

## 601 Lect 23: Recall From Last Time

**Prop:** QSAT, GEOGRAPHY, and SUCCINCT REACH are **PSPACE** complete.

**Prop:** HORN-SAT, EMPTY-CFL, AREACH, MCVP, and CVP are **P** complete.

# There are very few natural problems that are

- Known to be in NP, and
- Not known to be NP-complete, and
- Not known to be in P

## Examples:

- Factoring natural numbers
- Graph Isomorphism
- Model Checking the  $\mu$ -Calculus

**Theorem:** [Ladner] If  $P \neq NP$  then there exist problems that are in  $NP - P$  that are not NP complete.

**Theorem:** [Ladner] If  $\mathbf{P} \neq \mathbf{NP}$  then there exists a problem  $C \in \mathbf{NP} - \mathbf{P}$  that is not  $\mathbf{NP}$  complete.

**Proof:** Assume that  $\mathbf{P} \neq \mathbf{NP}$ .

We will construct  $C$  by a method called “delayed diagonalization”.

The construction will make sure that:

•  $C$  is **easy**:  $\mathbf{SAT} \not\leq C$ .  $R_1, R_3, R_5, \dots$

•  $C$  is **hard**:  $C \notin \mathbf{P}$ .  $R_2, R_4, R_6, \dots$

$R_{2k+1}$  : “ $M_k$  isn’t a  $\mathbf{DSPACE}[k \log n]$  reduction from  $\mathbf{SAT}$  to  $C$ ”

$R_{2k+2}$  : “ $M_k$  isn’t a  $\mathbf{DTIME}[kn^k]$  recognizer of  $C$ ”

**Observation:** If all the  $R_i$ ’s are met, then we’re done!

SAT

$R_2$

$C$

$R_4$

$\emptyset$

$R_1$

$R_3$

**Conditions to Satisfy:**  $R_i, \quad i = 1, \dots, \infty$

$R_{2k-1}$  : “ $M_k$  isn't a **DSPACE** $[k \log n]$  reduction from SAT to  $C$ ”

$R_{2k}$  : “ $M_k$  isn't a **DTIME** $[kn^k]$  recognizer of  $C$ ”

On input  $w$ , recursively and **deterministically**  $C(w)$  does following:

1. **do** for  $|w|$  steps {
2.     **for**  $i = 1 \dots \infty$  **do** {
3.         **for**  $x = 1 \dots \infty$  **do** {
4.             **if** ( $R_i$  verified on input  $x$ ) **then** next  $i$
5.         } } }
6. **if** ( $i$  is even and  $w \in$  SAT) **then** ACCEPT
7. **else** REJECT



# Isn't that a nice proof?

Oh, you didn't understand?

Here's a bit more detail: When we say " $R_i$  verified on input  $x$ " we mean the following: If  $i = 2k - 1$ , then  $R_i$  asserts that " $M_k$  isn't a **DSPACE** $[k \log n]$  reduction from **SAT** to **C**". We therefore deterministically check if  $x \in \text{SAT}$  – note that this could take  $O(2^n)$  steps, where  $n = |x|$  – and then we run  $M_k(x)$  using space at most  $k \log n$  and time at most  $2^{k \log n}$ . If  $M_k$  requires too much space, or is in a loop then we have proved condition  $R_i$ . Otherwise, we get an output,  $z$ , and we check **recursively** whether  $C(z)$  holds. Note that  $|z| < |w|$  because we are only running this whole algorithm for at most  $|w|$  steps when the original input is  $w$ . If this part of the computation completes, then we check that  $x \in \text{SAT} \Leftrightarrow z \in C$ . If not, then we have proved  $R_i$  and with what time remains we go on to the next  $i$ , otherwise with what time remains we go on to the next  $x$ .

# Isn't that a nice proof? (continued)

If  $i = 2k$  then  $R_i$  asserts that “ $M_k$  isn't a **DTIME** $[kn^k]$  recognizer of  $C$ ”. We therefore run  $M_k(x)$  for at most  $k|x|^k$  steps. If it fails to output a boolean value in this time, then we have verified condition  $R_i$ . Otherwise we recursively call  $C(x)$  and check that  $C(x) = M_k(x)$ . If not, then we have verified condition  $R_i$  and with what time remains we go on to the next  $i$ , if so, then with what time remains we go on to the next  $x$ .

Now, please go back two slides and say, “Oh, now I get it, that is a nice proof!”

$$\text{PRIME} = \{m \in \mathbf{N} \mid m \text{ is prime}\}$$

**Prop:**  $\overline{\text{PRIME}} \in \mathbf{NP}$

**Proof:**

$$m \in \overline{\text{PRIME}} \Leftrightarrow m < 2 \vee \exists xy (1 < x < m \wedge x \cdot y = m)$$



**Question:** Is  $\text{PRIME} \in \mathbf{NP}$ ?

**Fact: (Fermat's Little Thm):** Let  $p$  be prime and  $0 < a < p$ , then,  $a^{p-1} \equiv 1 \pmod{p}$ .

$$\mathbf{Z}_n^* = \{a \in \{1, 2, \dots, n-1\} \mid \text{GCD}(a, n) = 1\}$$

$Z_n^*$  is the multiplicative group of integers mod  $n$  that are relatively prime to  $n$ .

**Euler's phi function:**  $\varphi(n) = |\mathbf{Z}_n^*|$

**Prop:** If  $n = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$  is the prime factorization of  $n$ , then

$$\varphi(n) = n(p_1 - 1)(p_2 - 1) \cdots (p_k - 1) / (p_1 p_2 \cdots p_k)$$

**Euler's Thm:** For any  $n$  and any  $a \in \mathbf{Z}_n^*$ ,  $a^{\varphi(n)} \equiv 1 \pmod{n}$ .

**Fact:** Let  $p > 2$  be prime. Then  $\mathbf{Z}_p^*$  is a cyclic group of order  $p - 1$ . That is,

$$\mathbf{Z}_p^* = \{a, a^2, a^3, \dots, a^{p-1}\}$$

$$m \in \text{PRIME} \quad \Leftrightarrow \quad \exists a \in \mathbf{Z}_m^* (\text{ord}(a) = m - 1)$$

**Pratt's Thm:**    **PRIME**  $\in$  **NP**.

**Proof:** Given  $m$ ,

1. Guess  $a$ ,  $1 < a < m$
2. Check  $a^{m-1} \equiv 1 \pmod{m}$  by repeated squaring.
3. Guess prime factorization:  $m - 1 = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$
4. Check for  $1 \leq i \leq k$ ,  $a^{m-1/p_i} \not\equiv 1 \pmod{m}$
5. Recursively check that  $p_1, p_2, \dots, p_k$  are prime.

**Divide and Conquer NP Algorithm:**

$$T(n) = O(n^2) + T(n - 1)$$

$$T(n) = O(n^3)$$

□

**Cor:** PRIME and FACTORING are in  $\mathbf{NP} \cap \mathbf{co-NP}$ .

**Proof:** PRIME: immediately from Pratt's Thm.

FACTORING is the problem of given  $N$ , find its prime factorization:  $N = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$ .

Think of this as a decision problem by putting the factorization in a standard form, e.g.,  $p_1 < p_2 < \cdots < p_k$ , and asking if bit  $i$  of the factorization is "1".

This is in  $\mathbf{NP} \cap \mathbf{co-NP}$  because an  $\mathbf{NP}$  or  $\mathbf{co-NP}$  machine can guess the unique prime factorization, and check that it is correct. □

# More Primality Testing

$a \in \mathbf{Z}_m^*$  is a **quadratic residue** mod  $m$  iff,  $\exists b (b^2 \equiv a \pmod{m})$

For  $p$  prime let,

$$\left(\frac{a}{p}\right) = \begin{cases} 1 & \text{if } a \text{ is a quadratic residue mod } p \\ -1 & \text{otherwise} \end{cases}$$

Generalize to  $\left(\frac{a}{m}\right)$  when  $m$  is not prime,

$$\left(\frac{a}{mn}\right) = \left(\frac{a}{m}\right) \left(\frac{a}{n}\right)$$

$$\left(\frac{a}{m}\right) = \left(\frac{(a \bmod m)}{m}\right)$$

**Quadratic Reciprocity Thm: [Gauss]** For odd  $a, m$ ,

$$\left(\frac{a}{m}\right) = \begin{cases} \left(\frac{m}{a}\right) & \text{if } a \equiv 1 \pmod{4} \text{ or } m \equiv 1 \pmod{4} \\ -\left(\frac{m}{a}\right) & \text{if } a \equiv 3 \pmod{4} \text{ and } m \equiv 3 \pmod{4} \end{cases}$$

$$\left(\frac{2}{m}\right) = \begin{cases} 1 & \text{if } m \equiv 1 \pmod{8} \text{ or } m \equiv 7 \pmod{8} \\ -1 & \text{if } m \equiv 3 \pmod{8} \text{ or } m \equiv 5 \pmod{8} \end{cases}$$

Thus, we can calculate  $\binom{a}{m}$  efficiently.

$$\begin{aligned}\binom{107}{351} &= -\binom{351}{107} = -\binom{30}{107} \\ &= -\binom{2}{107} \binom{15}{107} = -\binom{107}{15} \\ &= -\binom{2}{15} = -1\end{aligned}$$

$$107 \equiv 351 \equiv 15 \equiv 3 \pmod{4}$$

$$107 \equiv 3 \pmod{8}; \quad 15 \equiv 7 \pmod{8}$$

**Fact:**[Gauss] For  $p$  prime,  $a \in \mathbf{Z}_p^*$ ,  $\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \pmod{p}$ .

**Fact:** If  $m$  not prime then,

$$\left| \left\{ a \in \mathbf{Z}_m^* \mid \left(\frac{a}{m}\right) \equiv a^{\frac{m-1}{2}} \pmod{m} \right\} \right| < \frac{m-1}{2}$$

## Solovay-Strassen Primality Algorithm:

1. Input is odd number  $m$
2. For  $i := 1$  to  $k$  **do** {
3.     choose  $a < m$  at random
4.     **if**  $\text{GCD}(a, m) \neq 1$  **return**("not prime")
5.     **if**  $\left(\frac{a}{m}\right) \not\equiv a^{\frac{m-1}{2}} \pmod{m}$  **return**("not prime")
6. }
7. **return**("probably prime")

## Thm:

- If  $m$  is prime then Solovay-Strassen( $m$ ) returns “probably prime”.
- If  $m$  is not prime, then the probability that Solovay-Strassen( $m$ ) returns “probably prime” is less than  $1/2^k$ .

**Cor:** PRIME  $\in$  “Truly Feasible”

## Thm:

- If  $m$  is prime then Solovay-Strassen( $m$ ) returns “probably prime”.
- If  $m$  is not prime, then the probability that Solovay-Strassen( $m$ ) returns “probably prime” is less than  $1/2^k$ .

**Cor:** PRIME  $\in$  “Truly Feasible”

**Fact:** [Agrawal, Kayal, and Saxena, 2002] PRIME  $\in$  P

**Def:** A decision problem  $S$  is in **BPP** (Bounded Probabilistic Polynomial Time) iff there is a probabilistic, polynomial-time algorithm  $A$  such that for all inputs  $w$ ,

$$\mathbf{if} (w \in S) \mathbf{then} \text{Prob}(A(w) = 1) \geq \frac{2}{3}$$

$$\mathbf{if} (w \notin S) \mathbf{then} \text{Prob}(A(w) = 1) \leq \frac{1}{3}$$

**Prop:** If  $S \in \mathbf{BPP}$  then there is a probabilistic, polynomial-time algorithm  $A'$  such that for all  $n$  and all inputs  $w$  of length  $n$ ,

$$\mathbf{if} (w \in S) \mathbf{then} \text{Prob}(A'(w) = 1) \geq 1 - \frac{1}{2^n}$$

$$\mathbf{if} (w \notin S) \mathbf{then} \text{Prob}(A'(w) = 1) \leq \frac{1}{2^n}$$

**Proof:** Iterate  $A$  polynomially many times and answer with the majority. Probability the mean is off by  $\frac{1}{3}$  decreases exponentially with  $n$  — Chernoff bounds. □

Is **BPP** equal to **P**???

Probably, because pseudo-random number generators are good.

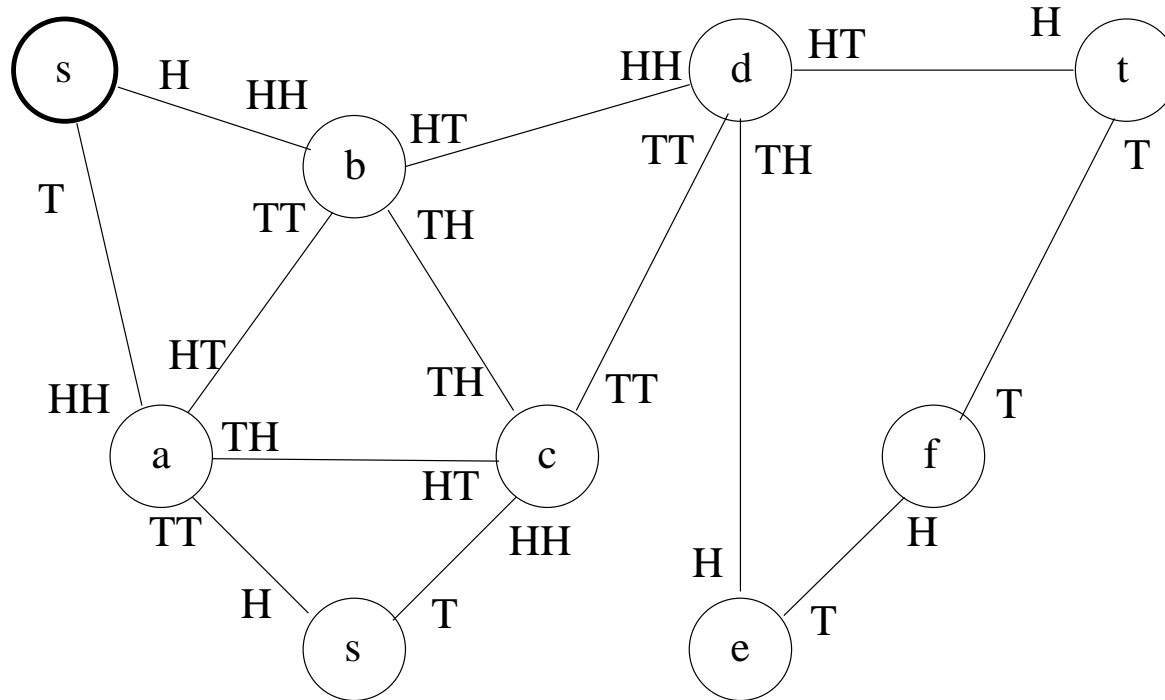
Is randomness ever useful?

Yes: *Theory of Games and Economic Behavior*, by John Von Neumann, and Oskar Morgenstern, Princeton university press, 1944.

Colonel Kelly:  
Which base to inspect?

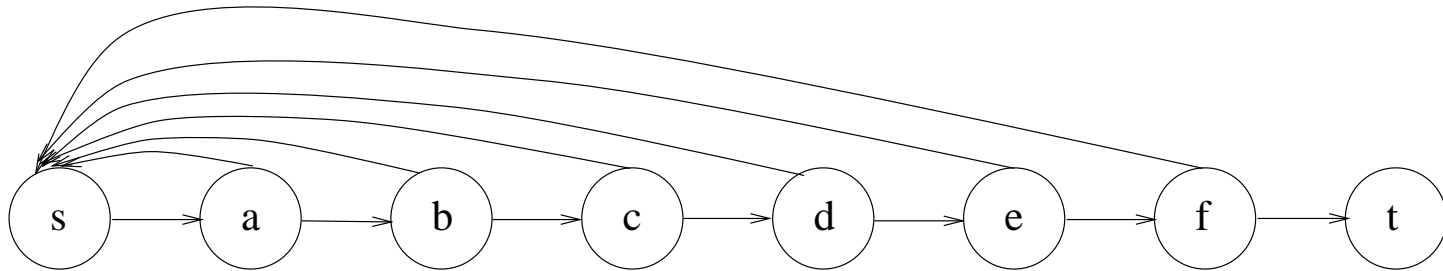
If we randomize, then our opponent cannot know what we will do.

$$\text{UREACH} = \left\{ G, \text{ undirected} \mid s \xrightarrow[G]{*} t \right\}$$



**Fact:** Let  $T(i)$  be the expected number of steps in a random walk to visit all vertices in connected graph  $G$ , starting from  $i$ . Then,  $T(i) \leq 2e(n - 1)$ .

**Cor:** UREACH  $\in$  BPL.

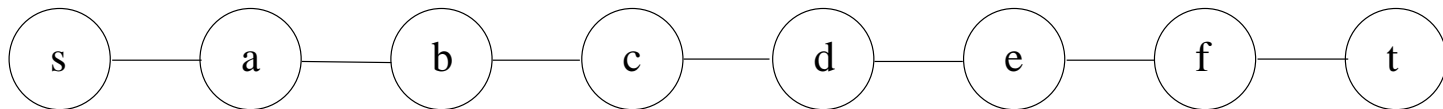


A *universal traversal sequence* for graphs on  $n$  nodes, is a sequence of instructions

$$q = a_1 a_2 a_3 \cdots a_t \in \{1, \dots, n-1\}^*$$

Such that for any **undirected** graph on  $n$  nodes, if we start at  $s$  in  $G$  and follow  $q$ , then we will visit every vertex in the connected component of  $s$ .

**Fact:**  $\exists$  universal traversal sequences of length  $O(n^3)$ .



**Not true for directed graphs.**

**Fact:** [Reingold, 2004]  $\text{UREACH} \in \mathbf{L}$

**Proof idea:** derandomization of universal traversal sequences using expander graphs. □

**Cor:**  $\text{Symmetric-L} = \mathbf{L}$