Lecture B.1

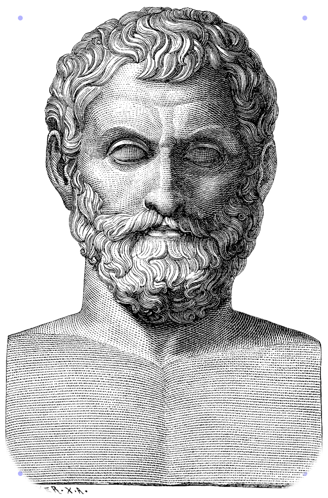
Intro to PCPs

(Probabilistically Checkable Proofs)

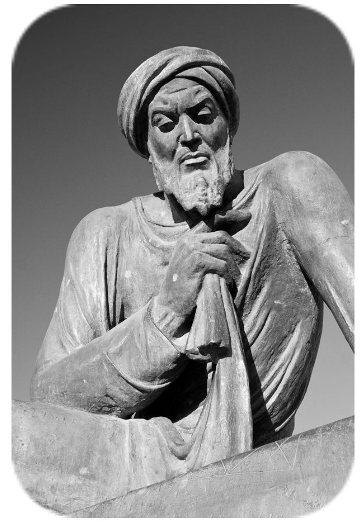
Tom Gur

MSRI Summer Graduate School on
Foundations and Frontiers of Probabilistic Proofs
July 26, 2021

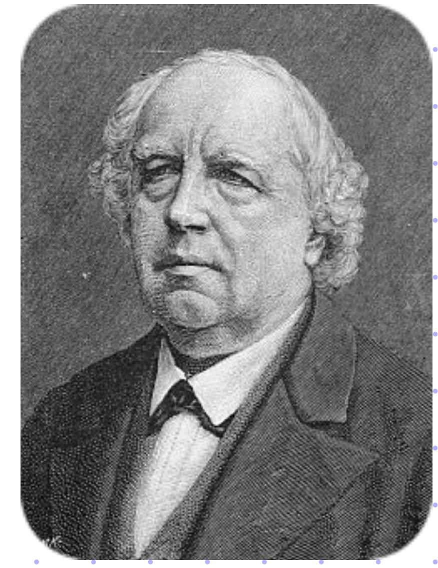
Evolution of mathematical proofs



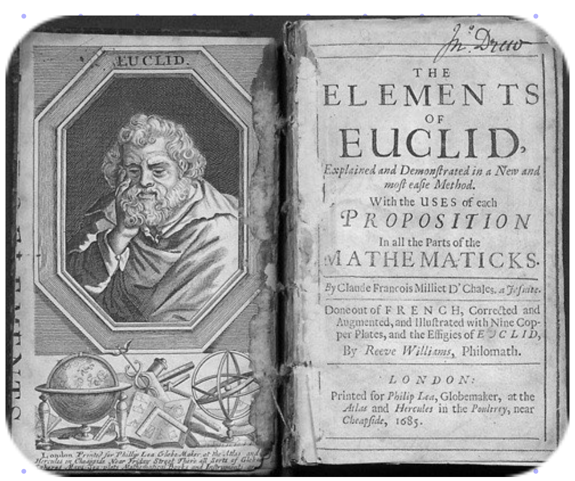
Thales of Miletus



al-Khwarizmi



Weierstrass



Euclid

Madhava of Sangamagrama

Hilbert

Checking a proof without reading it?


Science Times

The New York Times

Biologists Tell a Tale Of Interfering In-Laws

Asteroid Defense: 'Risk Is Real,' Planners Say

New Short Cut Found For Long Math Proofs

A proof can be tested by checking just a part, inventors say.

Proofs have many interpretations:

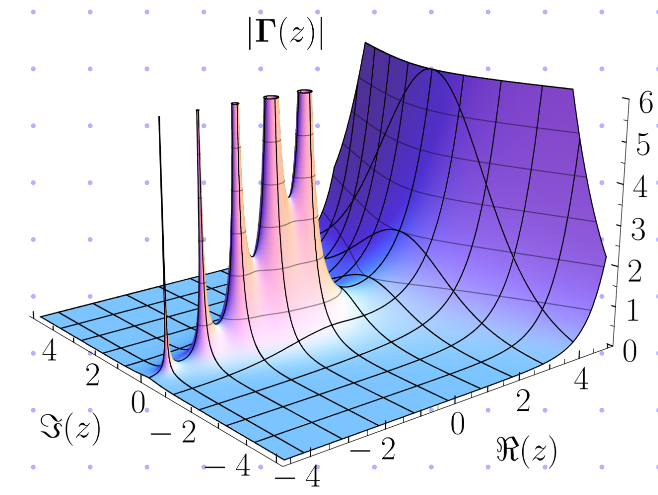
- "logical derivations from axioms" - Zermelo
- "approximation of understanding" - Dinur
- "Whatever that convinces me" - Even

Key idea: check proofs probablistically allowing negligable error

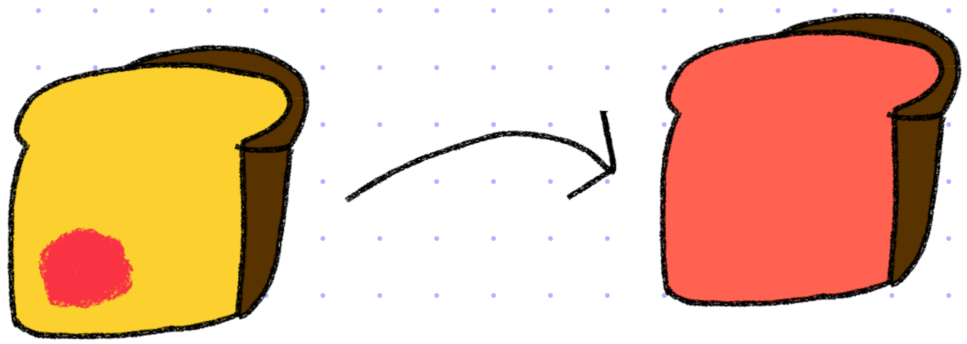
Caveat: What if only one line of the proof is wrong?

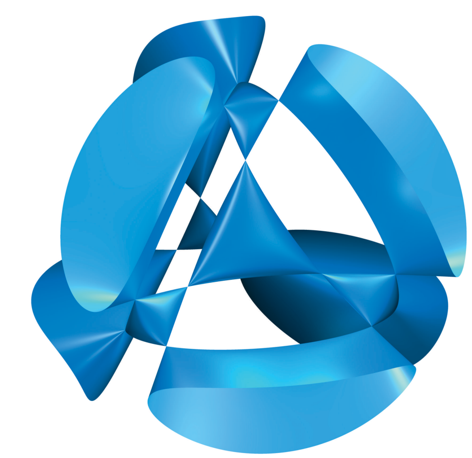
Local-to-global phenomena

Idea: endow proof with a rich structure that allows checking global properties via local constraints!



aka, the "Jam principle"



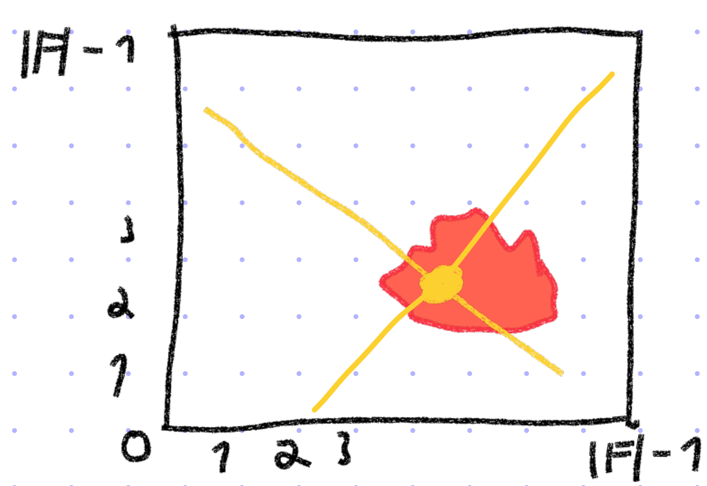


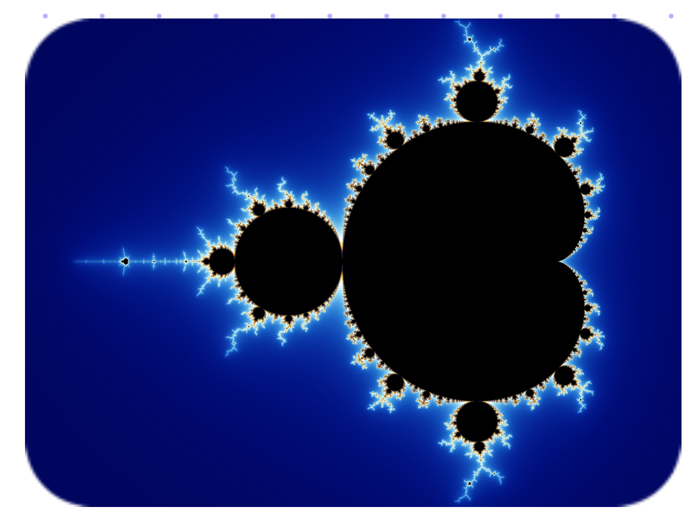
Informal example: low-degree polynomials

$p \in \mathbb{F}[X,Y]$, $\deg(p) \le d$

$p_\ell(t) := p(\bar{x} + t\bar{y}) \in \mathbb{F}[X]$

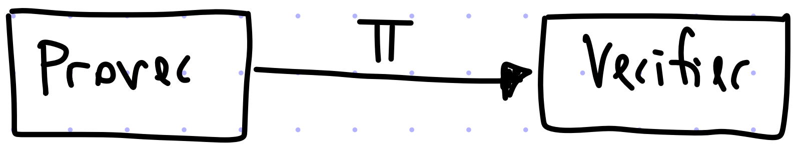
$\deg(p_\ell) \le d$



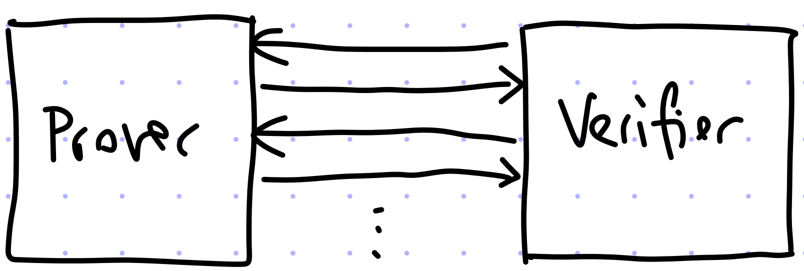


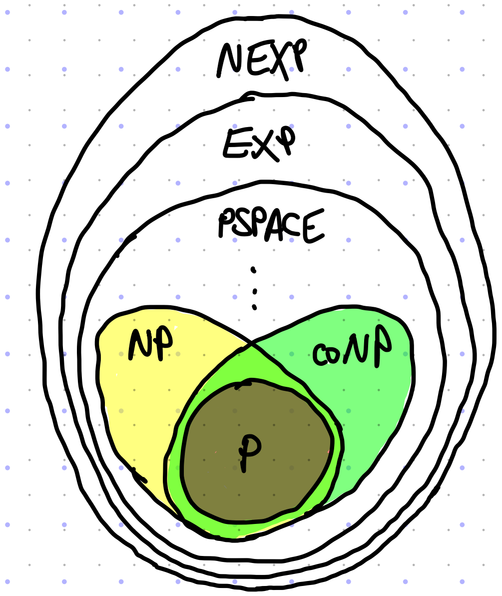
A New Model: Probabilistically Checkable Proofs

- NP represents proofs having a deterministic polynomial-time verifier



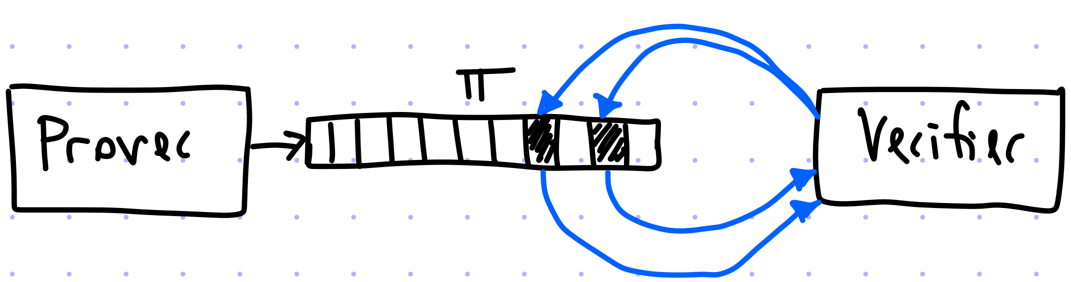
- IP represents proofs where the polynomial-time verifier has two new resources:
  ① randomness , and ② interaction





Today we study a new model:

- PCP represents proofs where the polynomial-time verifier has two new resources:
  ① randomness , and ② oracle access to proof



Definition of PCP

Let P be an all-powerful prover and V a ppt oracle algorithm.
We say that (P,V) is a PCP system for a language L with completeness error ε_c and soundness error ε_s if the following holds:

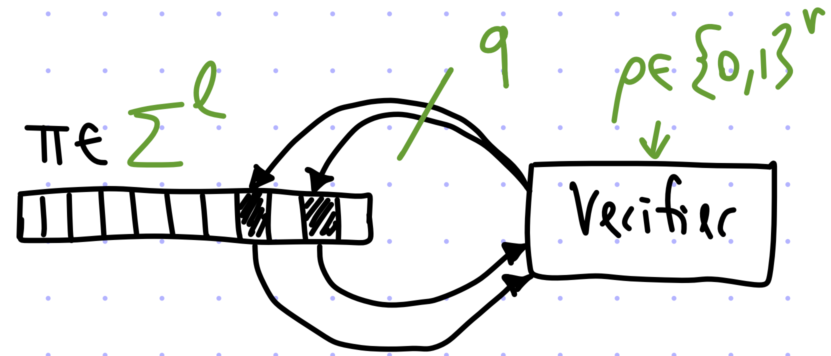
① completeness: $\forall x \in L$, for $\pi := P(x)$, $\Pr_{\rho}[V^{\pi}(x;\rho)=1] \geq 1-\varepsilon_c$

② soundness: $\forall x \notin L\ \forall \tilde{\pi}\ \Pr_{\rho}[V^{\tilde{\pi}}(x;\rho)=1] \leq \varepsilon_s$

We call π a "PCP", and can view it as a "robust encoding" of a witness, which admits verification without reading all its symbols.

For IPs we cared about: round complexity, communication complexity, ...
For PCPs we have a somewhat different set of parameters:

- Σ : proof alphabet
- ℓ : proof length
- q : verifier query complexity
- r : verifier randomness complexity



[typically queries to π will be non-adaptive]

Some Special Cases

We wish to understand $PCP[\varepsilon_c, \varepsilon_s, \Sigma, \ell, q, r, \ldots]$ in different regimes.
Let's start with some special cases to warm up.

Suppose there is no proof $(q=0)$:

- $PCP[q=0, r=0] = P$ ← if there is no proof and no randomness then the verifier is just a polytime algorithm
- $PCP[q=0, r=O(\log n)] = P$ ← logarithmically-many random bits don't help
- $PCP[q=0, r=poly(n)] = BPP$ ← if there is randomness but no proof then the verifier is just a ppt algorithm

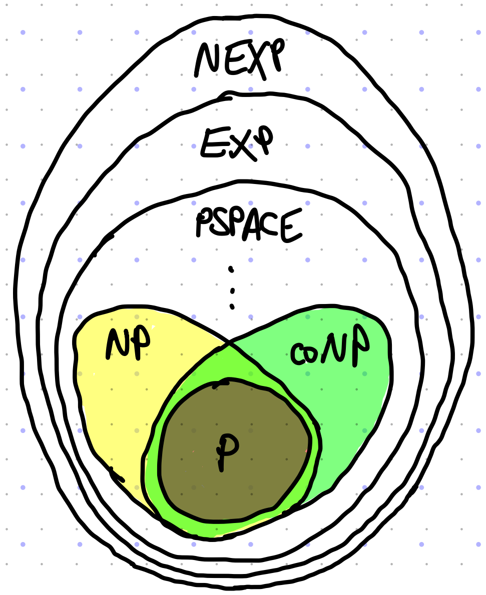
Suppose there is no randomness $(r=0)$:

- $PCP[q=poly(n), r=0] = NP$ ← verifier can read in full a poly-size witness

We denote by PCP the complexity class with no restrictions beyond "V is ppt".
This means that $q=poly(n)$, $r=poly(n)$ and allows for $\ell = exp(n)$, $|\Sigma| = exp(n)$.

Questions

- Which languages have PCPs (beyond NP & BPP)? — more than PSPACE
- Do PCPs have benefits for NP languages? — yes
  (E.g. query complexity sublinear in witness size)
- Do PCPs have benefits for tractable languages? — yes
  (E.g. PCP verification faster than execution)
- Are there ZK PCPs for NP languages? — yes

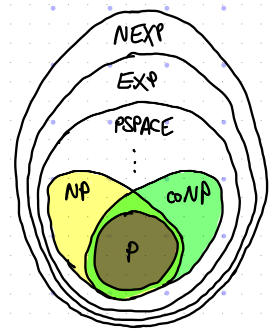


Many good news!

But the PCP model is weird (PCP verifier has oracle access to a large proof).
How are PCPs useful?

① lead to interactive arguments (& other crypto proofs) with strong efficiency features
② lead to hardness & approximation results

Upper Bound on PCPs



- Σ : proof alphabet
- ℓ : proof length
- q : verifier query complexity
- r : verifier randomness complexity

theorem: $PCP \subseteq NEXP$

lemma: (i) $\ell \leq 2^r q$ for non-adaptive verifiers
(ii) $\ell \leq 2^r |\Sigma|^q q$ for adaptive verifiers

[in constructions ℓ is usually smaller than these upper bounds]

proof of (i): there are at most 2^r different query sets
proof of (ii): each answer from the proof can lead to a different next query

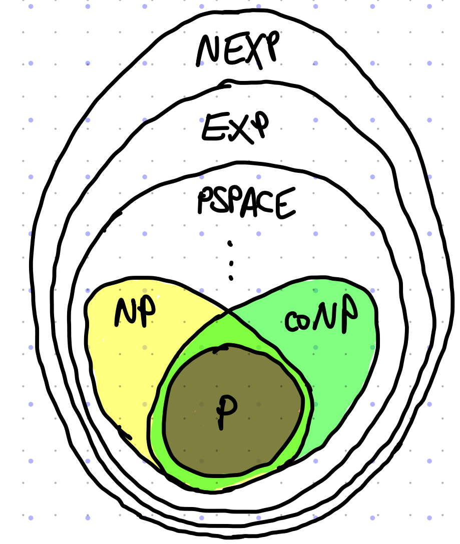
lemma: $PCP[\ell, r] \subseteq NTIME\left((2^r+\ell)\cdot poly(n)\right)$.

proof: Suppose (P,V) is a PCP system for L where the PCP verifier uses r random bits to query a proof of length ℓ. Consider this decider:

$D(x, \pi) :=$ For every $\rho \in \{0,1\}^r$ compute $b_\rho := V^\pi(x;\rho)$ and output 1 iff $\sum_\rho b_\rho / 2^r \geq 1-\varepsilon_c$.
($x \in \{0,1\}^n$, $\pi \in \{0,1\}^\ell$)

If $x \in L$ then $\exists \pi$ s.t. $D(x,\pi)=1$. If $x \notin L$ then $\forall \pi$ $D(x,\pi)=0$. ∎

A Simple Inclusion: PSPACE



theorem: PSPACE $\subseteq$ PCP

lemma: IP $\subseteq$ PCP

proof: Suppose that (P,V) is a public-coin IP for L. (Public coin comes WLOG.)
Consider proofs in this format: $\Pi = \{a_{r_1}\}_{r_1} \cup \{a_{r_1,r_2}\}_{r_1,r_2} \cup \cdots \cup \{a_{r_1,\dots,r_k}\}_{r_1,\dots,r_k}$.
The PCP verifier samples $r_1,\dots,r_k$ and accepts if the IP verifier accepts:

$$V(x, a_{r_1}, a_{r_1,r_2}, \dots, a_{r_1,\dots,r_k}; r_1 \dots r_k) \stackrel{?}{=} 1.$$

Completeness: consider the honest proof $\Pi := \{P(x,r_1)\}_{r_1} \cup \{P(x,r_1,r_2)\}_{r_1,r_2} \cup \cdots \cup \{P(x,r_1,\dots,r_k)\}_{r_1,\dots,r_k}$.
Soundness: any proof in the above format corresponds to an "unrolled" IP prover ■

In sum: PSPACE $\subseteq$ PCP $\subseteq$ NEXP. We will see that PCP = NEXP by recycling techniques (arithmetization, sumcheck) and using new ones (low-degree testing).
We will also see how to "scale down" to get PCPs for NP.

Course outline

Local-to-global phenomena

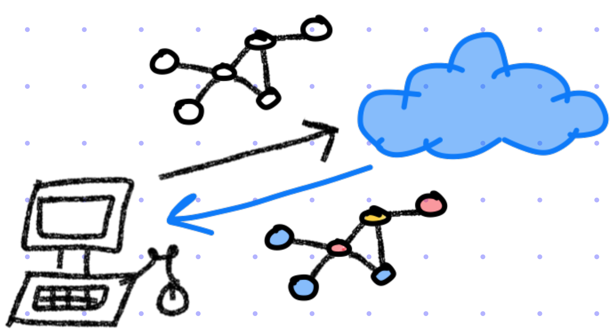
- Linearity testing
- Low-degree testing
- FFT-based testing of univariate polynomials

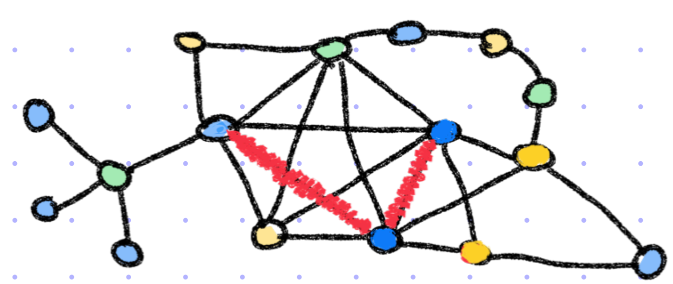
PCP constuctions

- exp-size, $O(1)$-local PCPs
- poly-size, polylog-local PCPs
- PCP composition
- Sublinear-time verification

Applications

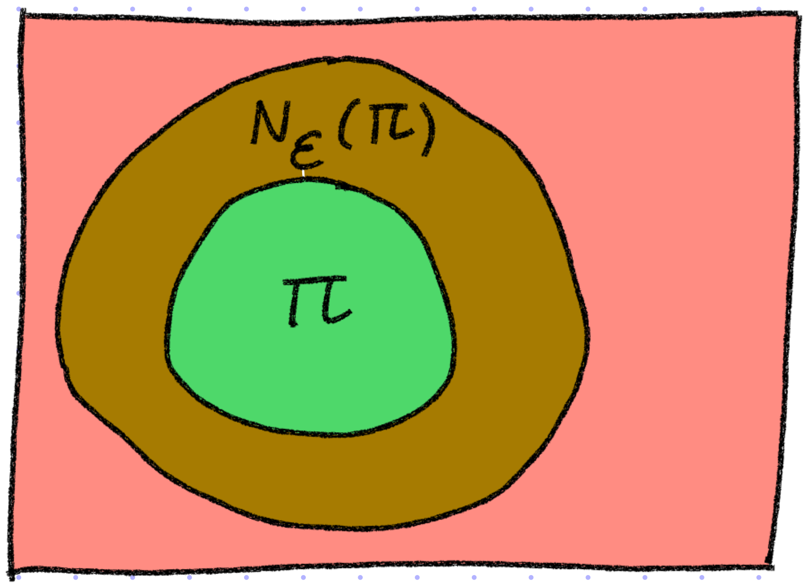
- Delegation of computation
- Hardness of approximation





Conceptual perspectives

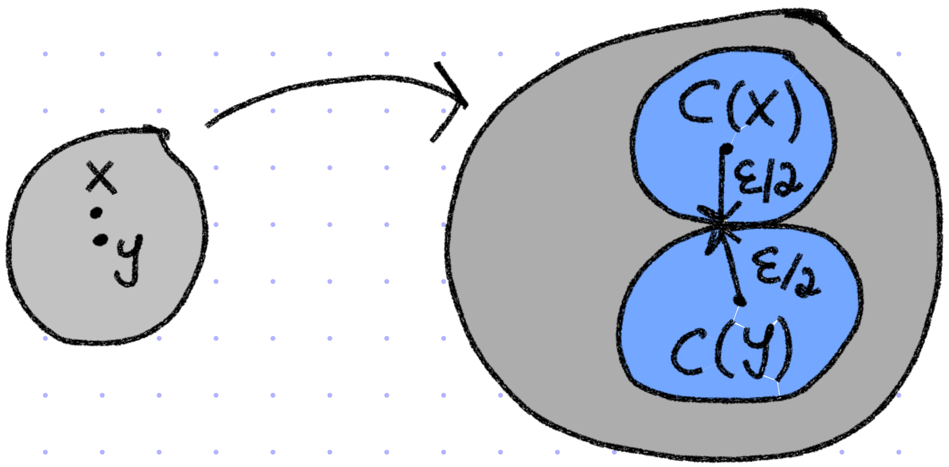
Property testing



Is $f \in \Pi$ (e.g. $\Pi = \mathbb{F}^{\leq d}[X]$)

or $\delta(f, \Pi) > \varepsilon$?

Coding theory



For every $x \neq y$, we have

$\delta(C(x), C(y)) > \varepsilon$

e.g., univariate polynomials

low-degree polynomials

Linear func. on hypercube