**Theorem:**

- 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 $1/2^k$.

**Corollary:** PRIME $\in$ "Truly Feasible"

**Definition:** 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 $w$,

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

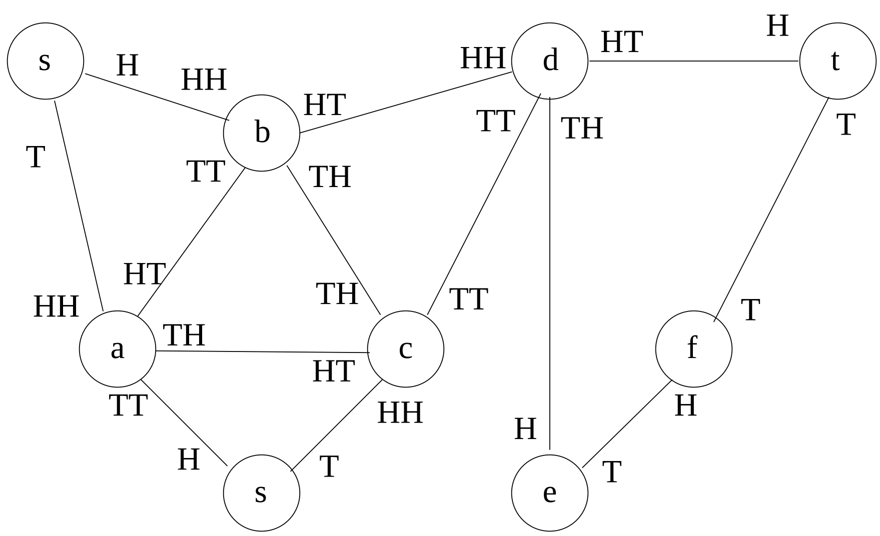
**Equivalently,** there is a probabilistic, polynomial-time algorithm $A'$ such that for all $n$ and all inputs $w$ of length n,

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

**Other Randomized Classes:**

- **NP**: $\text{Prob}(A(w) = 1) > 0$ iff $w \in S$
- **RP**: $\text{Prob}(A(w) = 1) \geq 1/2$ for $w \in S$, else $\text{Prob}(A(w) = 1) = 0$
- **PP**: $\text{Prob}(A(w) = 1) > 1/2$ for $w \in S$, else $\text{Prob}(A(w) = 1) < 1/2$

$$\text{REACH}_u \quad = \quad \{G, \text{ undirected} \mid s \underset{G}{\overset{\star}{\to}} t\}$$

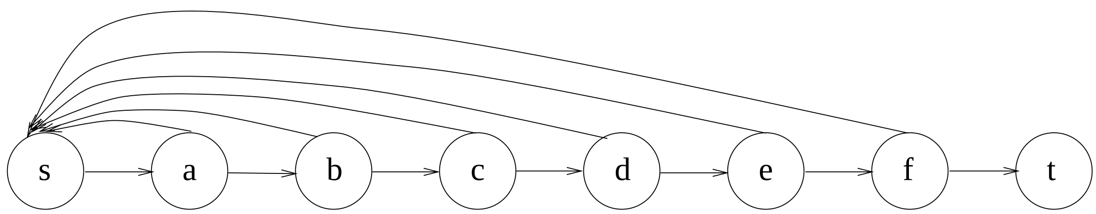



**Fact 26.1** *Let $T(i)$ be the expected number of steps in a random walk to visit all vertices in connected graph $G$, starting from $i$. Then,*

$$T(i) \;\leq\; 2e(n-1)$$

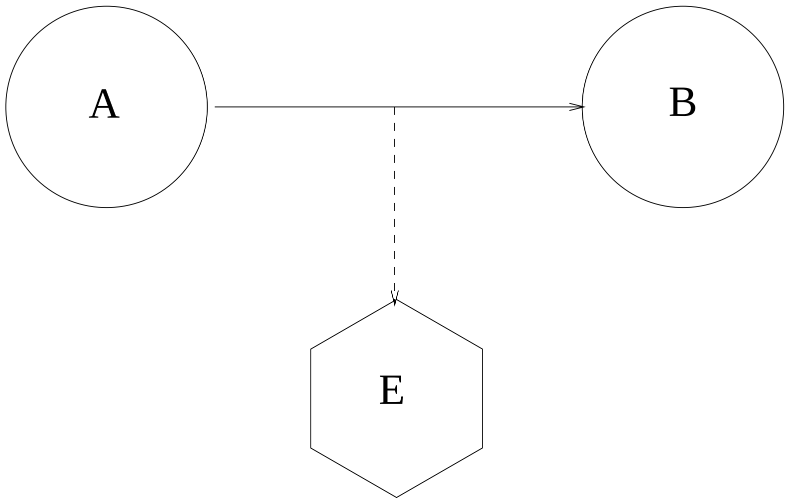
**Corollary 26.2**

$$\text{REACH}_u \;\in\; \text{BPL}$$




A look at this *directed* graph should convince you that a random walk on it is *not* likely to reach all vertices in polynomial time. To get to vertex $t$ from $s$ you would have to guess right about $n$ times in a row.

It's very plausible that $\mathrm{REACH}_u$ is in **L**, and one might hope to prove it by derandomizing the random walk. (There must exist a single sequence of choices of size $O(n^3)$ that visits every node of *any* undirected labelled $n$-node graph.) But randomization doesn't seem to help much with the general REACH problem.




**One-Time Pad:** $p \in \{0,1\}^n; m \in \{0,1\}^n$

$$E(p,x) = p \oplus x$$

$$D(p,x) = p \oplus x$$

$$D(p, E(p,m)) = p \oplus (p \oplus m) \quad = \quad m$$

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $p$ | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| $m$ | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |
| $E(p, m)$ | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 |
| $D(p, E(p, m))$ | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 |

**Theorem 26.3** *If $p$ is chosen at random and known only to $A$ and $B$ then*

1. *$E(p, m)$ provides no information to $E$ about $m$ except perhaps its length.*
2. *Better not use $p$ more than once! XOR of two messages with same pad is plaintext XOR plaintext, easy to attack.*

$B$ chooses $p, q$ $n$-bit primes,

$B$ chooses GCD$(e, \varphi(pq)) = 1$;   $\varphi(pq) = (p-1)(q-1)$.

$B$ **publishes**: $pq, e$;   keeps $p, q$ **secret**.

$B$ computes $d, k$, s.t. $ed + k\varphi(pq) = 1$

Break message into pieces shorter than $2n$ bits

$$\begin{aligned}
E_B(x) &\equiv x^e \pmod{pq} \\
D_B(x) &\equiv x^d \pmod{pq}
\end{aligned}$$

$$\begin{aligned}
D_B(E_B(m)) &\equiv (m^e)^d \pmod{pq} \\
&\equiv m^{1-k\varphi(pq)} \pmod{pq} \\
&\equiv m \cdot (m^{\varphi(pq)})^{-k} \pmod{pq} \\
&\equiv m \pmod{pq} \\
&\equiv E_B(D_B(m)) \pmod{pq}
\end{aligned}$$

For **sufficiently large** $n$, [$n \geq 128$ is fine in 2002],

**It is widely believed that:** $E_B(m)$ divulges no useful information about $m$ to anyone not knowing $p, q$, or $d$.

**Message signing:**

Let $m =$ "$B$ promises to give $A$ $10 by 5/17/02."

Let $m' = m \circ r$ where $r$ is nonce or current date and time

**It is widely believed that:** $D_B(m')$ could be produced only by $B$. Thus it can be used as a contract signed by $B$!

Useful for proving authenticity!

[Goldwasser, Micali, Rackoff], [Babai]

Decision problem: $D$; input string: $x$

Two players:

**Prover — Merlin** is computationally all-powerful. Wants to convince **Verifier** that $x \in D$.

**Verifier — Arthur**: probabilistic polynomial-time TM. Wants to know the truth about whether $x \in D$.

$$\text{Input} = x; \quad n = |x|; \quad t = n^{O(1)}$$

| | | |
|---|---|---|
| 0. | **A** has $x$ | **M** has $x$ |
| 1. | flip $\sigma_1$, compute $m_1 \longrightarrow$ | |
| 2. | | $\longleftarrow m_2$ |
| 3. | flip $\sigma_3$, compute $m_3 \longrightarrow$ | |
| 4. | | $\longleftarrow m_4$ |
| $\vdots$ | $\vdots$ | $\vdots$ |
| $2t$. | | $\longleftarrow m_{2t}$ |
| $2t+1$. | flip $\sigma_{2t+1}$, accept or reject | |

**Definition 26.4** $D \in$ IP iff there is such a polynomial-time interactive protocol

1. If $x \in D$, then there exists a strategy for $\mathbf{M}$

$$\text{Prob}\{\mathbf{A}\text{ accepts}\} \ > \ \frac{2}{3}$$

2. If $x \notin D$, then for all strategies for $\mathbf{M}$

$$\text{Prob}\{\mathbf{A}\text{ accepts}\} \ < \ \frac{1}{3}$$

♠

**Observation 26.5** *Iterating makes probabilities of error exponentially small.*

**Special Cases of** IP**:**

- Deterministic Arthur = **NP**
- No Merlin = **BPP**

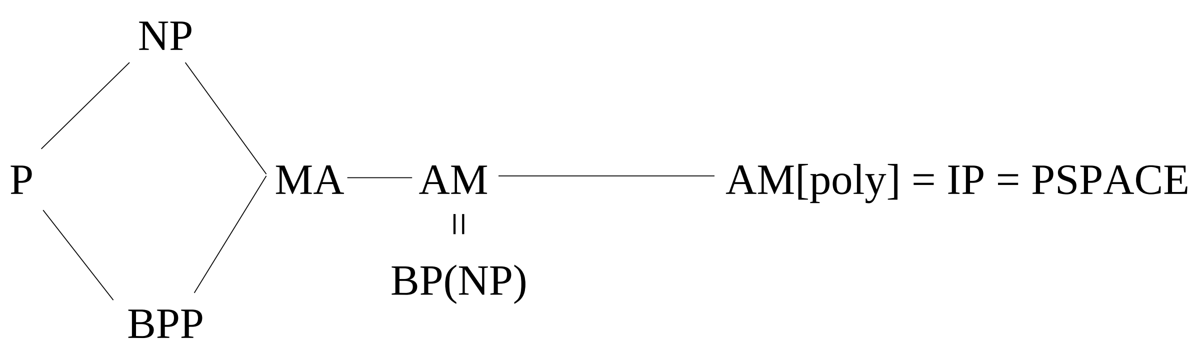
**Definition 26.6** MA is the set of decision problems admitting two step proofs where Merlin moves first.

AM is the set of decision problems admitting two step proofs where Arthur moves first.

$$\mathrm{AM}[2k] \quad = \quad \underbrace{AM\,AM \cdots AM}_{2k} \qquad \spadesuit$$

**Fact 26.7** [Babai] *For all $k \geq 2$,*

$$\mathrm{AM}[k] \quad = \quad \mathrm{AM}$$



**Fact 26.8** [Goldwasser,Sipser] *The power of interactive proofs is unchanged if* $\mathbf{M}$ *knowns* $\mathbf{A}$*'s coin tosses. For all* $k$*,*

$$\mathrm{IP}[k] \quad = \quad \mathrm{AM}[k]; \qquad \mathrm{IP} \quad = \quad \mathrm{AM}[n^{O(1)}]$$

## Graph Non-Isomorphism $\in$ AM

Input = $G_0, G_1$, $n = \|G_0\| = \|G_1\|$

| | A | M |
|---|---|---|
| 0. | **A** has $G_0, G_1$ | **M** has $G_0, G_1$ |
| 1. | flip $\kappa : \{1, \ldots, r\} \to \{0,1\}$<br>flip $\pi_1, \ldots, \pi_r \in S_n$<br>$\pi_1(G_{\kappa(1)}), \ldots, \pi_r(G_{\kappa(r)}) \longrightarrow$ | |
| 2. | | $\longleftarrow m_2 \in \{0,1\}^r$ |
| 3. | accept iff $\kappa = m_2$ | |

**Proposition 26.9** *Graph Non-Isomorphism* $\in$ AM

**Proof:** If $G_0 \not\cong G_1$, then **A** will accept with probability 1.

If $G_0 \cong G_1$, then **A** will accept with probability $\leq 2^{-r}$. ♠

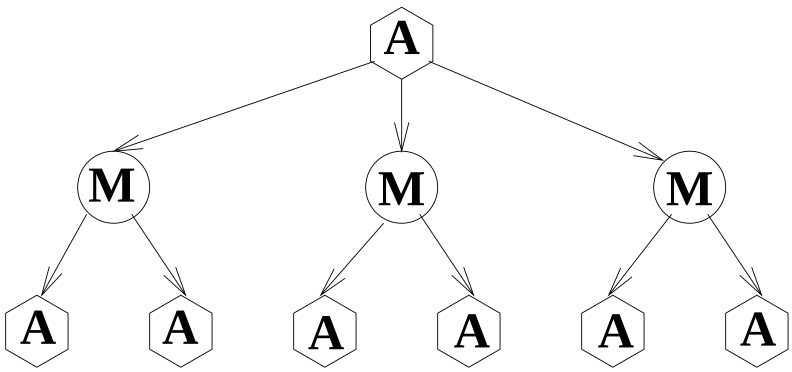
**Corollary 26.10** *If Graph Isomorphism is* **NP**-*complete then* PH *collapses to* $\Sigma_2^p$.

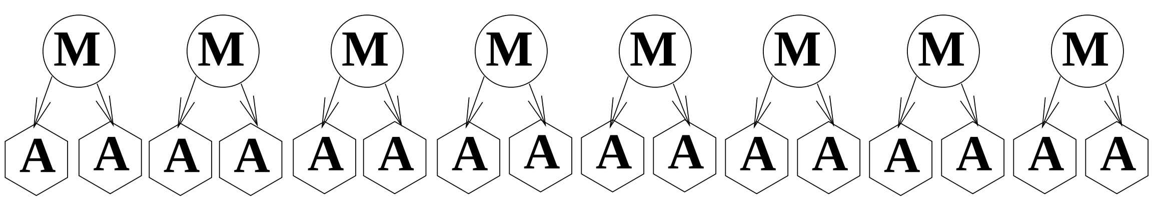
**Fact 26.11 Shamir's Theorem:** IP $=$ **PSPACE**

**proof that** IP $\subseteq$ **PSPACE:** Evaluate the game tree.

For M's moves choose the maximum value.

For **A**'s moves choose the average value.




**Hard Direction:** Construct an interactive proof that a string is in QSAT. There are proofs in [P] and in Sipser.

Any decision problem $D \in$ **NP** has a deterministic, polynomial-time verifier.

By adding randomness to the verifier, we can greatly restrict its computational power and the number of bits of $\Pi$ that it needs to look at, while still enabling it to accept all of **NP**.

We say that a verifier **A** is $(r(n), q(n))$-*restricted* iff for all inputs of size $n$, and all proofs $\Pi$, **A** uses at most $O(r(n))$ random bits and examines at most $O(q(n))$ bits of its proof, $\Pi$.

Let PCP$(r(n), q(n))$ be the set of boolean queries that are accepted by $(r(n), q(n))$-restricted verifiers.

**Fact 26.12 (PCP Theorem)** **NP** $=$ PCP$[\log n, O(1)]$

**Fact 26.13 [Hastad]** **NP** $=$ PCP$[\log n, 3]$

MAX-3-SAT: given a 3CNF formula, find a truth assignment that maximizes the number of true clauses.

$$(x_1 \vee x_2 \vee \overline{x_3}) \wedge (x_1 \vee x_4 \vee \overline{x_5}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_4}) \wedge (x_2 \vee \overline{x_3} \vee \overline{x_4})$$
$$\wedge (\overline{x_2} \vee x_3 \vee x_5) \wedge (\overline{x_3} \vee \overline{x_4} \vee \overline{x_5}) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3) \wedge (\overline{x_2} \vee \overline{x_4} \vee x_5)$$

**Proposition 26.14** MAX-3-SAT *has a polynomial-time* $\epsilon = \frac{1}{2}$ *approximation algorithm.*

**Proof:** Be greedy. ♠

**Open for Years:** Assuming **NP** $\neq$ **P** is there some $\epsilon$, $0 < \epsilon < 1$ s.t. MAX-3-SAT has no PTIME $\epsilon$-approximation algorithm?

**Theorem 26.15** *The PCP theorem (***NP** $=$ PCP$[\log n, 1]$*)*
*is equivalent to the fact that*
*If* **P** $\neq$ **NP**, *then*
*For some* $\epsilon$, $1 > \epsilon > 0$,
MAX-3-SAT *has no polynomial-time,* $\epsilon$*-approximation algorithm.*

**Fact 26.16** MAX-3-SAT *has a PTIME approximation algorithm with* $\epsilon = \frac{1}{8}$ *and no better ratio can be achieved unless* **P** $=$ **NP**.

**References:**


- *Approximation Algorithms for NP Hard Problems*, Dorit Hochbaum, ed., 1997, PWS.
- Sanjeev Arora, "The Approximability of NP-hard Problems", STOC 98, www.cs.princeton.edu/$\sim$arora.