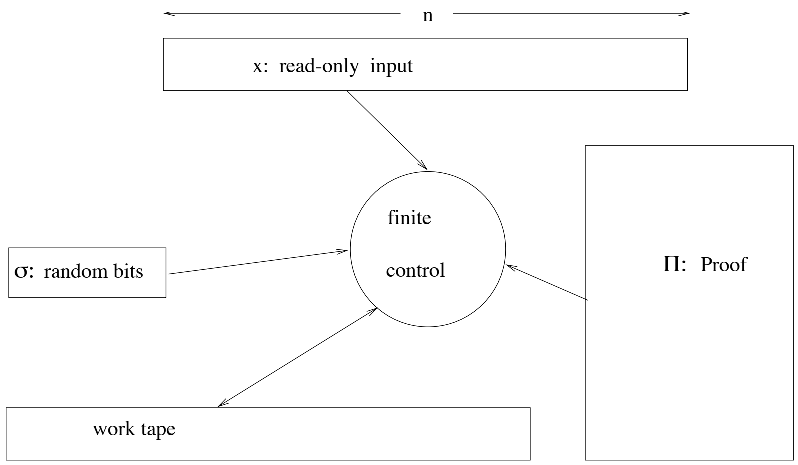
Interactive Proofs



Merlin-Arthur games (MA) [Babai]

Decision problem: D; input string: x

Merlin — Prover — chooses the polynomial-length string Π that **Maximizes** the chances of convincing Arthur that x is an element of D.

Arthur — Verifier — computes the **Average** value of his possible computations on Π, x. Arthur is a polynomial-time, probabilistic Turing machine.

Definition 17.1 We say that **Arthur accepts** D iff the following conditions hold:

1. If $x \in D$, there exists a proof Π_x, such that **Arthur** accepts for every random string σ,

$$Pr_\sigma\left[\mathbf{Arthur}^{\Pi_x}(x,\sigma) = Accept\right] = 1$$

2. If $x \notin D$, for every proof Π, **Arthur** rejects for most of the random strings σ,

$$Pr_\sigma\left[\mathbf{Arthur}^{\Pi}(x,\sigma) = Accept\right] < \frac{1}{2}$$

□

Proposition 17.2 NP $\subseteq$ MA.

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

Verifier **Arthur** is $(r(n), q(n))$-**restricted** iff **Arthur** always uses at most $O(r(n))$ random bits and examines at most $O(q(n))$ bits of its proof, Π.

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

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

Proof: Be greedy: choose the literal that occurs most often and make it true; repeat. □

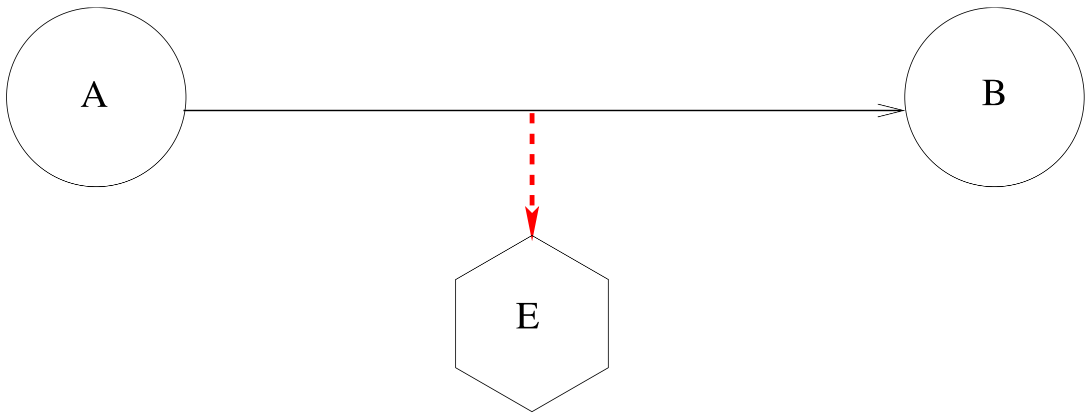
Had Been Open for Years: Assuming NP $\neq$ P is there some ϵ, $0 < \epsilon < 1$, s.t. MAX-3-SAT has no PTIME ϵ-approximation algorithm?

Theorem 17.4 (PCP Theorem [ALMSS) NP $=$ PCP$[\log n, 1]$

Corollary 17.5 *If* P $\neq$ NP, *Then* $\exists\epsilon\,.\ 0 < \epsilon < 1$, MAX-3-SAT *has no ptime,* ϵ*-approximation algorithm.*

Theorem 17.6 ([Hastad]) *In the PCP theorem, looking at 3 bits of the proof are necessary and sufficient. Thus, the best possible PTIME approximation ration for* MAX-3-SAT *is* $\frac{1}{8}$ *(and this is acheivable).*

Cryptography



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

$$\begin{aligned} E(p,x) &= p \oplus x \\ D(p,x) &= p \oplus x \\ D(p,E(p,m)) &= p \oplus (p \oplus m) = m \end{aligned}$$

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

Thm: If p is chosen at random and known only to A and B Then $E(p,m)$ provides no information to E about m except perhaps its length.

Better not use p more than once!

Public-Key Cryptography

Idea: [Diffie, Hellman, 1976] Using computational complexity, I may be able to publish a key for sending secret messages to me, that are intractable to decode. Example: Diffie-Hellman key exchange.

Realization: [Rivest, Shamir, Adleman, 1976] This is the Public-Key Algorithm that is used today in the SSL algorithm that lets your browser generate a key to send an order to Amazon.com without, **we believe**, divulging any **useful** information about your credit card number, or what you bought.

RSA

B **chooses** p, q n-bit primes, $e,$ s.t. $\text{GCD}(e, \varphi(pq)) = 1$;

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

Using Euclid's algorithm, B **computes** d, k, s.t. $ed + k\varphi(pq) = 1$

[Break message into pieces shorter than $2n$ bits]

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

For **sufficiently large** n, [$n \geq 300$ bits is fine in 2005],

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 5/17/05."

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

Interactive Proofs

[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.	**Arthur** has x	**Merlin** has x
1.	flip σ_1, compute m_1 $\longrightarrow$	
2.		$\longleftarrow m_2$
3.	flip σ_3, compute m_3 $\longrightarrow$	
4.		$\longleftarrow m_4$
$\vdots$	$\vdots$	$\vdots$
$2t$.		$\longleftarrow m_{2t}$
$2t+1$.	flip σ_{2t+1}, accept or reject	

Definition 17.7 $D \in$ AM iff there is a PTIME interactive protocol

1. If $x \in D$, then there exists a strategy for **Merlin**

$$\text{Prob}\{\textbf{Arthur} \text{ accepts } x\} \;=\; 1$$

2. If $x \notin D$, then for all strategies for **Merlin**

$$\text{Prob}\{\textbf{Arthur} \text{ accepts } x\} \;<\; \frac{1}{2}$$

$\square$

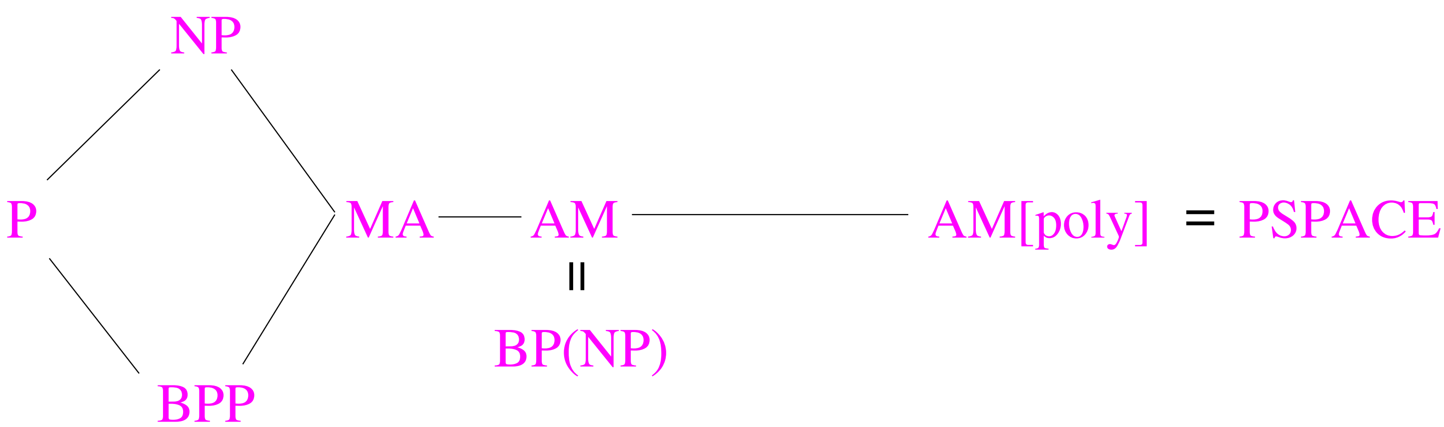
Observation 17.8 *As for* BPP*, by iterating we can make probability of error exponentially small.*

Definition 17.9 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. For $k \geq 2$,

$$\mathrm{AM}[k] \quad = \quad \underbrace{\textbf{Arthur}\textbf{Merlin}\textbf{Arthur}\cdots}_{k} \qquad \square$$

Fact 17.10 *[Babai] For all $k \geq 2$,* $\quad \mathrm{AM}[k] \quad = \quad \mathrm{AM}.$



Fact 17.11 [Goldwasser,Sipser] *The power of interactive proofs is unchanged if* $\mathbf{M}$ *knowns* $\mathbf{A}$*'s coin tosses. For all k,*

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

(Originally, **Arthur**'s coin tosses were public, but Verifier's were secret. Now it's whichever is more convenient.)

Graph Isomorphism $\in$ NP; Is it in co-NP? Consider the following AM game, where **Arthur** keeps his coin tosses secret:

$$\text{Input} = G_0, G_1, \quad n = \|G_0\| = \|G_1\|$$

0.	**Arthur** has G_0, G_1	**Merlin** has G_0, G_1
1.	flip $\kappa : \{1, \ldots, r\} \to \{0,1\}$ flip $\pi_1, \ldots, \pi_r \in S_n$ $\pi_1(G_{\kappa(1)}), \ldots, \pi_r(G_{\kappa(r)}) \longrightarrow$	
2.		$\longleftarrow m_2 \in \{0,1\}^r$
3.	accept iff $\kappa = m_2$	

The above game shows:

Proposition 17.12 **Prop:** *Graph Isomorphism* $\in$ *co*-AM

Proof: If $G_0 \ncong G_1$, then **Arthur** will accept with probability 1.

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

Corollary 17.13 *If Graph Isomorphism is* NP-*complete then* PH *collapses.*