Last time: Finished randomness unit
  \begin{itemize}
  \item $\text{BPP} \subseteq \text{P/poly} \ (\text{Adleman})$
  \item $\text{BPP} \subseteq \Sigma^p_2 \cap \Pi^p_2 \ (\text{Sipser - Gacs - Lautemann})$
  \end{itemize}

Readings: Pap. 11.2, AB 7.4, 9.5.4

Today: start unit on complexity of counting problems
  \begin{itemize}
  \item motivation, basics, $\#P$, $FP$
  \item $\#P$-completeness, $\text{PERMANENT}$
  \item Readings: Pap. 18.1, AB 17.1-17.3
  \end{itemize}

Midterm: back soon.

Questions?

Motivation / basics

Counting problem: output a $\#$ (count objects)

Ex:
  \begin{itemize}
  \item $\#\text{SAT}$: input is $\varphi$, a Boolean formula $\varphi(x_1, \ldots, x_n)$
    \vspace{.1in}
    \begin{itemize}
    \item $Q$: how many $x = (x_1, \ldots, x_n) \in \{0,1\}^n$ have $\varphi(x) = 1$?
    \end{itemize}
    \vspace{.1in}
    \begin{itemize}
    \item hard to find...
    \end{itemize}
  \item $\#\text{CYCLES}$: input $G = (V, E)$ undirected graph
    \vspace{.1in}
    \begin{itemize}
    \item $Q$: how many simple cycles?
    \end{itemize}
    \vspace{.1in}
    \begin{itemize}
    \item easy to find if exist
    \end{itemize}
  \item $\#\text{PATHS}$: input is $G = (V, E)$ directed graph, $s$, $t \in V$
  \end{itemize}
Q: How many directed acyclic paths?

Motivation:

• Connected to other fields:
  - Statistical physics (_configuration of a physical system?)
  - Combinatorial enumeration

• Applications in various areas:
  - Network reliability
  - AI/decision making under uncertainty

Robot, int vars $x_1, \ldots, x_n$

Robot should act if $\Psi(x_1, \ldots, x_n)$

Uncertainty about variables... $x_i \in \{0, 1\}$

$\Pr[\text{should act}] = \frac{\#s.t. \Psi}{2^n}$.

• Counting problems can be hard even if corresponding decision problems are easy (#CYCLES, #PATHS).
• Only interested in counting "easy to verify/recognize" objects.

$\text{#P} :$ the class of counting problems.

_Definition:_ A function $f: \Sigma^* \rightarrow \mathbb{N}$ is in $\text{#P}$ if

1. There's a polytime NTM $N$ s.t. $\forall x,$
   
   $f(x) = \# \text{ of acc. computations of } N \text{ on } x.$

   Or, if you like,
Alt: \( f \in \#P \) if there's a poly-time verifier \( V(\cdot, \cdot) \) for some lang. \( L \in \text{NP} \), s.t. \( \forall x, f(x) = |\{w: V(x, w) \text{ accepts}\}|. \)

- Recall: "\( V \) is a poly-time verif. for \( L \)" means:
  - \( V(x, w) \) runs in poly(\(|x|\)) time \( \forall x, w, \) \( V \)
  - \( \forall x, x \in L \) iff \( \exists w \text{ s.t } V(x, w) \text{ accepts}. \)

Ex: I claim mult. is in \#P.

Input: \((a, b)\)

NTM: guess \( 1 \leq x \leq a \)

" \( 1 \leq y \leq b \)

verifies \( x \cdot y \leq a \cdot b. \)

Note: the \( L \) corr. to the NTM \( N \) (or verif. \( V \)) needn't be a "hard" lang. in NP (CYCLES, PATH).

- and there are "easy" counting problems in \#P.

Returning to \#SAT:

\[ V(\varphi, z) : \quad z = \text{ ass to vars}, \]

\[ x \cdot w \quad \varphi = \text{ a Bool formula} \]

\[ V(\varphi, z) \text{ acc. iff } \varphi(z) = 1. \]

Similar for \#PATHS

\[ V(G, (u_1, u_2, \ldots, u_n)) \]

\[ V \text{ checks } s = u, t = u, u_i \rightarrow u_{i+1}, \text{ edge is present} \]
in G for i=1,...,r-1.

Fact: Let \( L \in \text{NP} \) (\( \forall N \text{ corr. NTM} \)),
\( \exists f \) be the fn in \( \#P \) corr. to \( N \).
For all \( x \), \( f(x) \geq 0 \) if \( x \in L \).

So computing \( f(x) \) is at least as hard as deciding \( x \in L \).

Def: \( FP = \{ \text{all det. poly-time computable fns } f : \Sigma^* \rightarrow \mathbb{N} \} \).

"function-P"
"easy problems"

Cor: If \( \#P \leq FP \) then \( P = NP \).

Completeness
right notion

Recall NPC:
\( L_1 \leq_p L_2 \text{ poly-time reduc} \)

For counting, oracles provide the "right" notion of reduc.
Recall: oracle for lang. $L$: black box

$\chi \rightarrow L_{\text{oracle}}$

$Y/N$

$x \in L \iff x \notin L$

$P^L: L \in P^L \iff \exists f \text{ TM } M^f \text{ s.t. } M^f \text{ dec. } L$

---

Function oracles: $f: \Sigma^* \rightarrow \mathbb{N}$

$TM M$ writes a string $\varepsilon$ on "oracle tape"; enters $2$ oracles in one time step, the value $f(\varepsilon)$ is written on "oracle response tape".

Space complexity: # cells of worktapes, oracle tapes visited by TM.

$M^f: TM$ with function oracle $f$

$P^f$: class of lang. dec. by some poly time $M$ equipped with an $f$-oracle

$FP^f$: class of functions computed by some poly time $M$ equipped with an $f$-oracle

---

Reminder: it's okay for $M^f$ to call the $f$-oracle multiple times.

---

Def: A function $f: \Sigma^* \rightarrow \mathbb{N}$ is $\#P$-complete if

1. $f \in \#P$
2. for every $g \in \#P$, have $g \in FP^f$. 
If \( f \in \text{FP} \), then \( \text{FP}^f = \text{FP} \), so

Claim: if \( f \) is \( \#P \)-complete \& \( f \in \text{FP} \), then \( \text{FP} = \#P \).

Recall: \( \#3\text{CNF} \): input is a 3CNF, output is \( \#\text{sat} \). assts.

Thm: \( \#3\text{CNF} \) is \( \#P \)-complete.

Proof sketch: ① Easy to see \( \#3\text{CNF} \in \#P \).
② must show \( \forall g \in \#P \), have \( g \in \text{FP} \) \( \#3\text{CNF} \).

\[ \xrightarrow{(\text{N accepts})} \]

Fix \( g \in \#P \). So \( \exists \text{NTM} \ N \), \( \lambda \), \( L \) s.t. \( g(x) = \#\text{acc comp. of } N \text{ on } x \).

Reduce to \( \#3\text{CNF} \) in two steps:
① \( L \leq_P \text{CIRCUIT-SAT} \) \( \Rightarrow \) languages.
② \( \text{CIRCUIT-SAT} \leq 3\text{CNF} \)

Each \( \text{reduce} \). ② \( \text{preserves } \#\text{accepting comp.} \).

②: Cook-Levin reduc. \( R \) from \( L \) to \( \text{CUT-SAT} \): given \( x \), constructs \( C \) s.t. \( x \) of \( \text{each assignment to inputs} \) \( \iff \) \( N \) makes \( n \) nondet steps that of \( C \)

\( C \) accepts partie. asst. \( \iff \) \( N \) accepts under that parte.
So \( g(x) = \# \text{sat. asssts. to ckt } C = R_{CC}(x) \).

(b) Second reduc. \( R_{3CNF} \) from \( \text{CKT-SAT to 3CNF} \): given as input ckt \( C \), \( R_{3CNF} \) constructs a 3CNF \( \psi(x, y) \) s.t.

\[
C(x) = 1 \implies \exists! \text{ assst to y-vars s.t. } \psi(x, y) = 1
\]

\[
C(x) = 0 \implies \text{no assst to y-vars causes } \psi(x, y) = 1
\]

(either some gate's integrity is violated, or output = 0)

\[
\text{Thm 9.34 Sipser}
\]

So \( \# \text{sat. of } C = \# \text{sat. of } \psi \).

So \( g \in \text{FP}^{\#3CNF} \): on input \( x \), perform \( R_{CC} \) to get ckt \( C \); then perform \( R_{3CNF} \) on \( C \) to get \( \psi \), a 3CNF whose \( \# \text{of sat. asssts = acc. comp. of } L \text{ on } x \).

Call \( \#3CNF \) oracle on \( \psi \), return result.

\( \#3CNF \): existence hard, counting hard.

A problem where \( \nabla \text{ easy, } \forall \text{ }.
3DNF: OR of ANOS of width 3

a 3DNF \( x_1, x_2, \ldots, x_9 \) \( \lor \) \( x_2 x_4 x_5 \lor x_6 x_7 x_8 \lor x_8 x_9 \)

3DNF sat. question: trivial to determine existence. But...

Thm: \( \#3\text{DNF} \) is \#P-complete.

Pf: Given \( \varphi \) a 3DNF, \( \overline{\varphi} \) is a 3CNF.

So given \( \varphi \in \text{3CNF} \), write down the 3DNF \( \overline{\varphi} \).

Call \( \#3\text{DNF} \) oracle, get answer \( m \), return \( 2^m - m \).

\( \#3\text{DNF} \) in \#P;

above shows every \( \text{P} \in \#\text{P} \) is \text{rd.} to \( \#3\text{DNF} \).

(b/c every \( \text{P} \in \#\text{P} \) is \text{rd.} to \( \#3\text{CNF}, \#3\text{CNF} \text{rd.} to \#\text{DNF} \))

Next time: PER (\& more.)