Today

- NP $\subseteq$ P$/_{\text{poly}} \Rightarrow \Sigma_3^P = \Pi_3^P$.
- SAT $\in$ L $\Rightarrow$ $\exists \epsilon > 0$ s.t. SAT $\notin$ DTIME$(n^{1+\epsilon})$

Recall Circuit complexity

- Given Boolean function family $\{f_n\}_n$ with $f_n : \{0,1\}^n \to \{0,1\}$ show lower bounds on smallest circuit computing f_n.
- Hope: Can show NP $\neq$ P by showing NP $\not\subseteq$ P$/_{\text{poly}}$.
- Wait - what?
- P$/_{\text{poly}}$ includes undecidable languages!
- Why should it not just contain NP, if it is so powerful!
- Karp-Lipton: Non-uniformity is not too powerful in deciding uniform languages. Specifically:

Thm: If NP $\subseteq$ P$/_{\text{poly}}$ then PH collapses.

Intuition: Why non-uniformity is not needed

- Suppose C is a circuit that decides SAT.
- Then can guess C using existential quantifier.
- Can verify C is right using a universal quantification.
- To decide ϕ, can now compute $C(\phi)$.
- Conclude: NP is in Σ_2^P (Er ... not too impressive!)

Still helps in the hierarchy

- Say trying to decide $\Sigma 4^P$-complete language, L
  - $L = \{\phi | \exists x_1, \forall x_2, \exists x_3, \forall x_4 \phi(x_1, x_2, x_3, x_4) = 1\}$.
- Lets guess the circuit C needed to decide the language $L' = \{(\phi, x_1, x_2, x_3) | \forall x_4 \phi(x_1, x_2, x_3, x_4) = 1\}$.
- Crucial point: Don't need to know which ϕ, x_1, x_2, x_3 this circuit is going to be applied to, to guess and verify it. So can do it in parallel to our computation.

Formally

Can guess C while guessing x_1 and verify while enumerating x_2 ... and so:

$\phi \in L$ iff
$$\begin{aligned}
&\exists x_1, C\\
&\forall x_2, \phi', x_1', x_2', x_3', y_4'\\
&\exists x_3, x_4' \text{ s.t.}\\
&C(\phi', x_1', x_2', x_3') \Rightarrow \phi'(x_1', x_2', x_3', x_4')\\
&\bigwedge \phi'(x_1', x_2', x_3', y_4') \Rightarrow C(\phi', x_1', x_2', x_3')\\
&\bigwedge C(\phi, x_1, x_2, x_3) = 1.
\end{aligned}$$

This language is clearly in Σ_3^P.

Fortnow's theorem

Theorem: [Fortnow '97] If SAT $\in$ L, then $\exists \epsilon > 0$ s.t. SAT $\notin$ Time$(n^{1+\epsilon})$.

Proof: Assume SAT $\in$ L, and SAT $\in \cap_{\epsilon > 0}$ Time$(n^{1+\epsilon})$.

Then .... will get contradication (after few slides).

Proof Idea

1. SAT in Time$(n^{1+\epsilon})$, implies non-determinism is not very powerful, & so alternation is not very powerful.
2. SAT is complete for NTIME(n) implies SAT is very powerful.
3. SAT in L implies small space computation is very powerful.
4. Savitch's theorem implies alternation is powerful in small space ccomputation, and hence very powerful for all computation.
5. Contradiction to (1)!

How to formalize all this? Use (Time) Hierarchy theorem.

Fortnow: Step 1

Fact 1: If SAT $\in$ L, then TIME$(T(n)) \subseteq$ SPACE$c \cdot \log T(n)$

Proof: Padding + completeness of SAT under Logspace reductions.

Fortnow: Step 2

Fact 2: SPACE$(s) \subseteq$ ATIME$[i, i2^{s/i}s]$.

Proof:

- Draw depth i tree of width w having 2^s leaves.
- At top level, Guess w intermediate configurations $c_1, \ldots, c_w$ and for all successive pairs c_j, c_{j+1} verify reach from c_j to c_{j+1} in w^{i-1} steps.

Corollary: (with TIME$(T) \subseteq$ ATIME$[i, (T)^{c/i}]$.

Fortnow: Step 3

Fact 3: If,say, SAT $\in$ TIME$(n^{1+\epsilon})$, then ATIME[a,t] $\subseteq$ TIME$t^{(1+\epsilon)^{2i}}$.

Proof:

- Induction on # alternations.
- Use strong form of Cook's theorem at every step.
- Take care to make sure numbers work out.

Contradiction?

Have

Time$(T(n) = 2^{2^{\sqrt{\log n}}})$
$\subseteq (\log T)$
$\subseteq$ ATime$[i, T^{c/i}]$
$\subseteq$ Time$(T^{(c/i)(1+\epsilon)^{2i}})$.

Contradicts if $(c/i)(1+\epsilon)^{2i} < 1$. Can be arranged by picking $i = 10c$ and $\epsilon = 1/(2i)$.