Robust PCPs

Today we construct robust PCPs for $NP$, which are used as an "outer PCP" in the proof of the PCP Theorem via proof composition.

Recall the definition of robust PCPs:

- **Definition**: $(P, V)$ is a PCP system for a language $L$ with robustness parameter $\sigma$ if:
  1. **Completeness**: $\forall x \in L$, for $\Pi := P(x)$, $\Pr_P[V^\pi(x; \rho) = 1] \geq 1 - \varepsilon_c$
  2. **Robust Soundness**: $\forall x \not\in L \forall \Pi \exists \sigma' \Pr_P[\Delta(\Pi; \Omega(x, \rho)), R(V)(x, \rho)] \leq \varepsilon_s$

The above is for non-adaptive PCPs $(V^\pi(x; \rho) = D(x, \rho, \Pi(\Omega(x, \rho)))$ and $R(V) = \{(x, \rho, \alpha) | \alpha \in \Sigma^{O(\sqrt{n}) \wedge D(x, \rho, \alpha) = 1}\}$.

Robustness $\sigma \in \{0, \sigma\}$ (wrt Hamming distance over $\Sigma$) is trivial.

The challenge is to achieve $\sigma = \Omega(1)$ even if $\sigma$ is super-constant.

We show how to achieve a robust analogue of the polynomial-size PCPs we constructed:

- **Theorem**: $NP \subseteq \text{PCP}[\varepsilon_c = 0, \varepsilon_s = 1/2, \Sigma = \{0, 1\}, \ell = \text{poly}(n), q = \text{polylog}(n), \varepsilon = O(\log n), \sigma = \Omega(1)]$
Proof Plan

We prove the theorem in two steps, starting from "standard" PCPs for \( \text{NP} \). [ The construction for quadratic equations that uses the sumcheck protocol and low-degree test. ]

\[
\text{NP} \subseteq \text{PCP} \left[ \varepsilon_c = 0, \varepsilon_s = \frac{1}{2}, \Sigma = \{0,1\}, \ell = \text{poly}(n), q = \text{polylog}(n), r = O(\text{log} n) \right]
\]

**Step 1: query bundling** reduce query complexity to constant at the expense of alphabet size

\[
\text{NP} \subseteq \text{PCP} \left[ \varepsilon_c = 0, \varepsilon_s = \frac{1}{2}, \Sigma = \{0,1,3^{\text{polylog}(n)}\}, \ell = \text{poly}(n), q = O(1), r = O(\text{log} n) \right]
\]

**Step 2: robustification** achieve constant robustness over \( \{0,1,3\} \) at the expense of query complexity

**Theorem:** \( \text{NP} \subseteq \text{PCP} \left[ \varepsilon_c = 0, \varepsilon_s = \frac{1}{2}, \Sigma = \{0,1,3\}, \ell = \text{poly}(n), q = \text{polylog}(n), r = O(\text{log} n), \sigma = O(1) \right] \)

We study each step in turn.
Robustification

We wish to achieve good robustness over the binary alphabet, starting from a large-alphabet PCP.

**Idea:** break each large-symbol query into multiple bit queries

This preserves completeness and soundness, and indeed reduces the alphabet to binary.

**Problem:** the resulting PCP may have trivial robustness \( \sigma \in [0, \frac{1}{q \cdot \log |\Sigma|}) \)

This is because many local views in the large-alphabet PCP could be 1 symbol (out of \( q \)) away from accepting, and so in the binary-alphabet PCP this could translate to 1 bit (out of \( q \cdot \log |\Sigma| \)) away from accepting.

This simple idea can be fixed to achieve this lemma:

**Lemma:** \( \text{PCP}[\varepsilon_s, \Sigma, \ell, q, \delta] \leq \text{PCP}[\varepsilon_s, \Sigma' = \{0,1\}, \ell' = O(\ell \cdot \log |\Sigma|), q' = O(q \cdot \log |\Sigma|), r' = r, \delta' = \delta(\frac{1}{q})] \)
Robustification

Lemma: \( PCP(\varepsilon, \Sigma, l, q, \delta, r) \subseteq PCP(\varepsilon, \Sigma', l' = \Theta(l \cdot \log |\Sigma|), q' = \Theta(q \log |\Sigma|), r' = \epsilon, \delta' = \Omega(l/q)) \)

Proof: Encode each proof symbol via a good code to "sparsify" accepting local views.

Let \( \text{Enc}: \Sigma \to \{0,1\}^N \) be an injective map with linear block length \( N \) (i.e., \( \Theta(\log |\Sigma|) \)) and constant relative distance \( \delta \) (\( \Delta(\text{Enc}(u), \text{Enc}(v)) \geq \delta > 0 \) for every two distinct \( u, v \in \Sigma \)). This is a standard tool known as a good code (it can be constructed via code concatenation).

\[ \pi: [l] \to \Sigma \]

\[ \Pi: [lN] \to \{0,1\}^N \]

\[ q' = q \cdot N \]

**P_\varepsilon(x)**

1. \( \pi := P(x) \in \Sigma^l \)
2. For \( i \in [l]: \Pi_i = \text{Enc}(\pi_i) \in \{0,1\}^N \)
3. Output \( \Pi = (\Pi_1, ..., \Pi_l) \in \{0,1\}^{N \cdot l} \)

**V_\varepsilon^{\Pi}(x)**

Run \( V(x) \) by answering a query \( i \in [l] \) by returning \( \text{Enc}(\Pi_i) \). (Reject if any 1.)

Completeness: If \( x \in L \) then \( V(x) \) accepts \( \Pi = P(x) \) w.p. at least 1-\( \varepsilon_c \). Hence \( V_\varepsilon(x) \) accepts \( \Pi := (\text{Enc}(\pi_1), ..., \text{Enc}(\pi_\ell)) \) w.p. at least 1-\( \varepsilon_c \) because query \( i \in [l] \) is answered with \( \text{Enc}(\text{Enc}(\pi_i)) = \pi_i \).

Robust soundness: next slide.
Suppose that $x \notin L$ and fix $\overline{\Pi} = (\overline{\Pi}_1, ..., \overline{\Pi}_e) \in \{0,1\}^N$.

Let $E$ be the event that the local view contains a string that is $\delta' -$ far from the code.

Then

$$P_p \left[ \Delta(\overline{\Pi}, \pi_{\text{r}}(x,p), R(V_r)(x,p)) \leq \frac{\delta}{2 \cdot 9} \right] \quad \text{(i.e., robustness parameter } \delta = \frac{\delta}{2 \cdot 9} \text{)}$$

$$\leq P_p \left[ \Delta(\overline{\Pi}, \pi_{\text{r}}(x,p), R(V_r)(x,p)) \leq \frac{\delta}{2 \cdot 9} | E \right] + P_p \left[ \Delta(\overline{\Pi}, \pi_{\text{r}}(x,p), R(V_r)(x,p)) \leq \frac{\delta}{2 \cdot 9} | \overline{\Pi} \right]$$

$$\leq \varepsilon_5 + \varepsilon_5.$$  \hspace{0.5cm} \bullet \hspace{0.5cm} \bullet \hspace{0.5cm} \bullet

(\text{b) If } \overline{\Pi} \text{ contains a string that is } \delta' - \text{ far from the code, then } \overline{\Pi} \text{ is } \frac{\delta}{2 \cdot 9} - \text{ far from any accepting local view because all accepting local views consist of } q \text{ strings in the code.}

(\text{b) The "correction" of } \overline{\Pi} \text{ is the string } \overline{\Pi} = (\overline{\Pi}_1, ..., \overline{\Pi}_e \text{ where } \overline{\Pi}_i \text{ is closest codeword to } \overline{\Pi}_i \in \{0,1\}^N \text{ (breaking ties arbitrarily) and its decoding is } \pi = (\pi_1, ..., \pi_e) = (\text{Enc}(\overline{\Pi}_1), ..., \text{Enc}(\overline{\Pi}_e)) \in \Sigma^e. \text{ By the soundness of } V; \quad P_p \left[ V \cdot \overline{\Pi}(x) = 1 \right] = P_p \left[ V \cdot \pi(x) = 1 \right] \leq \varepsilon_5.

Whenever every string in $\overline{\Pi}$ is $\delta' -$ close to the code, then $\overline{\Pi}$ is the only string of $q$ codewords that is $\frac{\delta}{2 \cdot 9}$ - close to $\overline{\Pi}$. So

$$P_p \left[ \Delta(\overline{\Pi}, \pi_{\text{r}}(x,p), R(V_r)(x,p)) \leq \frac{\delta}{2 \cdot 9} | E \right] \leq P_p \left[ \overline{\Pi} \in R(V_r)(x,p) \right] = P_p \left[ V \cdot \overline{\Pi}(x) = 1 \right]. \hspace{2cm} \blacksquare$$
Bundling Queries

We are given a PCP system over a small alphabet, e.g., $\Sigma = \{0, 1\}$.

Goal: a new PCP system with $O(1)$ queries over a larger alphabet, e.g., $\Sigma' = \Sigma^{\omega_9}$.

We restrict our attention to non-adaptive PCPs: $V^\pi(x; \rho) = D(x, \rho, \Pi[Q(x, \rho)])$

Idea: write answers to query sets as large symbols and add a consistency test

The new PCP is $\Pi_b := (\pi: [Q] \to \Sigma, (\alpha_p: [Q] \to \Sigma)_{p \in \{0, 1\}^r})$

The new PCP verifier $V_b(x)$ is:

1. Sample $p \in \{0, 1\}^r$ and $i \in [q]$.
2. Read $\alpha_p \in \Sigma^q$ and $\Pi[Q(x, \rho)[i]]$. 
3. Check that $\alpha_p[i] = \Pi[Q(x, \rho)[i]]$. 
4. Check that $V(x; \rho) = 1$ when answering $j$-th query with $\alpha_p[i]$.

Lemma: PCP $[E_6, \Sigma, e, q, r] \leq$ PCP $[E_5' = 1 - (1 - \omega_9)^{1/9}, \Sigma' = \Sigma^{\omega_9}, l' = O(l + 2q'), q' = 2, r' = r + \log q]$ does not suffice for us: we need constant soundness error even when $q$ is super constant.
Bundling Queries

Fix a field $F$, subset $H \subseteq F$, and number of variables $m \in \mathbb{N}$. Assume that $|H| > \max\{|\mathbb{Z}|, q\}$. We identify $[q]$ with $H^m$ by setting $m := \log q / \log |H|$. Similarly, we can view a query set $\alpha(x, p) \subseteq [q]$ as $q$ elements of $H^m$.

Two changes from prior approach:

- replace $\pi: [q] \to \Sigma$ with its $(F, H, m)$-extension $\hat{\pi}: F^m \to F$
- replace $\alpha_p: [q] \to \Sigma$ with $\hat{\alpha}_p(z) := \hat{\pi}(Q_p(z)) \in F^{\langle q \cdot m \cdot H^m \rangle}$, where $Q_p: F \to F^m$ is $m$ polynomials of degree less than $q$ s.t. $\forall \ j \in [q] \ \ Q_p(j) = \alpha(x, p)[j] \in H^m$.

(Here we use $1, 2, \ldots, q$ to denote any $q$ distinct elements in $F$.)

Warm up:

$P_b(x) := (\hat{\pi}: F^m \to F, (\hat{\alpha}_p(z))_{p \in \{0, 1\}^r})$

$V_b(x)$

1. Sample $p \in \{0, 1\}^r$ and $y \in F$.
2. Read $\hat{\alpha}_p \in F[z]$ and $\hat{\pi}(Q_p(z))$.
3. Check that $\hat{\alpha}_p(y) = \hat{\pi}(Q_p(y))$.
4. Check that $V(x; y) = 1$ when answering $j$-th query with $\hat{\alpha}_p(j)$. 

Bundling Queries

\[ \Pi_b := (\tilde{\pi}, \tilde{\alpha}_p(z) \rho \in \Sigma_{0,1}^r) \]

\[ V_b(x) \]

1. Sample \( \rho \in \Sigma_{0,1}^r \) and \( \tilde{\pi} \in \Pi \).
2. Read \( \tilde{\alpha}_p \in \Pi \) and \( \tilde{\pi}(Q_p(z)) \).
3. Check that \( \tilde{\alpha}_p(\tilde{\pi}) = \tilde{\pi}(Q_p(z)) \).
4. Check that \( V(x; \rho) = 1 \) when answering \( j \)-th query with \( \tilde{\alpha}_p(j) \).

For this warm-up case (\( \tilde{\pi} \) is guaranteed to be the low degree extension), we can prove:

**Claim:** The soundness error is at most \( 1 - (1 - \varepsilon_5) \cdot \left( 1 - \frac{g_m \|h\|_1}{1 + f} \right) \). [Improving on \( 1 - (1 - \varepsilon_5) \cdot \frac{1}{q} \)]

**Proof:** Suppose that \( x \in L \) and fix a PCP \( \Pi_b = (\tilde{\pi}, (\tilde{\alpha}_p)_{\rho \in \Sigma_{0,1}^r}) \).

\[ \Pr[V_b(x) = 1] = \Pr_{\rho, z}[D(x, \rho, \tilde{\alpha}_p(z), \tilde{\alpha}_p(z)); 1] \land \tilde{\alpha}_p(\tilde{\pi}) = \tilde{\pi}(Q_p(z)) \]

\[ = 1 - \Pr_{\rho, z}[D(x, \rho, \tilde{\alpha}_p(z), \tilde{\alpha}_p(z)); 0] \land \tilde{\alpha}_p(\tilde{\pi}) \neq \tilde{\pi}(Q_p(z))] \leq 1 - (1 - \varepsilon_5) \cdot \left( 1 - \frac{g_m \|h\|_1}{1 + f} \right) \]

**Problem:** How to handle the case where \( \tilde{\pi} \) is noisy?

The LDT that we say makes \( w(i) \) queries to \( f \). Moreover, the query \( Q_p(z) \) to \( f \) is biased.
Bundling Queries

• We rely on a large-alphabet constant-query low-degree test:

\[ \text{Theorem [line vs. point test]} \quad \forall f : \mathbb{F}^m \rightarrow \mathbb{F} \quad \forall \text{ lines oracle \( \{ \hat{a}_{0,1}, \hat{a}_{0,2} : \mathbb{F}^m \rightarrow \mathbb{F} \} \) of degree \( \Delta \)} \]

\[ \text{if } f \text{ is } \delta\text{-far from total degree } \Delta \text{ then } \operatorname{Pr}_{a,b \in \mathbb{F}^m, \mu \in \mathbb{F}} \{ f(\alpha + \beta) = \hat{a}_{0,1}(\mu) \} \leq \varepsilon_{\text{EUF}}(\delta) \]

• We randomize the query to \( \hat{f} \): \( \tilde{Q}_{\rho, \nu}(z) \) is the polynomial s.t. \( \tilde{Q}_{\rho, \nu}(q+1) = \nu \) for random \( \nu \in \mathbb{F}^m \), so that \( \forall \nu \in \mathbb{F} \setminus \{ q \} \), \( \tilde{Q}_{\rho, \nu}(z) \) is random in \( \mathbb{F}^m \)

Here is the final construction:

\( P_0(x) \)

1. \( \Pi := P(x) \in \mathbb{F} \mathbb{F}^m \)

2. \( \hat{\Pi} : \mathbb{F}^m \rightarrow \mathbb{F} \) is \((\mathbb{F}, \mathbb{F}, m)\)-extension of \( \Pi \)

3. For \( p \in \{ 0, 1 \} \) and \( \nu \in \mathbb{F}^m \):
   \( \hat{Q}_{\rho, \nu}(z) := \hat{f}(Q_{\rho, \nu}(z)) \)

4. For \( a, b \in \mathbb{F}^m \):
   \( \hat{a}(z) := \hat{f}(az + b) \)

5. Output \( \Pi_b := (\hat{f}, \hat{Q}_{\rho, \nu})_{p \in \{ 0, 1 \}} \bigcup (a_{ab} \in \mathbb{F}^m) \)

\( V_0(x) \)

1. Sample \( a, b \in \mathbb{F}^m \) and \( \mu \in \mathbb{F} \), and check \( \hat{a}_{0,1}(\mu) \neq \hat{f}(\alpha + \beta) \).

2. Sample \( p \in \{ 0, 1 \} \) and \( \nu \in \mathbb{F} \setminus \{ q \} \), and check \( \hat{Q}_{\rho, \nu}(z) \neq \hat{f}(Q_{\rho, \nu}(z)) \).

3. Check that \( V(x; \rho) = 1 \) when answering \( j \)-th query with \( \hat{Q}_{\rho}(j) \).

One can prove that the new error is

\[ \varepsilon' = \max \{ \varepsilon_{\text{EUF}}(\delta), 1 - (1 - \varepsilon) \left( 1 - \frac{m \mathbb{M}^H}{\mathbb{F}^1 - q} \right)^\delta \} \]

Lemma: \( \text{PCP} [\varepsilon, \Sigma, \ell, q, r] \subseteq \text{PCP} [\varepsilon', \Sigma' = \Sigma' = q^m \mathbb{M}^H, \ell' = \ell + 2^{m} \mathbb{M}^H + \mathbb{M}^H, q' = O(1), r' = r + O(\log \mathbb{M}^H)] \)

We can then set \( \mathbb{M}^H = \log \ell \), \( m = \frac{\log \mathbb{M}^H}{\log \ell} = \frac{\log \ell}{\log \log \ell} \), \( \mathbb{M}^H = O(q^m \mathbb{M}^H) \) for the regime of interest.