Ideals, Gröbner Bases, and PCPs

Sophus Valentin Willumsgaard

Prashanth Amireddy, Amik Raj Behera, Srikanth Srinivasan,
Madhu Sudan

July 15, 2026

Outline

Informal Introduction

In a PCP protocol, you have a prover and a verifier.

Informal Introduction

In a PCP protocol, you have a prover and a verifier.

certificate $\longrightarrow$

- The prover Loki sends a certificate that a decision problem is true (eg. a graph is 3-colourable).

Informal Introduction

In a PCP protocol, you have a prover and a verifier.

certificate →

← (accept, reject)

- The prover Loki sends a certificate that a decision problem is true (eg. a graph is 3-colourable).
- The verifier looks at a few randomly chosen bits of the certificate, and guess whether or not the certificate is valid.

Informal Introduction

In a PCP protocol, you have a prover and a verifier.

certificate →

← (accept, reject)

- The prover Loki sends a certificate that a decision problem is true (eg. a graph is 3-colourable).
- The verifier looks at a few randomly chosen bits of the certificate, and guess whether or not the certificate is valid.
- The prover will try and trick the verifier!

Introduction to PCP

PCP (Probabilistically Checkable Proof) is characterized by 3 parameters

Introduction to PCP

PCP (Probabilistically Checkable Proof) is characterized by 3 parameters

- The number of random bits $r(n)$ the verifier has access to.
- The number of queries $\ell(n)$ the verifier uses
- The alphabet size $\alpha(n)$ of each query (the number of bits provided in each query).

Introduction to PCP

PCP (Probabilistically Checkable Proof) is characterized by 3 parameters

- The number of random bits $r(n)$ the verifier has access to.
- The number of queries $\ell(n)$ the verifier uses
- The alphabet size $\alpha(n)$ of each query (the number of bits provided in each query).

Definition (PCP class)

A decision problem L is in $\mathrm{PCP}[r(n), \ell(n), \alpha(n)]$, if it has a PCP verifier $\mathcal{V}$ satisfying:

- **Completeness:** For every $x \in L$, there exist a proof Π, such that verifier accepts (x, Π) with probability $\frac{3}{4}$.
- **Soundness:** For every $x \notin L$, and every proof Π, the verifier rejects (x, Π) with probability $\frac{3}{4}$.

PCP theorem

Theorem (PCP theorem [ALMSS98])

$$\mathrm{NP} = \mathrm{PCP}[\mathcal{O}(\log n), \mathcal{O}(1), \mathcal{O}(1)].$$

Ie. every problem in NP *has a* PCP*-protocol with* $\mathcal{O}(\log n)$ *random bits, constant queries and constant alphabet size.*

PCP theorem

Theorem (PCP theorem [ALMSS98])

$$\mathrm{NP} = \mathrm{PCP}[\mathcal{O}(\log n), \mathcal{O}(1), \mathcal{O}(1)].$$

Ie. every problem in NP *has a* PCP*-protocol with* $\mathcal{O}(\log n)$ *random bits, constant queries and constant alphabet size.*

- The PCP theorem was first proved in 92', following a productive line of results in the 80s and 90s.
- All proofs use a technique **composition of PCP's**, where multiple PCP's are used in succession to get a stronger PCP.

PCP theorem

Theorem (PCP theorem [ALMSS98])

$$\mathrm{NP} = \mathrm{PCP}[\mathcal{O}(\log n), \mathcal{O}(1), \mathcal{O}(1)].$$

Ie. every problem in NP *has a* PCP*-protocol with* $\mathcal{O}(\log n)$ *random bits, constant queries and constant alphabet size.*

- The PCP theorem was first proved in 92', following a productive line of results in the 80s and 90s.
- All proofs use a technique **composition of PCP's**, where multiple PCP's are used in succession to get a stronger PCP.
- In our work, we give the first proof of the PCP theorem using only **1** composition of two PCP's.

PCP theorem

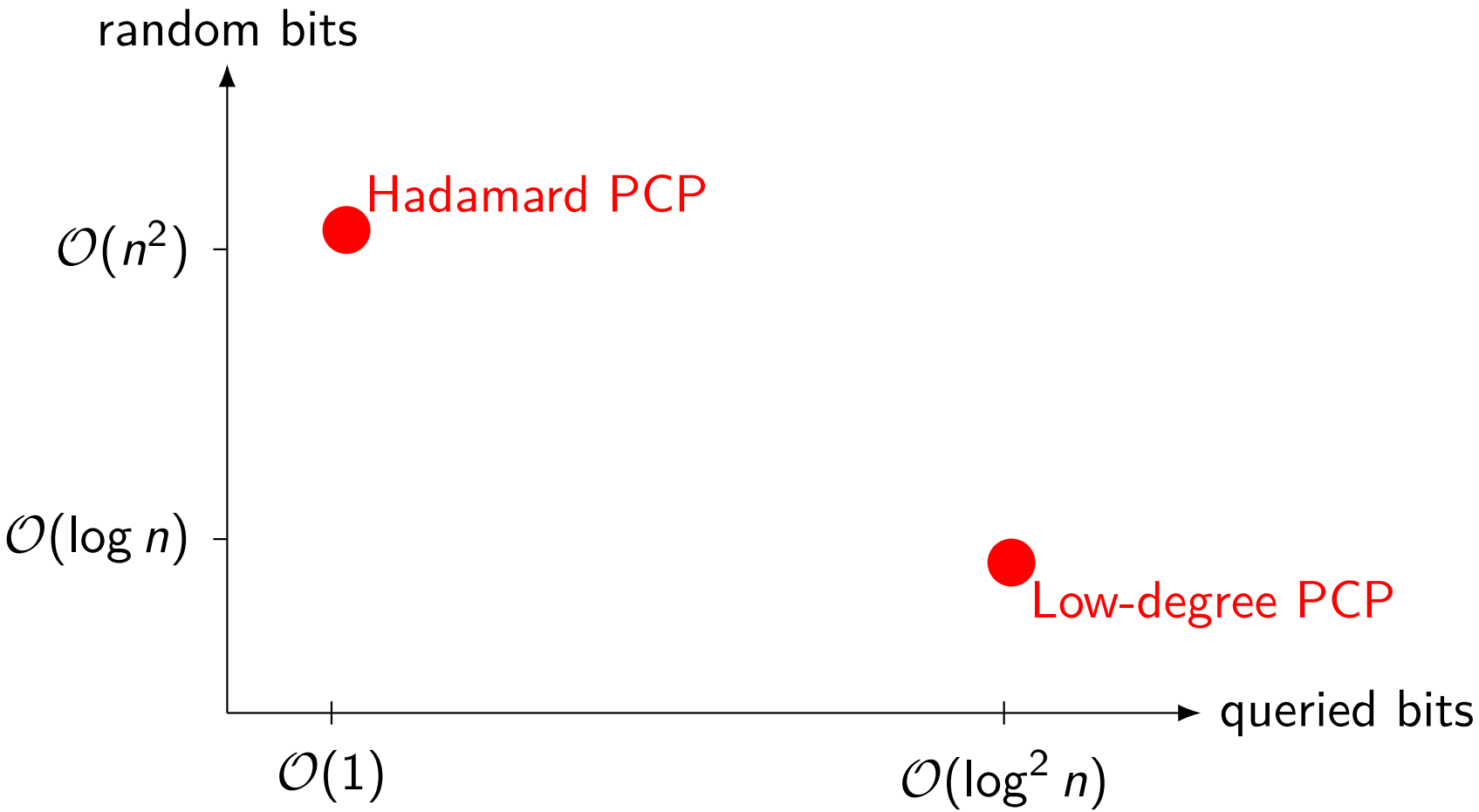
Theorem (PCP theorem [ALMSS98])

$$\mathrm{NP} = \mathrm{PCP}[\mathcal{O}(\log n), \mathcal{O}(1), \mathcal{O}(1)].$$

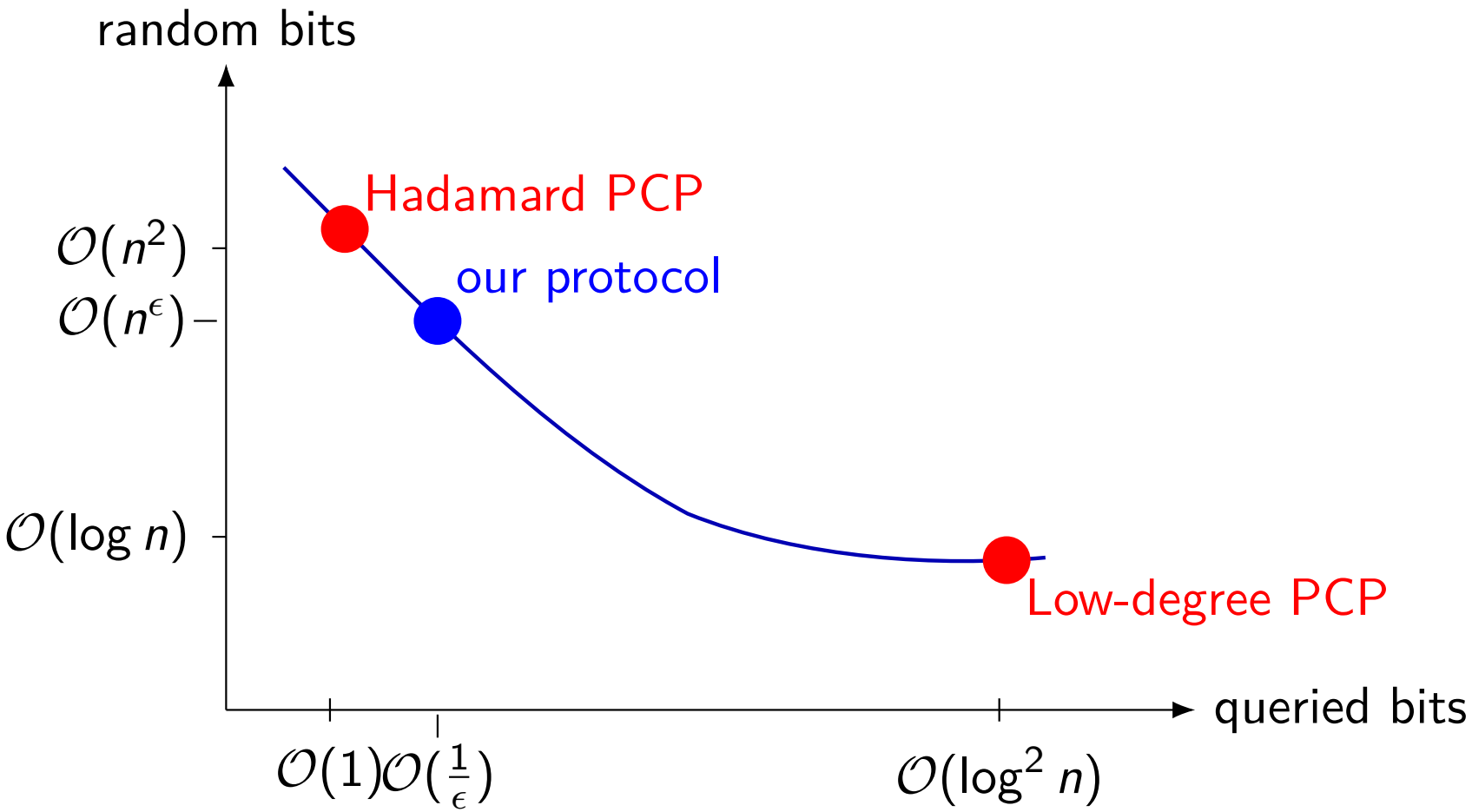
Ie. every problem in NP *has a* PCP*-protocol with* $\mathcal{O}(\log n)$ *random bits, constant queries and constant alphabet size.*

- The PCP theorem was first proved in 92', following a productive line of results in the 80s and 90s.
- All proofs use a technique **composition of PCP's**, where multiple PCP's are used in succession to get a stronger PCP.
- In our work, we give the first proof of the PCP theorem using only **1** composition of two PCP's.
- In this talk we will present our new PCP-protocol, and how it differs from previous protocols.

Results

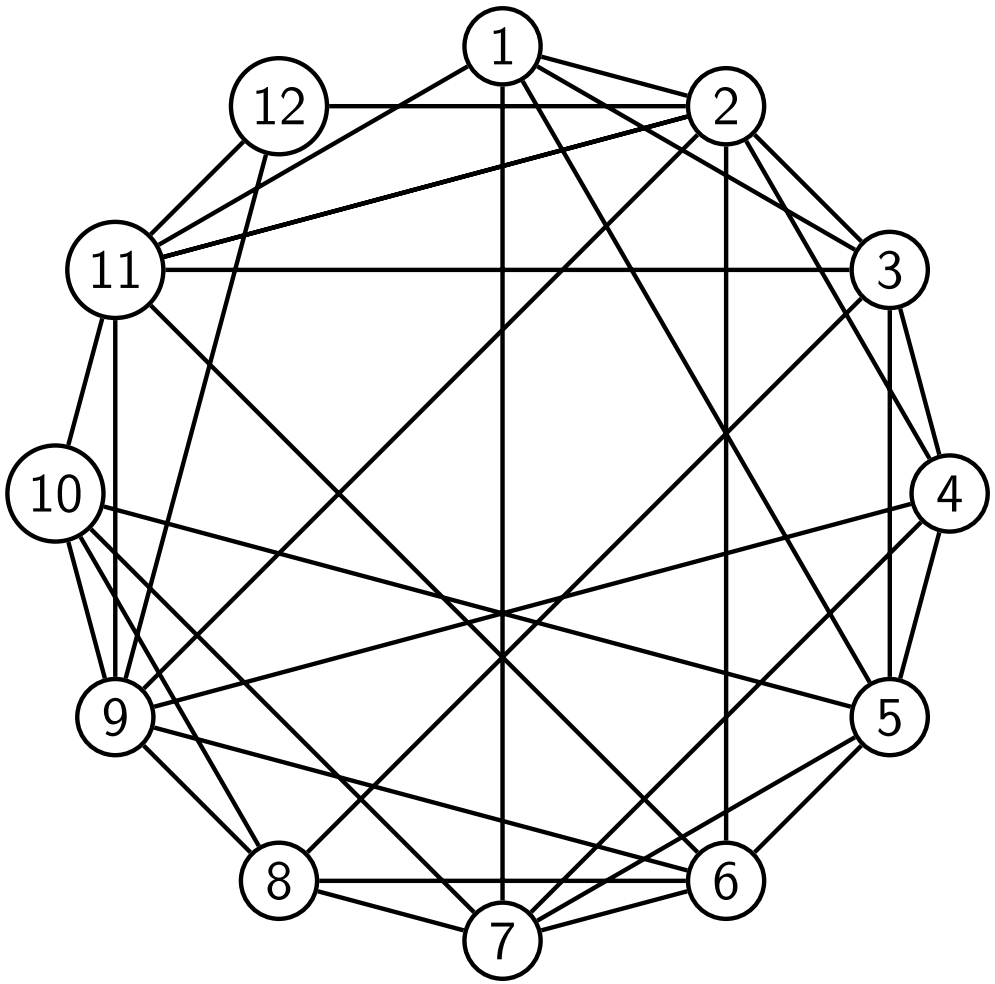

random bits
$\mathcal{O}(n^2)$
Hadamard PCP
$\mathcal{O}(\log n)$
Low-degree PCP
queried bits
$\mathcal{O}(1)$
$\mathcal{O}(\log^2 n)$

Results



We obtain a **new family of PCP protocols** whose parameters depend on a choice of a subset $S \subseteq \mathbb{F}_q^m$.

3-coloring of Graphs

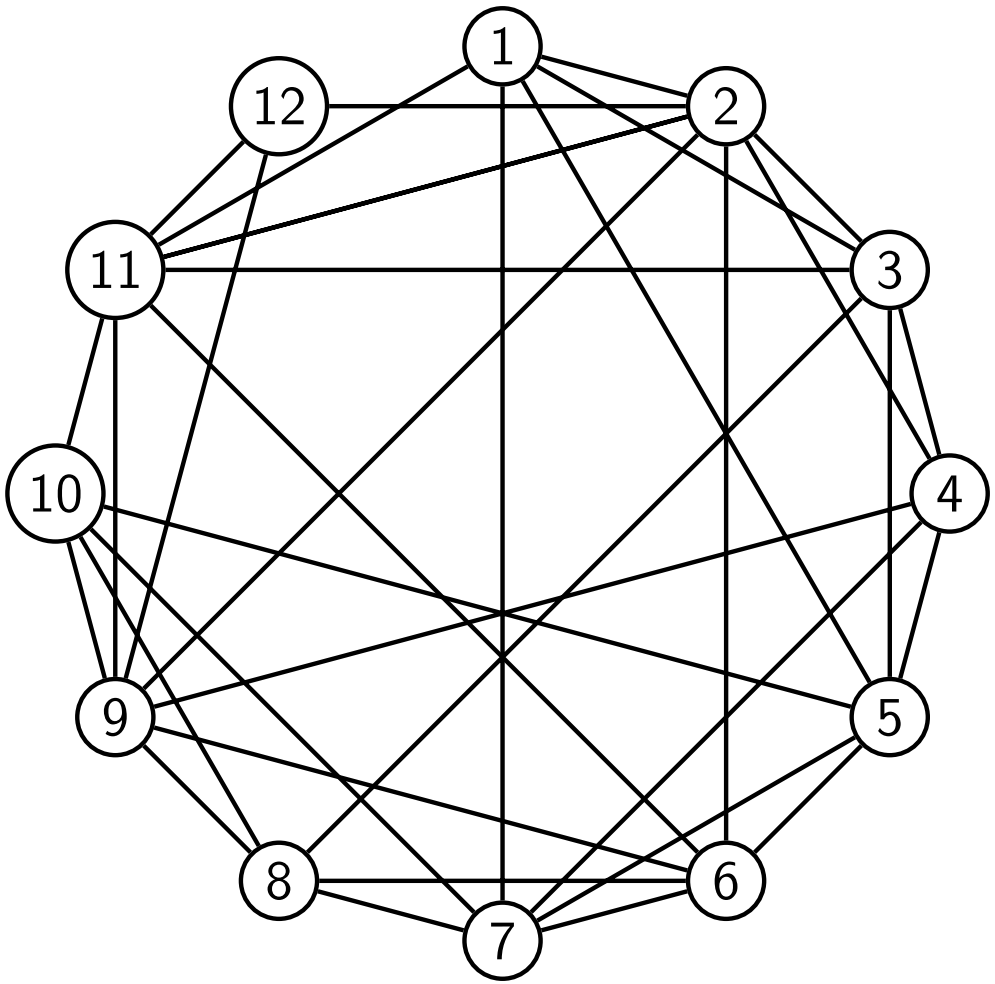


The NP-complete problem our protocol tackles is **3-coloring of a graph**:

3-coloring of a graph

Given a graph G on n vertices, does there exist a coloring of the vertices using 3 colors, where no neighbours share a color?

3-coloring of Graphs



The NP-complete problem our protocol tackles is **3-coloring of a graph**:

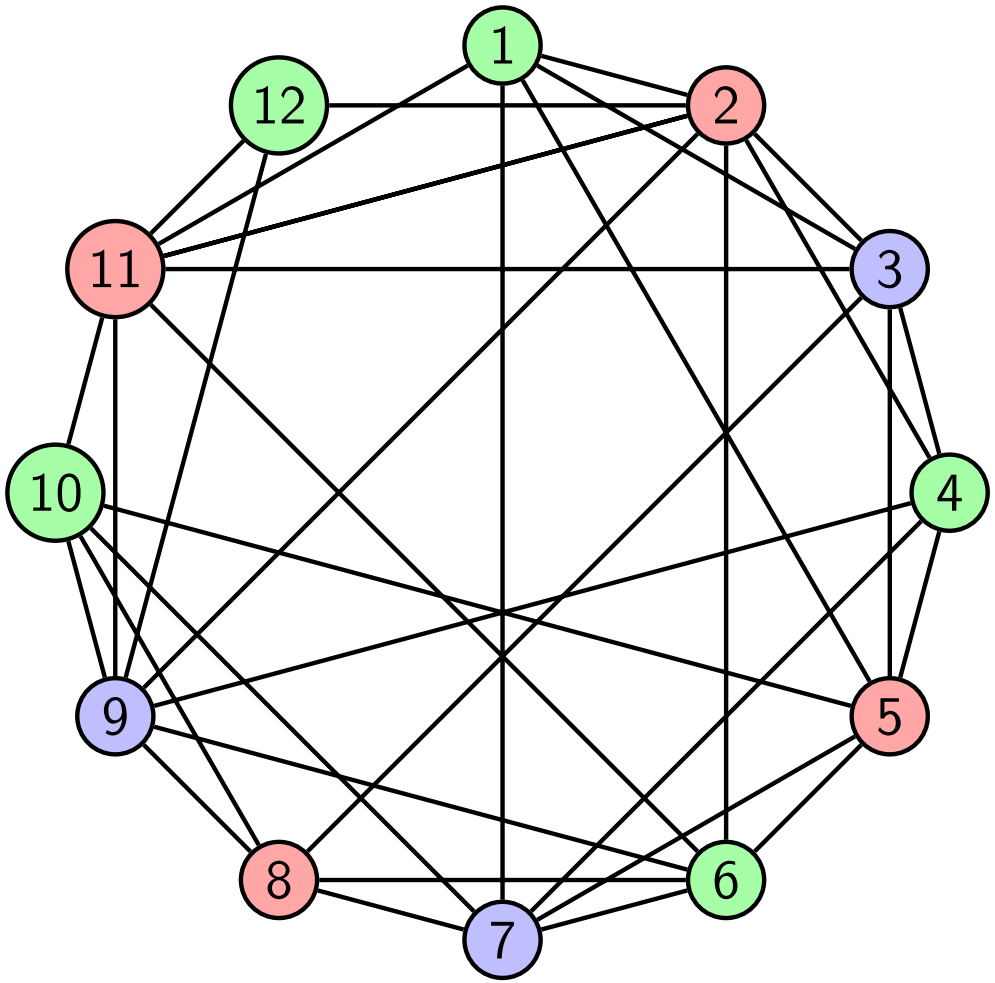
3-coloring of a graph

Given a graph G on n vertices, does there exist a coloring of the vertices using 3 colors, where no neighbours share a color?

Theorem ([Lov73])

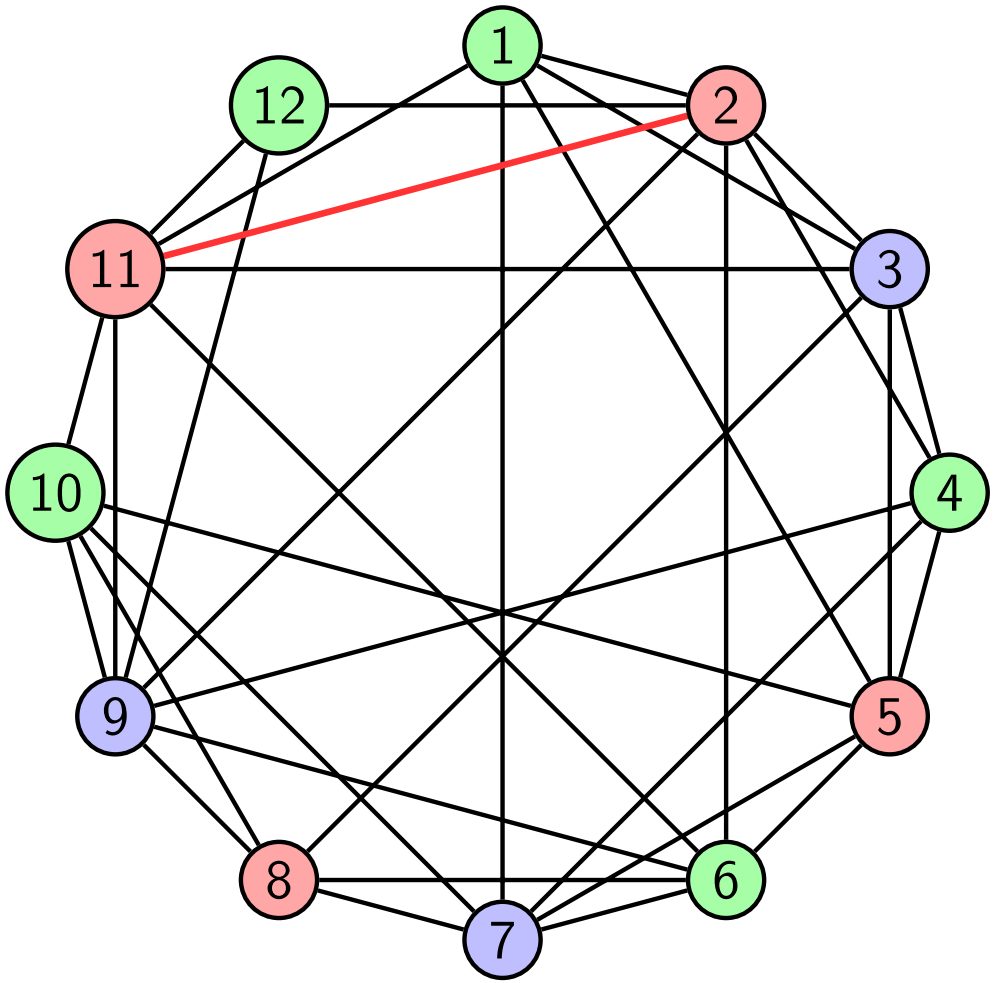
3-coloring is NP-complete.

3-coloring of Graphs



There is an obvious NP-certificate by giving a valid 3-coloring.

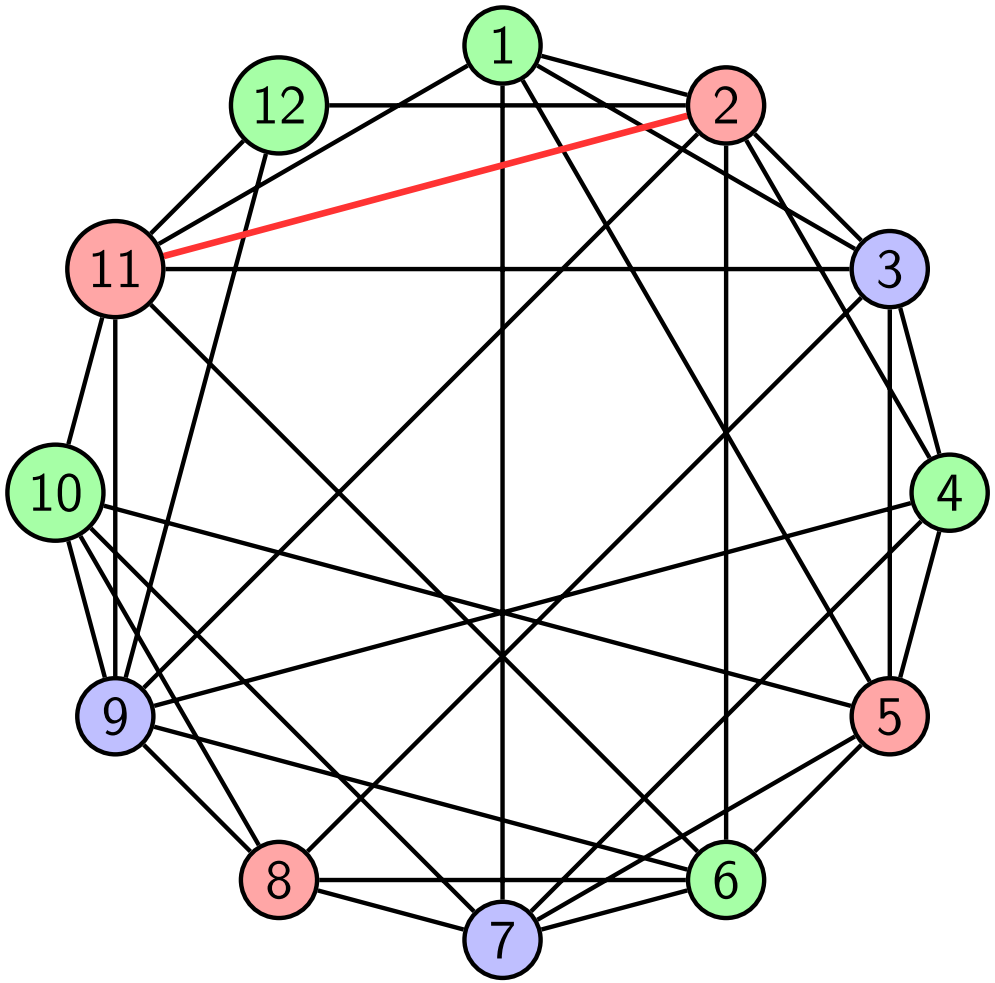
3-coloring of Graphs



There is an obvious NP-certificate by giving a valid 3-coloring.

However, if we only look at a few of the colors, it is hard to detect errors.

3-coloring of Graphs



There is an obvious NP-certificate by giving a valid 3-coloring.

However, if we only look at a few of the colors, it is hard to detect errors.

We will encode the coloring, so it is easier to detect errors

From coloring to polynomials

Assume that we have a coloring of a graph $G = (V, E)$ with n vertices

$$C : V \to \{-1, 0, 1\}$$
$$E : V \times V \to \{0, 1\}$$

From coloring to polynomials

Assume that we have a coloring of a graph $G = (V, E)$ with n vertices

$$C : V \to \{-1, 0, 1\}$$
$$E : V \times V \to \{0, 1\}$$

Pick a finite field $\mathbb{F}_q$, and identify

$$V \subseteq \mathbb{F}_q^m$$

From coloring to polynomials

Assume that we have a coloring of a graph $G = (V, E)$ with n vertices

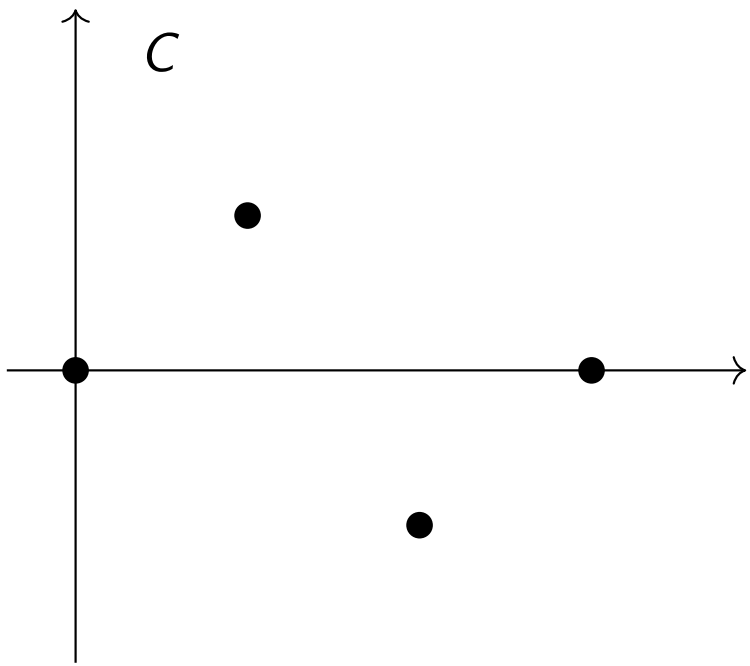
$$C : V \to \{-1, 0, 1\}$$
$$E : V \times V \to \{0, 1\}$$

Pick a finite field $\mathbb{F}_q$, and identify

$$V \subseteq \mathbb{F}_q^m$$

Polynomial interpolation on C, E

$$\tilde{C} : \mathbb{F}_q^m \to \mathbb{F}_q$$
$$\tilde{E} : \mathbb{F}_q^m \times \mathbb{F}_q^m \to \mathbb{F}_q$$



From coloring to polynomials

Assume that we have a coloring of a graph $G = (V, E)$ with n vertices

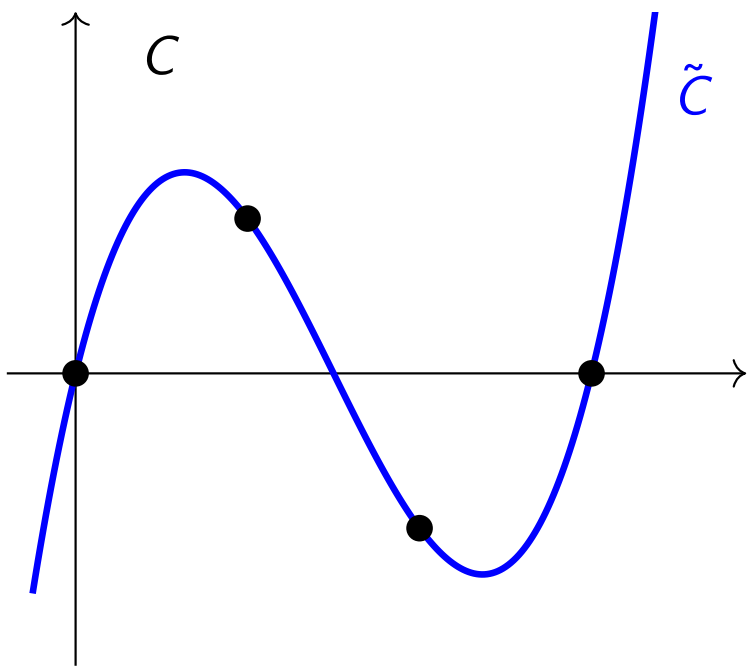
$$C : V \to \{-1, 0, 1\}$$
$$E : V \times V \to \{0, 1\}$$

Pick a finite field $\mathbb{F}_q$, and identify

$$V \subseteq \mathbb{F}_q^m$$

Polynomial interpolation on C, E

$$\tilde{C} : \mathbb{F}_q^m \to \mathbb{F}_q$$
$$\tilde{E} : \mathbb{F}_q^m \times \mathbb{F}_q^m \to \mathbb{F}_q$$



From coloring to polynomials

Assume that we have a coloring of a graph $G = (V, E)$ with n vertices

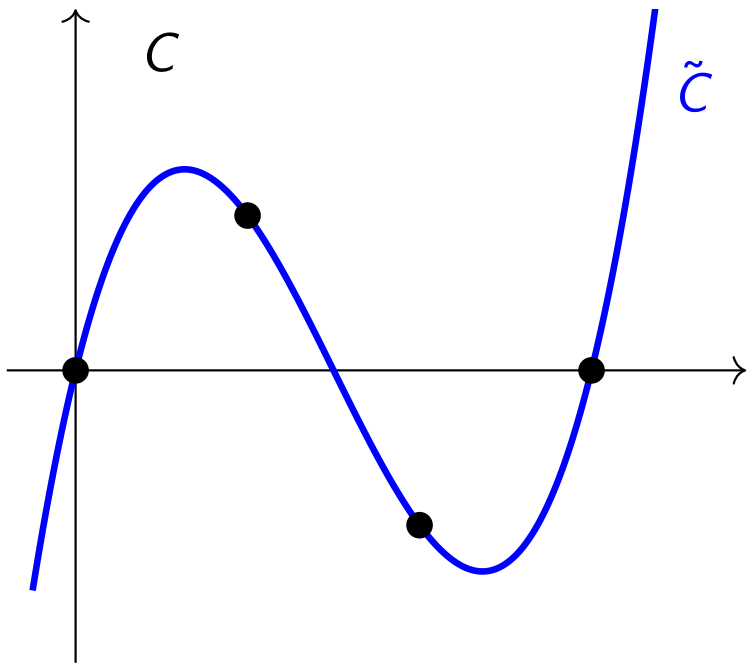
$$C : V \to \{-1, 0, 1\}$$
$$E : V \times V \to \{0, 1\}$$

Pick a finite field $\mathbb{F}_q$, and identify

$$V \subseteq \mathbb{F}_q^m$$

Polynomial interpolation on C, E

$$\tilde{C} : \mathbb{F}_q^m \to \mathbb{F}_q \qquad \deg(\tilde{C}) \leq d$$
$$\tilde{E} : \mathbb{F}_q^m \times \mathbb{F}_q^m \to \mathbb{F}_q \qquad \deg(\tilde{E}) \leq 2d$$



Upshot: Polynomials are much more structured than functions!

When is the coloring valid?

The 3-coloring $\tilde{C}$ is valid if it satifies two criteria:

When is the coloring valid?

The 3-coloring $\tilde{C}$ is valid if it satifies two criteria:
Using valid colors:

$$A(v) := \left(\tilde{C}(v) + 1\right) \cdot \tilde{C}(v) \cdot \left(\tilde{C}(v) - 1\right)$$

$$v \text{ has a valid color} \iff A(v) = 0.$$

When is the coloring valid?

The 3-coloring $\tilde{C}$ is valid if it satifies two criteria:
Using valid colors:

$$A(v) := \left(\tilde{C}(v) + 1\right) \cdot \tilde{C}(v) \cdot \left(\tilde{C}(v) - 1\right)$$

$$v \text{ has a valid color} \iff A(v) = 0.$$

Neighbours do not share colors:

$$B(v, w) := \tilde{E}(v, w) \cdot \prod_{\alpha=\{\pm 1, \pm 2\}} \left(\tilde{C}(v) - \tilde{C}(w) - \alpha\right)$$

$$v, w \text{ are not neighbours or do not share colors} \iff B(v, w) = 0.$$

When is the coloring valid?

The 3-coloring $\tilde{C}$ is valid if it satifies two criteria:
Using valid colors:

$$A(v) := \left(\tilde{C}(v)+1\right)\cdot\tilde{C}(v)\cdot\left(\tilde{C}(v)-1\right) \qquad \deg(A)\leq 3d$$

$$v \text{ has a valid color} \iff A(v)=0.$$

Neighbours do not share colors:

$$B(v,w) := \tilde{E}(v,w)\cdot\prod_{\alpha=\{\pm1,\pm2\}}\left(\tilde{C}(v)-\tilde{C}(w)-\alpha\right) \quad \deg(B)\leq 6d$$

$$v,w \text{ are not neighbours or do not share colors} \iff B(v,w)=0.$$

A, B are also low degree polynomials! In this talk, we focus on $B(v,w)=0$, checking $A(v)=0$ is identical.

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

- B **is correct:** $B(x, y) = \tilde{E}(x, y) \cdot \prod_{\alpha \in \{\pm 1, \pm 2\}} (\tilde{C}(x) + \tilde{C}(y) - \alpha)$.

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

- B **is correct:** $B(x, y) = \tilde{E}(x, y) \cdot \prod_{\alpha \in \{\pm 1, \pm 2\}} (\tilde{C}(x) + \tilde{C}(y) - \alpha)$.

The prover can trick us by giving us non-low degree polynomials!

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

- B **is correct:** $B(x, y) = \tilde{E}(x, y) \cdot \prod_{\alpha \in \{\pm 1, \pm 2\}} (\tilde{C}(x) + \tilde{C}(y) - \alpha)$.

The prover can trick us by giving us non-low degree polynomials!

- **Oracles are low degree:**

$$\deg(\tilde{C}) \leq d, \deg(B) \leq 6d.$$

We start with how to check B is correct.

Telling polynomials apart

Theorem (Polynomial Distance Lemma)

Fix a field $\mathbb{F}_q$ of size q. If two non-zero polynomials P, Q of degree d are not equal, then we have

$$\Pr_{\mathbf{a} \sim \mathbb{F}_q^m}[P(\mathbf{a}) = Q(\mathbf{a})] \leq \frac{d}{q}.$$

Telling polynomials apart

Theorem (Polynomial Distance Lemma)

Fix a field $\mathbb{F}_q$ of size q. If two non-zero polynomials P, Q of degree d are not equal, then we have

$$\Pr_{\mathbf{a} \sim \mathbb{F}_q^m}[P(\mathbf{a}) = Q(\mathbf{a})] \leq \frac{d}{q}.$$

This gives the following simple test for checking if two polynomials are the same.

Algorithm 2 Polynomial Equality test

1: Sample $a \leftarrow \mathbb{F}_q^m$ uniformly at random; query $P(a), Q(a)$.
2: **return** $P(a) = Q(a)$ ▷ ACCEPT iff true, else REJECT

Algorithm

Algorithm 2 Polynomial Equality test

1: Sample $a \leftarrow \mathbb{F}_q^m$ uniformly at random; query $P(a), Q(a)$.
2: **return** $P(a) = Q(a)$ ▷ Accept iff true, else Reject

Query Complexity	Alphabet Length	Randomness Complexity
2	$\log q$	$m \log q$

Algorithm

Algorithm 2 Polynomial Equality test

1: Sample $a \leftarrow \mathbb{F}_q^m$ uniformly at random; query $P(a), Q(a)$.
2: **return** $P(a) = Q(a)$ ▷ ACCEPT iff true, else REJECT

Query Complexity	Alphabet Length	Randomness Complexity
2	$\log q$	$m \log q$

The test fails with probability

$$\Pr_{\mathbf{a} \sim \mathbb{F}_q^m}[P(\mathbf{a}) = Q(\mathbf{a})] \leq \frac{d}{q}.$$

So as long as q is big enough (depends on V!), we can test whether or not B is correct with low error rate.

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

✓ B **is correct:** $B(x, y) = \tilde{E}(x, y) \cdot \prod_{\alpha \in \{\pm 1, \pm 2\}} (\tilde{C}(x) + \tilde{C}(y) - \alpha)$.

The prover can trick us by giving us non-low degree polynomials!

- **Oracles are low degree:**

$$\deg(\tilde{C}) \leq d, \deg(B) \leq 6d.$$

We now show how to test if the oracles are low degree.

Idea for Low degree test

Goal

Build a test which takes a function $f : \mathbb{F}_q^m \to \mathbb{F}_q$ such that:

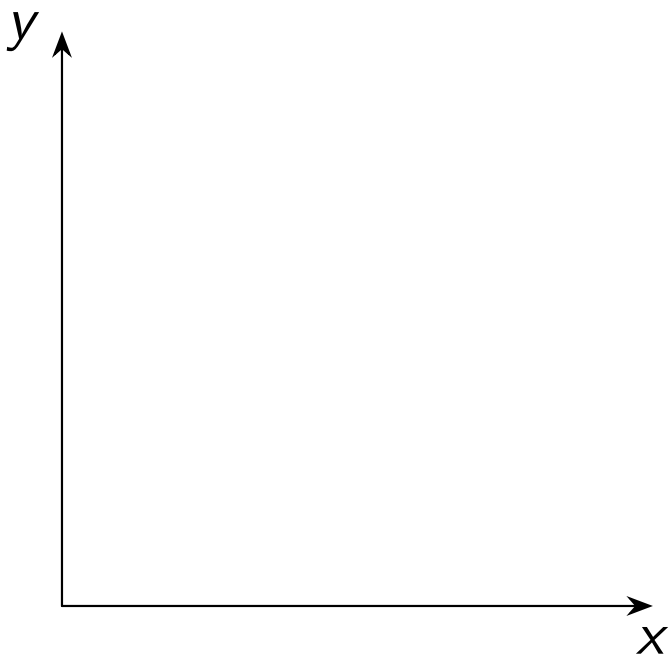
- Always accepts if f is a degree d polynomial.
- Rejects with high probability if f is far away from being a polynomial.

We start with one variable $f : \mathbb{F}_q \to \mathbb{F}_q$, where we can use interpolation

Low-Degree Test for one variable

Algorithm 3 1D Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q \to \mathbb{F}_q$, degree bound d

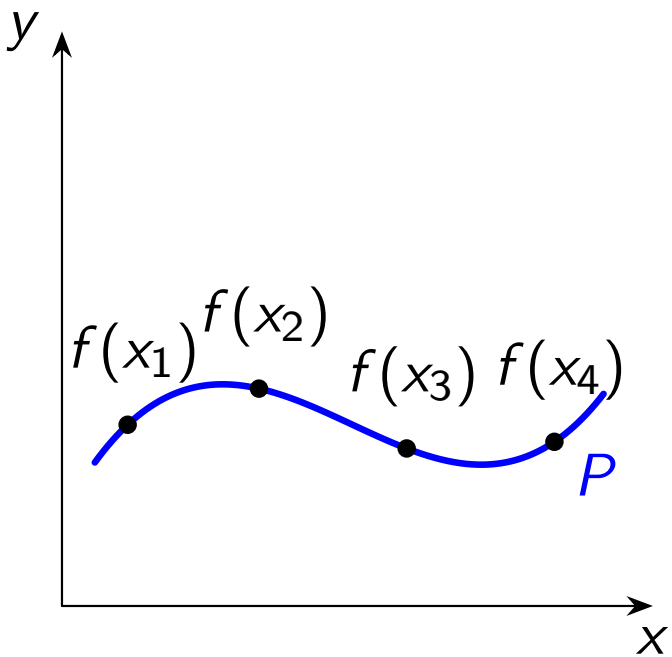


Low-Degree Test for one variable

Algorithm 3 1D Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q \to \mathbb{F}_q$, degree bound d

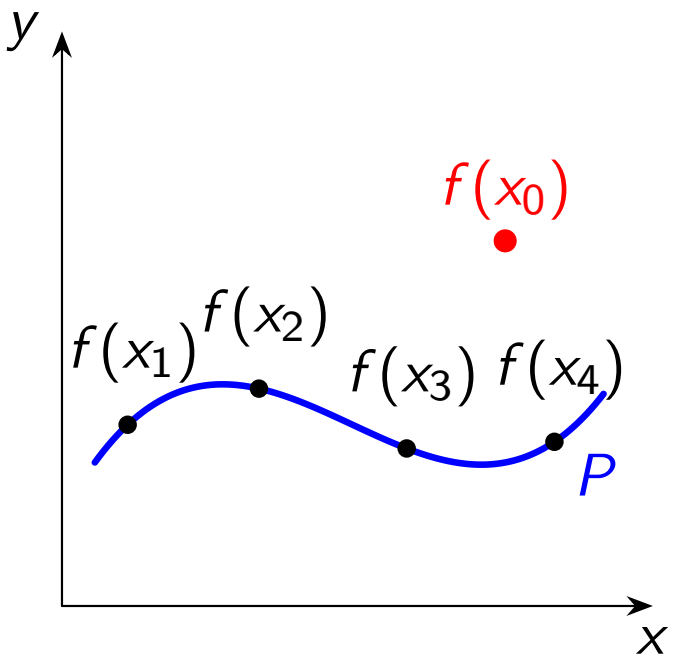
2: Pick $d+1$ points in $\mathbb{F}_q$, and interpolate to get a polynomial P.



Low-Degree Test for one variable

Algorithm 3 1D Low-Degree Test

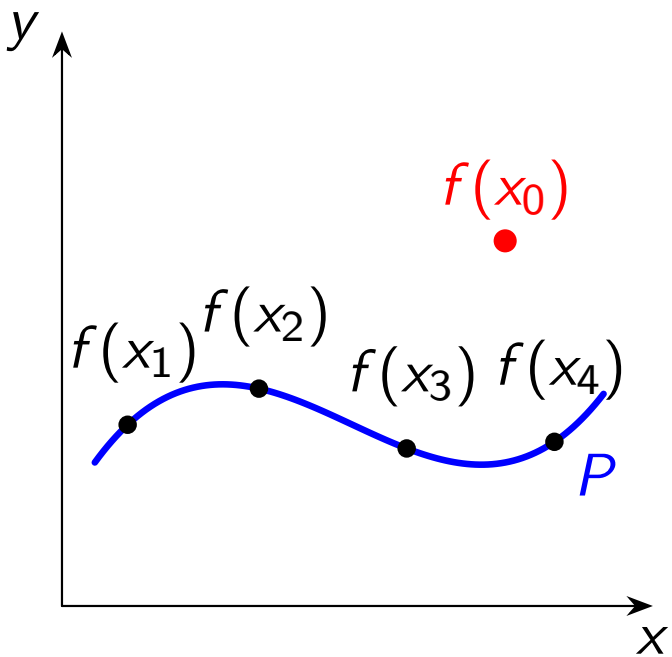
1: **Input:** function $f : \mathbb{F}_q \to \mathbb{F}_q$, degree bound d
2: Pick $d+1$ points in $\mathbb{F}_q$, and interpolate to get a polynomial P.
3: **return** Pick a new point x_0 and ACCEPT if $f(x_0) = P(x_0)$



Low-Degree Test for one variable

Algorithm 3 1D Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q \to \mathbb{F}_q$, degree bound d
2: Pick $d+1$ points in $\mathbb{F}_q$, and interpolate to get a polynomial P.
3: **return** Pick a new point x_0 and ACCEPT if $f(x_0) = P(x_0)$

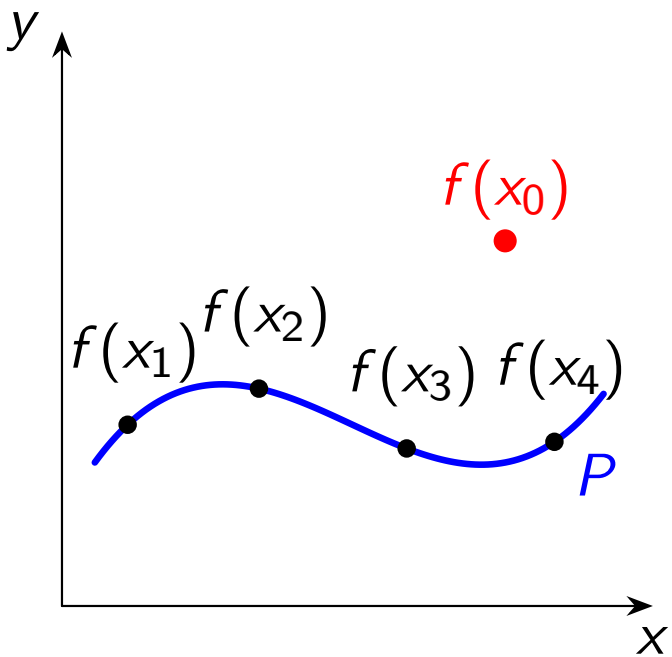


If f disagrees with every degree d polynomial with a fraction δ, then test rejects with probability δ.

Low-Degree Test for one variable

Algorithm 3 1D Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q \to \mathbb{F}_q$, degree bound d
2: Pick $d+1$ points in $\mathbb{F}_q$, and interpolate to get a polynomial P.
3: **return** Pick a new point x_0 and ACCEPT if $f(x_0) = P(x_0)$

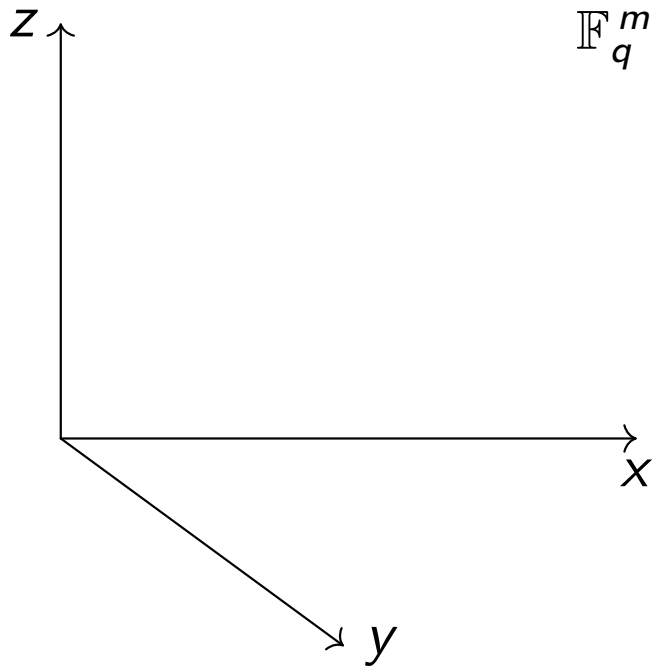


If f disagrees with every degree d polynomial with a fraction δ, then test rejects with probability δ.
We generalize to multiple variables.

Low-Degree Test

Algorithm 3 Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q^m \to \mathbb{F}_q$, degree bound d

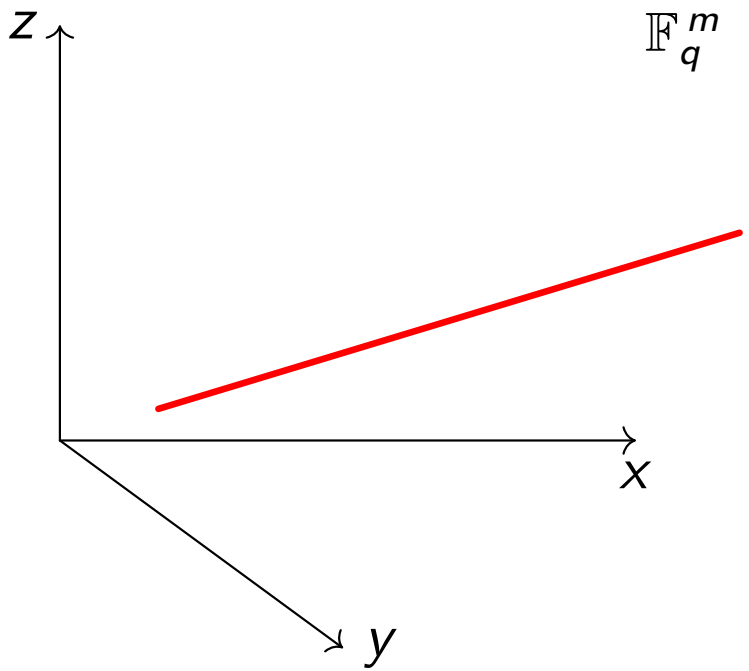


Low-Degree Test

Algorithm 3 Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q^m \to \mathbb{F}_q$, degree bound d

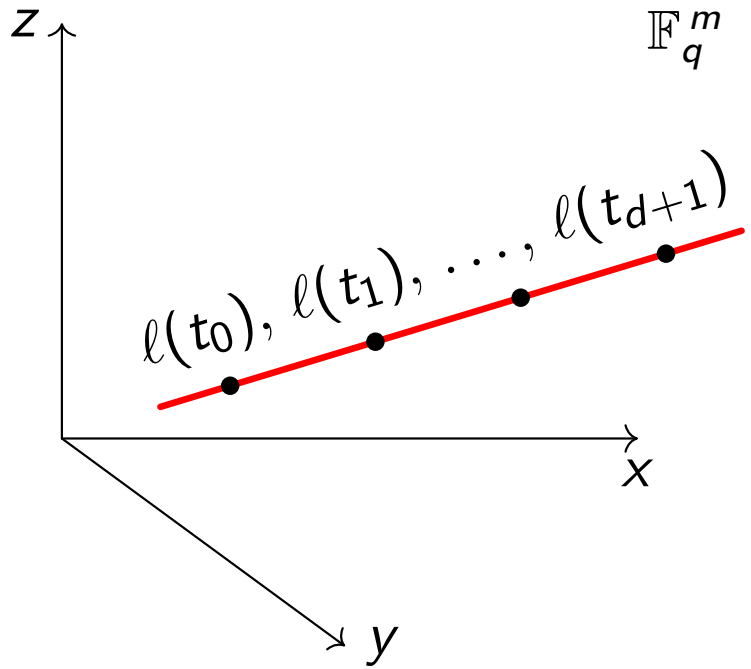
2: Sample $\mathbf{a}, \mathbf{b} \leftarrow \mathbb{F}_q^m$; set $\ell(t) = \mathbf{a} + t\mathbf{b}$



Low-Degree Test

Algorithm 3 Low-Degree Test

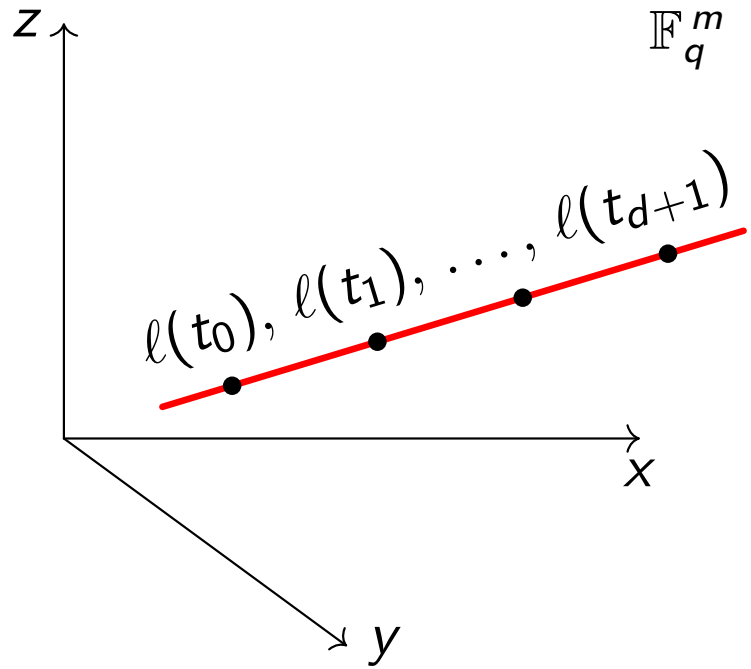
1: **Input:** function $f : \mathbb{F}_q^m \to \mathbb{F}_q$, degree bound d
2: Sample $\mathbf{a}, \mathbf{b} \leftarrow \mathbb{F}_q^m$; set $\ell(t) = \mathbf{a} + t\mathbf{b}$
3: Perform 1D-Low-Degree-Test on $f(\ell(t))$



Low-Degree Test

Algorithm 3 Low-Degree Test

1: **Input:** function $f : \mathbb{F}_q^m \to \mathbb{F}_q$, degree bound d
2: Sample $\mathbf{a}, \mathbf{b} \leftarrow \mathbb{F}_q^m$; set $\ell(t) = \mathbf{a} + t\mathbf{b}$
3: Perform 1D-Low-Degree-Test on $f(\ell(t))$



Query Complexity	Alphabet Complexity	Randomness Complexity
$d + 2$	$\log q$	$(2m + d + 2) \log q$

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}$

- **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

✓ B **is correct:** $B(x, y) = \tilde{E}(x, y) \cdot \prod_{\alpha \in \{\pm 1, \pm 2\}} (\tilde{C}(x) + \tilde{C}(y) - \alpha)$.

The prover can trick us by giving us non-low degree polynomials!

✓ **Oracles are low degree:**

$$\deg(\tilde{C}) \leq d, \deg(B) \leq 6d.$$

Lastly we show how to test if a polynomial vanishes on V.

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$, *prove* $P(v) = 0$ *for* $v \in V$.

Polynomial distance lemma does not hold over V.

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$, prove $P(v) = 0$ for $v \in V$.

Polynomial distance lemma does not hold over V. We assume that $V = \{0,1\}^m \subseteq \mathbb{F}_q^m$.

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$, *prove* $P(v) = 0$ *for* $v \in V$.

Polynomial distance lemma does not hold over V. We assume that $V = \{0,1\}^m \subseteq \mathbb{F}_q^m$.

Theorem (Hilbert's Nullstellensatz)

For a P *be a degree* d *polynomial, we have*

$$P(v) = 0 \text{ for } v \in \{0,1\}^m \iff P(\mathbf{x}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

where $\deg(h_i) \leq d - 2$.

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$ *of degree* d*, prove there exist* h_i *such*

$$P(\mathbf{x}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

Idea

Prover gives us query access to P *and* $\sum_{i=1}^{m} h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$ *and apply* Polynomial-Equality.

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$ of degree d, prove there exist h_i such

$$P(\mathbf{x}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

Idea

Prover gives us query access to P and $\sum_{i=1}^{m} h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$ and apply Polynomial-Equality.

Problem

A cheating prover, can just give us P twice!

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$ of degree d, prove there exist h_i such

$$P(\mathbf{x}) = \sum h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

Trick: Instead of $\sum_{i=1}^{m} h_i(\mathbf{x}) (x_i^2 - x_i)$ directly, we introduce new variables

$$\mathcal{M}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot y_i$$

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$ of degree d, prove there exist h_i such

$$P(\mathbf{x}) = \sum h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

Trick: Instead of $\sum_{i=1}^{m} h_i(\mathbf{x}) (x_i^2 - x_i)$ directly, we introduce new variables

$$\mathcal{M}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot y_i$$

1) run Polynomial-Equality$(P, \mathcal{M})$ with $y_i \leftarrow x_i^2 - x_i$.

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$ of degree d, prove there exist h_i such

$$P(\mathbf{x}) = \sum h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

Trick: Instead of $\sum_{i=1}^{m} h_i(\mathbf{x}) (x_i^2 - x_i)$ directly, we introduce new variables

$$\mathcal{M}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot y_i$$

1) run $\textsc{Polynomial-Equality}(P, \mathcal{M})$ with $y_i \leftarrow x_i^2 - x_i$.
What if a cheating prover gives us P?

Zero-on-subset test

Problem

Given a polynomial $P : \mathbb{F}_q^m \to \mathbb{F}_q$ of degree d, prove there exist h_i such

$$P(\mathbf{x}) = \sum h_i(\mathbf{x}) \cdot (x_i^2 - x_i)$$

Trick: Instead of $\sum_{i=1}^{m} h_i(\mathbf{x})\,(x_i^2 - x_i)$ directly, we introduce new variables

$$\mathcal{M}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{m} h_i(\mathbf{x}) \cdot y_i$$

1) run $\textsc{Polynomial-Equality}(P, \mathcal{M})$ with $y_i \leftarrow x_i^2 - x_i$.
What if a cheating prover gives us P?
We run $\textsc{Polynomial-Equality}(0, \mathcal{M})$ with $y_i \leftarrow 0$, ensuring $\mathcal{M}$ depends on y-variables.

Zero-on-subset test

Algorithm 3 Zero-on-subset test

1: **Input:** Polynomials $P(\mathbf{x})$, $\mathcal{M}(\mathbf{x}, \mathbf{y})$
2: $\textsc{Polynomial-Equality}(0, \mathcal{M}(\mathbf{x}, 0))$
3: $\textsc{Polynomial-Equality}\big(P(\mathbf{x}), \mathcal{M}(\mathbf{x}, x_i^2 - x_i)\big)$
4: **return** $\textsc{Accept}$ if both **true**

Zero-on-subset test

Algorithm 3 Zero-on-subset test

1: **Input:** Polynomials $P(\mathbf{x})$, $\mathcal{M}(\mathbf{x}, \mathbf{y})$
2: $\textsc{Polynomial-Equality}(0, \mathcal{M}(\mathbf{x}, 0))$
3: $\textsc{Polynomial-Equality}(P(\mathbf{x}), \mathcal{M}(\mathbf{x}, x_i^2 - x_i))$
4: **return** $\textsc{Accept}$ if both **true**

Also works for other subsets if we replace $x_i^2 - x_i$ with other polynomials. Parameters of test depends on choice of subset V.

PCP for 3-coloring

Certificates: polynomials $B, \tilde{C}, \mathcal{M}_B$

- ✓ **Neighbours do not share colors:** $B(v, w) = 0$ for all $v, w \in V$.

The prover can trick us by picking fake B!

- ✓ B **is correct:** $B(x, y) = \tilde{E}(x, y) \cdot \prod_{\alpha \in \{\pm 1, \pm 2\}} (\tilde{C}(x) + \tilde{C}(y) - \alpha)$.

The prover can trick us by giving us non-low degree polynomials!

- ✓ **Oracles are low degree:**

$$\deg(\tilde{C}) \leq d, \deg(B) \leq 6d, \deg(\mathcal{M}_B) \leq 6d.$$

Doing the 3 tests above for both A, B gives the PCP protocol of 3-colorability of graphs.

How good is our PCP?

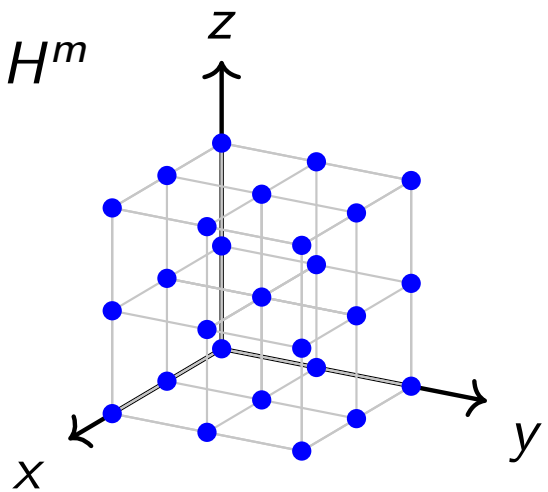
The parameters depends on choice of subset V, not a clear best choice.

Subset	**Random Bits**	**Queries**	**Alphabet Size**

How good is our PCP?

The parameters depends on choice of subset V, not a clear best choice.

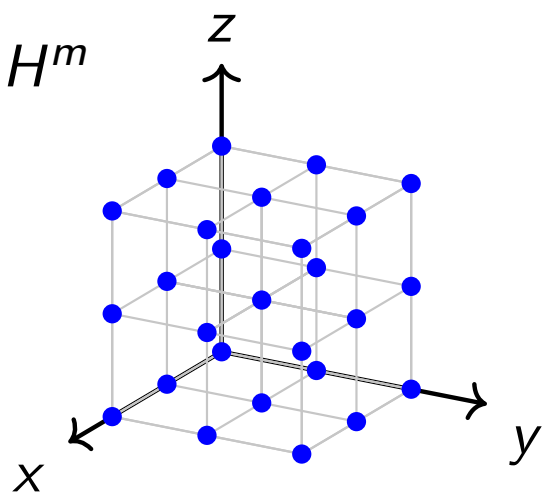
Subset	Random Bits	Queries	Alphabet Size
$\{0, \ldots, \log n\}^{\frac{\log n}{\log \log n}}$	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(\log^2 n / \log \log n)$

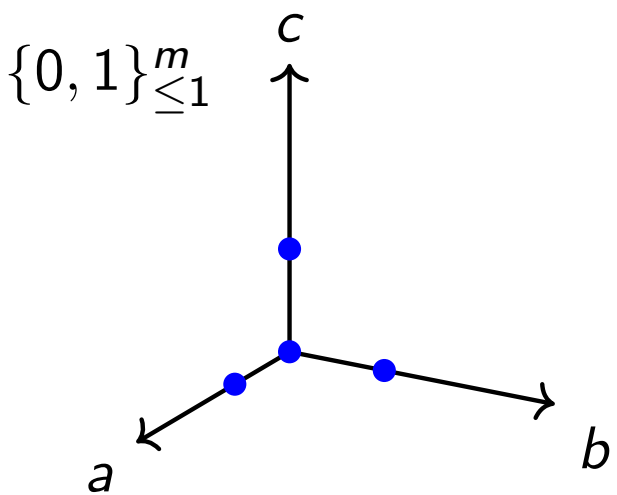


How good is our PCP?

The parameters depends on choice of subset V, not a clear best choice.

Subset	Random Bits	Queries	Alphabet Size
$\{0,\ldots,\log n\}^{\frac{\log n}{\log\log n}}$	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(\log^2 n/\log\log n)$
$\left(\{0,1\}^{n^{0.25}}_{\leq 1}\right)^4$	$\mathcal{O}(\sqrt{n})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$

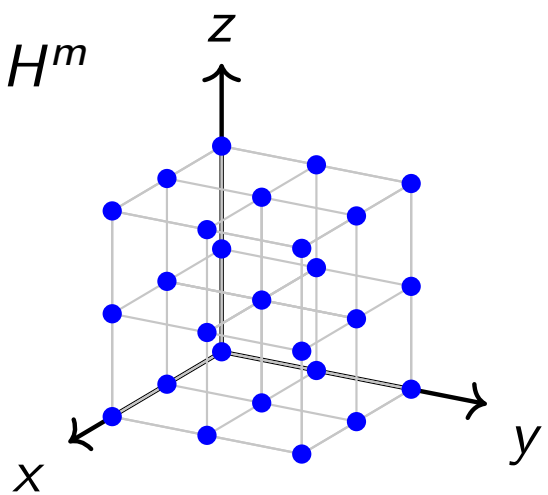


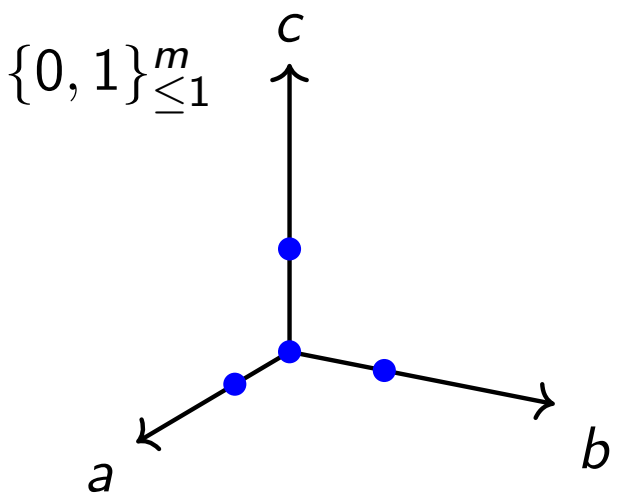


How good is our PCP?

The parameters depends on choice of subset V, not a clear best choice.

Subset	Random Bits	Queries	Alphabet Size
$\{0,\ldots,\log n\}^{\frac{\log n}{\log\log n}}$	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(\log^2 n/\log\log n)$
$\left(\{0,1\}^{n^{0.25}}_{\leq 1}\right)^4$	$\mathcal{O}(\sqrt{n})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Composition	$\mathcal{O}(\log n)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$





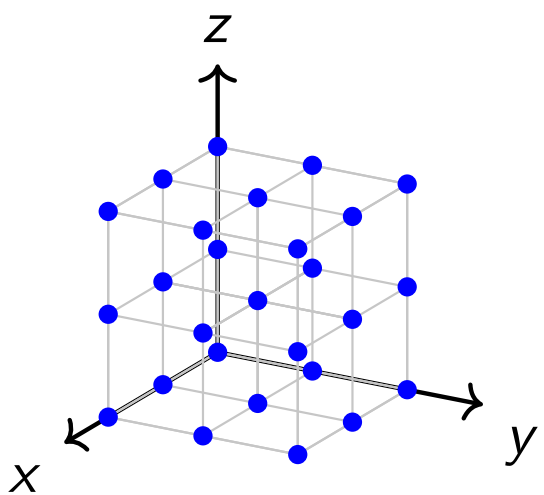
What makes a subset good

A good subset has two properties:

What makes a subset good

A good subset has two properties:

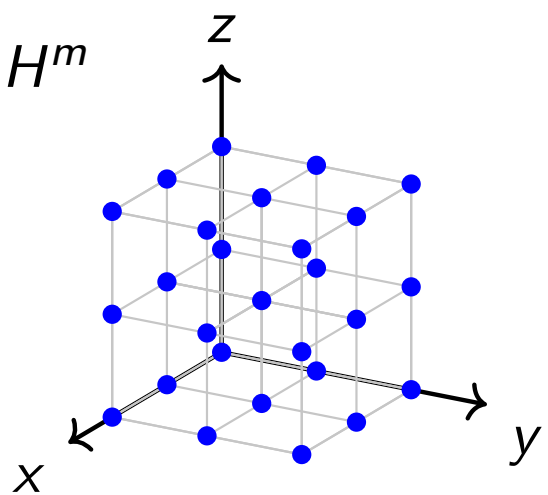
- Defined using few variables, and simple to describe with equations (like the grid with $x_i^2 - x_i$) less randomness

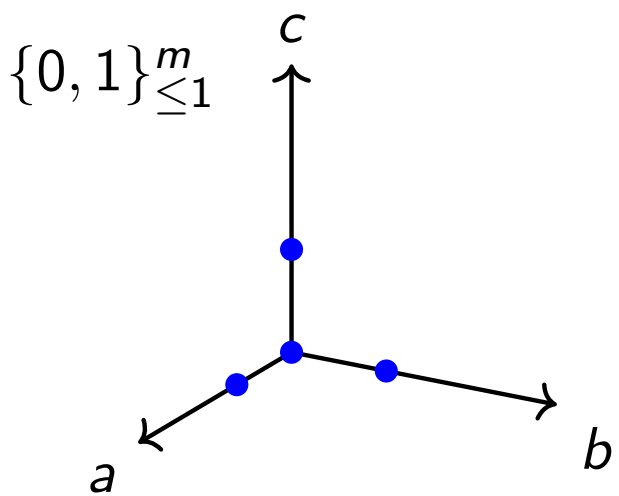


What makes a subset good

A good subset has two properties:

- Defined using few variables, and simple to describe with equations (like the grid with $x_i^2 - x_i$) less randomness
- functions $f : V \to \mathbb{F}_q$ can be extended to low-degree polynomials (like the Boolean slice) smaller alphabet size/query complexity

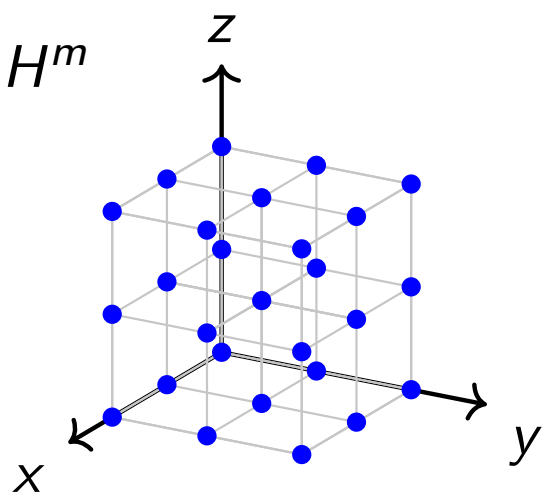


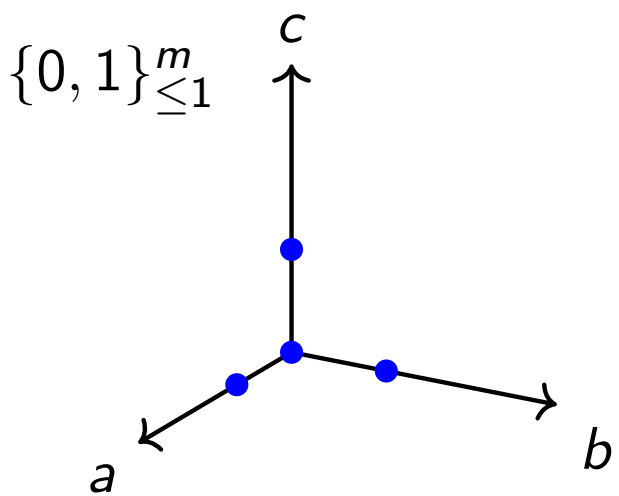


What makes a subset good

A good subset has two properties:

- Defined using few variables, and simple to describe with equations (like the grid with $x_i^2 - x_i$) less randomness
- functions $f : V \to \mathbb{F}_q$ can be extended to low-degree polynomials (like the Boolean slice) smaller alphabet size/query complexity





These properties are mostly opposed to each other

What is next!

Further research directions

- Proving the PCP theorem without composition (hard!)

What is next!

Further research directions

- Proving the PCP theorem without composition (hard!)
- Finding other applications of the Zero-on-subset protocol (IP, MIP)

What is next!

Further research directions

- Proving the PCP theorem without composition (hard!)
- Finding other applications of the Zero-on-subset protocol (IP, MIP)

If you want to hear more about PCP's then come to ITU on friday for lectures by Srikanth and Amik!

What is next!

Further research directions

- Proving the PCP theorem without composition (hard!)
- Finding other applications of the Zero-on-subset protocol (IP, MIP)

If you want to hear more about PCP's then come to ITU on friday for lectures by Srikanth and Amik!

Thanks for listening!