- We defined Σ_i & Π_i like NP, but with i alternating quantifiers on top of a poly-time TM.

▷ $PH = \bigcup_{i \geq 0} \Sigma_i = \bigcup_{i \geq 0} \Pi_i \subseteq$ Pspace.

constant many alternations — arbitrary many alternations

- Defn: If $\exists i$, $PH = \Sigma_i$ then we shall say that PH collapses to the i-th level.

PH-conjecture: PH does not collapse.

"It is a generalization of "$P \neq NP$?".

- We now show several separations that follow from this conjecture.

Theorem 1: If for an $i \geq 1$, $\Sigma_i = \Pi_i$, then PH collapses to the i-th level.

Proof: • Say, $\Sigma_i = \Pi_i$. What is Σ_{i+1}?

• An $L \in \Sigma_{i+1}$ iff $\exists$ poly-time TM M & a $c>0$ s.t. $\forall x$,

$x \in L$ iff $\exists u_1 \forall u_2 \ldots Q_{i+1} u_{i+1}$

u's of length $|x|^c$

$M(x, u_1, \ldots, u_{i+1}) = 1$.

• Define a related language $L' :=$

$\{(y,z) \mid \forall u_2 \exists u_3 \ldots Q_{i+1} u_{i+1}\ M(y, z, u_2, \ldots, u'_{i+1}) = 1\}$.

z & u's of length $|y|^c$

• Clearly, $L' \in \Pi_i = \Sigma_i$.

• Also, we see that: $x \in L$ iff $\exists u_1, (x, u_1) \in L'$.

• The above two observations together mean:

$L \in \Sigma_i$

$\Rightarrow \Sigma_{i+1} \subseteq \Sigma_i \Rightarrow \Sigma_{i+1} = \Sigma_i$

$\Rightarrow \Sigma_{i+1} = \Sigma_i = \Pi_i = \Pi_{i+1}$.

• By induction, $\forall j \geq i$, $\Sigma_j = \Sigma_i = \Pi_i = \Pi_j$

$\Rightarrow PH = \Sigma_i$. $\square$

Corollary: If for an $i \geq 0$, $\Sigma_i = \Sigma_{i+1}$ then PH collapses to the i-th level.

Proof: • Let $\Sigma_i = \Sigma_{i+1}$.

$\Rightarrow \Pi_i = \Pi_{i+1}$

• We know $\Pi_i \cup \Sigma_i \subseteq \Pi_{i+1} \cap \Sigma_{i+1}$.

$\Rightarrow \Sigma_{i+1} = \Pi_{i+1} = \Sigma_i = \Pi_i$.

• By Theorem 1 we get $PH = \Sigma_i$. $\square$

Corollary: $P = NP \iff PH = P$.

Complete problems in PH

- Suppose A is a PH-complete problem (under poly-time reductions).
- Then, $\exists i$, $A \in \Sigma_i$.

Implying $\Sigma_i = PH$!

▷ PH-conjecture ⇒ PH $\subsetneq$ Pspace.

Proof: • Assuming the PH-conjecture, we deduce, as above, that there are no PH-complete problems.

• On the other hand, there is a Pspace-complete problem.

$\Rightarrow$ PH $\subsetneq$ Pspace. $\square$

Σ_i-complete problems

Defn: For $i \geq 1$, define $\underline{\Sigma_i \text{Sat}}$:=
$\{ \phi(u_1, \ldots, u_i) \mid \phi(\bar{x})$ be a boolean CNF formula with a partition of $x_1, \ldots, x_n$ into $u_1, \ldots, u_i$ s.t. $\exists u_1 \forall u_2 \cdots Q_i u_i$, $\phi(u_1, \ldots, u_i) = 1 \}$.

not quite a QBF

Theorem: Σ_iSat is Σ_i-complete. ← Σ_1Sat is Sat.

Proof:

• By defn, $\Sigma_i \text{Sat} \in \Sigma_i$.

- Any $L \in \Sigma_i$ has a corresponding poly-time TM M.
- The computation of M can be captured in a CNF formula φ.

  (by Cook-Levin reduction)
- This reduces the qn. $x \in L$ to the truth of the quantified formula $\exists u_1 \forall u_2 \cdots Q_i u_i \; \varphi(x, u_1, \ldots, u_i)$.

$\Rightarrow$

Σ_iSat is Σ_i-hard as well. $\square$

<u>Defn:</u> For $i \geq 1$, define <u>Π_iSat</u> $:= \{\varphi(u_1, \ldots, u_i) \mid$ φ is a boolean DNF formula with a partition of $x_1, \ldots, x_n$ into $u_1, \ldots, u_i$ s.t. $\forall u_1 \exists u_2 \cdots Q_i u_i \; \varphi(u_1, \ldots, u_i) = 1\}$.

<u>Corollary:</u> Π_iSat is Π_i-complete.

PH via oracle machines

- Like NP represents computation on non-deterministic TMs.
  What does PH represent?

<u>Defn:</u> • For complexity classes C_1, C_2 we define the class $C_1^{C_2} := \bigcup_{L \in C_2} C_1^{L}$.

▷ $P^{NP} = P^{SAT}$.

▷ $NP^{NP} = NP^{SAT}$.

- We intend to show $NP^{NP} = \Sigma_2$!

<u>Theorem:</u> $\forall i \geq 2$, $\Sigma_i = NP^{\Sigma_{i-1}Sat}$.

<u>Proof sketch:</u>

• We exhibit the ideas by taking $i = 2$.

• Let $L \in \Sigma_2$.
  Then there is a poly-time TM M &

a constant $c>0$ s.t.

(1) - - - - $\forall x, \quad x \in L$ iff $\exists u_1 \forall u_2 \; M(x, u_1, u_2) = 1$.

in $\{0,1\}^{|x|^c}$

- The associated language $L' :=$
$$\{(y,z) \mid \forall u_2 \in \{0,1\}^{|y|^c}, M(y,z,u_2)=1\}$$
is in Π_1.
- Thus, L' can be decided by an oracle to sat.

$\Rightarrow \quad L' \in P^{SAT}$.

- We could rewrite eqn.(1) as:

$x \in L$ iff $\exists u_1, (x, u_1) \in L'$.

$\Rightarrow \quad L \in NP^{SAT}$

$\Rightarrow \quad \Sigma_2 \subseteq NP^{SAT}$.

- Let $L \in NP^{SAT}$. Say, L is decided by a poly-time NDTM N using SAT oracle.
- N makes <u>choices</u> in its execution path & <u>queries</u> the oracle on CNF formulas.