Lecture 7

The PCP theorem and hardness of approximation

Scribe: Thomas Vidick

The PCP theorem was proved as the result of a long, intense line of work in the 90s exploring the power of interaction. Since then the theorem has been given a second proof, and has found many applications, in particular to hardness of approximation. In this lecture we give two equivalent formulations of the theorem.

7.1 Proof checking version

Our first formulation gives the theorem its name (PCP = Probabilistically Checkable Proof). It states that, provided that one is willing to settle for a probabilistic decision process that errs with small probability, all languages in NP have proofs that can be verified very efficiently: only a constant number of symbols of the proof need to be evaluated! To state this formally we first need the notion of a PCP verifier.

Definition 7.1 (PCP verifier)**.** Let $r, q : \mathbb{N} \to \mathbb{N}$ and let $\Sigma = (\Sigma_n)_{n\in\mathbb{N}}$ be a sequence of finite sets.[1] A $(r,q)_\Sigma -$ PCP verifier V is a probabilistic polynomial-time algorithm that takes as input a string $x \in \{0,1\}^*$ and has access to a special tape containing a *proof* $\Pi \in \Sigma_n^*$, where $n = |x|$ is the length of x. The verifier flips at most $r(n)$ random coins, makes at most $q(n)$ queries to the proof, and outputs a single "accept" or "reject" symbol. Given the specification of a verifier V and $x \in \{0,1\}^*$ we let $\omega(V_x)$ be the maximum acceptance probability of V, when its input is x and the maximum is taken over all possible proofs $\Pi \in \Sigma^*$.

We define an associated complexity class:

Definition 7.2. Let $c, s : \mathbb{N} \to [0,1]$ be arbitrary (computable) functions and r, q, Σ as in Definition 7.1. A language L is in $\mathrm{PCP}_{c,s}[r,q]_\Sigma$ if there exists a $(r,q)_\Sigma -$ PCP verifier V such that, for every $x \in \{0,1\}^*$:

- (Completeness.) If $x \in L$ then $\omega(V_x) \geq c$, i.e. there exists a proof Π such that V_x accepts Π with probability at least c,
- (Soundness.) If $x \notin L$ then $\omega(V_x) \leq s$, i.e. for every proof Π, V_x accepts Π with probability at most s.

When $c = 1$, $s = 1/2$ and $\Sigma = \{0,1\}$ we use the shorthand $\mathrm{PCP}[r,q]$ for $\mathrm{PCP}_{c,s}[r,q]_\Sigma$.

[1] Most often we will have $\Sigma_n = \Sigma$ a constant-sized set for all n.

The PCP theorem states that all languages in NP have very efficient verifiers, namely ones that only read a *constant* number of bits from the proof:

Theorem 7.3 (PCP, proof-checking variant)**.** *The inclusion* $\mathrm{NP} \subseteq \mathrm{PCP}(O(\log n), O(1))$ *holds. Equivalently, given a* $(O(\log n), O(1))_{\{0,1\}}$*-PCP verifier V the problem of deciding between $\omega(V_x) = 1$ and $\omega(V_x) \leq 1/2$ is* NP*-hard.*

7.2 Constraint satisfaction problems

Our second formulation of the PCP theorem uses the language of constraint satisfaction problems, and is the most useful for applications to hardness of approximation.

Definition 7.4 (CSP)**.** Let $m, q : \mathbb{N} \to \mathbb{N}$ and let $\Sigma = (\Sigma_n)$ be a sequence of finite sets. A $(m,q)_\Sigma - \mathrm{CSP}$ φ is a collection of m constraints C_j, each acting on at most q out of a total of n variables. If $m = \mathrm{poly}(n)$ and $\Sigma = \{0,1\}$ we refer to $(m,q)_\Sigma - \mathrm{CSPs}$ as $q - \mathrm{CSPs}$.

Given a CSP φ, its value $\omega(\varphi)$ is the maximum fraction of constraints that an assignment can satisfy:

$$\omega(\varphi) \;=\; \max_{x_1,\dots,x_n} \frac{\#\{j :\; C_j \text{ is satisfied by } (x_i)\}}{m}.$$

The Cook-Levin theorem states that 3-SAT is NP-complete. In other words, the problem of deciding whether $\omega(\varphi) = 1$ or $\omega(\varphi) \leq 1 - 1/m$ for $(m,3)_{\{0,1\}} - \mathrm{CSPs}$ is NP-complete. The PCP theorem shows that an a priori much easier, approximation version of this problem remains just as hard:

Theorem 7.5 (PCP, CSP variant)**.** *For any $L \in$ NP there exists a polynomial-time mapping $x \in \{0,1\}^* \mapsto \varphi_x$ from strings x to $(m,q) - \mathrm{CSP}$s φ_x, where $m = \mathrm{poly}(n)$ and $q = O(1)$, such that $x \in L \implies \omega(\varphi_x) = 1$ and $x \notin L \implies \omega(\varphi_x) \leq 1/2$. Equivalently, there exists a constant q such that given a $q - \mathrm{CSP}$ φ the problem of deciding between $\omega(\varphi) = 1$ and $\omega(\varphi) \leq 1/2$ is* NP*-complete.*

With some more work, Hastad showed the following:

Theorem 7.6. *Let $\varepsilon > 0$. Given a* 3*-SAT formula φ, it is* NP*-hard to distinguish between $\omega(\varphi) = 1$ and $\omega(\varphi) \leq 7/8 + \varepsilon$.*

The theorem is optimal in the sense that given any 3-SAT formula there *always* exists an assignment satisfying a fraction $7/8$ of the clauses. To see this, note that a randomly chosen assignment satisfies any 3-SAT clause with probability exactly $7/8$. By linearity of expectation, the expected fraction of clauses satisfied by a random assignment is $7/8$, hence there must exist an assignment satisfying as many clauses.

7.3 Equivalence between the two versions

We show the implication Theorem 7.3 $\implies$ Theorem 7.5 in detail. Let L be any language in NP and $x \in \{0,1\}^*$. By Theorem 7.3 we know that there exists a PCP verifier $V = V_x$ that flips $r = r(|x|)$ random coins, makes $q = q(|x|)$ queries to a proof Π, and is such that, if $x \in L$ then there exists a proof Π that V_x accepts with probability 1, whereas if $x \notin L$ any proof Π is accepted by V_x with probability at most $1/2$.

Consider the following constraint satisfaction problem φ_x. φ_x has as many variables as there are locations in the proof Π_x, which without loss of generality is at most $q2^r = \mathrm{poly}(|x|)$. For each possible string $R \in \{0,1\}^r$ we introduce a constraint C_R which corresponds to the check performed by V_x on this choice of random bits. Precisely, whenever the verifier's random bits correspond to R, he looks up q entries of the proof Π and accepts if and only if these entries satisfy a certain constraint. The constraint C_j is defined to evaluate to 1 if and only if the variables on which C_j acts are assigned values that would have made the verifier V_x accept, if the entries of Π he looks up were assigned matching values.

If $x \in L$ we know that there exists a proof Π that is accepted by V_x. In other words, there is an assignment to the variables of φ_x (the corresponding entries of Π) that satisfies all the clauses (the verifier's test on any possible random string). Hence $\omega(\varphi_x) = 1$.

If $x \notin L$ then every proof Π is rejected by V_x with probability at least $1/2$. Suppose for contradiction that $\omega(\varphi_x) > 1$. This means that there would exist an assignment to all variables that satisfies strictly more than $1/2$ of the constraints. We can define a corresponding proof Π that lists all the variables' values. By definition of φ_x from V_x the proof Π would satisfy strictly more than $1/2$ of the verifier's possible tests, contradicting our assumption that V_x rejects any proof with probability at least $1/2$. Hence it must be that $\omega(\varphi_x) \leq 1/2$, which concludes the proof.

The reverse implication is not hard to see. The PCP verifier will choose a clause uniformly at random, query all variables that appear in it, and accept if and only if it is satisfied. We omit the details.

7.4 A "toy" version of the PCP Theorem

The original PCP Theorem in its proof-checking version demonstrates that for any $L \in$ NP there exists a verifier that uses only $O(\log(n))$ random bits, queries only a constant number of positions in the proof, and correctly answers the question $x \in L$? with constant probability. A simpler version only requires the number of random bits to be polynomial in the input size:

Theorem 7.7. NP $\subseteq$ PCP$(r = O(poly(n)), q = O(1))$.

This version has an exponential blowup in the maximal proof size that is $O(2^{poly(n)})$ compared to $O(poly(n))$ from the original PCP Theorem. Despite being a weaker result, it will allow us to demonstrate tools and ideas used in the original version.

In order to prove Theorem 7.7, we use the NP-complete problem "Quadratic Equations" (**QUADEQ**) that is defined next.

Definition 7.8. (**QUADEQ**) An instance φ of **QUADEQ** is given by m constraints C_j over n boolean variables x_i of the form:

$$C_j : \sum_i \alpha_i^{(j)} x_i + \sum_{i,k} \beta_{i,k}^{(j)} x_i x_k \equiv \gamma^{(j)} \mod 2,$$

or equivalently

$$\alpha^{(j)} \cdot x + \beta^{(j)} \cdot (x \otimes x) \equiv \gamma^{(j)} \mod 2,$$

where

$$x = (x_i)_{i=1,\dots,n} \in \{0,1\}^n,$$
$$\alpha^{(j)} = (\alpha_i^{(j)})_{i=1,\dots,n} \in \{0,1\}^n,$$
$$\beta^{(j)} = (\beta_{ik}^{(j)})_{i,k=1,\dots,n} \in \{0,1\}^{n^2} \text{ and}$$
$$\gamma^{(j)} \in \{0,1\}.$$

The instance φ belongs to **QUADEQ** if and only if there is an assignment x that satisfies all constraints.

The following is an example of a **QUADEQ** instance.

$$\begin{cases} C_1: & x_1 + x_2 + x_4x_5 + x_2x_7 \equiv 1 \mod 2 \\ C_2: & x_7 + x_1x_2 \equiv 0 \mod 2 \\ \vdots & \\ C_m: & x_9 + x_5x_6 \equiv 1 \mod 2 \end{cases} \tag{7.1}$$

The goal is to describe a PCP verifier for **QUADEQ** and as we advance some tools are established. The first such tool is a test that fails with probability $\frac{1}{2}$ if a **QUADEQ** instance φ is infeasible, and always accepts otherwise.

Given coefficients $a = (a_i)_{i=1,\ldots,m} \in \{0,1\}$ chosen independently and uniformly at random, form an equation by combining the constraints of φ as follows:

$$\mathrm{E} = \mathrm{E}(a) : \sum_j a_j(\alpha^{(j)} \cdot x + \beta^{(j)} \cdot (x \otimes x)) = \sum_j a_j \gamma^{(j)}.$$

Claim 7.9. *For a uniformly random choice of the coefficients a, it holds:*

(i) If x satisfies all constraints, then x satisfies $\mathrm{E}(a)$,

(ii) If x does not satisfy all constraints, $\Pr_a[x \textit{ satisfies } E(a)] \leq \frac{1}{2}$.

Proof. Item (i) is clear, as any assignment that satisfies all equations individually must also satisfy the sum.

For item (ii), we introduce the error vector given by

$$e = \begin{pmatrix} \alpha^{(1)} \cdot x + \beta^{(1)} \cdot (x \otimes x) - \gamma^{(1)} \\ \vdots \\ \alpha^{(m)} \cdot x + \beta^{(m)} \cdot (x \otimes x) - \gamma^{(m)} \end{pmatrix} \tag{7.2}$$

Since not all the constraints of φ are satisfiable, the vector e has at least one not zero component. Note that the inner product of the random vector a with the error vector e checks the parity of the elements a_i for which $e_i = 1$. As the elements a_i are drawn independently and uniformly at random this parity is 1 with probability exactly $\frac{1}{2}$. Moreover, the probability that x does not satisfy E is $\Pr_a[e \cdot a = 1] \leq \frac{1}{2}$. $\square$

Now, We are ready for our first attempt to solve the simplified PCP Theorem 7.7. We assume that the verifier has access to a proof $\Pi = (\Pi^1, \Pi^2)$ where $\Pi^1 \in \{0,1\}^{2^n}$ and $\Pi^2 \in \{0,1\}^{2^{n^2}}$. Ideally, we would like to have Π to be composed of

- $(\Pi^1)_\alpha = \alpha \cdot x$, and
- $(\Pi^2)_\beta = \beta x \cdot (\otimes x)$.

for some $x \in \{0,1\}^n$.

In words, the proof Π^1 encodes in each position α the value of the inner product with a fixed x (similarly to Π^2). If $\varphi \in$ **QUADEQ**, the bit string x would be the satisfying assignment.

It is important to note that all combination of the constraints given by any random a will lead to a new value for α and β whose inner product with x is encoded in Π. This is a key point that allows us to use Claim 4. A first attempt at designing a verifier for **QUADEQ** is given below.

```
1 Choose $a = (a_i)_{i=1,\dots,m} \in \{0,1\}$ uniformly at random ;
2 Compute $\begin{cases} \alpha = \sum_j a_j \alpha^{(j)} \in \{0,1\}^n \\ \beta = \sum_j a_j \beta^{(j)} \in \{0,1\}^{n^2} \\ \gamma = \sum_j a_j \gamma^{(j)} \in \{0,1\} \end{cases}$ ;
3 Make two queries $(\Pi^1)_\alpha$ and $(\Pi^2)_\beta$ ;
4 Accept iff $(\Pi^1)_\alpha + (\Pi^2)_\beta = \gamma$ ;
```
Algorithm 1: Verifier V for **QUADEQ**

The problem of this verifier is that it expects the proof to be in a particular format. Provided this is the case, it follows from Claim that the verifier V has completeness 1 and soundness at most $\frac{1}{2}$. However, it can not rely on receiving this exact format, or otherwise the system may loose its constant soundness as the proof Π is given by an adversarial prover.

The proofs Π^1 and Π^2 should encode the evaluation of a linear function (the inner product with a fixed x, or $x \otimes x$) over all possible inputs. Fortunately, this is a strong property that we can exploit to ensure that Π is "close" to having the desired format. For this, we devise a linearity test that has oracle access to a function $f : \{0,1\}^n \to \{0,1\}$ and whose goal is to check that f is linear. (By linearity we mean that there is $c \in \{0,1\}^n$ such that $f(\alpha) = c_1\alpha_1 + \cdots + c_n\alpha_n \mod 2 = c \cdot \alpha$ for every α.)

Testing if f is exact linear would require querying its value on all inputs. Nevertheless, the next simple test can enforce that it is "almost" linear.

```
1 Choose $\alpha, \alpha' \in \{0,1\}^n$ at random;
2 Query $f(\alpha), f(\alpha'), f(\alpha + \alpha')$;
3 Accept iff $f(\alpha + \alpha') = f(\alpha) + f(\alpha')$;
```
Algorithm 2: BLR Linearity Test

The next theorem makes precise our notion of "almost linear". If the linearity test succeeds with high probability, f agrees with a single linear function on a large fraction of inputs.

Theorem 7.10 (BLR)**.** *The BLR linearity test satisfies:*

(i) If f is linear, then $\Pr[f \textit{ passes BLR test}] = 1$.

(ii) Suppose $\Pr[f \textit{ passes BLR test}] \geq 1 - \epsilon$ *for some* $\epsilon > 0$*, then there is a coefficient vector c such that $f(\alpha) = c \cdot \alpha$ for $1 - \epsilon$ fraction of $\alpha \in \{0,1\}^n$.*

Assuming the theorem, which is a classic in complexity theory but whose proof we do not include, we conclude the description of our PCP verifier for **QUADEQ**. Recall that the a correct proof $\Pi = (\Pi^1, \Pi^2)$ must consist of $(\Pi^1)_\alpha = \alpha \cdot x$ and $(\Pi^2)_\beta = \beta \cdot (x \otimes x)$ for some satisfying assignment x of the given **QUADEQ** instance φ.

To verify the proof Π, the verifier performs the each of the following with probability $1/4$.

1. Linearity test on Π^1. That is, we use Theorem 7.11 with $f = (-1)^{\Pi^1}$. This is equivalent to querying $(\Pi^1)_\alpha$, $(\Pi^1)_{\alpha'}$ and $(\Pi^1)_{\alpha+\alpha'}$ for $\alpha, \alpha' \in_u \{0,1\}^n$, and testing whether $(\Pi^1)_\alpha + (\Pi^1)_{\alpha'} = (\Pi^1)_{\alpha+\alpha'}$.

2. Linearity test on Π^2

3. Consistency of tensor product: Choose $\alpha, \alpha' \in \{0,1\}^n$, $\beta \in \{0,1\}^{n^2}$ uniformly at random. Then check $(\Pi^1)_\alpha (\Pi^1)_{\alpha'} = (\Pi^2)_{(\alpha\otimes\alpha')+\beta} + (\Pi^2)_\beta$.

4. As in the previous lecture, construct an equation at random, say with coefficients (α, β, γ), and check $(\Pi^1)_\alpha + (\Pi^2)_\beta = \gamma$.

We will skip the detailed analysis of this verifier. But, we have presented all the required elements to establish the correctness and soundness of this verifier. Overall, if φ is satisfiable then there exists a proof Π that is accepted with probability 1. On the other hand, if φ is not satisfiable then any proof is rejected with probability at least 0.001 (the exact constant is not relevant here). This implies that we have soundness at most 0.999. Repeating the verifier's test a large enough (constant) number of times to amplify the soundness, we obtain Theorem 7.7: NP $\subseteq$ PCP$_{1,1/2}(O(poly(n)), O(1))$. Note that the length of the proof Π is $2^n + 2^{n^2}$ bits, and the number of random bits required by the verifier is at most $2 + 2n^2$.

Recall that the PCP theorem is a stronger version of Theorem 7.7 that claims that NP $\subseteq$ PCP$(O(\log n), O(1))$. In order to get to the PCP theorem we need to save on randomness, or equivalently make the proof much shorter. We can accomplish this by extending our framework to low-degree polynomials over $\mathbb{F}_p$.

7.5 Proof of Theorem 7.10

For completeness, we include a proof of Theorem 7.10. This material was not covered in class. For convenience, we restate the theorem here using slightly different notation.

Theorem 7.11 (BLR). *Suppose a function $f : \{\pm 1\}^n \to \{\pm 1\}$ satisfies*

$$\Pr_{\alpha,\alpha' \in_U \{\pm 1\}} \left[f(\alpha) f(\alpha') = f(\alpha\alpha') \right] = 1 - \varepsilon,$$

where the product $\alpha\alpha'$ is taken componentwise and $\in_U$ means that α, α' are chosen uniformly at random. Then there exists a set $S \subseteq [n]$ such that $f(\alpha) = \prod_{i \in S} \alpha_i$ for at least a fraction $1 - \varepsilon$ of all α.

The connection between the theorem and the linearity test is obtained by setting $f(\alpha) = (-1)^{\Pi}_{\alpha}$. Now, if f satisfies $f(\alpha) = \prod_{i \in S} \alpha_i$ then proof Π satisfies the required linear form, i.e., $\Pi_\alpha = \sum_{i \in S} \alpha_i = c_S \cdot \alpha$, where c_s is the indicator vector for set S. To prove the BLR theorem we will need a little bit of Fourier analysis over $\mathbb{F}_2^n$.

Basics of Fourier analysis over $\mathbb{F}_2^n$

In this section we present some basic definitions and results from Fourier analysis over $\mathbb{F} = (\{\pm 1\}, \times)$. These results are used in the next section to prove Theorem 7.11.

Note that the set $V = \{f : \{\pm 1\}^n \to \mathbb{R}\}$ is a vector space. The canonical basis for this vector space consists of the delta functions δ_x. Specifically, for all $x \in \{\pm 1\}^n$ write

$$\begin{aligned} \delta_x : y \mapsto 1 &\quad \text{if } y = x \\ 0 &\quad \text{otherwise.} \end{aligned}$$

We have a natural inner product in this vector space that is defined as follows:

$$\langle f, g \rangle := \frac{1}{2^n} \sum_{x \in \{\pm 1\}^n} f(x) g(x).$$

In addition to the canonical basis, for this vector space of functions we have a *Fourier* basis that consists of the following set of functions $\{\chi_S : x \mapsto \prod_{i \in S} x_i \mid S \subseteq [n]\}$. Note that this basis is orthonormal with respect to the inner product $\langle , \rangle$:

$$\begin{aligned}\langle \chi_S, \chi_T \rangle &= \frac{1}{2^n} \sum_x \chi_S(x)\chi_T(x) \\ &= \frac{1}{2^n} \sum_x \prod_{i\in S} x_i \prod_{i \in T} x_i \\ &= \frac{1}{2^n} \sum_x \chi_{S\Delta T}(x), \qquad \text{(since } x_i \in \{\pm 1\}, x_i^2 = 1).\end{aligned}$$

where $S\Delta T$ denotes the symmetric difference between the sets S and T. For $S \neq T$ we have $\sum_x \chi_{S\Delta T}(x) = 0$; since we are summing over all $x \in \{\pm 1\}^n$. For $S = T$, the inner product $\langle \chi_S, \chi_S \rangle$ is equal to one. Hence, we get orthonormality. In particular, the characters χ_S form a complete basis for our vector space V, and every function $f \in V$, $f : \{\pm 1\} \to \mathbb{R}$ can be decomposed as

$$f(x) = \sum_{S \subseteq [n]} \hat{f}(S) \cdot \chi_S(x),$$

where $\hat{f}(S) \in \mathbb{R}$ is called the Fourier coefficient at S. Note that by orthogonality of the Fourier basis we have $\hat{f}(S) = \langle f, \chi_S \rangle$. Next we state a useful property of the Fourier coefficients.

Proposition 7.12. *For any functions $f, g : \{\pm 1\} \to \mathbb{R}$ we have $\langle f, g \rangle = \sum_S \hat{f}(S)\hat{g}(S)$.*

This proposition implies a useful corollary—called Parseval's identity—for ± 1 valued functions. Note that when f is ± 1 valued, the inner product $\langle f, f \rangle = 1$. This along with the proposition above, $\langle f, f \rangle = \sum_S \hat{f}(S)^2$, gives us Parseval's identity:

Corollary 7.13. *For any ± 1-valued function f, we have $\sum_S \hat{f}(S)^2 = 1$.*

Proof of Theorem 7.11

Using Fourier analytic tools we will now prove Theorem 7.11. The assumption of the theorem implies that

$$\begin{aligned}1 - 2\varepsilon &= \frac{1}{2^{2n}} \sum_{\alpha, \alpha'} f(\alpha) f(\alpha') f(\alpha\alpha') \\ &= \frac{1}{2^{2n}} \sum_{\alpha\alpha'} \sum_{S,T,U} \hat{f}(S)\hat{f}(T)\hat{f}(U)\chi_S(\alpha)\chi_T(\alpha')\chi_U(\alpha\alpha') \qquad \text{(via Fourier expansion)} \\ &= \frac{1}{2^{2n}} \sum_{\alpha\alpha'} \sum_{S,T,U} \hat{f}(S)\hat{f}(T)\hat{f}(U)\chi_S(\alpha)\chi_T(\alpha')\chi_U(\alpha)\chi_U(\alpha')\end{aligned}$$

Here, the last equality follows from the fact that $\chi_U(\alpha\alpha') = \chi_U(\alpha)\chi_U(\alpha')$; this equality is obtained just from the definition of χ_U. Rewriting the right-hand-side of the above equation we obtain:

$$\begin{aligned}1 - 2\varepsilon &= \sum_{S,T,U} \hat{f}(S)\hat{f}(T)\hat{f}(U) \left(\frac{1}{2^n} \sum_\alpha \chi_S(\alpha)\chi_T(\alpha) \right) \left(\frac{1}{2^n} \sum_\alpha \chi_T(\alpha')\chi_U(\alpha') \right) \\ &= \sum_{S,T,U} \hat{f}(S)\hat{f}(T)\hat{f}(U) \langle \chi_S, \chi_T \rangle \, \langle \chi_T, \chi_U \rangle\end{aligned}$$

The orthogonality of χs implies that only the summands where $S = T = U$ are non-zero, in fact are equal to one. The remaining summands, where either $S \neq T$ or $T \neq U$, are equal to zero. Therefore, the following inequality holds:

$$\begin{aligned} 1 - 2\varepsilon &\leq \sum_U \hat{f}(U)^3 \\ &\leq \max_W \hat{f}(W) \sum_U \hat{f}(U)^2 \\ &= \max_W \hat{f}(W) \qquad \text{(by Parseval's identity)}. \end{aligned}$$

Write $S_0 = \arg\max_S \hat{f}(S)$. By the inequality above, $\hat{f}(S_0) \geq 1 - 2\varepsilon$.

We can now prove Theorem 7.11. Recall that the Fourier coefficient $\hat{f}(S_0) = \frac{1}{2^n} \sum_\alpha f(\alpha)\chi_{S_0}(\alpha)$. But, the inequality $\hat{f}(S_0) \geq 1 - 2\varepsilon$ holds iff $f(\alpha) = \chi_{S_0}(\alpha)$ for at least $(1-\varepsilon)2^n$ α's; both f and χ_{S_0} are ± 1 valued. In other words, for at most $\varepsilon 2^n$ many α's for f and χ_{S_0} differ. This completes the proof of the theorem.