Penn State University
CSE 564 Complexity of Combinatorial Problems

Instructor: Pei Wu

Module
3

Interactive proofs and Merlin-Arthur games

In the previous lectures we studied the polynomial hierarchy (PH) as a generalization of NP obtained by allowing alternating quantifiers. This lecture introduces a different axis along which we can enrich NP: instead of a single static proof string, we allow an all-powerful prover to *interact* with a probabilistic polynomial-time verifier. The resulting model—*interactive proofs*—captures many natural "proofs of knowledge" that do not fit the NP paradigm and, at the same time, enables probabilistic verification of statements that are not known to have short certificates.

After defining the class IP, we prove the basic containment $\mathsf{IP} \subseteq \mathsf{PSPACE}$ by viewing an interactive protocol as a game tree in which the verifier averages over its random coins and the prover plays a best response. We then demonstrate that interaction genuinely adds power by giving the classic two-message interactive protocol for *graph non-isomorphism*.

The second half of the lecture focuses on *public-coin* interaction. We define the classes MA and AM, show that $\mathsf{MA} \subseteq \mathsf{AM}$, and sketch why any constant number of public-coin rounds collapses to one. We also discuss perfect completeness for MA. Finally, we present the Goldwasser–Sipser approach to proving $\mathsf{GNI} \in \mathsf{AM}$ via hashing/approximate counting, and we sketch the Goldwasser–Sipser theorem that private coins can be made public at the cost of two additional rounds: $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$.

Note: This is only a list of contents mentioned in the class. All proofs are sketching high-level ideas. Details can be found in AB's book, hopefully. (Ch. 8)

3.1 Interactive proof systems

Recall the familiar NP "proof" picture: a prover P sends a witness π and a deterministic polynomial-time verifier V checks it. Interactive proofs generalize this to allow multiple messages and randomness.

Definition 3.1 (interactive protocol). Fix an input length n. An *r-round interactive protocol* (with $r = \mathrm{poly}(n)$) consists of:

- A prover strategy P given by functions $\pi_1, \ldots, \pi_r$, where each π_i outputs the prover's i-th message as a function of the input $x \in \{0,1\}^n$ and the transcript so far. Concretely,

if verifier messages are $\sigma_1, \ldots, \sigma_r$ and prover messages are $\pi_1, \ldots, \pi_r$, then

$$\pi_i : \{0,1\}^{q_1} \times \cdots \times \{0,1\}^{q_i} \times \{0,1\}^{p_1} \times \cdots \times \{0,1\}^{p_{i-1}} \times \{0,1\}^n \to \{0,1\}^{p_i}.$$

- A probabilistic polynomial-time verifier V, which we can view as a sequence of randomized machines $V_1, \ldots, V_r$. The verifier's i-th message σ_i is sampled as

$$\sigma_i \leftarrow V_i\big(x, \pi_1, \ldots, \pi_{i-1}; \sigma_1, \ldots, \sigma_{i-1}\big).$$

  After r rounds, the verifier outputs *accept* or *reject*.

Definition 3.2 (IP). A language $L \subseteq \{0,1\}^*$ is in IP if there exists a polynomial-round interactive protocol (P, V) such that

- **Completeness:** if $x \in L$, then $\Pr[\langle P, V\rangle(x) \text{ accepts}] \geq 2/3$.
- **Soundness:** if $x \notin L$, then for every (possibly cheating) prover P^*, $\Pr[\langle P^*, V\rangle(x) \text{ accepts}] \leq 1/3$.

The probability is over the verifier's random coins.

We also write $\mathsf{IP}[k]$ for protocols with k total messages ("k-turn") or, equivalently, a constant number of rounds; the exact convention is not important for this lecture.

Remark 3.3 (deterministic interaction). If the verifier is deterministic (no random coins), interaction does not buy anything beyond NP. Intuitively, a deterministic verifier's future messages are completely determined by the transcript, so a prover can simply send an accepting transcript and the verifier can check it.

Theorem 3.4. *$\mathsf{IP} \subseteq \mathsf{PSPACE}$.*

Proof. Fix an IP protocol (P, V) for a language L with r rounds. Consider the (exponentially large) game tree of all possible transcripts. At a verifier node we branch over the verifier's random choices and take an *average*; at a prover node we branch over possible prover messages and take the *maximum* (the prover's best response).

We can compute the optimal acceptance probability by recursion. Let $\ell \in \{1, 2, \ldots, 2r+1\}$ be the position in the transcript. Define Rec-IP(ℓ, transcript prefix) to return the acceptance probability from that point onward assuming optimal play by the prover. The recursion is:

- *Base case:* if $\ell = 2r+1$ (past the last prover message), return the verifier's acceptance probability given the full transcript.
- *Prover move:* if $\ell = 2k$ for some k, maximize over all possible prover messages π_k:

$$v \leftarrow \max_{\pi_k} \textsc{Rec-IP}(\ell + 1, \text{prefix}, \pi_k).$$

- *Verifier move:* if $\ell = 2k - 1$ for some k, average over the verifier's randomness in generating σ_k:

$$v \leftarrow \mathbb{E}_{\sigma_k \leftarrow V_k(\cdot)}\big[\textsc{Rec-IP}(\ell + 1, \text{prefix}, \sigma_k)\big].$$

Finally, accept iff $\textsc{Rec-IP}(1, \varepsilon) \geq 2/3$.

The recursion depth is $O(r) = \text{poly}(n)$. We can evaluate it in polynomial space by a depth-first traversal: we only need to store the current recursion stack and the current transcript prefix (both polynomial length), reusing space across branches. Thus membership in L can be decided in PSPACE. □

3.2 Power of interactive proofs

The prototypical example showing that interaction helps is *graph non-isomorphism*.

Definition 3.5 (GI and GNI). GI (graph isomorphism) is $\{(G_0, G_1) : G_0 \cong G_1\}$. GNI (graph non-isomorphism) is $\{(G_0, G_1) : G_0 \ncong G_1\}$.

It is immediate that $\mathsf{GI} \in \mathsf{NP}$: a witness can be a relabeling of vertices (a permutation) exhibiting the isomorphism. In contrast, it is not clear how to give short certificates of non-isomorphism.

Theorem 3.6. *$\mathsf{GNI} \in \mathsf{IP}$.*

Proof. Given (G_0, G_1), the verifier performs:

1. Choose a random bit $i \in \{0, 1\}$.
2. Choose a uniformly random permutation ρ of the vertices. Let $\widetilde{G}$ be the graph obtained from G_i by relabeling with ρ.
3. Send $\widetilde{G}$ to the prover.
4. The prover replies with a bit $\hat{\imath} \in \{0, 1\}$.
5. The verifier accepts iff $\hat{\imath} = i$.

Completeness. If $G_0 \ncong G_1$, then $\widetilde{G}$ is isomorphic to exactly one of G_0, G_1. An unbounded prover can test isomorphism and determine i with certainty.
Soundness. If $G_0 \cong G_1$, then $\widetilde{G}$ is distributed identically whether $i = 0$ or $i = 1$. Hence no prover can guess i with probability better than $1/2$.

Repeating the protocol independently and taking the majority outcome reduces error from $1/2$ to $2^{-\Omega(t)}$ after t repetitions. □

3.3 Public-coin interaction: MA and AM

Interactive proofs allow the verifier to keep its random coins private. Two important restricted models are MA and AM, which are one-round interactions.

Definition 3.7 (MA). A language L is in MA if there exists a probabilistic polynomial-time verifier V such that:

- (Completeness) If $x \in L$, then there exists a proof π with $\Pr_r[V(x, \pi; r) = 1] \geq 2/3$.
- (Soundness) If $x \notin L$, then for all proofs π, $\Pr_r[V(x, \pi; r) = 1] \leq 1/3$.

Here r denotes the verifier's random string.

DEFINITION 3.8 (AM). A language L is in AM if there exists a deterministic polynomial-time verifier V such that:

- (Completeness) If $x \in L$, then $\Pr_r[\exists \pi\ V(x, \pi; r) = 1] \geq 2/3$.
- (Soundness) If $x \notin L$, then $\Pr_r[\exists \pi\ V(x, \pi; r) = 1] \leq 1/3$.

The key point is that the random string r is an explicit message from Arthur to Merlin (public coin).

More generally, AM[k] denotes a k-turn public-coin protocol with an alternating pattern such as AMA, AMAM, etc.

PROPOSITION 3.9. *MA $\subseteq$ AM.*

Proof. Let $L \in$ MA with verifier V. By standard amplification, for any parameter m we can modify V (by independent repetition and majority) so that

$$x \in L \implies \exists \pi\ \Pr_r[V(x, \pi; r) = 1] \geq 1 - 2^{-m},$$
$$x \notin L \implies \forall \pi\ \Pr_r[V(x, \pi; r) = 1] \leq 2^{-m}.$$

Consider the AM protocol where Arthur samples r uniformly, sends it to Merlin, Merlin replies with a proof π, and Arthur accepts iff $V(x, \pi; r) = 1$.
Completeness. If $x \in L$, then for the good π we have $\Pr_r[\exists \pi\ V(x, \pi; r) = 1] \geq 1 - 2^{-m}$.
Soundness. If $x \notin L$, then for every fixed π, $\Pr_r[V(x, \pi; r) = 1] \leq 2^{-m}$. Let Π be the set of all possible proofs of length $|\pi|$; then

$$\Pr_r[\exists \pi\ V(x, \pi; r) = 1] \leq \sum_{\pi \in \Pi} \Pr_r[V(x, \pi; r) = 1] \leq 2^{|\pi|} \cdot 2^{-m} = 2^{|\pi| - m}.$$

Choosing $m \geq |\pi| + 2$ makes this at most $1/4$, and a constant-factor gap can be amplified to the usual $1/3$ by repetition. □

COROLLARY 3.10. *For every constant k, AM[k] = AM.*

Proof sketch. We only look at the basic "round collapsing" step AMA $\subseteq$ AAM, and then observe that AAM = AM because Arthur can send both random strings at once.

At a high level, one amplifies the protocol so that completeness is very close to 1 and soundness is very small. Then, by averaging over Arthur's first random string r_1, most choices of r_1 are "typical" in the sense that the remaining interaction behaves well as an MA-type statement over the later random string(s). This lets one swap the order of an Arthur message with the adjacent Merlin message, losing only a constant in completeness/soundness. Iterating a constant number of times yields the collapse.

The same argument does *not* immediately imply AM[poly] $\subseteq$ AM: with polynomially many rounds, the constant losses in parameters would accumulate, the running time become exponential. □

3.4 Perfect completeness in MA

The completeness parameter in MA can be boosted all the way to 1.

PROPOSITION 3.11 (perfect completeness for MA). *$MA_{2/3,1/3} = MA_{1,1/3}$; that is, MA has perfect completeness.*

Proof idea. Amplify an MA protocol so that on yes-instances $x \in L$ there exists a proof π accepted with probability $1 - 2^{-m}$ over the verifier's coins. Let $A \subseteq \{0,1\}^m$ be the set of accepting random strings. Then $|A| \geq (1 - 2^{-m})2^m$, so the complement is tiny.

Using the Sipser–Gács/Lautemann "shifting" idea for BPP, one can show that there exists a polynomial-size list of shifts $S_1, \ldots, S_t \in \{0,1\}^m$ such that

$$\forall r \in \{0,1\}^m \; \exists j \in [t] \text{ with } r \oplus S_j \in A.$$

Merlin's proof now consists of π together with the list of shifts. Arthur picks a uniform r and accepts iff $V(x, \pi; r \oplus S_j) = 1$ for some j. On yes-instances this accepts for every r, giving perfect completeness. On no-instances, amplification ensures the accepting set is too small for any polynomial list of shifts to cover all r, yielding soundness bounded away from 1. □

REMARK 3.12 (no perfect soundness). In contrast, MA cannot have perfect soundness unless it collapses to NP. If a protocol had soundness 0, then on no-instances no random string r could make the verifier accept. On a yes-instance, Merlin could then simply provide a pair (π, r) that makes the verifier accept, and a deterministic verifier could check it. In particular, $\mathsf{MA}_{2/3,0} \subseteq \mathsf{NP}$.

3.5 GNI in AM via hashing (Goldwasser–Sipser)

We already saw that GNI has a simple IP protocol. Surprisingly, GNI also admits a public-coin protocol. The key tool is a way to distinguish whether a set is "large" or "small" using only public randomness and a prover who supplies witnesses.

Step 1: reduce GNI to a set-size gap. Given graphs (G_0, G_1) on n vertices, define

$$Q \;:=\; \{\widetilde{G} : \widetilde{G} \cong G_0 \text{ or } \widetilde{G} \cong G_1\}.$$

If $G_0 \ncong G_1$, then $|Q| = 2 \cdot n!$ (every relabeling of each graph). If $G_0 \cong G_1$, then $|Q| = n!$. Thus GNI reduces to deciding whether $|Q|$ is "large" or "small" given a promise gap by a factor of 2.

Step 2: a generic "large vs. small" protocol. Let U be a universe (think $U = \{0,1\}^m$) and let $S \subseteq U$. Assume a promise that either $|S| \geq s$ (*large*) or $|S| \leq s/2$ (*small*).

If we could choose a truly random function $f : U \to [s]$, then picking a random bin $i \in [s]$ would give

$$\Pr[\exists x \in S : f(x) = i] \leq |S|/s \leq 1/2 \quad \text{in the small case,}$$

while in the large case the probability is bounded away from $1/2$. However, a truly random f has exponentially large description.

The fix is to use a small, efficiently samplable family of *pairwise independent* hash functions.

DEFINITION 3.13 (pairwise independent hash family). A family $\mathcal{H}$ of functions $h : U \to [s]$ is *pairwise independent* if:

1. For every $x \in U$ and $i \in [s]$, $\Pr_{h \leftarrow \mathcal{H}}[h(x) = i] = 1/s$.
2. For every $x \neq y \in U$ and every $i, j \in [s]$, $\Pr_{h \leftarrow \mathcal{H}}[h(x) = i \wedge h(y) = j] = 1/s^2$.

FACT 3.14 (explicit constructions). *There are explicit pairwise independent families with succinct description. One standard construction is: interpret U as a finite field $\mathbb{F}$ of size $|\mathbb{F}| \approx 2^m$, choose $a, b \in \mathbb{F}$ uniformly, and set $h_{a,b}(x) = ax + b$.*

The AM protocol. The notes use $2s$ bins. Arthur samples a random $h \in \mathcal{H}$ mapping to $[2s]$ and a random $i \in [2s]$, and sends (h, i) to Merlin. Merlin must reply with an element $x \in S$ such that $h(x) = i$. Arthur checks membership $x \in S$ (in our application, that $x \in Q$) and the hash condition.
Soundness. If $|S| \leq s/2$, then $\Pr_{h,i}[\exists x \in S : h(x) = i] \leq |S|/(2s) \leq 1/4$.
Completeness. If $|S| \geq s$, then by inclusion–exclusion and pairwise independence,

$$\begin{aligned}\Pr_{h,i}[\exists x \in S : h(x) = i] &\geq \sum_{x \in S} \Pr[h(x) = i] \quad - \sum_{\substack{x,y \in S \\ x \neq y}} \Pr[h(x) = i \wedge h(y) = i] \\ &= |S| \cdot \frac{1}{2s} - \binom{|S|}{2} \cdot \frac{1}{(2s)^2}.\end{aligned}$$

Taking $|S| = s$ gives $\geq 1/2 - 1/8 = 3/8 > 1/4$. Repeating in parallel amplifies the gap.

Apply to GNI. We apply this set-size test to $S = Q$ with threshold $s = n!$ (up to constant factors), distinguishing $|Q| = n!$ from $|Q| = 2n!$. This yields $\mathsf{GNI} \in \mathsf{AM}$.

3.6 From private coins to public coins: $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$

The Goldwasser–Sipser theorem shows that private coins can be made public without losing power, at the cost of two extra messages.

THEOREM 3.15 (Goldwasser–Sipser, special case). *Every two-message (private-coin) interactive proof can be simulated by a four-message public-coin protocol: $\mathsf{IP}[2] \subseteq \mathsf{AM}[4]$. More generally, $\mathsf{IP}[k] \subseteq \mathsf{AM}[k+2]$.*

Proof sketch for $\mathsf{IP}[2] \subseteq \mathsf{AM}[4]$. Let $L \in \mathsf{IP}[2]$ with verifier V. On input x, the private-coin verifier samples random coins r, computes a challenge $q = q(x, r)$, sends q to the prover, receives a reply π, and then accepts according to a predicate $V(x, \pi, r)$.

For each challenge q, let $R(q) = \{r : q(x, r) = q\}$ be the set of random strings consistent with q. For a candidate prover message π, define the conditional acceptance rate

$$a(q, \pi) := \Pr_{r \sim R(q)}[V(x, \pi, r) = 1].$$

The protocol's completeness and soundness can be expressed (up to standard constant-gap amplification) as

$$\mathbb{E}_q\big[\max_{\pi}\ a(q,\pi)\big]\ \begin{cases}\geq 1-\varepsilon & (x\in L),\\ \leq \varepsilon & (x\notin L).\end{cases}$$

By averaging, in the yes case most q's admit some π with large $a(q,\pi)$ (such as $4/5$), whereas in the no case most q's have $a(q,\pi)$ small for every π (e.g. at most $1/5$).

The public-coin protocol has four messages:

$$\begin{array}{ccc} \mathrm{M} & \xleftarrow{\ r\ } & \mathrm{Arthur}\\ & \xrightarrow{\ (q,\pi)\ } & \\ & \xleftarrow{\ (h,i)\ } & \\ & \xrightarrow{\ r'\ } & \end{array}$$

where:

- Arthur sends uniform r. Merlin must respond with $q = q(x,r)$ (which Arthur can recompute and check) together with a proposed prover message π. This makes Arthur's message uniformly random, as required for public coins.
- Arthur now wants to verify that $a(q,\pi)$ is large. Let $\mathrm{Good}(q,\pi) = \{r' \in R(q) : V(x,\pi,r') = 1\}$. Using the same hashing idea as in the GNI–in–AM protocol, Arthur samples a pairwise independent hash h and a random bucket index i, and asks Merlin to produce an $r' \in \mathrm{Good}(q,\pi)$ with $h(r') = i$. Arthur checks that $q(x,r') = q$, that $h(r') = i$, and that $V(x,\pi,r') = 1$. If $a(q,\pi)$ is large then $\mathrm{Good}(q,\pi)$ is a large fraction of $R(q)$, so Merlin succeeds with probability bounded away from $1/4$; if $a(q,\pi)$ is small then Merlin succeeds with probability at most $\approx 1/4$. Repeating amplifies.

When the sets $|R(q)|$ are uniform, we know the size of $|R(q)|$. Then what does it mean to be dense, or sparse is explicit, say, maybe we are distinguishing sets of size $\geq 4|R(q)|/5$ of $\leq |R(q)|/5$. In general, we don't know the size of $R(q)$. Then we can't set up the range of our hashing function. In general, Goldwasser-Sipser uses a *bucketing* argument to partition the randomness space so that the relevant conditional distributions behave nearly uniformly, and then applies the above hashing test within each bucket. □