H\) generated, using \(\Omega\), in total polynomial time, and \(\text{NEXT}(G, H, \Omega)\) to be \textbf{NP-hard}, but we do not know (yet) such examples.
\textbf{Remarks Cont'd}

\begin{center}
\begin{tabular}{|l|}
\hline
NEXT($G, H, \Omega$) \\
\hline
Given a hypergraph $H$ represented by a membership oracle $\Omega$, and an explicitly listed subfamily $G \subseteq H$, \textbf{decide} whether $G = H$ or not, and \textbf{if not, find} a new set $H \in H \setminus G$. \\
\hline
\end{tabular}
\end{center}

\textbf{Theorem.} The hypergraph $H$ can be generated in \textit{incremental polynomial time} using the oracle $\Omega$ if and only if the above decision problem $\text{NEXT}(G, H, \Omega)$ can be solved in \textit{polynomial time} for every subfamily $G \subseteq H$.

- In principle, it could be possible to have $H$ generated, using $\Omega$, in \textit{total polynomial time}, and $\text{NEXT}(G, H, \Omega)$ to be \textbf{NP-hard}, but we do not know (yet) such examples.

- There are problems solvable in \textit{incremental polynomial time} for which we \textbf{do not know} a \textit{polynomial delay} solution. We \textbf{do not know either} how to show that a problem cannot be solved with \textit{polynomial delay}. 
For self-reducible classes of problems:

- **Total** and **incremental** (polynomial) time generation are equivalent.

- **Generation** is easier than **counting**. In other words, if there is a polynomial time algorithm to determine $|\mathcal{H}|$ (using only the oracle $\Omega$) for all problems in the given self-reducible class of problems, then $\mathcal{H}$ can also be generated with polynomial delay, using only $\Omega$.

  (??Dr. Folklore, Late-Pleistocene??)