Computational Complexity: A Modern Approach

Sanjeev Arora and Boaz Barak
Princeton University

http://www.cs.princeton.edu/theory/complexity/

complexitybook@gmail.com

Chapter 22

Proofs of PCP Theorems and the Fourier Transform Technique

We saw in Chapter 11 that the **PCP** Theorem implies that computing approximate solutions to many optimization problems is **NP**-hard. This chapter gives a complete proof of the **PCP** Theorem. In Chapter 11 we also mentioned that the **PCP** Theorem does not suffice for proving several other similar results, for which we need stronger (or simply different) "**PCP** Theorems". In this chapter we survey some such results and their proofs. The two main results are Raz's *parallel repetition theorem* (see Section 22.3) and Håstad's 3-bit **PCP** theorem (Theorem 22.16). Raz's theorem leads to strong hardness results for the 2CSP problem over large alphabets. Håstad's theorem shows that certificates for **NP** languages can be probabilistically checked by examining only 3 bits in them. One of the consequences of Håstad's result is that computing a $(7/8 + \epsilon)$-approximation for the MAX-3SAT problem is **NP**-hard for every $\epsilon > 0$. Since we know that 7/8-approximation is in fact possible in polynomial time (see Example 11.2 and Exercise 11.3), this shows (assuming $\mathbf{P} \neq \mathbf{NP}$) that the approximability of MAX-3SAT has an abrupt transition from easy to hard at 7/8. Such a result is called a *threshold* result, and threshold results are now known for a few other problems.

Håstad's result builds on the other results we have studied, including the (standard) **PCP** Theorem, and Raz's theorem. It also uses Håstad's method of analysing the verifier's acceptance probability using *Fourier transforms*. Such Fourier analysis has also proved useful in other areas in theoretical computer science. We introduce this technique in Section 22.5 by using it to show the correctness of the linearity testing algorithm of Section 11.5, which completes the proof of the result $\mathbf{NP} \subseteq \mathbf{PCP}(\text{poly}(n), 1)$ in Section 11.5. We then use Fourier analysis to prove Håstad's 3-bit **PCP** Theorem.

In Section 22.8 we prove the hardness of approximating the SET-COVER problem. In Section 22.2.3 we prove that computing $n^{-\epsilon}$-approximation to MAX-INDSET in **NP**-hard. In Section 22.9 we briefly survey other **PCP** Theorems that have been proved, including those that assume the so-called *unique games conjecture*.

22.1 Constraint satisfaction problems with non-binary alphabet

In this chapter we will often use the problem $q\mathsf{CSP}_W$, which is defined by extending the definition of qCSP in Definition 11.11 from binary alphabet to an alphabet of size W.

Definition 22.1 *($q\mathsf{CSP}_W$)* For integers $q, W \geq 1$ the $q\mathsf{CSP}_W$ problem is defined analogously to the qCSP problem of Definition 11.11, except the underlying alphabet is $[W] = \{1, 2, \ldots, W\}$ instead of $\{0, 1\}$. Thus constraints are functions mapping $[W]^q$ to $\{0, 1\}$.

For $\rho < 1$ we define $\_\mathsf{GAP}q\mathsf{CSP}W\rho$ analogously to the definition of ρ-GAPqCSP for binary alphabet (see Definition 11.13). ♢

Example 22.2
3SAT is the subcase of qCSP$_W$ where $q = 3$, $W = 2$, and the constraints are OR's of the involved literals.
Similarly, the **NP**-complete problem 3COL can be viewed as a subcase of 2CSP$_3$ instances where for each edge (i, j), there is a constraint on the variables u_i, u_j that is satisfied iff $u_i \neq u_j$. The graph is 3-colorable iff there is a way to assign a number in $\{0, 1, 2\}$ to each variable such that all constraints are satisfied.

22.2 Proof of the PCP Theorem

This section proves the PCP Theorem. We present Dinur's proof [Din06], which simplifies half of the original proof of [AS92, ALM$^+$92]. Section 22.2.1 gives an outline of the main steps. Section 22.2.2 describes one key step, Dinur's gap amplification technique. Section 22.2.5 describes the other key step, which is from the original proof of the **PCP** Theorem [ALM$^+$92] and its key ideas were presented in the proof of **NP** $\subseteq$ **PCP**(poly(n), 1) in Section 11.5.

22.2.1 Proof outline for the PCP Theorem.

As we have seen, the **PCP** Theorem is equivalent to Theorem 11.14, stating that ρ-GAPqCSP is **NP**-hard for some constants q and $\rho < 1$. Consider the case that $\rho = 1 - \epsilon$ where ϵ is not necessarily a constant but can be a function of m (the number of constraints). Since the number of satisfied constraints is always a whole number, if φ is unsatisfiable then val(φ) $\leq 1 - 1/m$. Hence, the gap problem (1–$1/m$)-GAP3CSP is a generalization of 3SAT and is **NP** hard. The idea behind the proof is to start with this observation, and iteratively show that (1–ϵ)-GAPqCSP is **NP**-hard for larger and larger values of ϵ, until ϵ is as large as some absolute constant independent of m. This is formalized in the following definition and lemma.

Definition 22.3 Let f be a function mapping CSP instances to CSP instances. We say that f is a *CL-reduction* (short for *complete linear-blowup reduction*) if it is polynomial-time computable and for every CSP instance φ, satisfies:

Completeness: If φ is satisfiable then so is $f(\varphi)$.

Linear blowup: If m is the number of constraints in φ then the new qCSP instance $f(\varphi)$ has at most Cm constraints and alphabet W, where C and W can depend on the arity and the alphabet size of φ (but not on the number of constraints or variables). $\diamondsuit$

Lemma 22.4 *(**PCP** Main Lemma)*
There exist constants $q_0 \geq 3$, $\epsilon_0 > 0$, and a CL-reduction f such that for every q_0CSP-instance φ with binary alphabet, and every $\epsilon < \epsilon_0$, the instance $\psi = f(\varphi)$ is a q_0CSP (over binary alphabet) satisfying

$$\mathsf{val}(\varphi) \leq 1 - \epsilon \Rightarrow \mathsf{val}(\psi) \leq 1 - 2\epsilon$$

Lemma 22.4 can be succinctly described as follows:

	Arity	Alphabet	Constraints	Value
Original	q_0	binary	m	$1-\epsilon$
	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
Lemma 22.4	q_0	binary	Cm	$1-2\epsilon$

This lemma allows us to easily prove the **PCP** Theorem.

Proving Theorem 11.5 from Lemma 22.4. Let $q_0 \geq 3$ be as stated in Lemma 22.4. As already observed, the decision problem q_0CSP is **NP**-hard. To prove the **PCP** Theorem we give a reduction from this problem to GAP q_0CSP. Let φ be a q_0CSP instance. Let m be the number of constraints in φ. If φ is satisfiable then $\mathsf{val}(\varphi) = 1$ and otherwise $\mathsf{val}(\varphi) \leq 1 - 1/m$. We use Lemma 22.4 to amplify this gap. Specifically, apply the function f obtained by Lemma 22.4 to φ a total of $\log m$ times. We get an instance ψ such that if φ is satisfiable then so is ψ, but if φ is not satisfiable (and so $\mathsf{val}(\varphi) \leq 1 - 1/m$) then $\mathsf{val}(\psi) \leq 1 - \min\{2\epsilon_0, 1 - 2^{\log m}/m\} = 1 - 2\epsilon_0$. Note that the size of ψ is at most $C^{\log m} m$, which is polynomial in m. Thus we have obtained a gap-preserving reduction from L to the $(1–2\epsilon_0)$-GAP q_0CSP problem, and the **PCP** theorem is proved. ■

The rest of this section proves Lemma 22.4 by combining two transformations: the first transformation amplifies the gap (i.e., fraction of violated constraints) of a given CSP instance, at the expense of increasing the alphabet size. The second transformation reduces back the alphabet to binary, at the expense of a modest reduction in the gap. The transformations are described in the next two lemmas.

Lemma 22.5 *(Gap Amplification* [Din06]*)*
For every $\ell, n \in \mathbb{N}$, there exist numbers $W \in \mathbb{N}, \epsilon_0 > 0$ and a CL-reduction $g_{\ell,q}$ such that for every qCSP instance φ with binary alphabet, the instance $\psi = g_{\ell,q}(\varphi)$ has arity only 2, uses alphabet of size at most W and satisfies:

$$\mathsf{val}(\varphi) \leq 1 - \epsilon \Rightarrow \mathsf{val}(\psi) \leq 1 - \ell\epsilon$$

for every $\epsilon < \epsilon_0$.

Lemma 22.6 *(Alphabet Reduction)*
There exists a constant q_0 and a CL- reduction h such that for every CSP instance φ, if φ had arity two over a (possibly non-binary) alphabet $\{0..W–1\}$ then $\psi = h(\varphi)$ has arity q_0 over a binary *alphabet and satisfies:*

$$\mathsf{val}(\varphi) \leq 1 - \epsilon \Rightarrow \mathsf{val}(h(\varphi)) \leq 1 - \epsilon/3$$

Lemmas 22.5 and 22.6 together imply Lemma 22.4 by setting $f(\varphi) = h(g_{6,q_0}(\varphi))$. Indeed, if φ was satisfiable then so will $f(\varphi)$. If $\mathsf{val}(\varphi) \leq 1-\epsilon$, for $\epsilon < \epsilon_0$ (where ϵ_0 the value obtained in Lemma 22.5 for $\ell = 6$, $q = q_0$) then $\mathsf{val}(g_{6,q_0}(\varphi)) \leq 1 - 6\epsilon$ and hence $\mathsf{val}(h(g_{6,q_0}(\varphi))) \leq 1 - 2\epsilon$. This composition is described in the following table:

	Arity	Alphabet	Constraints	Value
Original	q_0	binary	m	$1-\epsilon$
	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
Lemma 22.5 ($\ell = 6$, $q = q_0$)	2	W	Cm	$1-6\epsilon$
	$\Downarrow$	$\Downarrow$	$\Downarrow$	$\Downarrow$
Lemma 22.6	q_0	binary	$C'Cm$	$1-2\epsilon$

22.2.2 Dinur's Gap Amplification: Proof of Lemma 22.5

To prove Lemma 22.5, we need to exhibit a function g that maps a qCSP instance to a 2CSP$_W$ instance over a larger alphabet $\{0..W–1\}$ in a way that increases the fraction of violated constraints. In the proof verification viewpoint (Section 11.3), the fraction of violated constraints is merely the soundness parameter. So at first sight, our task merely seems to be reducing the "soundness" parameter of a **PCP** verifier, which as already noted (in the Remarks following Theorem 11.5) can be easily done by repeating the verifier's operation 2 (or more generally, k) times. The problem with this trivial idea is that the CSP instance corresponding to k repeated runs of the verifier is not another 2CSP instance, but an instance of arity $2k$ since the verifier's decision depends upon $2k$ different entries in the proof. In the next chapter, we will see another way of "repeating" the verifier's operation using *parallel repetition*, which does result in 2CSP instances, but greatly increases the size of the CSP instance. By contrast, here we desire a CL-reduction, which means the size must only increase by a constant factor. The key to designing such a CL-reduction is walks in expander graphs, which we describe separately first in Section 22.2.3 since it is of independent interest.

22.2.3 Expanders, walks, and hardness of approximating INDSET

Dinur's proof uses expander graphs, which are described in Chapter 21. Here we recap the facts about expanders used in this chapter, and as illustration we use them to prove a hardness result for MAX-INDSET.

In Chapter 21 we define a parameter $\lambda(G) \in [0,1]$, for every regular graph G (see Definition 21.2). For every $c \in (0,1)$, we call a regular graph G satisfying $\lambda(G) \leq c$ a *c-expander graph*. If $c < 0.9$, we drop the prefix c and simply call G an *expander graph*. (The choice of the constant 0.9 is arbitrary.) As shown in Chapter 21, for every constant $c \in (0,1)$ there is a constant d and an algorithm that given input $n \in N$, runs in poly(n) time and outputs the adjacency matrix of an n-vertex d-regular c-expander (see Theorem 21.19).

The main property we need in this chapter is that for every regular graph $G = (V,E)$ and every $S \subseteq V$ with $|S| \leq |V|/2$,

$$\Pr_{(u,v)\in E}[u \in S, v \in S] \leq \frac{|S|}{|V|}\left(\frac{1}{2} + \frac{\lambda(G)}{2}\right) \tag{1}$$

(Exercise 22.1) Another property we use is that $\lambda(G^\ell) = \lambda(G)^\ell$ for every $\ell \in \mathbb{N}$, where G^ℓ is obtained by taking the adjacency matrix of G to the ℓ^{th} power (i.e., an edge in G^ℓ corresponds to an (ℓ–1)-step path in G). Thus (1) also implies that

$$\Pr_{(u,v)\in E(G^\ell)}[u \in S, v \in S] \leq \frac{|S|}{|V|}\left(\frac{1}{2} + \frac{\lambda(G)^\ell}{2}\right) \tag{2}$$

Example 22.7
As an application of random walks in expanders, we describe how to prove a stronger version of the hardness of approximation result for INDSET in Theorem 11.15. This is done using the next Lemma, which immediately implies (since $m = \text{poly}(n)$) that there is some $\epsilon > 0$ such that computing $n^{-\epsilon}$-approximation to MAX-INDSET is **NP**-hard in graphs of size n. (See Section 22.9.2 for a survey of stronger hardness results for MAX-INDSET.) Below, $\tilde{\alpha}(F)$ denotes the fractional size of the largest independent set in F. It is interesting to note how this Lemma gives a more efficient version of the "self-improvement" idea of Theorem 11.15.

Lemma 22.8 *For every $\lambda > 0$ there is a polynomial-time computable reduction f that maps every n-vertex graph F into an m-vertex graph H such that*

$$(\tilde{\alpha}(F) - 2\lambda)^{\log n} \leq \tilde{\alpha}(H) \leq (\tilde{\alpha}(F) + 2\lambda)^{\log n}$$

Proof: We use random walks to define a more efficient version of the "graph product" used in the proof of Corollary 11.17. Let G be an expander on n nodes that is d-regular (where d is some constant independent of n) and let $\lambda = \lambda(G)$. For notational ease we assume G, F have the same set of vertices. We will map F into a graph H of $nd^{\log n - 1}$ vertices in the following way:

- The vertices of H correspond to all the $(\log n-1)$-step paths in the λ-expander G.
- We put an edge between two vertices u, v of H corresponding to the paths $\langle u_1, \ldots, u_{\log n}\rangle$ and $\langle v_1, \ldots, v_{\log n}\rangle$ if there exists an edge in G between two vertices in the set $\{u_1, \ldots, u_{\log n}, v_1, \ldots, v_{\log n}\}$.

It is easily checked that for any independent set in H if we take all vertices of F appearing in the corresponding walks, then that constitutes an independent set in F. From this observation the proof is concluded using Exercise 22.2. ■ $\diamondsuit$

22.2.4 Dinur's Gap-amplification

We say that a $q\mathsf{CSP}_W$ instance φ is "nice" if it satisfies the following properties:

Property 1: The arity q of φ is 2 (though the alphabet may be non binary).

Property 2: Let the *constraint graph* of φ be the graph G with vertex set $[n]$ where for every constraint of φ depending on the variables u_i, u_j, the graph G has the edge (i, j). We allow G to have parallel edges and self-loops. Then G is d-regular for some constant d (independent of the alphabet size) and at every node, half the edges incident to it are self-loops.

Property 3: The constraint graph is an expander. That is, $\lambda(G) \leq 0.9$.

It turns out that when proving Lemma 22.5 we may assume without loss of generality that the CSP instance φ is nice, since there is a relatively simple CL reduction mapping arbitrary qCSP instances to "nice" instances. (See Section 22.A; we note that these CL reductions will actually *lose* a factor depending on q in the soundness gap, but we can regain this factor by choosing a large enough value for t in Lemma 22.9 below.) The rest of the proof consists of a "powering" operation for nice 2CSP instances. This is described in the following lemma:

Lemma 22.9 *(Powering) There is an algorithm that given any $2\mathsf{CSP}_W$ instance ψ satisfying Properties 1 through 3 and an integer $t \geq 1$ produces an instance ψ^t of 2CSP such that:*

1. *ψ^t is a $2\mathsf{CSP}_{W'}$-instance with alphabet size $W' < W^{d^{5t}}$, where d denote the degree of ψ's constraint graph. The instance ψ^t has $d^{t+\sqrt{t}+1}n$ constraints, where n is the number of variables in ψ.*
2. *If ψ is satisfiable then so is ψ^t.*
3. *For every $\epsilon < \frac{1}{d\sqrt{t}}$, if $\mathsf{val}(\psi) \leq 1-\epsilon$ then $\mathsf{val}(\psi^t) \leq 1-\epsilon'$ for $\epsilon' = \frac{\sqrt{t}}{10^5 dW^4}\epsilon$.*
4. *The formula ψ^t is produced from ψ in time polynomial in m and W^{d^t}.* $\diamondsuit$

Proof: Let ψ be a $2\mathsf{CSP}_W$-instance with n variables and $m = nd$ constraints, and as before let G denote the *constraint graph* of ψ. To prove Lemma 22.9, we first show how we construct the formula ψ^t from ψ. The main idea is a certain "powering" operation on constraint graphs using random walks of a certain length.

Construction of ψ^t. The formula ψ^t will have the same number of variables as ψ. The new variables $\mathbf{y} = y_1, \ldots, y_n$ take values over an alphabet of size $W' = W^{d^{5t}}$, and thus a value of a new variable y_i is a d^{5t}-tuple of values in $\{0..W{-}1\}$. To avoid confusion in the rest of the proof, we reserve the term "variable" for these new variables, and say "old variables" if we mean the variables of ψ.

We will think of a value of variable y_i as giving a value in $\{0..W{-}1\}$ to every old variable u_j where j can be reached from i using a path of at most $t+\sqrt{t}$ steps in G (see Figure 22.1). In other words it gives an assignment for every u_j such that j is in the ball of radius $t+\sqrt{t}$ and center i in G. Since graph G is d-regular, the number of such nodes is at most $d^{t+\sqrt{t}+1}$, which is less than d^{5t}, so this information can indeed be encoded using an alphabet of size W'.

Below, we will often say that an assignment to y_i "claims" a certain value for the old variable u_j. Of course, the assignment to a different variable $y_{i'}$ could claim a different value for u_j; these potential inconsistences make the rest of the proof nontrivial. In fact, the constraints in the $2\mathsf{CSP}_{W'}$ instance ψ^t are designed to reveal such consistencies.

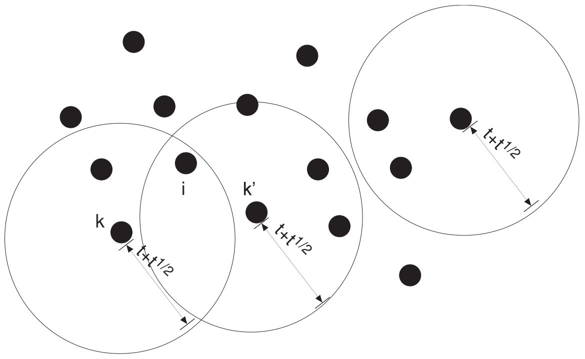

Figure 22.1 The CSP ψ^t consists of n variables taking values in an alphabet of size $W^{d^{5t}}$. An assignment to a new variable y_i encodes an assignment to all old variables of ψ corresponding to nodes that are in a ball of radius $t+\sqrt{t}$ around i in ψ's constraint graph. An assignment $y_1,\ldots,y_n$ to ψ^t may be *inconsistent* in the sense that if j falls in the intersection of two such balls centered at i and i', then y_i and $y_{i'}$ may claim different values for u_i.

For every $(2t+1)$-step path $p=\langle i_1,\ldots,i_{2t+2}\rangle$ in G, we have one corresponding constraint C_p in ψ^t (see Figure 22.2). The constraint C_p depends on the variables y_{i_1} and $y_{i_{2t+2}}$ (so we do indeed produce an instance of $2\mathsf{CSP}_{W'}$) and outputs FALSE if (and only if) there is some $j\in[2t+1]$ such that:

1. i_j is in the $t+\sqrt{t}$-radius ball around i_1.
2. i_{j+1} is in the $t+\sqrt{t}$-radius ball around i_{2t+2}
3. If w denotes the value y_{i_1} claims for u_{i_j} and w' denotes the value $y_{i_{2t+2}}$ claims for $u_{i_{j+1}}$, then the pair (w,w') violates the constraint in ψ that depends on u_{i_j} and $u_{i_{j+1}}$.

A few observations are in order. First, the time to produce this $2\mathsf{CSP}_{W'}$ instance is polynomial in m and W^{d^t}, so part 4 of Lemma 22.5 is trivial. Second, for every assignment to the old variables $u_1,u_2,\ldots,u_n$ we can "lift" it to a *canonical* assignment to $y_1,\ldots,y_n$ by simply assigning to each y_i the vector of values assumed by u_j's that lie in a ball of radius $t+\sqrt{t}$ and center i in G. If the assignment to the u_j's was a satisfying assignment for ψ, then this canonical assignment satisfies ψ^t, since it will satisfy all constraints encountered in walks of length $2t+1$ in G. Thus part 2 of Lemma 22.5 is also trivial. This leaves part 3 of the Lemma, the most difficult part. We have to show that if $\mathsf{val}(\psi)\le 1-\epsilon$ then every assignment to the y_i's satisfies at most $1-\epsilon'$ fraction of constraints in ψ^t, where $\epsilon<\frac{1}{d\sqrt{t}}$ and $\epsilon'=\frac{\sqrt{t}}{10^5 dW^4}\epsilon$.

The plurality assignment. To prove part 3 of the lemma, we show how to transform every assignment $\mathbf{y}$ for ψ^t into an assignment $\mathbf{u}$ for ψ and then argue that if $\mathbf{u}$ violates a "few" (i.e., ϵ fraction) of ψ's constraints then $\mathbf{y}$ violates "many" (i.e., $\epsilon'=\Omega(\sqrt{t}\epsilon)$ fraction) of constraints of ψ^t.

From now on, let us fix some arbitrary assignment $\mathbf{y}=y_1,\ldots,y_n$ to ψ^t's variables. As already noted, the values y_i's may be mutually *inconsistent* and not correspond to any

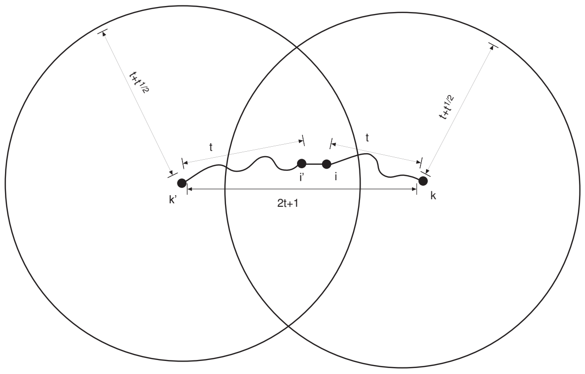

Figure 22.2 ψ^t has one constraint for every path of length $2t+1$ in ψ's constraint graph, checking that the views of the balls centered on the path's two endpoints are consistent with one another and the constraints of ψ.

obvious assignment for the old variable u_j's. The following notion is key because it tries to extract a single assignment for the old variables.

For every variable u_i of ψ, we define the random variable Z_i over $\{0, \ldots, W-1\}$ to be the result of the following process: starting from the vertex i, take a t step random walk in G to reach a vertex k, and output the value that y_k claims for u_i. We let z_i denote the most likely value of Z_i. If more than one value is most likely, we break ties arbitrarily. We call the assignment $z_1, \ldots, z_n$ the *plurality assignment* (see Figure 22.3). Note that $Z_i = z_i$ with probability at least $1/W$.

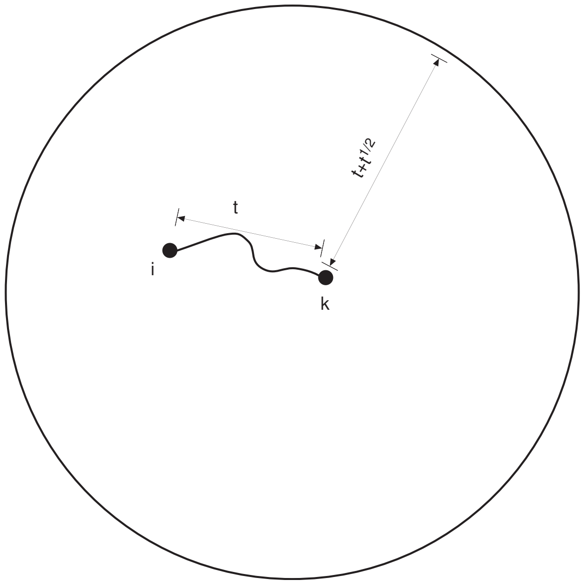

Figure 22.3 An assignment y for ψ^t induces a plurality assignment u for ψ in the following way: u_i gets the most likely value that is claimed for it by y_k, where k is obtained by taking a t-step random walk from i in the constraint graph of ψ.

Since $\mathsf{val}(\psi) \leq 1-\epsilon$, *every* assignment for ψ fails to satisfy ϵ fraction of the constraints, and this is therefore also true for the plurality assignment. Hence there exists a set F of $\epsilon m = \epsilon nd/2$ constraints in ψ that are violated by the assignment $\mathbf{z} = z_1, \ldots, z_n$. We will use this set F to show that at least an ϵ' fraction of ψ^t's constraints are violated by the assignment $\mathbf{y}$.

Analysis. The rest of the proof defines events in the following probability space: we pick a $(2t+1)$-step path, denoted $\langle i_1, \ldots, i_{2t+2}\rangle$, in G from among all such paths (in other words, pick a random constraint of ψ^t). For $j \in \{1, 2, \ldots, 2t+1\}$, say that the jth edge in the path, namely (i_j, i_{j+1}), is *truthful* if y_{i_1} claims the plurality value for i_j and $y_{i_{2t+2}}$ claims the plurality value for i_{j+1}. Observe that if the path has an edge that is truthful and also lies in F, then by definition of F the constraint corresponding to that path is unsatisfied. Our goal is to show that at least ϵ' fraction of the paths have such edges.

The proof will follow the following sequence of claims:

Claim 22.10 *For each edge e of G and each $j \in \{1, 2, \ldots, 2t+1\}$,*

$$\Pr[e \textit{ is the } j\textit{'th edge of the path}] = \frac{1}{|E|} = \frac{2}{nd}.$$

PROOF: It is easily checked that in a d-regular graph if we take a random starting point i_1 and pick a random path of length $2t+1$ going of it, then the j'th edge on a random path is also a random edge of G. ■

The next claim shows that edges that are roughly in the middle of the path (specifically, in the interval of size $\delta\sqrt{t}$ in the middle) are quite likely to be truthful.

Claim 22.11 *Let $\delta < \frac{1}{100W}$. For each edge e of G and each $j \in \{t, t, \ldots, t+\delta\sqrt{t}\}$,*

$$\Pr[j\textit{th edge of path is truthful } | e \textit{ is the } j\textit{th edge}] \geq \frac{1}{2W^2}.$$

PROOF: The main intuition is that since half the edges of G are self-loops, a random walk of length in $[t-\delta\sqrt{t}, t+\delta\sqrt{t}]$ is statistically very similar to a random walk of length t.

Formally, the lemma is proved by slightly inverting the viewpoint of how the path is picked. By the previous claim the set of walks of length $2t+1$ that contain $e = (i_j, i_{j+1})$ at the jth step can be generated by concatenating a random walk of length j out of i_j and a random walk of length $2t - j$ out of i_{j+1} (where the two walks are chosen independently). Let i_1 and i_{2t+2} denote the endpoints of these two walks. Then we need to show that

$$\Pr[y_{i_1} \text{ claims plurality value for } i_j \bigwedge y_{i_{2t+2}} \text{ claims plurality value for } i_{j+1}] \geq \frac{1}{2W^2}. \quad (3)$$

Since the plurality assignment was defined using walks of length exactly t, it follows that if j is precisely t, then the expression on the left hand side in (3) is at least $1/W \times 1/W = 1/W^2$. (This crucially uses that the walks to y_{i_1} and $y_{i_{2t+2}}$ are independently chosen.)

However, here j varies in $\{t, t+1, \ldots, t+\delta\sqrt{t}\}$, so these random walks have lengths between $t-\delta\sqrt{t}$ and $t+\delta\sqrt{t}$. We nevertheless show that the expression cannot be too different from $1/W^2$ for each j.

Since half the edges incident to each vertex are self-loops, we can think of an ℓ-step random walk from a vertex i as follows: **(1)** throw ℓ coins and let S_ℓ denote the number of the coins that came up "heads" **(2)** take S_ℓ "real" (non self-loop) steps on the graph. Recall that S_ℓ, the number of heads in ℓ tosses, is distributed according to the familiar *binomial distribution.*

It can be shown that the distributions S_t and $S_{t+\delta\sqrt{t}}$ are within statistical distance at most 10δ for every δ, t (see Exercise 22.3). In other words,

$$\frac{1}{2}\sum_m \left|\Pr[S_t = m] - \Pr[S_{t+\delta\sqrt{t}} = m]\right| \leq 10\delta.$$

It follows that the distribution of the endpoint of a t-step random walk out of e will be *statistically close* to the endpoint of a $(t+\delta\sqrt{t})$-step random walk, and the same is true for the $(t-\delta\sqrt{t})$-step random walk. Thus the expression on the left hand side of (3) is at least

$$(\frac{1}{W} - 10\delta)(\frac{1}{W} - 10\delta) \geq \frac{1}{2W^2},$$

which completes the proof. ■

Now let V be the random variable denoting the number of edges among the middle $\delta\sqrt{t}$ edges that are truthful and in F. Since it is enough for a path to contain one such edge for the corresponding constraint to be violated, our goal is to to show that $\Pr[V > 0] \geq \epsilon'$.

The previous two claims imply that the chance that any particular one of the edges in the interval of size $\delta\sqrt{t}$ is truthful and in F is $\frac{|F|}{|E|} \times \frac{1}{2W^2}$. Hence linearity of expectations implies:

$$\mathsf{E}[V] \geq \delta\sqrt{t} \times \frac{|F|}{|E|} \times \frac{1}{2W^2} = \frac{\delta\sqrt{t}\epsilon}{2W^2}.$$

This shows that $\mathsf{E}[V]$ is high, but we are not done since the expectation could be high and yet V could still be 0 for most of the walks. To rule this out, we consider the second moment. This calculation is the only place we use the fact that the contraint graph is an expander.

Claim 22.12 $\mathsf{E}[V^2] \leq 30\epsilon\delta\sqrt{t}d$. ♢

Proof: Let random variable V' denote the number of edges in the middle interval that are in F. Since V counts the number of edges that are in F *and* are truthful, $V \leq V'$. It suffices to show $E[V'^2] \leq 30\epsilon\delta\sqrt{t}d$. To prove this we use the mixing property of expanders and the fact that F contains ϵ fraction of the edges.

Specifically, for $j \in \{t, t, \ldots, t + \delta\sqrt{t}\}$ let I_j be an indicator random variable that is 1 if the jth edge is in F and 0 otherwise. Then $V' = \sum_{j\in\{t,t,\ldots,t+\delta\sqrt{t}\}} I_j$. Let S be the set of vertices that have at least one end point in F, implying $|S|/n \leq d\epsilon$.

$$\begin{aligned}
\mathsf{E}[V'^2] &= \mathsf{E}[\sum_{j,j'} I_j I_{j'}] \\
&= \mathsf{E}[\sum_j I_j^2] + \mathsf{E}[\sum_{j\neq j'} I_j I_{j'}] \\
&= \epsilon\delta\sqrt{t} + \mathsf{E}[\sum_{j\neq j'} I_j I_{j'}] \quad \text{(linearity of expectation and Claim 22.10)} \\
&= \epsilon\delta\sqrt{t} + 2\sum_{j<j'} \Pr[(j\text{th edge is in } F) \wedge (j'\text{th edge is in } F)] \\
&\leq \epsilon\delta\sqrt{t} + 2\sum_{j<j'} \Pr[(j\text{th vertex of walk lies in } S) \wedge (j'\text{th vertex of walk lies in } S)] \\
&\leq \epsilon\delta\sqrt{t} + 2\sum_{j<j'} \epsilon d(\epsilon d + (\lambda(G))^{j'-j}) \quad \text{(using (2))} \\
&\leq \epsilon\delta\sqrt{t} + 2\epsilon^2\delta\sqrt{t}d^2 + 2\epsilon\delta\sqrt{t}d\sum_{k\geq 1}(\lambda(G))^k \\
&\leq \epsilon\delta\sqrt{t} + 2\epsilon^2\delta\sqrt{t}d^2 + 20\epsilon\delta\sqrt{t}d \quad \text{(using } \lambda(G) \leq 0.9) \\
&\leq 30\epsilon\delta\sqrt{t}d \quad \text{(using } \epsilon < \tfrac{1}{d\sqrt{t}}\text{, an assumption of Lemma 22.9)}.
\end{aligned}$$

■

Finally, since $\Pr[V > 0] \geq \mathsf{E}[V]^2/\mathsf{E}[V^2]$ for any nonnegative random variable (see Exercise 22.4), we conclude that $\Pr[V > 0] \geq \frac{\sqrt{t}}{10^5 dW^4}\epsilon = \epsilon'$, and Lemma 22.9 is proved. ■

22.2.5 Alphabet Reduction: Proof of Lemma 22.6

Interestingly, the main component in the proof of Lemma 22.6 is the exponential-sized **PCP** of Section 11.5 (An alternative proof is explored in Exercise 22.5.)

Let φ be a 2CSP instance as in the statement of Lemma 22.6, with n variables $u_1, u_2, \ldots, u_n$, alphabet $\{0..W-1\}$ and m constraints $C_1, C_2, \ldots, C_m$. Think of each variable as taking values that are bit strings in $\{0,1\}^{\log W}$. Then if constraint C_s involves variables say u_i, u_j we may think of it as a circuit applied to the bit strings representing u_i, u_j where the constraint is said to be satisfied iff this circuit outputs 1. Say ℓ is an upper bound on the size of this circuit over all constraints. Note that ℓ is at most $2^{2\log W} < W^4$. We will assume without loss of generality that all circuits have the same size.

The idea in alphabet reduction will be to write a small CSP instance for each of these circuits, and replace each old variable by a set of new variables. This technique from [AS92] was called *verifier composition*, and more recently, a variant was called *PCP's of proximity,* and both names stem from the "proof verification" view of **PCP**'s (see Section 11.2). We state the result (a simple corollary of Theorem 11.19) first in the verification viewpoint and then translate into the CSP viewpoint.

Corollary 22.13 *(***PCP** *of proximity) There exists a verifier V that given any circuit C with $2k$ input wires and size ℓ has the following property:*

1. *If $\mathbf{u_1}, \mathbf{u_2}$ are strings of k bits each such that $\mathbf{u_1} \circ \mathbf{u_2}$ is a satisfying assignment for circuit C, then there is a string π_3 of size $2^{\mathrm{poly}(\ell)}$ such that V accepts $\mathsf{WH}(\mathbf{u_1}) \circ \mathsf{WH}(\mathbf{u_2}) \circ \pi_3$ with probability* 1.

2. *For every three bit strings π_1, π_2, π_3, where π_1 and π_2 have size 2^k, if V accepts $\pi_1 \circ \pi_2 \circ \pi_3$ with probability at least $1/2$, then π_1, π_2 are 0.99-close to $\mathsf{WH}(\mathbf{u_1})$, $\mathsf{WH}(\mathbf{u_2})$ respectively for some k-bit strings $\mathbf{u_1}, \mathbf{u_2}$ where $\mathbf{u_1} \circ \mathbf{u_2}$ is a satisfying assignment for circuit C.*

3. *V runs in $\mathrm{poly}(\ell)$ time, uses $\mathrm{poly}(\ell)$ random bits and examines only $O(1)$ bits in the provided strings.* $\diamondsuit$

Before giving the proof, we describe how it allows us to do alphabet reduction, as promised. First we note that in the CSP viewpoint of Corollary 22.13,(see Table 11.1) the variables are the bits of π_1, π_2, π_3, and V can be represented as a CSP instance of size $2^{\mathrm{poly}(\ell)}$ in these new variables. The arity of the constraints is the number of bits that the verifier reads in the proof, which is *some fixed constant independent of W and ϵ.* The fraction of satisfied constraints is the acceptance probability of the verifier.

Returning to the instance whose alphabet size we want to reduce, we replace each original variable u_i from the alphabet $\{0, \ldots, W-1\}$ by a sequence $U_i = (U_{i,1}, \ldots, U_{i,2^W})$ of 2^W binary-valued variables, which in a valid assignment would be an encoding of u_i using the Walsh-Hadamard code. For each old constraint $C_s(u_i, u_j)$ we apply the constraint satisfaction view of Corollary 22.13, using C_s as the circuit whose assignment is being verified. Thus for each original constraint C_s we have a vector of $2^{\mathrm{poly}(\ell)}$ new binary-valued variables Π_s, which plays the role of π_3 in Corollary 22.13, whereas U_i, U_j play the role of π_1, π_2 respectively. The set of new constraints corresponding to C_s is denoted $\mathcal{C}_s$. As already noted the arity of the new constraints is some fixed constant independent of W, ϵ.

The overall CSP instance is the union of these constraints $\cup_{s=1}^{m} \mathcal{C}_s$; see Figure 22.4. Clearly, if the old instance was satisfiable then so is this union. Now we show that if some assignment satisfies more than $1 - \epsilon/3$ fraction of the new constraints, then we can construct an assignment for the original instance that satisfies more than $1-\epsilon$ fraction of its constraints. This is done by "decoding" the assignment for each each set of new variables U_i by the following rule: if U_i is 0.99-close to some linear function $\mathsf{WH}(a_i)$ then use a_i as the assignment for the old variable u_i, and otherwise use an arbitrary string. Now consider how well we did on any old constraint $C_s(u_i, u_j)$. If the decodings a_i, a_j of U_i, U_j do not satisfy C_s then Corollary 22.13 implies that at least $1/2$ the constraints of $\mathcal{C}_s$ were not satisfied anyway. Thus if δ is the fraction of old constraints that are not satisfied, then $\delta/2 \le \epsilon/3$, implying $\delta < 2\epsilon/3$, and the Lemma is proved.

To finish, we prove Corollary 22.13.

Proof: (of Corollary 22.13) The proof uses the reduction from CKT-SAT to QUADEQ (see Section 11.5.2 and Exercise 11.15). This reduction transforms a circuit C with ℓ wires (where "inputs" are considered as wires in the circuit) to an instance of QUADEQ of with ℓ variables and $O(\ell)$ equations where the variables in the QUADEQ instance correspond to

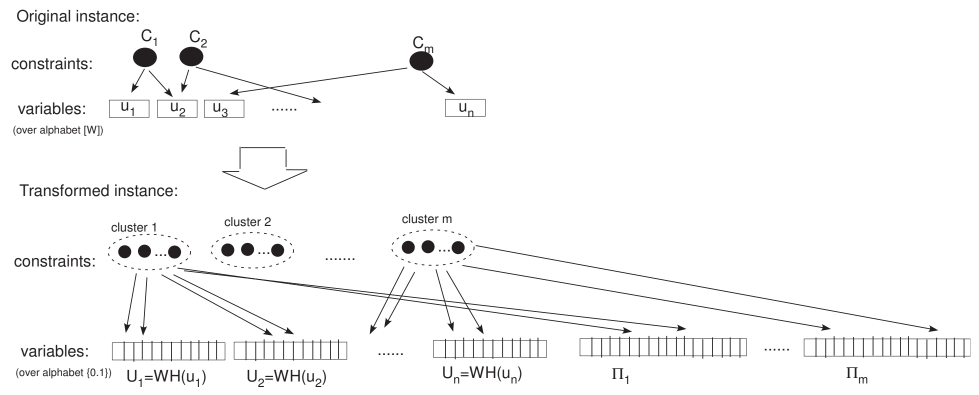

Figure 22.4 The alphabet reduction transformation maps a 2CSP instance φ over alphabet $\{0..W{-}1\}$ into a qCSP instance ψ over the binary alphabet. Each variable of φ is mapped to a block of binary variables that in the correct assignment will contain the Walsh-Hadamard encoding of this variable. Each constraint C_ℓ of φ depending on variables u_i, u_j is mapped to a cluster of constraints corresponding to all the **PCP** of proximity constraints for C_ℓ. These constraint depend on the encoding of u_i and u_j, and on additional auxiliary variables that in the correct assignment contain the **PCP** of proximity proof that these are indeed encoding of values that make the constraint C_ℓ true.

values of wires in the circuit. Thus every solution to the QUADEQ instance has ℓ bits, of which the first k bits give a satisfying assignment to the circuit.

The verifier expects π_3 to contain whatever our verifier of Theorem 11.19 expects in the proof for this instance of QUADEQ, namely, a linear function f that is WH(w), and another linear function g that is WH$(w \otimes w)$ where w satisfies the QUADEQ instance. The verifier checks these functions as described in the proof of Theorem 11.19.

However, in the current setting our verifer is also given strings π_1, π_2, which we think of as functions $\pi_1, \pi_2 : \mathrm{GF}(2)^k \to \mathrm{GF}(2)$. The verifier checks that both are 0.99-close to linear functions, say $\tilde{\pi_1}, \tilde{\pi_2}$. Then to check that $\tilde{f}$ encodes a string whose first $2k$ bits are the same as the string encoded by $\tilde{\pi_1}, \tilde{\pi_2}$, the verifier does the following *concatenation test*, which uses the properties of the Walsh-Hadamard code.

Concatenation test. We are given three linear functions π_1, π_2, f that encode strings of lengths k, k, and ℓ respectively. Denoting by $\mathbf{u}$ and $\mathbf{v}$ the strings encoded by π_1, π_2 respectively (that is, $\pi_1 = \mathsf{WH}(\mathbf{u})$ and $\pi_2 = \mathsf{WH}(\mathbf{v})$), and by w the string encoded by f, we have to check by examining only $O(1)$ bits in these functions that $\mathbf{u} \circ \mathbf{v}$ is the same as the first $2k$ bits of w. By the random subsum principle, the following simple test rejects with probability $1/2$ if this is not the case. Pick random $\mathbf{x}, \mathbf{y} \in \{0,1\}^k$, and denote by $\mathbf{XY} \in \mathrm{GF}(2)^\ell$ the string whose first k bits are $\mathbf{x}$, the next k bits are $\mathbf{y}$ and the remaining bits are all 0. Accept if $f(\mathbf{XY} = \pi_1(\mathbf{x}) + \pi_2(\mathbf{y})$ and else reject. ■

22.3 Hardness of 2CSP$_W$: Tradeoff between gap and alphabet size

The problem 2CSP$_W$ often plays a role in proofs of advanced **PCP** theorems. The (standard) **PCP** theorem implies that there is some constant W and some $\nu < 1$ such that computing ν-approximation to 2CSP$_W$ is **NP**-hard (see Definition 22.1).

Corollary 22.14 *(of* **PCP** *Theorem) There is some $\nu < 1$ and some W such that GAP 2CSP$_W(\nu)$ is* **NP***-hard.* ♢

For advanced **PCP** theorems we would like to prove the same result for smaller ν, without making W too large. (Note: if W is allowed to be $\exp(n)$ then the problem is **NP**-hard even for $\nu = 0$!) At first glance the "gap amplification" of Lemma 22.5 seems relevant, but that doesn't suffice because first, it cannot lower ν below some fixed constant, and second, because it greatly increases the alphabet size. The next theorem gives the best tradeoff possible (up to the value of c) between these two parameters. For further constructions of **PCP**'s, it is useful to restrict attention to a special subclass of 2CSP instances, which have the so-called *projection* property. This means that for each constraint $\varphi_r(y_1, y_2)$ and each value of y_1, there is a *unique* value of y_2 such that $\varphi_r(y_1, y_2) = 1$. Another way to state this is that for each constraint φ_r there is a function $h : [W] \to [W]$ such that the constraint is satisfied by (u, v) iff $h(u) = v$.

A 2CSP instance is said to be *regular* if every variable appears in the same number of constraints.

Theorem 22.15 *(Raz [Raz95b]) There is a $c > 1$ such that for every $t > 1$, GAP $\mathsf{2CSP}_W(\epsilon)$ is* **NP***-hard for $\epsilon = 2^{-t}, W = 2^{ct}$, and this is true also for* 2CSP *instances that are regular and have the projection property.* ♢

A weaker version of this theorem, with a somewhat simpler proof, was obtained by Feige and Kilian [FK93]. This weaker version is sufficient for many applications, including for Håstad's 3-bit **PCP** theorem (see Section 22.4 below).

22.3.1 Idea of Raz's proof: Parallel Repetition

Let φ be the $\mathsf{2CSP}_W$ instances produced by the reduction of Corollary 22.14. For some $\nu < 1$ it has the property that either $\mathsf{val}(\varphi) = 1$ or $\mathsf{val}(\varphi) = \nu < 1$ but deciding which case holds is hard. There is an obvious "powering" idea for trying to lower the gap while maintaining the arity at 2. Let φ^{*t} denote the following instance. Its variables are t-tuples of variables of φ. Its constraints correspond to t-tuples of constraints, in the sense that for every t-tuple of constraints $\varphi_1(y_1, z_1), \varphi_2(y_2, 2), \ldots, \varphi_t(y_t, z_t)$ the new instance has a constraint of arity 2 involving the new variables $(y_1, y_2, \ldots, y_t)$ and $(z_1, z_2, \ldots, z_t)$ and the Boolean function describing this constraint is simply

$$\bigwedge_{i=1}^{t} \varphi_i(y_i, z_i).$$

(To put it in words, the new constraint is satisfied iff all the t constituent constraints are.)

In the verification viewpoint, this new 2CSP instance corresponds to running the verifier in parallel t times, hence Raz's theorem is also called the *parallel repetition theorem*.

It is easy to convert a satisfying assignment for φ into one for φ^{*t} by taking t-tuples of the values. Furthermore, given an assignment for φ that satisfies ν fraction of the constraints, it is easy to see that the assignment that forms t-tuples of these values satisfies at least ν^t fraction of the constraints of φ^{*t}. It seemed "obvious" to researchers that no assignment can do better. Then a simple counterexample was found, whereby more than ν^t fraction of constraints in φ^{*t} could be satisfied (see Exercise 22.6). Raz shows, however, that no assignment can satisfy more than ν^{ct} fraction of the constraints of φ^{*t}, where c depends upon the alphabet size W. The proof is quite difficult, though there have been some simplifications (see the chapter notes and the book's web site).

22.4 Håstad's 3-bit PCP Theorem and hardness of MAX-3SAT

In Chapter 11 we showed $\mathbf{NP} = \mathbf{PCP}(\log n, 1)$, in other words certificates for membership in **NP** languages can be checked by examining $O(1)$ bits in them. Now we are interested in keeping the number of query bits as low as possible, *while keeping the soundness around* 1/2. The next Theorem shows that the number of query bits can be reduced to 3, and furthermore the verifier's decision process consists of simply looking at the parity of these three bits.

Theorem 22.16 *(Håstad's 3-bit* **PCP** [Hås97]*)*
For every $\delta > 0$ and every language $L \in \mathbf{NP}$ there is a **PCP***-verifier V for L making three (binary) queries having completeness parameter $1 - \delta$ and soundness parameter at most $1/2 + \delta$.*
Moreover, the tests used by V are linear. *That is, given a proof $\pi \in \{0,1\}^m$, V chooses a triple $(i_1, i_2, i_3) \in [m]^3$ and $b \in \{0,1\}$ according to some distribution and accepts iff $\pi_{i_1} + \pi_{i_2} + \pi_{i_3} = b \pmod 2$.*

22.4.1 Hardness of approximating MAX-3SAT

We first note that Theorem 22.16 is intimately connected to the hardness of approximating a problem called MAX-E3LIN, which is a subcase of $3\mathsf{CSP}_2$ in which the constraints specify the *parity* of triples of variables. Another way to think of such an instance is that it gives a set of linear equations mod 2 where each equation has at most 3 variables. We are interested in determining the largest subset of equations that are simultaneously satisfiable. We claim that Theorem 22.16 implies that $(1/2 + \nu)$-approximation to this problem is **NP**-hard for every $\nu > 0$. This is a *threshold result* since the problem has a simple 1/2-approximation algorithm. (It uses observations similar to those we made in context of MAX-3SAT in Chapter 11; a random assignment satisfies, in the expectation, half of the constraints, and this observation can be turned into a deterministic algorithm that satisfies at least 1/2 of the equations.)

To prove our claim about the hardness of MAX-E3LIN, we convert the verifier of Theorem 22.16 into an equivalent CSP by the recipe of Section 11.3. Since the verifier imposes parity constraints on triples of bits in the proof, the equivalent CSP instance is an instance of MAX-E3LIN where either $1 - \delta$ fraction of the constraints are satisfiable, or at most $1/2 + \delta$ are. Since distinguishing between the two cases is **NP**-hard, we conclude that it is **NP**-hard to compute a ρ-approximation to MAX-E3LIN where $\rho = \frac{1/2+\delta}{1-\delta}$. Since $\delta > 0$ is allowed to be arbitrarily small, ρ can be arbitrarily close to 1/2 and we conclude that $(1/2 + \nu)$-approximation is **NP**-hard for every $\nu > 0$.

Also note that the fact that completeness is strictly less than 1 in Theorem 22.16 is inherent if $\mathbf{P} \neq \mathbf{NP}$, since determining if there is a solution satisfying *all* of the equations (in other words, the satisfiability problem for MAX-E3LIN) is possible in polynomial time using Gaussian elimination

Now we prove a hardness result for MAX-3SAT, which as mentioned earlier, is also a threshold result.

Corollary 22.17 *For every $\epsilon > 0$, computing $(7/8+\epsilon)$-approximation to* MAX-3SAT *is* **NP***-hard.* ◊

PROOF: We reduce MAX-E3LIN to MAX-3SAT. Take the instance of MAX-E3LIN produced by the above reduction, where we are interested in determining whether $(1 - \nu)$ fraction of the equations can be satisfied or at most $1/2+\nu$ are. Represent each linear constraint by four $3CNF$ clauses in the obvious way. For example, the linear constraint $a+b+c = 0 \pmod 2$ is equivalent to the clauses $(\overline{a} \vee b \vee c), (a \vee \overline{b} \vee c), (a \vee b \vee \overline{c}), (\overline{a} \vee \overline{b} \vee \overline{c})$. If a, b, c satisfy the linear constraint, they satisfy all 4 clauses and otherwise they satisfy at most 3 clauses. We conclude that in one case at least $(1 - \epsilon)$ fraction of clauses are simultaneously satisfiable, and in the other case at most $1 - (\frac{1}{2} - \nu) \times \frac{1}{4} = \frac{7}{8} + \frac{\nu}{4}$ fraction are. The ratio between the two cases tends to 7/8 as ν decreases. Since Theorem 22.16 implies that distinguishing between the two cases is **NP**-hard for every constant ν, the result follows. ■

22.5 Tool: the Fourier transform technique

Theorem 22.16 is proved using Fourier analysis. The continuous Fourier transform is extremely useful in mathematics and engineering. Likewise, the discrete Fourier transform has found many uses in algorithms and complexity, in particular for constructing and analyzing **PCP**'s. The Fourier transform technique for **PCP**'s involves calculating the maximum acceptance probability of the verifier using Fourier analysis of the functions presented in the proof string. (See Note 22.21 for a broader perspective of uses of discrete Fourier transforms in combinatorial and probabilistic arguments.) It is delicate enough to give "tight" inapproximability results for MAX-INDSET, MAX-3SAT, and many other problems.

To introduce the technique we start with a simple example: analysis of the linearity test over GF(2) (i.e., proof of Theorem 11.21). We then introduce the *Long Code* and show how to test for membership in it. These ideas are then used to prove Håstad's 3-bit PCP Theorem.

22.5.1 Fourier transform over $\mathrm{GF}(2)^n$

The Fourier transform over $\mathrm{GF}(2)^n$ is a tool to study functions on the Boolean hypercube. In this chapter, it will be useful to use the set $\{+1,-1\} = \{\pm 1\}$ instead of $\{0,1\}$. To transform $\{0,1\}$ to $\{\pm1\}$, we use the mapping $b \mapsto (-1)^b$ (i.e., $0 \mapsto +1$, $1 \mapsto -1$). Thus we write the hypercube as $\{\pm1\}^n$ instead of the more usual $\{0,1\}^n$. Note this maps the XOR operation (i.e., addition in GF(2)) into the multiplication operation over $\mathbb{R}$.

The set of functions from $\{\pm1\}^n$ to $\mathbb{R}$ defines a 2^n-*dimensional Hilbert space* (i.e., a vector space with an associated inner product) as follows. Addition and multiplication by a scalar are defined in the natural way: $(f+g)(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x})$ and $(\alpha f)(\mathbf{x}) = \alpha f(\mathbf{x})$ for every $f, g : \{\pm1\}^n \to \mathbb{R}$, $\alpha \in \mathbb{R}$. We define the inner product of two functions f, g, denoted $\langle f, g\rangle$, to be $\mathsf{E}_{\mathbf{x}\in\{\pm1\}^n}[f(\mathbf{x})g(\mathbf{x})]$. (This is the *expectation inner product*.)

The *standard basis* for this space is the set $\{\mathbf{e_x}\}_{\mathbf{x}\in\{\pm1\}^n}$, where $\mathbf{e_x}(\mathbf{y})$ is equal to 1 if $\mathbf{y} = \mathbf{x}$, and equal to 0 otherwise. This is an orthogonal basis, and every function $f : \{\pm1\}^n \to \mathbb{R}$ can be represented in this basis as $f = \sum_{\mathbf{x}} a_{\mathbf{x}}\mathbf{e_x}$. For every $\mathbf{x} \in \{\pm1\}^n$, the coefficient $a_{\mathbf{x}}$ is equal to $f(\mathbf{x})$.

The *Fourier basis* is an alternative orthonormal basis that contains, for every subset $\alpha \subseteq [n]$, a function χ_α where $\chi_\alpha(\mathbf{x}) = \prod_{i\in\alpha} x_i$. (We define $\chi_\emptyset$ to be the function that is 1 everywhere). This basis is actually the Walsh-Hadamard code (see Section 11.5.1) in disguise: the basis vectors correspond to the *linear* functions over GF(2). To see this, note that every linear function of the form $\mathbf{b} \mapsto \mathbf{a} \odot \mathbf{b}$ (with $\mathbf{a}, \mathbf{b} \in \{0,1\}^n$) is mapped by our transformation to the function taking $\mathbf{x} \in \{\pm1\}^n$ to $\prod_{i \text{ s.t. } a_i=1} x_i$. To check that the Fourier basis is indeed an orthonormal basis for $\mathbb{R}^{2^n}$, note that the random subsum principle implies that for every $\alpha, \beta \subseteq [n]$, $\langle \chi_\alpha, \chi_\beta\rangle = \delta_{\alpha,\beta}$ where $\delta_{\alpha,\beta}$ is equal to 1 iff $\alpha = \beta$ and equal to 0 otherwise.

Remark 22.18
Note that in the $\{-1,1\}$ view, the basis functions can be viewed as *multilinear polynomials* (i.e., multivariate polynomials whose degree in each variable is 1). Thus the fact that every real-valued function $f : \{-1,1\}^n$ has a Fourier expansion can also be phrased as "Every such function can be represented by a multilinear polynomial." This is very much in the same spirit as the polynomial representations used in Chapters 8 and 11.

Since the Fourier basis is an orthonormal basis, every function $f : \{\pm1\}^n \to \mathbb{R}$ can be represented as $f = \sum_{\alpha\subseteq[n]} \hat{f}_\alpha \chi_\alpha$. We call $\hat{f}_\alpha$ the α^{th} *Fourier coefficient* of f. We will often use the following simple lemma:

Lemma 22.19 *Every two functions $f, g : \{\pm1\}^n \to \mathbb{R}$ satisfy*

1. $\langle f, g\rangle = \sum_\alpha \hat{f}_\alpha \hat{g}_\alpha$.
2. (Parseval's Identity) $\langle f, f\rangle = \sum_\alpha \hat{f}_\alpha^2$.

PROOF: The second property follows from the first. To prove the first we expand

$$\langle f,g\rangle = \langle\sum_{\alpha}\hat{f}_\alpha\chi_\alpha,\sum_{\beta}\hat{g}_\beta\chi_\beta\rangle = \sum_{\alpha,\beta}\hat{f}_\alpha\hat{g}_\beta\langle\chi_\alpha,\chi_\beta\rangle = \sum_{\alpha,\beta}\hat{f}_\alpha\hat{g}_\beta\delta_{\alpha,\beta} = \sum_{\alpha}\hat{f}_\alpha\hat{g}_\alpha$$

■

Example 22.20
Some examples for the Fourier transform of particular functions:

1. The majority function on 3 variables (i.e., the function $MAJ(u_1,u_2,u_3)$ that outputs $+1$ if and only if at least two of its inputs are $+1$, and -1 otherwise) can be expressed as $1/2u_1 + 1/2u_2 + 1/2u_3 - 1/2u_1u_2u_3$. Thus, it has four Fourier coefficients equal to $1/2$ and the rest are equal to zero.
2. If $f(u_1,u_2,\ldots,u_n) = u_i$ (i.e., f is a *coordinate function*, a concept we will see again in context of long codes) then $f = \chi_{\{i\}}$ and so $\hat{f}_{\{i\}} = 1$ and $\hat{f}_\alpha = 0$ for $\alpha \neq \{i\}$.
3. If f is a random Boolean function on n bits, then each $\hat{f}_\alpha$ is a random variable that is a sum of 2^n binomial variables (equally likely to be $1,-1$) and hence looks like a normally distributed variable with standard deviation $2^{n/2}$ and mean 0. Thus with high probability, all 2^n Fourier coefficients have values in $[-\frac{\text{poly}(n)}{2^{n/2}},\frac{\text{poly}(n)}{2^{n/2}}]$.

22.5.2 The connection to PCPs: High level view

In the PCP context we are interested in *Boolean-valued* functions, i.e., those from $GF(2)^n$ to $GF(2)$. Under our transformation they turn into functions from $\{\pm1\}^n$ to $\{\pm1\}$. Thus, we say that $f:\{\pm1\}^n \to \mathbb{R}$ is *Boolean* if $f(\mathbf{x}) \in \{\pm1\}$ for every $\mathbf{x} \in \{\pm1\}^n$. Note that if f is Boolean then $\langle f,f\rangle = \mathsf{E}_{\mathbf{x}}[f(\mathbf{x})^2] = 1$.

On a high level, we use the Fourier transform in the soundness proofs for PCP's to show that if the verifier accepts a proof π with high probability then π is "close to" being "well-formed" (where the precise meaning of "close-to" and "well-formed" is context dependent). Usually we relate the acceptance probability of the verifier to an expectation of the form $\langle f,g\rangle = \mathsf{E}_{\mathbf{x}}[f(\mathbf{x})g(\mathbf{x})]$, where f and g are Boolean functions arising from the proof. We then use techniques similar to those used to prove Lemma 22.19 to relate this acceptance probability to the Fourier coefficients of f,g, allowing us to argue that if the verifier's test accepts with high probability, then f and g have few relatively large Fourier coefficients. This will provide us with some nontrivial useful information about f and g, since in a "generic" or random function, all the Fourier coefficient are small and roughly equal.

22.5.3 Analysis of the linearity test over $GF(2)$

We will now prove Theorem 11.21, thus completing the proof of the **PCP** Theorem. Recall that the linearity test is provided a function $f:\mathrm{GF}(2)^n \to \mathrm{GF}(2)$ and has to determine whether f has significant agreement with a linear function. To do this it picks $\mathbf{x},\mathbf{y} \in \mathrm{GF}(2)^n$ randomly and accepts iff $f(\mathbf{x}+\mathbf{y}) = f(\mathbf{x}) + f(\mathbf{y})$.

Now we rephrase this test using $\{\pm1\}$ instead of GF(2), so linear functions turn into Fourier basis functions. For every two vectors $\mathbf{x},\mathbf{y} \in \{\pm1\}^n$, we denote by $\mathbf{xy}$ their componentwise multiplication. That is, $\mathbf{xy} = (x_1y_1,\ldots,x_ny_n)$. Note that for every basis function $\chi_\alpha(\mathbf{xy}) = \chi_\alpha(\mathbf{x})\chi_\alpha(\mathbf{y})$.

For two Boolean functions f,g, their inner product $\langle f,g\rangle$ is equal to the fraction of inputs on which they *agree* minus the fraction of inputs on which they *disagree*. It follows that for every $\epsilon \in [0,1]$ and functions $f, g : \{\pm1\}^n \to \{\pm1\}$, f has agreement $\frac{1}{2} + \frac{\epsilon}{2}$ with g iff $\langle f, g\rangle = \epsilon$. Thus, if f has a large Fourier coefficient then it has significant agreement with some Fourier basis function, or in the GF(2) worldview, f is close to some linear function. This means that Theorem 11.21 concerning the correctness of the linearity test can be rephrased as follows:

Theorem 22.22 *Suppose that $f : \{\pm1\}^n \to \{\pm1\}$ satisfies $\Pr_{\mathbf{x},\mathbf{y}}[f(\mathbf{xy}) = f(\mathbf{x})f(\mathbf{y})] \geq \frac{1}{2}+\epsilon$. Then, there is some $\alpha \subseteq [n]$ such $\hat{f}_\alpha \geq 2\epsilon$.* ♢

Proof: We can rephrase the hypothesis as $E_{\mathbf{x},\mathbf{y}}[f(\mathbf{xy})f(\mathbf{x})f(\mathbf{y})] \geq (\frac{1}{2}+\epsilon) - (\frac{1}{2}-\epsilon) = 2\epsilon$. We note that from now on we do not need f to be Boolean, but merely to satisfy $\langle f, f\rangle = 1$.

Expressing f by its Fourier expansion,

$$2\epsilon \leq \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y}}[f(\mathbf{xy})f(\mathbf{x})f(\mathbf{y})] = \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y}}[(\sum_\alpha \hat{f}_\alpha\chi_\alpha(\mathbf{xy}))(\sum_\beta \hat{f}_\beta\chi_\beta(\mathbf{x}))(\sum_\gamma \hat{f}_\gamma\chi_\gamma(\mathbf{y}))].$$

Since $\chi_\alpha(\mathbf{x}y) = \chi_\alpha(\mathbf{x})\chi_\alpha(\mathbf{y})$ this becomes

$$= \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y}}[\sum_{\alpha,\beta,\gamma} \hat{f}_\alpha\hat{f}_\beta\hat{f}_\gamma\chi_\alpha(\mathbf{x})\chi_\alpha(\mathbf{y})\chi_\beta(\mathbf{x})\chi_\gamma(\mathbf{y})].$$

Using linearity of expectation:

$$= \sum_{\alpha,\beta,\gamma} \hat{f}_\alpha\hat{f}_\beta\hat{f}_\gamma \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y}}[\chi_\alpha(\mathbf{x})\chi_\alpha(\mathbf{y})\chi_\beta(\mathbf{x})\chi_\gamma(\mathbf{y})]$$
$$= \sum_{\alpha,\beta,\gamma} \hat{f}_\alpha\hat{f}_\beta\hat{f}_\gamma \mathop{\mathsf{E}}_{\mathbf{x}}[\chi_\alpha(\mathbf{x})\chi_\beta(\mathbf{x})] \mathop{\mathsf{E}}_{\mathbf{y}}[\chi_\alpha(\mathbf{y})\chi_\gamma(\mathbf{y})]$$

(because $\mathbf{x}, \mathbf{y}$ are independent).

By orthonormality $\mathsf{E}_{\mathbf{x}}[\chi_\alpha(\mathbf{x})\chi_\beta(\mathbf{x})] = \delta_{\alpha,\beta}$, so we simplify to

$$= \sum_\alpha \hat{f}_\alpha^3$$
$$\leq (\max_\alpha \hat{f}_\alpha) \times (\sum_\alpha \hat{f}_\alpha^2) = \max_\alpha \hat{f}_\alpha\,,$$

since $\sum_\alpha \hat{f}_\alpha^2 = \langle f, f\rangle = 1$. Hence $\max_\alpha \hat{f}_\alpha \geq 2\epsilon$ and the theorem is proved. ■

22.6 Coordinate functions, Long code and its testing

Håstad's 3-bit PCP Theorem uses a coding method called the *long code*. Let $W \in \mathbb{N}$. We say that $f : \{\pm1\}^W \to \{\pm1\}$ is a *coordinate function* if there is some $w \in [W]$, such that $f(x_1, x_2, \ldots, x_W) = x_w$; in other words, $f = \chi_{\{w\}}$.[1] (Aside: Unlike the previous section, here we use W instead of n for the number of variables; the reason is to be consistent with our use of W for the alphabet size in $2\mathsf{CSP}_W$ in Section 22.7.)

Definition 22.23 *(Long Code)* The *long code* for $[W]$ encodes each $w \in [W]$ by the table of all values of the function $\chi_{\{w\}} : \{\pm1\}^{[W]} \to \{\pm1\}$. ♢

[1]Some texts call such a function a *dictatorship* function, since one variable ("the dictator") completely determines the outcome. The name comes from social choice theory, which studies different election setups. That field has also been usefully approached using Fourier analysis ideas described in Note 22.21.

Note 22.21 *(Self-correlated functions, isoperimetry, phase transitions)*

Although it is surprising to see Fourier transforms used in proofs of PCP Theorems, in retrospect this is quite natural. We try to put this in perspective, and refer the reader to the survey of Kalai and Safra [KS06] and the web-based lecture notes of O'Donnell, Mossell and others for further background on this topic.

Classically, Fourier tranforms are very useful in proving results of the following form: "If a function is correlated with itself in some structured way, then it belongs to some small family of functions." In the PCP setting, the "self-correlation" of a function $f : \{0,1\}^n \to \{0,1\}$ means that if we run some designated verifier on f that examines only a few bits in it, then this verifier accepts with reasonable probability. For example, in the linearity test over GF(2), the acceptance probability of the test is $E_{x,y}[I_{x,y}]$ where $I_{x,y}$ is an indicator random variable for the event $f(x) + f(y) = f(x + y)$.

Another classical use of Fourier transforms is study of *Isoperimetry*, which is the study of subsets of "minimum surface area." A simple example is the fact that of all connected regions in $\mathbb{R}^2$ with a specified area, the circle has the minimum perimeter. Again, isoperimetry can be viewed as a study of "self-correlation", by thinking of the characteristic function of the set in question, and realizing that the "perimeter" of "surface" of the set consists of points in space where taking a small step in some direction causes the value of this function to switch from 1 to 0.

Håstad's "noise" operator of Section 22.7 appears in works of mathematicians Nelson, Bonamie, Beckner, and others on *hypercontractive estimates*, and the general theme is again one of identifying properties of functions based upon their "self-correlation" behavior. One considers the correlation of the function f with the function $T_\rho(f)$ obtained by (roughly speaking) computing at each point the average value of f in a small ball around that point. One can show that the norms of f and $T_\rho(f)$ are related — not used in Håstad's proof but very useful in the **PCP** Theorems surveyed in Section 22.9; see also Exercise 22.10 for a small taste.

Fourier transforms and especially hypercontractivity estimates have also proved useful in study of *phase transitions* in random graphs (e.g., see Friedgut [Fri99]). The simplest case is the graph model $G(n, p)$ whereby each possible edge is included in the graph independently with probability p. A *phase transition* is a value of p at which the graph goes from almost never having a certain property to almost always having that property. For example, it is known that there is some constant c such that around $p = c \log n/n$ the probability of the graph being connected suddenly jumps from close to 0 to close to 1. Fourier transforms are useful to study phase transition because a phase transition is as an isoperimetry problem on a "Graph" (with a capital G) where each "Vertex" is an n-vertex graph, and an "Edge" between two "Vertices" means that one of the graphs is obtained by adding a few edges to the graph. Note that adding a few edges corresponds to raising the value of p by a little.

Finally, we mention some interesting uses of Fourier transforms in the results mentioned in Sections 22.9.4 and 22.9.5. These involve isoperimetry on the hypercube $\{0,1\}^n$. One can study isoperimetry in a graph setting by defining "surface area" of a subset of vertices as the "number of edges leaving the set," or some other notion, and then try to study isoperimetry in such settings. The Fourier transform can be used to prove isoperimetry theorems about hypercube and hypercube-like graphs. The reason is that a subset $S \subseteq \{0,1\}^n$ is nothing but a Boolean function that is 1 on S and -1 elsewhere. Assuming the graph is D-regular, and $|S| = 2^{n-1}$

$$E_{(x,y):\text{ edge}}[f(x)f(y)] = \frac{1}{2^n D}\left(|E(S,S)| + \left|\overline{S},\overline{S}\right| - 2\left|E(S,\overline{S})\right|\right),$$

which implies that the fraction of edges of S that leave the set is $1/2 - E[f(x)f(y)]/2$. This kind of expression can be analysed using the Fourier transform; see Exercise 22.11.b.

Note that w, normally written using $\log W$ bits, is being represented using a table of 2^W bits, a doubly exponential blowup! This inefficiency is the reason for calling the code "long."

The problem of testing for membership in the Long Code is defined by analogy to the earlier test for the Walsh-Hadamard code. We are given a function $f : \{\pm 1\}^W \to \{\pm 1\}$, and wish to determine if f has good agreement with $\chi_{\{w\}}$ for some w, namely, whether $\hat{f}_{\{w\}}$ is significant. Such a test is described in Exercise 22.5, but it is not sufficient for the proof of Håstad's Theorem, which requires a test using only *three* queries. Below we show such a three query test albeit at the expense of achieving the following weaker guarantee: if the test passes with high probability then f has a good agreement with a function χ_α where $|\alpha|$ is small (but not necessarily equal to 1). This weaker conclusion will be sufficient in the proof of Theorem 22.16.

Let $\rho > 0$ be some arbitrarily small constant. The test picks two uniformly random vectors $\mathbf{x}, \mathbf{y} \in_{\mathrm{R}} \{\pm 1\}^W$ and then a vector $\mathbf{z} \in \{\pm 1\}^W$ according to the following distribution: for every coordinate $i \in [W]$, with probability $1-\rho$ we choose $z_i = +1$ and with probability ρ we choose $z_i = -1$. Thus with high probability, about ρ fraction of coordinates in $\mathbf{z}$ are -1 and the other $1-\rho$ fraction are $+1$. We think of $\mathbf{z}$ as a "noise" vector. The test accepts iff $f(\mathbf{x})f(\mathbf{y}) = f(\mathbf{xyz})$. Note that the test is similar to the linearity test except for the use of the noise vector $\mathbf{z}$.

Suppose $f = \chi_{\{w\}}$. Then since $b \cdot b = 1$ for $b \in \{\pm 1\}$,

$$f(\mathbf{x})f(\mathbf{y})f(\mathbf{xyz}) = x_w y_w (x_w y_w z_w) = 1 \cdot z_w .$$

Hence the test accepts iff $z_w = 1$ which happens with probability $1-\rho$. We now prove a certain converse:

Lemma 22.24 *If the test accepts with probability* $1/2 + \delta$ *then* $\sum_\alpha \hat{f}_\alpha^3 (1-2\rho)^{|\alpha|} \geq 2\delta$. $\diamondsuit$

PROOF: If the test accepts with probability $1/2 + \delta$ then $\mathsf{E}[f(\mathbf{x})f(\mathbf{y})f(\mathbf{xyz})] = 2\delta$. Replacing f by its Fourier expansion, we have

$$\begin{aligned} 2\delta &\leq \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y},\mathbf{z}}\left[\Big(\sum_\alpha \hat{f}_\alpha \chi_\alpha(\mathbf{x})\Big) \cdot \Big(\sum_\beta \hat{f}_\beta \chi_\beta(\mathbf{y})\Big) \cdot \Big(\sum_\gamma \hat{f}_\gamma \chi_\gamma(\mathbf{xyz})\Big) \right] \\ &= \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y},\mathbf{z}}\left[\sum_{\alpha,\beta,\gamma} \hat{f}_\alpha \hat{f}_\beta \hat{f}_\gamma \chi_\alpha(\mathbf{x})\chi_\beta(\mathbf{y})\chi_\gamma(\mathbf{x})\chi_\gamma(\mathbf{y})\chi_\gamma(\mathbf{z}) \right] \\ &= \sum_{\alpha,\beta,\gamma} \hat{f}_\alpha \hat{f}_\beta \hat{f}_\gamma \mathop{\mathsf{E}}_{\mathbf{x},\mathbf{y},\mathbf{z}} \left[\chi_\alpha(\mathbf{x})\chi_\beta(\mathbf{y})\chi_\gamma(\mathbf{x})\chi_\gamma(\mathbf{y})\chi_\gamma(\mathbf{z})\right]. \end{aligned}$$

Orthonormality implies the expectation is 0 unless $\alpha = \beta = \gamma$, so this is

$$= \sum_\alpha \hat{f}_\alpha^3 \mathop{\mathsf{E}}_{\mathbf{z}}[\chi_\alpha(z)]$$

Now $\mathsf{E}_{\mathbf{z}}[\chi_\alpha(\mathbf{z})] = \mathsf{E}_{\mathbf{z}}\left[\prod_{w\in\alpha} z_w\right]$ which is equal to $\prod_{w\in\alpha}\mathsf{E}[z_w] = (1-2\rho)^{|\alpha|}$ because each coordinate of $\mathbf{z}$ is chosen independently. Hence we get that

$$2\delta \leq \sum_\alpha \hat{f}_\alpha^3 (1-2\rho)^{|\alpha|}. \ \blacksquare$$

The conclusion of Lemma 22.24 is reminiscent of the calculation in the proof of Theorem 22.22, except for the extra factor $(1-2\rho)^{|\alpha|}$. This factor depresses the contribution of $\hat{f}_\alpha$ for large α, allowing us to conclude that the small α's must contribute a lot. This is formalized in the following corollary (which is a simple calculation and left as Exercise 22.8).

Corollary 22.25 *If f passes the long code test with probability* $1/2 + \delta$, *then for* $k = \frac{1}{2\rho}\log\frac{1}{\epsilon}$, *there exists* α *with* $|\alpha| \leq k$ *such that* $\hat{f}_\alpha \geq 2\delta - \epsilon$. $\diamondsuit$

22.7 Proof of Theorem 22.16

We now prove Håstad's' Theorem. The starting point is the $2\mathsf{CSP}_W$ instance φ given by Theorem 22.15, so we know that φ is either satisfiable, or we can satisfy at most ϵ fraction of the constraints, where ϵ is arbitrarily small. Let W be the alphabet size, n be the number of variables and m the number of constraints. We think of an assignment as a function π from $[n]$ to $[W]$. Since the 2CSP instance has the *projection* property, we can think of each constraint φ_r as being equivalent to some function $h : [W] \to [W]$, where the constraint is satisfied by assignment π iff $\pi(j) = h(\pi(i))$.

Håstad's verifier uses the long code, but expects these encodings to be *bifolded*, a technical property we now define and is motivated by the observation that coordinate functions satisfy $\chi_{\{w\}}(-\mathbf{v}) = -\chi_{\{w\}}(\mathbf{v})$ for every vector $\mathbf{v}$.

Definition 22.26 A function $f : \{\pm 1\}^W \to \{\pm 1\}$ is *bifolded* if for all $\mathbf{v} \in \{\pm 1\}^W$, $f(-\mathbf{v}) = -f(\mathbf{v})$. ♢

(Aside: In mathematics we would call such a function *odd* but the term "folding" is more standard in the **PCP** literature where it has a more general meaning.)

Whenever the **PCP** proof is supposed to contain a codeword of the long code, we may assume without loss of generality that the function is bifolded. The reason is that the verifier can identify, for each pair of inputs $\mathbf{v}, -\mathbf{v}$, one designated representative —say the one whose first coordinate is $+1$— and just define $f(-\mathbf{v})$ to be $-f(\mathbf{v})$. One benefit —though of no consequence in the proof— of this convention is that bifolded functions require only half as many bits to represent. We will use the following fact:

Lemma 22.27 *If $f : \{\pm 1\}^W \to \{\pm 1\}$ is bifolded and $\hat{f}_\alpha \neq 0$ then $|\alpha|$ must be an odd number (and in particular, nonzero).* ♢

Proof: By definition,

$$\hat{f}_\alpha = \langle f, \chi_\alpha \rangle = \mathop{\mathsf{E}}_{\mathbf{v}}[f(\mathbf{v}) \prod_{i \in \alpha} \mathbf{v}_i] .$$

If $|\alpha|$ is even then $\prod_{i\in\alpha} \mathbf{v}_i = \prod_{i\in\alpha}(-\mathbf{v}_i)$. So if f is bifolded, the contributions corresponding to $\mathbf{v}$ and $-\mathbf{v}$ cancel each other and the entire expectation is 0. ■

Håstad's verifier. Now we describe Håstad's verifier V_H. V_H expects the proof $\tilde{\pi}$ to consist of a satisfying assignment to φ where the value of each of the n variables is encoded using the (bifolded) long code. Thus the proof consists $n2^W$ bits (rather, $n2^{W-1}$ if we take the bifolding into account), which V_H treats as n functions $f_1, f_2, \ldots, f_n$ each mapping $\{\pm 1\}^W$ to $\{\pm 1\}$. The verifier V_H randomly picks a constraint, say $\varphi_r(i,j)$, in the $2\mathsf{CSP}_W$ instance. Then V_H tries to check (while reading only three bits!) that functions f_i, f_j encode two values in $[W]$ that would satisfy φ_r, in other words, they encode two values w, u satisfying $h(w) = u$ where $h : [W] \to [W]$ is the function describing constraint φ_r. Now we describe this test, which is reminiscent of the long code test we saw earlier.

The Basic Håstad Test.
Given: Two functions $f, g : \{\pm 1\}^W \to \{\pm 1\}$. A function $h : [W] \to [W]$.
Goal: Check if f, g are long codes of two values w, u such that $h(w) = u$.
Test: For $u \in [W]$ let $h^{-1}(u)$ denote the set $\{w : h(w) = u\}$. Note that the sets $\{h^{-1}(u) : u \in [W]\}$ form a partition of $[W]$. For a string $\mathbf{y} \in \{\pm 1\}^W$ we define $\mathcal{H}^{-1}(\mathbf{y})$ as the string in $\{\pm 1\}^W$ such that for every $w \in [W]$, the wth bit of $\mathcal{H}^{-1}(\mathbf{y})$ is $y_{h(w)}$. In other words, for each $u \in [W]$, the bit y_u appears in $\mathcal{H}^{-1}(\mathbf{y})$ in all coordinates corresponding to $h^{-1}(u)$. V_H chooses uniformly at random $\mathbf{v}, \mathbf{y} \in \{\pm 1\}^W$ and chooses $\mathbf{z} \in \{\pm 1\}^W$ by letting $z_i = +1$ with probability $1 - \rho$ and $z_i = -1$ with probability ρ. It then accepts if

$$f(\mathbf{v})g(\mathbf{y}) = f(\mathcal{H}^{-1}(\mathbf{y})\mathbf{v}\mathbf{z}) \tag{4}$$

and rejects otherwise.

Translating back from $\{\pm 1\}$ to $\{0,1\}$, note that V_H's test is indeed linear, as it accepts iff $\tilde{\pi}[i_1] + \tilde{\pi}[i_2] + \tilde{\pi}[i_3] = b$ for some $i_1, i_2, i_3 \in [n2^W]$ and $b \in \{0,1\}$. (The bit b can indeed equal 1 because of the way V_H ensures the bifolding property.)

Now since ρ, ϵ can be arbitrarily small the next claim suffices to prove the Theorem. (Specifically, making $\rho = \epsilon^{1/3}$ makes the completeness parameter at least $1 - \epsilon^{1/3}$ and the soundness at most $1/2 + \epsilon^{1/3}$.)

Claim 22.28 *(Main) If φ is satisfiable, then there is a proof which V_H accepts with probability $1-\rho$. If $\mathsf{val}(\varphi) \le \epsilon$ then V_H accepts no proof with probability more than $1/2 + \delta$ where $\delta = \sqrt{\epsilon/\rho}$.* ♢

The rest of the section is devoted to proving Claim 22.28.

Completeness part; easy. If φ is satisfiable, then take any satisfying assignment $\pi : [n] \to [W]$ and form a proof for V_H containing the bifolded long code encodings of the n values. (As already noted, coordinate functions are bifolded.) To show that V_H accepts this proof with probability $1-\rho$, it suffices to show that the Basic Håstad Test accepts with probability $1-\rho$ for every constraint.

Suppose f, g are long codes of two integers w, u satisfying $h(w) = u$. Then, using the fact that for $x \in \{\pm 1\}$, $x^2 = 1$,

$$\begin{aligned} f(\mathbf{v})g(\mathbf{y})f(\mathcal{H}^{-1}(\mathbf{y})\mathbf{v}\mathbf{z}) &= \mathbf{v}_w \mathbf{y}_u (\mathcal{H}^{-1}(\mathbf{y})_w \mathbf{v}_w \mathbf{z}_w) \\ &= \mathbf{v}_w \mathbf{y}_u (\mathbf{y}_{h(w)} \mathbf{v}_w \mathbf{z}_w) \qquad = \mathbf{z}_w. \end{aligned}$$

Hence V_H accepts iff $\mathbf{z}_w = 1$, which happens with probability $1 - \rho$.

Soundness of V_H; more difficult. We first show that if the Basic Håstad Test accepts two functions f, g with probability significantly more than $1/2$, then the Fourier transforms of f, g must be correlated. To formalize this we define for $\alpha \subseteq [W]$,

$$h_2(\alpha) = \{u \in [W] : \ |h^{-1}(u) \cap \alpha| \ \text{is odd}\} \tag{5}$$

Notice in particular that for *every* $t \in h_2(\alpha)$ there is at least one $w \in \alpha$ such that $h(w) = t$.

In the next Lemma δ is allowed to be negative. It is the only place where we use the bifolding property.

Lemma 22.29 *Let $f, g : \{\pm 1\}^W \to \{\pm 1\}$, be bifolded functions and $h : [W] \to [W]$ be such that they pass the Basic Håstad Test (4) with probability at least $1/2 + \delta$. Then*

$$\sum_{\alpha \subseteq [W], \alpha \neq \emptyset} \hat{f}_\alpha^2 \hat{g}_{h_2(\alpha)} (1-2\rho)^{|\alpha|} \ge 2\delta \tag{6}$$

♢

Proof: By hypothesis, f, g are such that $E[f(\mathbf{v})f(\mathbf{v}\mathcal{H}^{-1}(\mathbf{y})\mathbf{z})g(\mathbf{y})] \ge 2\delta$. Replacing f, g by their Fourier expansions we get:

$$\begin{aligned} 2\delta \le &= \mathop{\mathsf{E}}_{\mathbf{v},\mathbf{y},\mathbf{z}} \left[(\sum_\alpha \hat{f}_\alpha \chi_\alpha(\mathbf{v}))(\sum_\beta \hat{g}_\beta \chi_\beta(\mathbf{y}))(\sum_\gamma \hat{f}_\gamma \chi_\gamma(\mathbf{v}\mathcal{H}^{-1}(\mathbf{y})\mathbf{z})) \right] \\ &= \sum_{\alpha,\beta,\gamma} \hat{f}_\alpha \hat{g}_\beta \hat{f}_\gamma \mathop{\mathsf{E}}_{\mathbf{v},\mathbf{y},\mathbf{z}} \left[\chi_\alpha(\mathbf{v})\chi_\beta(\mathbf{y})\chi_\gamma(\mathbf{v})\chi_\gamma(\mathcal{H}^{-1}(\mathbf{y}))\chi_\gamma(\mathbf{z}) \right] . \end{aligned}$$

By orthonormality this simplifies to

$$\begin{aligned} &= \sum_{\alpha,\beta} \hat{f}_\alpha^2 \hat{g}_\beta \mathop{\mathsf{E}}_{\mathbf{y},\mathbf{z}} \left[\chi_\beta(\mathbf{y})\chi_\alpha(\mathcal{H}^{-1}(\mathbf{y}))\chi_\alpha(\mathbf{z}) \right] \\ &= \sum_{\alpha,\beta} \hat{f}_\alpha^2 \hat{g}_\beta (1-2\rho)^{|\alpha|} \mathop{\mathsf{E}}_{\mathbf{y}} \left[\chi_\alpha(\mathcal{H}^{-1}(\mathbf{y})\chi_\beta(\mathbf{y}) \right] \end{aligned} \tag{7}$$

since $\chi_\alpha(\mathbf{z}) = (1-2\rho)^{|\alpha|}$, as noted in our analysis of the long code test. Now we have

$$\begin{aligned}\mathop{\mathsf{E}}_{\mathbf{y}}[\chi_\alpha(\mathcal{H}^{-1}(\mathbf{y}))\chi_\beta(\mathbf{y})] &= \mathop{\mathsf{E}}_{\mathbf{y}}[\prod_{w\in\alpha}\mathcal{H}^{-1}(\mathbf{y})_w \prod_{u\in\beta}\mathbf{y}_u] \\ &= \mathop{\mathsf{E}}_{\mathbf{y}}[\prod_{w\in\alpha}\mathbf{y}_{h(w)}\prod_{u\in\beta}\mathbf{y}_u],\end{aligned}$$

which is 1 if $h_2(\alpha) = \beta$ and 0 otherwise. Hence (7) simplifies to

$$\sum_\alpha \hat{f}_\alpha^2 \hat{g}_{h_2(\alpha)}(1-2\rho)^{|\alpha|}.$$

Finally we note that since the functions are assumed to be bifolded, the Fourier coefficients $\hat{f}_\emptyset$ and $\hat{g}_\emptyset$ are zero. Thus those terms can be dropped from the summation and the Lemma is proved. ■

The following Lemma completes the proof of the Claim 22.28 and hence of Håstad's 3-bit PCP Theorem.

Lemma 22.30 *Suppose φ is an instance of $2\mathsf{CSP}_W$ such that $\mathsf{val}(\varphi) < \epsilon$. If ρ, δ satisfy $\rho\delta^2 > \epsilon$ then verifier V_H accepts any proof with probability at most $1/2 + \delta$.* ♢

Proof: Suppose V_H accepts a proof $\tilde{\pi}$ of length $n2^W$ with probability at least $1/2 + \delta$. We give a probabilistic construction of an assignment π to the variables of φ such that the expected fraction of satisfied constraints is at least $\rho\delta^2$, whence it follows by the probabilistic method that a specific assignment π exists that lives up to this expectation. This contradicts the hypothesis if $\rho\delta^2 > \epsilon$.

The distribution from which π is chosen. We can think of $\tilde{\pi}$ as providing, for every $i \in [n]$, a function $f_i : \{\pm 1\}^W \to \{\pm 1\}$. The probabilistic construction of assignment π comes in two steps: we first use f_i to define a distribution $\mathcal{D}_i$ over $[W]$ as follows: select $\alpha \subseteq [W]$ with probability $\hat{f}_\alpha^2$ where $f = f_i$ and then select w at random from α. This is well-defined because $\sum_\alpha \hat{f}_\alpha^2 = 1$ and (due to bifolding) the fourier coefficient $f_\emptyset$ corresponding to the empty set is 0. We then pick $\pi[i]$ by drawing a random sample from distribution $\mathcal{D}_i$. Thus the assignment π is a random element of the product distribution $\prod_{i=1}^m \mathcal{D}_i$. We wish to show

$$\mathop{\mathsf{E}}_{\pi}[\mathop{\mathsf{E}}_{r\in[m]}[\pi \text{ satisfies } r\text{th constraint}]] \geq \rho\delta^2. \tag{8}$$

The analysis. For every constraint φ_r where $r \in [m]$ denote by $1/2 + \delta_r$ the conditional probability that the Basic Håstad Test accepts $\tilde{\pi}$, conditioned on V_H having picked φ_r. (Note: δ_r could be negative.) Then the acceptance probability of V_H is $\mathsf{E}_r[\frac{1}{2} + \delta_r]$ and hence $\mathsf{E}_r[\delta_r] = \delta$. We show that

$$\Pr_\pi[\pi \text{ satisfies } \varphi_r] \geq \rho\delta_r^2, \tag{9}$$

whence it follows that the left hand side of (8) is (by linearity of expectation) at least $\rho E_{r\in[m]}[\delta_r^2]$. Since $E[X^2] \geq E[X]^2$ for any random variable, this in turn is at least $\rho(E_r[\delta_r])^2 \geq \rho\delta^2$. Thus to finish the proof it only remains to prove (9).

Let $\varphi_r(i,j)$ be the rth constraint and let h be the function describing this constraint, so that

$$\pi \text{ satisfies } \varphi_r \qquad \text{iff} \qquad h(\pi[i]) = \pi[j].$$

Let I_r be the indicator random variable for the event $h(\pi[i] = \pi[j])$. From now on we use the shorthand $f = f_i$ and $g = f_j$. What is the chance that a pair of assignments $\pi[i] \in_R D_i$ and $\pi[j] \in_R D_j$ will satisfy $\pi[j] = h(\pi[i])$? Recall that we pick these values by choosing α with probability $\hat{f}_\alpha^2$, β with probability $\hat{g}_\beta^2$ and choosing $\pi[i] \in_R \alpha, \pi[j] \in_R \beta$. Assume that α is picked first. The conditional probability that $\beta = h_2(\alpha)$ is $\hat{g}_{h_2(\alpha)}^2$. If $\beta = h_2(\alpha)$, we claim that the conditional probability of satisfying the constraint is at least $1/|\alpha|$. The reason is that by definition, $h_2(\alpha)$ consists of u such that $\left|h^{-1}(u) \cap \alpha\right|$ is odd, and an odd number cannot be 0! Thus regardless of which value $\pi[j] \in h_2(\alpha)$ we pick, there *exists* $w \in \alpha$ with $h(w) = \pi[j]$, and the conditional probability of picking such a w as $\pi[i]$ is at least $1/|\alpha|$. Thus, we have that

$$\sum_\alpha \frac{1}{|\alpha|} \hat{f}_\alpha^2 \hat{g}_{h_2(\alpha)}^2 \leq \mathop{\mathsf{E}}_{D_i,D_j}[I_r] \tag{10}$$

This is similar to (but not quite the same as) the expression in Lemma 22.29, according to which

$$2\delta_r \leq \sum_\alpha \hat{f}_\alpha^2 \hat{g}_{h_2(\alpha)} (1-2\rho)^{|\alpha|}.$$

However, since one can easily see that $(1-2\rho)^{|\alpha|} \leq \dfrac{2}{\sqrt{\rho\,|\alpha|}}$ we have

$$2\delta_r \leq \sum_\alpha \hat{f}_\alpha^2 \left|\hat{g}_{h_2(\alpha)}\right| \frac{2}{\sqrt{\rho\,|\alpha|}}.$$

Rearranging,

$$\delta_r \sqrt{\rho} \leq \sum_\alpha \hat{f}_\alpha^2 \left|\hat{g}_{h_2(\alpha)}\right| \frac{1}{\sqrt{|\alpha|}}.$$

Applying the Cauchy-Schwartz inequality, $\sum_i a_i b_i \leq (\sum_i a_i^2)^{1/2} (\sum_i b_i^2)^{1/2}$, with $\hat{f}_\alpha \left|\hat{g}_{\pi_2(\alpha)}\right| \frac{1}{\sqrt{|\alpha|}}$ playing the role of the a_i's and $\hat{f}_\alpha$ playing that of the b_i's, we obtain

$$\delta_r \sqrt{\rho} \leq \sum_\alpha \hat{f}_\alpha^2 \left|\hat{g}_{h_2(\alpha)}\right| \frac{1}{\sqrt{|\alpha|}} \leq \left(\sum_\alpha \hat{f}_\alpha^2\right)^{1/2} \left(\sum_\alpha \hat{f}_\alpha{}^2 \hat{g}_{h_2(\alpha)}^2 \frac{1}{|\alpha|}\right)^{1/2} \tag{11}$$

Since $\sum_\alpha \hat{f}_\alpha^2 = 1$, by squaring (11) and combining it with (10) we get that for every r,

$$\delta_r^2 \rho \leq \mathop{\mathsf{E}}_{\mathcal{D}_i,\mathcal{D}_j}[I_r],$$

which proves (9) and finishes the proof. ■

22.8 Hardness of approximating SET-COVER

In the SET-COVER problem we are given a ground set $\mathcal{U}$ and a collection of its subsets $S_1, S_2, \ldots, S_n$ whose union is $\mathcal{U}$, and we desire the smallest subcollection I such that $\cup_{i \in I} S_i = \mathcal{U}$. Such a subcollection is called a *set cover* and its *size* is $|I|$. An algorithm is said to ρ-approximate this problem, where $\rho < 1$ if for every instance it finds a set cover of size at most OPT/ρ, where OPT is the size of the smallest set cover.

Theorem 22.31 ([LY94]) *If for any constant $\rho > 0$ there is an algorithm that ρ-approximates* SET-COVER *then* $\mathbf{P} = \mathbf{NP}$. *Specifically for every $\epsilon, W > 0$ there is a polynomial-time transformation f from* 2CSP$_W$ *instances to instances of* SET-COVER *such that if the* 2CSP$_W$ *instance is regular and satisfies the projection property then*

$$\mathsf{val}(\varphi) = 1 \Rightarrow f(\varphi) \text{ has a set cover of size } N$$

$$\mathsf{val}(\varphi) < \epsilon \Rightarrow f(\varphi) \text{ has no set cover of size} \frac{N}{4\sqrt{\epsilon}},$$

where N depends upon φ. ♢

Actually one can prove a somewhat stronger result; see the note at the end of the proof.

The proof needs the following gadget.

Definition 22.32 *((k,ℓ)-set gadget)* A (k,ℓ)-*set gadget* consists of a ground set $\mathcal{B}$ and some of its subsets $C_1, C_2, \ldots, C_\ell$ with the following property: every collection of at most k sets out of $C_1, \overline{C_1}, C_2, \overline{C_2}, \ldots, C_\ell, \overline{C_\ell}$ that is a set cover for $\mathcal{B}$ must include both C_i and $\overline{C_i}$ for some i. ♢

The following Lemma is left as Exercise 22.13.

Lemma 22.33 *There is an algorithm that given any k, ℓ, runs in time* $\mathrm{poly}(m, 2^\ell)$ *and outputs a (k,ℓ)-set gadget.* ♢

We can now prove Theorem 22.31. We give a reduction from $2\mathsf{CSP}_W$, specifically, the instances obtained from Theorem 22.15.

Let φ be an instance of $2\mathsf{CSP}_W$ such that either $\mathsf{val}(\varphi) = 1$ or $\mathsf{val}(\varphi) < \epsilon$ where ϵ is some arbitrarily small constant. Suppose it has n variables and m constraints. Let Γ_i denote the set of constraints in which the ith variable is the first variable, and Δ_i the set of constraints in which it is the second variable.

The construction. Construct a (k, W)-set gadget $(\mathcal{B}; C_1, C_2, \ldots, C_W)$ where $k > 2/\sqrt{\epsilon}$. Since variables take values in $[W]$, we can associate a set C_u with each variable value u.

The instance of SET-COVER is as follows. The ground set is $[m] \times \mathcal{B}$, which we will think of as m copies of $\mathcal{B}$, one for each constraint of φ. The number of subsets is nW; for each variable $i \in [n]$ and value $u \in [W]$ there is a subset $S_{i,u}$ which is the union of the following sets: $\{r\} \times C_u$ for each $r \in \Delta_i$ and $\{r\} \times \mathcal{B} \setminus C_{h(u)}$ for each $r \in \Gamma_i$. The use of complementary sets like this is at the root of how the gadget allows 2CSP (with projection property) to be encoded as SET-COVER.

The analysis. If the $2\mathsf{CSP}_W$ instance is satisfiable then we exhibit a set cover of size n. Let $\pi : [n] \to W$ be any satisfying assignment where $\pi(i)$ is the value of the ith variable. We claim that the collection of n subsets given by $\left\{S_{i,\pi[i]} : i \in [n]\right\}$ is a set cover. It suffices to show that their union contains $\{r\} \times \mathcal{B}$ for each constraint r. But this is trivial since if i is the first variable of constraint r and j the second variable, then by definition $S_{j,\pi[j]}$ contains $\{r\} \times C_{\pi[j]}$ and $S_{i,\pi[i]}$ contains $\{r\} \times \mathcal{B} \setminus C_{\pi[j]}$, and thus $S_{i,\pi[i]} \cup S_{j,\pi[j]}$ contains $\{r\} \times \mathcal{B}$.

Conversely, suppose less than ϵ fraction of the constraints in the $2\mathsf{CSP}_W$ instance are simultaneously satisfiable. We claim that every set cover must have at least nT sets, for $T = \frac{1}{4\sqrt{\epsilon}}$. For contradiction's sake suppose a set cover of size less than nT exists. Let us probabilistically construct an assignment for the $2\mathsf{CSP}_W$ instance as follows. For each variable i, say that a value u is *associated* with it if $S_{i,u}$ is in the set cover. We pick a value for i by randomly picking one of the values associated with it. It suffices to prove the following claim since our choice of k ensures that $8T < k$.

CLAIM: *If $8T < k$ then the expected number of constraints satisfied by this assignment is more than $\frac{m}{16T^2}$.*

PROOF: Call a variable *good* if it has less than $4T$ values associated with it. The average number of values associated per variable is less than T, so less than $1/4$ of the variables have more than $4T$ values associated with them. Thus less than $1/4$ of the variables are not good.

Since the 2CSP instance is regular, each variable occurs in the same number of clauses. Thus the fraction of constraints containing a variable that is not good is less than $2 \times 1/4 = 1/2$. Thus for more than $1/2$ of the constraints both variables in them are good. Let r be such a constraint and i, j be the variables in it. Then $\{r\} \times \mathcal{B}$ is covered by the union of $\cup_u S_{i,u}$ and $\cup_v S_{j,v}$ where the unions are over values associated with the variables i, j respectively. Since $8T < k$, the definition of a (k, W)-set gadget implies that any cover of $\mathcal{B}$ by less than $8T$ sets must contain two sets that are complements of one another. We conclude that there are values u, v associated with i, j respectively such that $h(u) = v$. Thus when we randomly construct an assignment by picking for each variable one of the values associated with it, these values are picked with probability at least $1/4T \times 1/4T = 1/16T^2$, and then the rth constraint gets satisfied. The claim (and hence Theorem 22.31) now follows by linearity of expectation. ■

Remark 22.34
The same proof actually can be used to prove a stronger result: there is a constant $c > 0$ such that if there is an algorithm that α-approximates SET-COVER for $\alpha = c/\log n$ then $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{O(\log n)})$. The idea is to use Raz's parallel repetition theorem where the number of repetitions t is superconstant so that the soundness is $1/\log n$. However, the running time of the reduction is $n^{O(t)}$, which is slightly superpolynomial.

22.9 Other PCP Theorems: A Survey

As mentioned in the introduction, proving inapproximability results for various problems often requires proving new **PCP** Theorems. We already saw one example, namely, Håstad's 3-bit **PCP** Theorem. Now we survey some other variants of the **PCP** Theorem that have been proved.

22.9.1 PCP's with sub-constant soundness parameter

The way we phrased Theorem 22.15, the soundness is an arbitrary small constant 2^{-t}. From the proof of the theorem it was clear that the reduction used to prove this **NP**-hardness runs in time n^t (since it forms all t-tuples of constraints). Thus if t is larger than a constant, the running time of the reduction is superpolynomial. Nevertheless, several hardness results use superconstant values of t. They end up not showing **NP**-hardness, but instead showing the nonexistence of a good approximation algorithms assuming **NP** does not have say $n^{\log n}$ time deterministic algorithms (this is still a very believable assumption). We mentioned this already in Remark 22.34 at the end of Section 22.8.

It is still an open problem to prove the **NP**-hardness of 2CSP for a factor ρ that is smaller than any constant. If instead of 2CSP one looks at 3CSP or 4CSP then one can achieve low soundness using larger alphabet size, while keeping the running time polynomial [RS97]. Often these suffice in applications.

22.9.2 Amortized query complexity

Some applications require binary-alphabet **PCP** systems enjoying a tight relation between the number of queries (which can be an arbitrarily large constant) and the soundness parameter. The relevant parameter here turns out to be the *free bit complexity* [FK93, BS94]. This parameter is defined as follows. Suppose the number of queries is q. After the verifier has picked its random string, and picked a sequence of q addresses, there are 2^q possible sequences of bits that could be contained in those addresses. If the verifier accepts for only t of those sequences, then we say that the free bit parameter is $\log t$ (note that this number need not be an integer). In fact, for proving hardness result for MAX-INDSET and MAX-CLIQUE, it suffices to consider the *amortized free bit complexity* [BGS95]. This parameter is defined as $\lim_{s \to 0} f_s/\log(1/s)$, where f_s is the number of free bits needed by the verifier to ensure the soundness parameter is at most s (with completeness at least say $1/2$). Håstad constructed systems with amortized free bit complexity tending to zero [Hås96]. That is, for every $\epsilon > 0$, he gave a **PCP**-verifier for **NP** that uses $O(\log n)$ random bits and ϵ amortized free bits. The completeness is 1. He then used this **PCP** system to show (borrowing the basic framework from [FGL$^+$91, FK93, BS94, BGS95]) that MAX-INDSET (and so, equivalently, MAX-CLIQUE) is **NP**-hard to $n^{-1+\epsilon}$-approximate for arbitrarily small $\epsilon > 0$.

22.9.3 2-bit tests and powerful fourier analysis

Recent advances on Håstad's line of work consist of using more powerful ideas from Fourier analysis. The Fourier analysis in Håstad's proof hardly uses the fact that the functions being tested are Boolean. However, papers of Kahn, Kalai, Linial [KKL88], Friedgut [Fri99], and Bourgain [Bou02] have led to important new insights into the Fourier coefficients of Boolean functions, which in turn have proved useful in analysing **PCP** verifiers. (See also Note 22.21.) The main advantage is for designing verifiers that read only 2 bits in the proof, which arise while proving hardness results for a variety of graph problems such as VERTEX-COVER, MAX-CUT and SPARSEST-CUT.

These new results follow Håstad's overall idea, namely, to show that if the verifier accepts some provided functions with good probability, then the function has a few large fourier coefficients (see Corollary 22.25 for example). However, Håstad's analysis (even for the long code test in Section 22.6) inherently requires the verifier to query 3 bits in the proof, and we briefly try to explain why. For simplicity we focus on the long code test. We did a simple analysis of the long code test to arrive at the conclusion of Lemma 22.24:

$$\sum_{\alpha} \hat{f}_{\alpha}^{3}(1-2\rho)^{|\alpha|} \geq 2\delta,$$

where $1/2+\delta$ is the probability that the verifier accepts the function. From this fact, Corollary 22.25 concludes that at least one fourier coefficient has value at least $c = c(\delta, \rho) > 0$. This is a crucial step because it lets us conclude that f has some (admittedly very weak) connection with some small number of codewords in the long code.

One could design an analogous 2-bit test. The first part of the above analysis still goes through but in the conclusion the cubes get replaced by squares:

$$\sum_{\alpha} \hat{f}_{\alpha}^{2}(1-2\rho)^{|\alpha|} \geq 2\delta. \qquad (12)$$

For a non-Boolean function this condition is not enough to let us conclude that some fourier coefficient of f has value at least $c = c(\delta, \rho) > 0$. However, the following lemma of Bourgain allows such a conclusion if f is Boolean. We say that a function $f : \{0,1\}^n \to \{0,1\}$ is a *k-junta* if it depends only on k of the n variables. Note that Parseval's identity implies that at least one fourier coefficient of a k-junta is $1/2^{k/2}$. The next Lemma implies that if a boolean function f is such that the LHS of (12) is at least $1-\rho^t$ where $t > 1/2$, then f is close to a k-junta for a small k.

Lemma 22.35 *([Bou02]) For every $\epsilon, \delta > 0$ and integer r there is a constant $k = k(r, \epsilon, \delta)$ such that if*

$$\sum_{\alpha : |\alpha| > r} \hat{f}_{\alpha}^{2} < \frac{1}{r^{1/2+\epsilon}},$$

then f has agreement $1-\delta$ with a k-junta. $\diamondsuit$

We suspect that there will be many other uses of fourier analysis in **PCP** constructions.

22.9.4 Unique games and threshold results

Håstad's ideas led to determination of the approximation threshold for several problems. But the status of other problems such as VERTEX-COVER and MAX-CUT remained open. In 2002 Khot [Kho02] proposed a new complexity theoretic conjecture called the *unique games conjecture* (UGC) that is stronger than $\mathbf{P} \neq \mathbf{NP}$ but still consistent with current knowledge. This conjecture concerns a special case of $2\mathsf{CSP}_W$ in which the constraint function is a *permutation* on $[W]$. In other words, if the constraint φ_r involves variables i, j, the constraint function h is a bijective mapping from $[W]$ to $[W]$. Then assignment $u_1, u_2, \ldots, u_n$ to the variables satisfies this constraint iff $u_j = h(u_i)$. According to UGC, for every constants $\epsilon, \delta > 0$ there is a domain size $W = W(\epsilon, \delta)$ such that there is no polynomial-time algorithm that given such an instance of $2\mathsf{CSP}_W$ with $\mathsf{val}(\cdot) \geq 1 - \epsilon$ produces an assignment that satisfies δ fraction of constraints.[2]

Khot suggested that current algorithmic techniques seem unable to design such an algorithm (an insight that seems to have been confirmed by lack of progress in the last few years, despite much effort). He also showed that this conjecture implies several strong results about hardness of approximation. The reason in a nutshell is that the Fourier analysis technique of Håstad (fortified with the above-mentioned discoveries regarding Fourier analysis of Boolean functions) can be sharpened if one starts with the instances of $2\mathsf{CSP}_W$ with the uniqueness constraint.

Subsequently, a slew of results have shown optimal or threshold results about hardness of approximation (often using some of the advanced fourier analysis mentioned above) assuming UGC is true.f For instance UGC implies that there is no polynomial-time algorithm that computes a $1/2 + \delta$-approximation to VERTEX-COVER for any $\delta > 0$ [KR08], and similarly no algorithm that computes a 0.878-approximation to MAX-CUT [KKMO05, MOO05]. Both of these lead to threshold results since these ratios are also the ones achieved by the current approximation algorithms.

Thus it is of great interest to prove or disprove the unique games conjecture. Algorithms designers have tried to disprove the conjecture using clever tools from semidefinite programming, and currently the conjecture seems truly on the fine line between being true and being false. It is known that it will suffice to restrict attention to the further subcase where the constraint function h is linear —i.e., the constraints are linear equations mod W in two variables.

22.9.5 Connection to Isoperimetry and Metric Space Embeddings

A *metric space* (X, d) consists of set of points X and a function d mapping pairs of points to nonnegative real numbers satisfying (a) $d(i,j) = 0$ iff $i = j$. (b) $d(i,j) + d(j,k) \geq d(i,k)$ (*triangle inequality*). An *embedding* of space (X, d) into space (Y, d') is a function $f : X \to Y$. Its *distortion* is the maximum over all point pairs $\{i,j\}$ of the quantities $\frac{d'(f(i),f(j))}{d(i,j)}, \frac{d(i,j)}{d'(f(i),f(j))}$. It is of great interest in algorithm design (and in mathematics) to understand the minimum *distortion* required to embed one family of metric spaces into another. One interesting subcase is where the host space (Y, d') is a subset of the ℓ_1 metric space on $\Re^n$ for some n (in other words, distance d' is defined using the ℓ_1 norm). Bourgain showed that every n-point metric space embeds in ℓ_1 with distortion $O(\log n)$. This fact is important in design of algorithms for graph partitioning problems such as SPARSEST-CUT. In that context, a metric called ℓ_2^2 had been identified. Goemans and Linial conjectured that this metric would be embeddable in ℓ_1 with distortion $O(1)$. If the conjecture were true we would have an $O(1)$-approximation algorithm for SPARSEST-CUT. Khot and Vishnoi [KV05] disproved the conjecture, using a construction of an interesting ℓ_2^2 metric that is inspired by the advanced **PCP** Theorems discussed in this chapter. The main idea is that since there is an intimate relationship between ℓ_1 metrics and cuts, one has to construct a graph whose cut structure and isoperimetry properties are tightly controlled. So Khot and Vishnoi use a hypercube-like graph, and use Fourier analysis to show its isoperimetry properties. (See Note 22.21.)

Their work has inspired other results about lower bounds on the distortions of metric embeddings.

[2]In fact, Khot phrased the UGC as the even stronger conjecture that solving this problem is **NP**-hard.

Chapter notes and history

As mentioned in the notes to Chapter 11, the **PCP** Theorem was proved in 1992 in the early versions of the papers [AS92, ALM$^+$92]. The AS-ALMSS proof of the **PCP** Theorem resisted simplification for over a decade. The overall idea of that proof (as indeed in **MIP** = **NEXP**) is similar to the proof of Theorem 11.19. (Indeed, Theorem 11.19 is the only part of the original proof that still survives in our writeup.) However, in addition to using encodings based upon the Walsh-Hadamard code the proof also used encodings based upon low degree multivariate polynomials. These have associated procedures analogous to the linearity test and local decoding, though the proofs of correctness are a fair bit harder. The proof also drew intuition from the topic of self-testing and self-correcting programs [BLR90, RS92], which was surveyed in Section 8.6. The alphabet reduction in the AS-ALMSS proof was also somewhat more complicated. A draft writeup of the original proof is available on this book's website. (We dropped it from the published version in favor of Dinur's proof but feel it is interesting in its own right and may be useful in future research.)

Dinur's main contribution in simplifying the proof is the gap amplification lemma (Lemma 22.5), which allows one to iteratively improve the soundness parameter of the **PCP** from very close to 1 to being bounded away from 1 by some positive constant. This allows her to use a simpler alphabet reduction. In fact, the alphabet reduction is the only part of the proof that now uses the "proof verification" viewpoint, and one imagines that in a few years this too will be replaced by a purely combinatorial construction. A related open problem is to find a Dinur-style proof of **MIP** = **NEXP**.

We also note that Dinur's general strategy is somewhat reminiscent of the zig-zag construction of expander graphs and Reingold's deterministic logspace algorithm for undirected connectivity described in Chapter 20, which suggests that more connections are waiting to be made between these different areas of research.

As mentioned in the notes at the end of Chapter 11, Papadimitriou and Yannakakis [PY88] had shown around 1988 that if it is **NP**-hard to ρ-approximate MAX-3SAT for some $\rho < 1$ then it is also **NP**-hard to ρ'-approximate a host of other problems where ρ' depends upon the problem. Thus after the discovery of the **PCP** Theorem, attention turned towards determining the exact approximation threshold for problems; see for example [BS94, BGS95]. Håstad's threshold results for MAX-CLIQUE [Hås96] and MAX-3SAT [Hås97] came a few years later and represented a quantum jump in our understanding.

The issue of parallel repetition comes from the paper of Fortnow, Rompel, and Sipser [FRS88] that erroneously claimed that $\mathsf{val}(\varphi^{*t}) = \mathsf{val}(\varphi)^t$ for every 2CSP φ and $t \in \mathbb{N}$. However, Fortnow [For89] soon found a counter example (see Exercise 22.6). Papers Lapidot and Shamir [LS91], Feige-Lovasz [FL92], which predate Raz's paper, imply hardness results for 2CSP but the running time of the reduction is superpolynomial. Verbitsky [Ver94] and Feige and Kilian [FK93] proved weaker versions of Raz's Theorem (Theorem 22.15). Raz's proof of the parallel repetition is based on an extension of techniques developed by Razborov [Raz90] in the context of communication complexity. The proof is beautiful but quite complex, though recently Holenstein [Hol07] gave some simplifications for Raz's proof; a writeup of this simplified proof is available from this book's website.

The hardness result for INDSET in Lemma 22.8 can be improved so that for all $\epsilon > 0$, $n^{-1+\epsilon}$-approximation in **NP**-hard in graphs with n vertices. This result is due to [Hås96], which caps a long line of other work [FGL$^+$91, AS92, ALM$^+$92, BS94, BGS95]. The use of expanders in the reduction of Lemma 22.8 is from [AFWZ95]. Note that a $1/n$-approximation is trivial: just output a single vertex, which is always an independent set. Thus this result can be viewed as a *threshold* result.

The hardness of SET-COVER is due to Lund and Yannakakis [LY94], which was also the first paper to implicitly use 2CSP$_W$ with projection property; the importance of this problem was identified in [Aro94, ABSS93], where it was called *label cover* used to prove other results. This problem is now ubiquitous in **PCP** literature.

A threshold result for SET-COVER was shown by Feige [Fei96]: computing $(1+\delta)/\ln n$ approximation is hard for every $\delta > 0$, whereas we know a simple $1/\ln n$-approximation algorithm.

See Arora and Lund [AL95] for a survey circa 1995 of how to prove the basic results about hardness of approximation. See Khot [Kho05] for a more recent survey about the results that use fourier analysis.

Exercises

22.1 Prove Equation (1). H459

22.2 Let $G = (V, E)$ be a λ-expander graph for some $\lambda \in (0, 1)$. Let S be a subset of V with $|S| = \beta|V|$ for some $\beta \in (0, 1)$. Let $(X_1, \ldots, X_\ell)$ be a tuple of random variables denoting the vertices of a uniformly chosen (ℓ–1)-step path in G. Then, prove that

$$(\beta - 2\lambda)^k \leq \Pr[\forall_{i\in[\ell]} X_i \in S] \leq (\beta + 2\lambda)^k$$

H459

22.3 Let S_t be the binomial distribution over t balanced coins. That is, $\Pr[S_t = k] = \binom{t}{k}2^{-t}$. Prove that for every $\delta < 1$, the statistical distance (see Section A.2.6) of S_t and $S_{t+\delta\sqrt{t}}$ is at most 10δ. H459

22.4 Prove that for every non-negative random variable V, $\Pr[V > 0] \geq \mathsf{E}[V]^2/\mathsf{E}[V^2]$.

H459

22.5 In this problem we explore an alternative approach to the Alphabet Reduction Lemma (Lemma 22.6) using Long Codes instead of Welsh-Hadamard codes. We already saw that the *long-code* for a set $\{0, \ldots, W-1\}$ is the function $\mathsf{LC} : \{0, \ldots, W-1\} \to \{0, 1\}^{2^W}$ such that for every $i \in \{0..W{-}1\}$ and a function $f : \{0..W{-}1\} \to \{0, 1\}$, (where we identify f with an index in $[2^w]$) the f^{th} position of $\mathsf{LC}(i)$, denoted by $\mathsf{LC}(i)_f$, is $f(i)$. We say that a function $L : \{0, 1\}^{2^W} \to \{0, 1\}$ is a *long-code codeword* if $L = \mathsf{LC}(i)$ for some $i \in \{0..W{-}1\}$.

- **(a)** Prove that LC is an error-correcting code with distance half. That is, for every $i \neq j \in \{0..W{-}1\}$, the fractional Hamming distance of $\mathsf{LC}(i)$ and $\mathsf{LC}(j)$ is half.
- **(b)** Prove that LC is *locally-decodable*. That is, show an algorithm that given random access to a function $L : 2^{\{0,1\}^W} \to \{0, 1\}$ that is (1–ϵ)-close to $\mathsf{LC}(i)$ and $f : \{0..W{-}1\} \to \{0, 1\}$ outputs $\mathsf{LC}(i)_f$ with probability at least 0.9 while making at most 2 queries to L.
- **(c)** Let $L = \mathsf{LC}(i)$ for some $i \in \{0..W{-}1\}$. Prove the for every $f : \{0..W{-}1\} \to \{0, 1\}$, $L(f) = 1 - L(\overline{f})$, where $\overline{f}$ is the negation of f (i.e. , $\overline{f}(i) = 1 - f(i)$ for every $i \in \{0..W{-}1\}$).
- **(d)** Let T be an algorithm that given random access to a function $L : 2^{\{0,1\}^W} \to \{0, 1\}$, does the following:
  - **(a)** Choose f to be a random function from $\{0..W{-}1\} \to \{0, 1\}$.
  - **(b)** If $L(f) = 1$ then output TRUE.
  - **(c)** Otherwise, choose $g : \{0..W{-}1\} \to \{0, 1\}$ as follows: for every $i \in \{0..W{-}1\}$, if $f(i) = 0$ then set $g(i) = 0$ and otherwise set $g(i)$ to be a random value in $\{0, 1\}$.
  - **(d)** If $L(g) = 0$ then output TRUE; otherwise output FALSE.

  Prove that if L is a long-code codeword (i.e., $L = \mathsf{LC}(i)$ for some i) then T outputs TRUE with probability one.

  Prove that if L is a *linear function* that is non-zero and not a long code codeword then T outputs TRUE with probability at most 0.9.
- **(e)** Prove that LC is *locally testable*. That is, show an algorithm that given random access to a function $L : \{0, 1\}^W \to \{0, 1\}$ outputs TRUE with probability one if L is a long-code codeword and outputs FALSE with probability at least 1/2 if L is not 0.9-close to a long-code codeword, while making at most a constant number of queries to L. H460
- **(f)** Using the test above, give an alternative proof for the Alphabet Reduction Lemma (Lemma 22.6). H460

22.6 ([For89, Fei91]) Consider the following 2CSP instance φ on an alphabet of size 4 (which we identify with $\{0, 1\}^2$). The instance φ has 4 variables $x_{0,0}, x_{0,1}, x_{1,0}, x_{1,1}$ and four constraints $C_{0,0}, C_{0,1}, C_{1,0}, C_{1,1}$. The constraint $C_{a,b}$ looks at the variables $x_{0,a}$ and $x_{1,b}$ and outputs TRUE if and only if $x_{0,a} = x_{1,b}$ and $x_{0,a} \in \{0a, 1b\}$.

- (a) Prove that $\mathsf{val}(\varphi^{*2}) = \mathsf{val}(\varphi)$, where φ^{*t} denotes the 2CSP over alphabet W^t that is the t-times parallel repeated version of φ as in Section 22.3.1. H460
- (b) Prove that for every t, $\mathsf{val}(\varphi^{*t}) \geq \mathsf{val}(\varphi)^{t/2}$.

22.7 (Solvability of Unique Games) We encountered unique games in Section 22.9.4; it is a special case of $\mathsf{2CSP}_W$ in which the constraint function h is a *permutation* on $[W]$. In other words, if constraint φ_r involves variables i, j, then assignment $u_1, u_2, \ldots, u_n$ to the variables satisfies this constraint iff $u_j = h(u_i)$. Prove that there is a polynomial-time algorithm that given such an instance, finds a satisfying assignment if one exists.

22.8 Prove Corollary 22.25.

22.9 Prove that the **PCP** system resulting from the proof of Claim 22.36 (Chapter 11) satisfies the projection property.

22.10 This question explores the notion of noise-senstivity of Boolean functions, which ties in to the discussion in Note 22.21. Let $f : \{\pm 1\}^n \to \{\pm 1\}$ and let $I \subseteq [n]$. Let M_I be the following distribution: we choose $z \in_R M_I$ by for $i \in I$, choose z_i to be $+1$ with probability 1/2 and -1 with probability 1/2 (independently of other choices), for $i \notin I$ choose $z_i = +1$. We define the *variation of f on I* to be $\Pr_{\mathbf{x} \in_R \{\pm 1\}^n, \mathbf{z} \in_R M_I}[f(\mathbf{x}) \neq f(\mathbf{xz})]$.

Suppose that the variation of f on I is less than ϵ. Prove that there exists a function $g : \{\pm 1\}^n \to \mathbb{R}$ such that **(1)** g does not depend on the coordinates in I and **(2)** g is 10ϵ-close to f (i.e., $\Pr_{\mathbf{x} \in_R \{\pm 1\}^n}[f(\mathbf{x}) \neq g(\mathbf{x})] < 10\epsilon$). Can you come up with such a g that outputs values in $\{\pm 1\}$ only?

22.11 For $f : \{\pm 1\}^n \to \{\pm 1\}$ and $\mathbf{x} \in \{\pm 1\}^n$ we define $N_f(\mathbf{x})$ to be the number of coordinates i such that if we let y to be $\mathbf{x}$ flipped at the i^{th} coordinate (i.e., $y = x\mathbf{e}^i$ where $\mathbf{e}^i$ has -1 in the i^{th} coordinate and $+1$ everywhere else) then $f(\mathbf{x}) \neq f(\mathbf{y})$. We define the *average sensitivity* of f, denoted by $as(f)$ to be the expectation of $N_f(\mathbf{x})$ for $\mathbf{x} \in_R \{\pm 1\}^n$.

(a) Prove that for every balanced function $f : \{\pm 1\}^n \to \{\pm 1\}$ (i.e., $\Pr[f(\mathbf{x}) = +1] = 1/2$), $as(f) \geq 1$.

(b) Let f be balanced function from $\{\pm 1\}^n$ to $\{\pm 1\}$ with $as(f) = 1$. Prove that f is a coordinate function or its negation (i.e., $f(\mathbf{x}) = x_i$ or $f(\mathbf{x}) = -x_i$ for some $i \in [n]$ and for every $\mathbf{x} \in \{\pm 1\}^n$). (*Restatement using the language of isoperimetry as in Note 22.21*: If a subset of half the vertices of the hypercube $\{0,1\}^n$ has exactly 2^{n-1} edges leaving it, then there is some i such that this subset is simply the set of vertices where $x_i = 0$ (or $x_i = 1$).)

22.12 ([KM91]) This exercise asks you to give an alternative proof of the Goldreich-Levin Theorem 9.12 using Fourier analysis.

(a) For every function $f : \{\pm 1\}^n \to \mathbb{R}$, denote $\tilde{f}_{\alpha\star} = \sum_{\beta \in \{0,1\}^{n-k}} \hat{f}^2_{\alpha\circ\beta}$, where $\circ$ denotes concatenation and we identify strings in $\{0,1\}^n$ and subsets of $[n]$ in the obvious way. Prove that

$$\tilde{f}_{0^k\star} = \mathop{\mathsf{E}}_{\substack{\mathbf{x},\mathbf{x}' \in_R \{0,1\}^k \\ \mathbf{y} \in_R \{0,1\}^{n-k}}} [f(\mathbf{x} \circ \mathbf{y}) f(\mathbf{x}' \circ \mathbf{y})]$$

H460

(b) Prove that for every $\alpha \in \{0,1\}^k$,

$$\tilde{f}_{\alpha\star} = \mathop{\mathsf{E}}_{\substack{\mathbf{x},\mathbf{x}' \in_R \{0,1\}^k \\ \mathbf{y} \in_R \{0,1\}^{n-k}}} [f(\mathbf{x} \circ \mathbf{y}) f(\mathbf{x}' \circ \mathbf{y}) \chi_\alpha(\mathbf{x}) \chi_\alpha(\mathbf{x}')] \qquad (13)$$

H460

(c) Show an algorithm `Estimate` that given $\alpha \in \{0,1\}^k$, $\epsilon > 0$ and oracle access to $f : \{\pm 1\}^n \to \{\pm 1\}$, runs in time $\mathrm{poly}(n, 1/\epsilon)$ and outputs an estimate of f_α within ϵ accuracy with probability $1 - \epsilon$. H460

(d) Show an algorithm `LearnFourier` that given $\epsilon > 0$ and oracle access to $f : \{\pm 1\}^n \to \{\pm 1\}$, runs in $\mathrm{poly}(n, 1/\epsilon)$ time and outputs a set L of $\mathrm{poly}(1/\epsilon)$ strings such that with probability at least 0.9, for every $\alpha \in \{0,1\}^n$, if $|\hat{f}_\alpha| > \epsilon$ then $\alpha \in L$. H460

(e) Show that the above algorithm implies Theorem 9.12.

22.13 Prove Lemma 22.33, albeit using a randomized algorithm. H460

22.14 Derandomize the algorithm of the previous exercise. H460

22.15 ([ABSS93]) In Problem 11.16 we explored the approximability of the problem of finding the largest feasible subsystem in a system of linear equations over the rationals. Show that there is an $\epsilon > 0$ such that computing an $n^{-\epsilon}$-approximation to this problem is **NP**-hard. H460

22.16 ([PY88]) Suppose we restrict attention to MAX-3SAT in which each variable appears in at most 5 clauses. Show that there is still a constant $\rho < 1$ such that computing a ρ-aproximation to this problem is **NP**-hard. H460

22.17 ([PY88]) In the MAX-CUT problem we are given a graph $G = (V, E)$ and seek to partition the vertices into two sets $S, \overline{S}$ such that we maximize the number of edges $|E(S, \overline{S})|$ between them. Show that there is still a constant $\rho < 1$ such that computing a ρ-aproximation to this problem is **NP**-hard.

22.A Transforming qCSP instances into "nice" instances.

We can transform a qCSP-instance φ into a "nice" 2CSP-instance ψ through the following three claims:

Claim 22.36 *There is a CL- reduction mapping any qCSP instance φ into a $2\mathsf{CSP}_{2^q}$ instance ψ such that*

$$\mathsf{val}(\varphi) \le 1-\epsilon \Rightarrow \mathsf{val}(\psi) \le 1-\epsilon/q \qquad \diamondsuit$$

Proof: Given a qCSP-instance φ over n variables $u_1,\dots,u_n$ with m constraints, we construct the following $2\mathsf{CSP}_{2^q}$ formula ψ over the variables $u_1,\dots,u_n,y_1,\dots,y_m$. Intuitively, the y_i variables will hold the restriction of the assignment to the q variables used by the i^{th} constraint, and we will add constraints to check consistency: that is to make sure that if the i^{th} constraint depends on the variable u_j then u_j is indeed given a value consistent with y_i. Specifically, for every φ_i of φ that depends on the variables $u_1,\dots,u_q$, we add q constraints $\{\psi_{i,j}\}_{j\in[q]}$ where $\psi_{i,j}(y_i,u_j)$ is true iff y_i encodes an assignment to $u_1,\dots,u_q$ satisfying φ_i and u_j is in $\{0,1\}$ and agrees with the assignment y_i. Note that the number of constraints in ψ is qm.

Clearly, if φ is satisfiable then so is ψ. Suppose that $\mathsf{val}(\varphi) \le 1-\epsilon$ and let $u_1,\dots,u_k,y_1,\dots,y_m$ be any assignment to the variables of ψ. There exists a set $S \subseteq [m]$ of size at least ϵm such that the constraint φ_i is violated by the assignment $u_1,\dots,u_k$. For any $i \in S$ there must be at least one $j \in [q]$ such that the constraint $\psi_{i,j}$ is violated. ■

Claim 22.37 *There is an absolute constant d and a CL- reduction mapping any $2\mathsf{CSP}_W$ instance φ into a $2\mathsf{CSP}_W$ instance ψ such that*

$$\mathsf{val}(\varphi) \le 1-\epsilon \Rightarrow \mathsf{val}(\psi) \le 1-\epsilon/(100Wd).$$

and the constraint graph of ψ is d-regular. That is, every variable in ψ appears in exactly d constraints. $\diamondsuit$

Proof: Let φ be a $2\mathsf{CSP}_W$ instance, and let $\{G_n\}_{n\in\mathbb{N}}$ be an explicit family of d-regular expanders. Our goal is to ensure that each variable appears in φ at most $d+1$ times (if a variable appears less than that, we can always add artificial constraints that touch only this variable). Suppose that u_i is a variable that appears in k constraints for some $n > 1$. We will change u_i into k variables $y_i^1,\dots,y_i^k$, and use a different variable of the form y_i^j in the place of u_i in each constraint u_i originally appeared in. We will also add a constraint requiring that y_i^j is equal to $y_i^{j'}$ for every edge (j,j') in the graph G_k. We do this process for every variable in the original instance, until each variable appears in at most d equality constraints and one original constraint. We call the resulting 2CSP-instance ψ. Note that if φ has m constraints then ψ will have at most $m + dm$ constraints.

Clearly, if φ is satisfiable then so is ψ. Suppose that $\mathsf{val}(\varphi) \le 1-\epsilon$ and let $\mathbf{y}$ be any assignment to the variables of ψ. We need to show that $\mathbf{y}$ violates at least $\frac{\epsilon m}{100W}$ of the constraints of ψ. Recall that for each variable u_i that appears k times in φ, the assignment $\mathbf{y}$ has k variables $y_i^1,\dots,y_i^k$. We compute an assignment $\mathbf{u}$ to φ's variables as follows: u_i is assigned the plurality value of $y_i^1,\dots,y_i^k$. We define t_i to be the number of y_i^j's that *disagree* with this plurality value. Note that $0 \le t_i \le k(1-1/W)$ (where W is the alphabet size). If $\sum_{i=1}^n t_i \ge \frac{\epsilon}{4}m$ then we are done. Indeed, by (1) (see Section 22.2.3), in this case we will have at least $\sum_{i=1}^n \frac{t_i}{10W} \ge \frac{\epsilon}{40W}m$ equality constraints that are violated.

Suppose now that $\sum_{i=1}^n t_i < \frac{\epsilon}{4}m$. Since $\mathsf{val}(\varphi) \le 1-\epsilon$, there is a set S of at least ϵm constraints violated in φ by the plurality assignment $\mathbf{u}$. All of these constraints are also present in ψ and since we assume $\sum_{i=1}^n t_i < \frac{\epsilon}{4}m$, at most half of them are given a different value by the assignment $\mathbf{y}$ than the value given by $\mathbf{u}$. Thus the assignment $\mathbf{y}$ violates at least $\frac{\epsilon}{2}m$ constraints in ψ. ■

Claim 22.38 *There is an absolute constant d and a CL-reduction mapping any $2\mathsf{CSP}_W$ instance φ with d'-regular constraint graph for $d \ge d'$ into a $2\mathsf{CSP}_W$ instance ψ such that*

$$\mathsf{val}(\varphi) \le 1-\epsilon \Rightarrow \mathsf{val}(\psi) \le 1-\epsilon/(10d)$$

and the constraint graph of ψ is a $4d$-regular expander, with half the edges coming out of each vertex being self-loops. $\diamondsuit$

Proof: There is a constant d and an explicit family $\{G_n\}_{n\in\mathbb{N}}$ of graphs such that for every n, G_n is a d-regular n-vertex 0.1-expander graph (See Section 22.2.3).

Let φ be a 2CSP-instance as in the claim's statement. By adding self loops, we can assume that the constraint graph has degree d (this can at worst decrease the gap by factor of d). We now add "null" constraints (constraints that always accept) for every edge in the graph G_n. In addition, we add $2d$ null constraints forming self-loops for each vertex. We denote by ψ the resulting instance. Adding these null constraints reduces the fraction of violated constraints by a factor at most four. Moreover, because any regular graph H satisfies $\lambda(H) \leq 1$ and because of λ's subadditivity (see Exercise 21.7, Chapter 21), $\lambda(\psi) \leq \frac{3}{4} + \frac{1}{4}\lambda(G_n) \leq 0.9$ where by $\lambda(\psi)$ we denote the parameter λ of ψ's constraint graph. ■