Locally Expanding Hypergraphs and the Unique Games Conjecture

Eric Purdy[*]

September 25, 2008

Abstract

We examine the hardness of approximating constraint satisfaction problems with k-variable constraints, known as k-CSP's. We are specifically interested in k-CSP's whose constraints are unique, which means that for any assignment to any $k-1$ of the variables, there is a unique assignment to the last variable satisfying the constraint. One fundamental example of these CSP's is Ek-Lin-p, the problem of satisfying as many equations as possible from an over-determined system of linear equations modulo a prime p, where each equation contains exactly k variables.

The central question in much of the recent work on inapproximability has been the Unique Games Conjecture, which posits a very strong hardness of approximation for 2-CSP's with unique constraints, such as E2-Lin-p. Many strong inapproximability results have been proven assuming that it is true, including a recent result of Raghavendra ("Optimal algorithms and inapproximability results for every CSP?" in *STOC*. ACM, 2008.) giving an approximation algorithm for every CSP, whose performance is essentially optimal if the Unique Games Conjecture is true. To date, however, not much progress has been made on resolving the conjecture.

In this paper, we give a reduction from unique 3-CSP's to unique 2-CSP's which is sometimes approximation-preserving, depending on the combinatorial structure of the underlying hypergraph of the 3-CSP. The underlying hypergraph of a 3-CSP is the hypergraph in which each vertex represents a variable, and every hyperedge represents a constraint. Every constraint c yields a hyperedge between the three variables involved in c. The reduction only works when the underlying hypergraph of the 3-CSP satisfies a (hypergraph) expansion property, which we call local expansion.

We prove that the Unique Games Conjecture is equivalent to a hardness result for unique 3-CSP's whose underlying hypergraphs are local expanders. We give a precise formulation of the desired hardness result as a conjecture, which we call the Expanding Unique 3-CSP's Conjecture. We also give a restricted, but still equivalent, conjecture that E3-Lin-p is

[*]Department of Computer Science, University of Chicago. Email: epurdy@cs.uchicago.edu

hard to approximate on local expanders, in the interest of simplifying definitions slightly. We call this the Expanding Linear Equations Conjecture. The equivalence of these three conjectures is our first main result.

When our reduction is applied to 3-CSP's with small alphabets (small in relation to the error parameters) we produce unique 2-CSP's with similar-sized alphabets, which are tractable. Our second main result is an approximation algorithm for unique 3-CSP's that does well on instances whose underlying hypergraphs are local expanders. Our approximation algorithm does better on these instances than is possible for general (non-expanding) instances, since a theorem of Håstad gives a strong hardness of approximation result for E3-Lin-p. In particular, this result gives an answer to an open-ended question posed by Khot and Naor in 2007 about whether or not approximation algorithms for E3-Lin-2 can exploit the structure of the underlying hypergraph.

We also explore hypergraph expansion, proving that random hypergraphs of an appropriate density meet our expansion criteria with high probability, and giving two algebraic constructions (one fully explicit, and one semi-random) of such hypergraphs. This indicates that our definition of local expansion is a natural definition, and gives some indication of what a local expander looks like.

Our work is interesting in light of a recent paper of Arora, Khot, Kolla, Steurer, Tulsiani, and Vishnoi ("Unique games on expanding constraint graphs are easy," in *STOC*. ACM, 2008.), in which it is shown that unique 2-CSP's whose underlying graphs are expanders can be approximated well. Our work demonstrates that such a result for locally expanding 3-CSP's would disprove the Unique Games Conjecture. Our work can also be interpreted as providing a complete problem for the complexity class UGP of problems with hardness results following from the Unique Games Conjecture.

Contents

1 Introduction **3**
1.1 The Unique Games Conjecture 4
1.2 Our Approach 7
1.3 Statement of Results 9
1.4 Conclusion 10
1.4.1 Expansion vs. Uniqueness 10
1.4.2 How Natural is Local Expansion? 11
1.4.3 Hardness of Random Unique 3-CSP's 11
1.4.4 Unique Games as a Complexity Class 12
1.5 Organization of the Paper 14

2 k-CSP's and Unique k-CSP's **14**
2.1 Constraint Satisfaction Problems 14
2.2 Gap Problems 16
2.3 Basic Properties of Unique k-CSP's 18

2.4 Unique Constraints have Algebraic Structure 19

3 Expansion on Hypergraphs 21
3.1 Hypergraph Preliminaries 22
3.2 Local Expansion 23
3.3 Random Hypergraphs are Local Expanders 26
3.4 Explicit Constructions from Gowers's Theorem 30
3.5 Semi-explicit Constructions from Fourier Analysis 33

4 The Pair Construction 36
4.1 Definitions 36
4.2 The Honest Case 38
4.3 The Dishonest Case 41
4.4 The Expanding Unique 3-CSP's Conjecture implies the Unique Games Conjecture 44

5 Reducing Unique Games to Expanding Unique 3-CSP's 45
5.1 The Reverse Construction 46

6 Approximating unique 3-CSP's 50

7 Acknowledgments 52

1 Introduction

A *k-ary constraint satisfaction problem*, which we will abbreviate k-CSP, is a triple $P = (V, C, \Sigma)$, where V is a set of *variables*, C is a set of *constraints*, and Σ is an *alphabet*. Each variable will take *values* from Σ, An *assignment* to the variables will be a function $\sigma : V \to \Sigma$, where $\sigma(v)$ is the value assigned to the variable v.

Each constraint will be a predicate defined on k variables $v_1, \ldots, v_k \in V$, so that each $c \in C$ will be a function from Σ^k to {true, false}. We will say that c is satisfied by an assignment $\sigma : V \to \Sigma$ when $c(\sigma(v_1), \ldots, \sigma(v_k)) =$ true. The satisfiability of P will be the largest fraction of constraints in C that can be satisfied simultaneously.

Given a CSP, we wish to find an assignment σ satisfying the maximum number of constraints. Many of the most fundamental NP-hard problems can be phrased in this way, such as 3-SAT and 3-COLORABILITY. Therefore, finding a perfect, or even an optimal assignment for a CSP is NP-hard for many interesting classes of CSP's, and we are forced to adapt. We would like to find approximation algorithms, which find close-to-optimal solutions, so that we can satisfy, say, 98% of constraints if we know that the satisfiability is at least 99%.

We are especially interested in *unique constraints*. A constraint c on k variables $v_1, \ldots, v_k$ is unique if, for any values assigned to any $k-1$ of the variables, there is a unique value for the remaining variable that will satisfy c. A unique

CSP is a CSP, all of whose constraints are unique. Unique constraints on three variables have been studied before, in [10].

When $k = 2$, this gives us permutation constraints: for each value of v_1, there is a unique value for v_2 that satisfies c, and vice versa, so that we can associate a permutation π to c such that c is satisfied if and only if $v_1 = \pi(v_2)$ or, equivalently, $v_2 = \pi^{-1}(v_1)$. The class of 2-CSP's with permutation constraints is called UG, for unique games. We will denote this problem by $UG(L)$, where L is the size of the alphabet.

Problem 1.1 (Unique Games, $UG(L)$)**.**
Variables: $v_1, \ldots, v_n$, taking values from $\{1, \ldots, L\}$
Constraints: $c_1, \ldots, c_m$, where each c_i is a permutation constraint between two variables v_{j_i} and v_{k_i}.

A concrete example of unique k-CSP's is provided by over-determined systems of linear equations modulo p, where p is prime:

Problem 1.2 (Ek-Lin-p)**.**
Variables $x_1, \ldots, x_n$, taking values in $\{0, \ldots, p-1\}$.
Constraints $c_1, \ldots, c_m$, where c_i is a constraint of the form

$$a_1 x_{i_i} + \cdots + a_k x_{i_k} \equiv b \mod p,$$

where each $a_i \neq 0 \mod p$.

Since none of the a_i are zero, and p is prime, we will always be able to find a unique solution for the remaining x_{i_j} when we give values to $k-1$ of the variables. For instance, $x_{i_1} = a_1^{-1}(b - a_2 x_{i_2} - \cdots - a_k x_{i_k})$.

This problem is general enough to stand in for all unique k-CSP's throughout most of this paper. Our results and conjectures about unique k-CSP's will be equivalent (or simply less general) if the reader only considers E2-Lin-p and E3-Lin-p, so there is not much harm in doing so.

1.1 The Unique Games Conjecture

In 2002, Khot [28] made the following conjecture, known as the Unique Games Conjecture, which says that for sufficiently large alphabets, these problems have essentially no useful approximation algorithms, unless P=NP:

Conjecture 1.3 (Unique Games Conjecture, Khot [28])**.** *For all $\epsilon, \delta > 0$, and sufficiently large $L = L(\delta, \epsilon)$, given an instance of $UG(L)$ it is NP-hard to distinguish between the cases:*

- *There exists an assignment for $v_1, \ldots, v_n$ that satisfies at least a $1 - \epsilon$ fraction of the constraints.*
- *No assignment satisfies more than a δ fraction of the constraints.*

Problem	Best Algorithm	Best Impossibility Result	
		with UGC	without UGC
Max-Cut	0.87856 [22]	0.87856 [29]	0.941 [41, 25]
Max 2-Sat	0.94016 [34]	0.94016 [7]	0.955 [41, 25]
Max 2-CSP	0.87401 [34]	0.87435 [8]	0.9 [41, 25]
Vertex Cover	k**	$k - \epsilon$** [31]	$k - 1 - \epsilon$** [16]
Sparsest Cut	$O(\sqrt{\log k} \log \log k)$* [5]	$\omega(1)$ [12]	> 1 [13, 14]
Multicut	$O(\log(k))$* [33, 21, 6, 35]	$\omega(1)$ [12]	$\Omega(1)$ [15]

Table 1: The best known approximability results, with and without assuming the Unique Games Conjecture. (Mostly from [8]). (* k is the number of demand pairs) (** For k-uniform hypergraphs)

If no polynomial-time algorithm can tell 99%-satisfiable instances from 1%-satisfiable instances, then no polynomial-time algorithm can hope to give a close-to-optimal solution on 99%-satisfiable instances.

The Unique Games Conjecture is the central question in much of the recent work on inapproximability. If it could be proven, a wealth of results would follow, including many tight hardness of approximation results. (See Table 1 for a partial list.) Moreover, in a recent paper [38], it is shown that, assuming the Unique Games Conjecture, a single algorithm based on semidefinite programming yields an essentially optimal approximation algorithm for every CSP, with a matching UGC-based hardness of approximation result. Regardless of whether the conjecture is true or false, it has a central and seemingly deep connection to approximability, so we will need to come up with some understanding of it.

It is proven in [29, 37] that the Unique Games Conjecture is equivalent to the special case of E2-Lin-p, i.e., to the assertion that, for all $\delta, \epsilon > 0$, it is NP-hard to tell apart $(1 - \epsilon)$-satisfiable instances and δ-satisfiable instances of E2-Lin-p, for some p depending on δ and ϵ. p will have to get larger as δ and ϵ go to zero, but all p above a certain number will work for a fixed δ and ϵ.

To date, the main progress on proving the Unique Games Conjecture (as opposed to disproving it) is the following theorem of Feige and Reichman:

Theorem 1.4 (Feige and Reichman, [18]). *For every $\epsilon > 0$, there are some constants η, p, with $0 < \eta < 1$ and p prime, such that it is NP-hard to distinguish between instances of $UG(p)$ with satisfiability greater than η and instances with satisfiability less than $\epsilon \cdot \eta$.*

This proves that UG is hard to approximate on instances with low satisfiability. By contrast, proving inapproximability for instances with high satisfiability is equivalent to the Unique Games Conjecture, by a recent result of Rao:

Theorem 1.5 (Rao, [39, 40]). *The Unique Games Conjecture (Conjecture 1.3) is equivalent to the following statement: For every $\epsilon, \delta > 0$, there is an alphabet size $L = L(\epsilon)$, such that it is NP-hard to distinguish between instances of $UG(L)$ with satisfiability greater than $1 - \epsilon^2$ and instances with satisfiability less than $1 - \epsilon^{1-\delta}$.*

So, for example, if we let $\delta = \frac{1}{2}$ and $\eta = \epsilon^2$, the Unique Games Conjecture is equivalent to the NP-hardness of distinguishing between instances with satisfiability greater than $1 - \eta$ and instances with satisfiability $1 - \sqrt[4]{\eta}$.

The main progress on disproving the Unique Games Conjecture (i.e., finding an efficient approximation algorithm for $UG(L)$) has been a series of algorithms based on semidefinite programming relaxations. There is a natural semidefinite programming relaxation of the problem, and most of the work has been in concocting and analyzing various rounding schemes to recover normal assignments from the vector assignments that SDP solutions generate. The best algorithm thus far is that of Charikar, Makarychev, and Makarychev [11], which satisfies a $1 - O(\sqrt{\epsilon \log L})$ fraction of constraints, given that the instance is $(1 - \epsilon)$-satisfiable.

By contrast, the approximability situation for unique k-CSP's with $k > 2$ is basically completely resolved by the following theorem of Håstad:

Theorem 1.6 (Håstad, [24, 25]). *For every $k \geq 3$, and for every $\epsilon > 0$, it is NP-hard to distinguish between instances of* Ek-Lin-p *with satisfiability greater than $1 - \epsilon$ and instances with satisfiability less than $\frac{1}{p} + \epsilon$.*

Since we can satisfy a $\frac{1}{p}$ fraction of constraints by taking a random assignment, this result is the best possible, and the best approximation that can be achieved for this problem is given by the trivial algorithm that just picks a random assignment.

The difference between the state of our knowledge for unique k-CSP's when $k = 2$ and when $k > 2$ seems to result from some sort of fundamental difficulty in making hardness reductions into unique 2-CSP's. There does not seem to be any way to naturally encode information and constraints in unique 2-CSP's, at least in the way that would be necessary for dealing with the Unique Games Conjecture. In the context of the Unique Games Conjecture, we want our instances with high satisfiability to have satisfiability arbitrarily close to one. (There are plenty of hardness reductions into unique 2-CSP's that do not meet this criteria: in [24], for instance, there is a hardness reduction into E2-Lin-2, but it only creates instances with relatively high satisfiability, not instances with arbitrarily high satisfiability.) There are not as many techniques for encoding information in binary constraints; in particular, the powerful Fourier methods of [24, 25] rely on checking that the sum of two variables is equal to a third, which cannot be done with only two variables per constraint.

There is one trick (used in the Raz verifier) to make non-binary constraints look binary, which is to make one new variable for each old variable, and one new variable for each old constraint, which has one value for each of the old variables involved in the constraint. The new constraints pick an old constraint and one of the involved variable, and check that the corresponding constraint-variable receives a satisfying assignment, and that this assignment is consistent with the assignment to the variable-variable. But the constraints created by this trick are not unique, and no approach similar to this one would create unique constraints.

1.2 Our Approach

In this paper, we give a reduction from unique 3-CSP's to unique 2-CSP's that is sometimes approximation-preserving. It has the property that it keeps the satisfiability close to one, so that we can hope to get hardness results for UG that have satisfiability arbitrarily close to one. On instances with low satisfiability, it is not as well-behaved, and we have to make assumptions about the combinatorial structure of our 3-CSP. We will explain this shortly.

Our reduction is based on the simple observation that we can reduce any 3-CSP to a 2-CSP through the following PCP reduction, which we refer to as the Pair Construction:

Protocol 1.7 (Pair Construction, PCP Protocol Version).
Verifier

1. Pick a constraint c, incident on variables u, v, w to check.
2. Send u, v to Prover 1.
3. Send v, w to Prover 2.
4. Accept if Prover 1 and Prover 2 send the same value for v, and if the values that they send for u, v, w satisfy the constraint c.

This construction maps unique constraints on 3 variables to permutation constraints on 2 variables. In the PCP protocol, for every answer sent by Prover 1, say $\sigma(u), \sigma(v)$ for u and v, there is exactly one correct answer for Prover 2, namely $\sigma(v)$ for v and the unique value $s \in \Sigma$ for w that causes $\sigma(u), \sigma(v), s$ to satisfy c. We go through this reduction in a more combinatorial way in Section 4.

The pair construction does not just work as a reduction in all cases, however. We cannot simply take a hard instance a unique 3-CSP and apply the pair construction to get a hard instance of UG, since the provers are not limited to using consistent assignments for all the pairs. A prover receiving queries u, v knows more about what question is being asked than does a prover who receives u alone. A dishonest prover could thus base his assignment for u on the knowledge that it will be in a constraint with v. We give an example in Section 4.3 that shows how this can cause the reduction to fail in certain cases. We show, however, as part of our main technical lemma, Lemma 1.12, that the pair construction is a valid reduction if the underlying hypergraph of the 3-CSP has an expansion property called local expansion, which we discuss in Section 3.

Briefly, a 3-uniform hypergraph $G = (V, E)$ is an (α, β)-local expander at u if, for every subset S of V for which

- an α fraction of the triples containing u also contain an element of S
- an α fraction of the triples containing u also contain an element of $V \setminus S$

we must have that a β fraction of the triples containing u also contain one element from S and one element not from S. We will say that G is an (α, β)-local expander if it satisfies this condition at every u.

This expansion property is a sort of local isoperimetric condition - the triples containing u cannot be cleanly divided into two classes with little interaction. This turns out to be exactly what we need to defeat cheating provers in the above PCP protocol. Drawing our questions to the provers from an (α, β)-local expander guarantees, roughly speaking, that provers who try to cheat an α fraction of the time will get caught a β fraction of the time.

Our work therefore shows that this expansion property is an important property to examine in a 3-CSP, particularly a unique one. This motivates us to restrict our attention to the class of unique 3-CSP's on local expanders:

Definition 1.8. $\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$ is the problem of satisfying the maximum number of constraints in a unique 3-CSP over an alphabet of size L whose underlying hypergraph is an (α, β)-local expander. This is a promise problem: on instances which do not meet our expansion criteria, algorithms are allowed to answer arbitrarily.

More concretely, we have:

Definition 1.9. $\textsc{Exp}_{\alpha,\beta}$-E3-$\textsc{Lin}$-$p$ is the problem of satisfying the maximum number of equations in an instance of E3-$\textsc{Lin}$-p whose underlying hypergraph is an (α, β)-local expander. On instances which do not meet our expansion criteria, algorithms are allowed to answer arbitrarily.

These have to be promise problems, since it is NP-hard to decide whether or not a given hypergraph is an (α, β)-local expander for a given α and β. This can be shown quite easily by a trivial reduction from Balanced Separator.

Using the Pair Construction and another, simpler, reduction, we prove that the Unique Games Conjecture is equivalent to the following conjecture about $\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$:

Conjecture 1.10 (Expanding Unique 3-CSP's Conjecture)**.** *For some fixed $\mu < 1$, and for every $\epsilon > 0$, there exist L, $\alpha < \frac{1-\mu}{8}$, and $\beta > 0$ such that it is NP-hard to distinguish between instances of $\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$ with satisfiability less than μ and instances with satisfiability greater than $1 - \epsilon$.*

It is easy to see that, if this conjecture is true for some ϵ and L, then it will be true for all sufficiently large L. Later, we will prove that L must become arbitrarily large as ϵ goes to zero. In the interest of putting a slightly friendlier face on this conjecture, we also restate it in the more specific context of E3-$\textsc{Lin}$-p:

Conjecture 1.11 (Expanding Linear Equations Conjecture)**.** *For some fixed $\mu < 1$, and for every $\epsilon > 0$, there exist a sufficiently large prime p, $\alpha < \frac{1-\mu}{8}$, and $\beta > 0$ such that it is NP-hard to distinguish between instances of $\textsc{Exp}_{\alpha,\beta}$-E3-$\textsc{Lin}$-$p$ with satisfiability less than μ and instances with satisfiability greater than $1 - \epsilon$.*

Again, for a fixed ϵ, this conjecture will be true for all sufficiently large L if it is true for any L, and L must go to infinity as ϵ goes to zero. A priori, the second conjecture is less likely to be true, since it posits hardness for a smaller class of problems. We will show, however, that the two are equivalent, since they are both equivalent to the Unique Games Conjecture.

1.3 Statement of Results

Our main technical lemma is the following:

Lemma 1.12. *Let $\mu, \epsilon > 0$. There exists some function $c(\alpha, \beta, \mu)$, and a polynomial time reduction F from unique 3-*CSP*'s to UG such that, if P is a unique 3-*CSP*:*

- *if P has an assignment satisfying at least a $1 - \epsilon$ fraction of constraints, $F(P)$ also has such an assignment.*
- *if no assignment for P satisfies more than a μ fraction of P's constraints, and the underlying hypergraph of P is an (α, β)-local expander, then no assignment for $F(P)$ can satisfy more than a $c(\alpha, \beta, \mu)$ fraction of $F(P)$'s constraints.*

Moreover, $c = \max\{\mu + 8\alpha, 1 - \alpha\beta\}$, so $c < 1$ when $\alpha < \frac{1-\mu}{8}$ and $\beta > 0$.

Using this reduction, and some other simpler lemmas, we arrive at our first main result:

Theorem 1.13. *The Expanding Unique 3-*CSP*'s Conjecture (Conjecture 1.10) is equivalent to the Unique Games Conjecture (Conjecture 1.3). Both are equivalent to the Expanding Linear Equations Conjecture (Conjecture 1.11)*

Again using our reduction, together with an algorithm of Charikar, Makarychev, and Makarychev [11] based on rounding a semidefinite programming relaxation of the problem, we get an algorithm that does well on unique 3-CSP's when the underlying hypergraph is a local expander. This theorem is our second main result:

Theorem 1.14. *There is a polynomial-time algorithm (namely Algorithm 6.2) which does the following: Let P be a unique 3-*CSP* with satisfiability $1 - \epsilon$. Let $\alpha_{min} \leq \frac{1}{2}$ be the largest value such that for all $\alpha \in [\alpha_{min}, \frac{1}{2}]$, the underlying hypergraph of P is an $(\alpha, \frac{1}{4}\alpha)$-local expander. The algorithm outputs an assignment satisfying a*

$$1 - \max\{C\sqrt[4]{\epsilon \log(L)}, 8\alpha_{min}\}$$

fraction of constraints, for some absolute constant C.

This gives an approximation algorithm for expanding unique 3-CSP's when the alphabet size L is sub-exponential in $1/\epsilon$. To put this in context, a random hypergraph with $\Omega(n^2 \log n)$ edges satisfies the above conditions with $\alpha_{min} \leq$

$\frac{1}{\log n}$ with high probability. Thus, when $\alpha_{min} = o(1)$ as a function of n, and L is a constant, the above algorithm has the simpler approximation guarantee of $1 - O\left(\sqrt[4]{\epsilon \log(L)}\right)$ We show this in Section 3, Theorem 3.7.

In particular, Theorem 1.14 gives one interesting answer to an open-ended question of Khot and Naor, asked in 2007:

Question 1.15 (Khot and Naor, [30])**.** Can we get better approximation guarantees for Max E3Lin2 in terms of the combinatorial structure of the underlying hypergraph of the problem? The results of [2] suggest that it might be possible to achieve a better approximation guarantee in the presence of additional structural information (such as chromatic number or clique number).

Our work shows that we can indeed get a better approximation guarantee for this problem and many similar problems, when the underlying hypergraph is a local expander.

1.4 Conclusion

Our main result gives a completely equivalent statement of the Unique Games Conjecture, in terms of the Expanding Unique 3-CSP's Conjecture. This may make it easier to approach the conjecture from a different angle, so to speak. It seems that it is in general easier to make reductions to 3-CSP's than to 2-CSP's, but it is by no means clear that current techniques for proving inapproximability for 3-CSP's can be adapted to produce local expanders.

1.4.1 Expansion vs. Uniqueness

Theorem 1.14 gives an approximation algorithm for expanding unique 3-CSP's when the alphabet size L is sub-exponential in $1/\epsilon$ (i.e., when approximation algorithms for UG exist). Conjecture 1.10 asks "Can we prove a hardness of approximation result for expanding unique 3-CSP's with large alphabets?" It seems plausible: Håstad's theorem (Theorem 1.6) gives it for non-expanding unique 3-CSP's with a constant-sized alphabet, and the same theorem gives it for non-unique expanding 3-CSP's, also with a constant-sized alphabet. (The latter is because we can add trivial, always-satisfied constraints to our CSP to make it an expander. This may not seem like it should change the approximation behavior of the 3-CSP, but it has the effect of adding consistency checks to the Pair Construction.) There is apparently some interaction between these two conditions, so that it is difficult to make both hold simultaneously without making the problem easy.

This conjecture (and the fact that it is equivalent to the Unique Games Conjecture) is especially intriguing in light of [4], where it is shown that expanding unique 2-CSP's (i.e., instances of UG with expanding constraint graphs) can be approximated in polynomial time. It seems unlikely that their work could be easily adapted to deal with expanding unique 3-CSP's, but it does add some support to the intuition that expansion properties and unique constraints do not mix well.

1.4.2 How Natural is Local Expansion?

The importance of Theorem 1.13 depends largely on how natural a problem $\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$ is. This, in turn, depends on how natural the definition of local expansion is. We address this question in Section 3, where we explore the concept of local expansion. Specifically, we show that random hypergraphs of a high enough edge density are local expanders with high probability. This is important, since expansion is usually considered a pseudo-randomness property, and any definition of expansion should satisfy this criterion. Our definition is thus somewhat consistent with the more standard definitions of expansion.

We also give some explicit and semi-explicit constructions of local expanders, using techniques from algebra. This allows us to actually construct local expanders, and gives us some idea of what they can look like. These constructions create hypergraphs which are globally pretty well connected, which is not always the case for local expanders.

Our proof that the Unique Games Conjecture implies the Expanding Linear Equations Conjecture also requires the construction of local expanders. In this case, our construction is much more combinatorial, and it creates a different sort of local expander. The hypergraphs created in this method are poorly connected on a global level, and well connected on a local level. It seems more likely that CSP's with this weaker kind of structure will be hard, since [4] seems to imply that we should not expect globally well-connected instances to be hard when our constraints are unique.

1.4.3 Hardness of Random Unique 3-CSP's

Our theorem that random hypergraphs of high enough density are local expanders with high probability suggests that we should examine the relationship between hardness of approximation and average-case hardness, as is done in Feige's Conjecture [17]. Specifically, if unique 3-CSP's are hard to approximate on randomly generated instances, then it must also be the case that expanding unique 3-CSP's are hard to approximate, and the Expanding Unique 3-CSP's Conjecture would be true. (And thus the Unique Games Conjecture would also be true.) Thus, we could say that some conjecture similar to Feige's conjecture (say, hardness of approximation for random E3-$\textsc{Lin}$-p instances for large enough p) would imply the Unique Games Conjecture.

The only problem with this line of reasoning is that the edge density must be fairly high before a random hypergraph becomes a local expander: Feige's Conjecture deals with CSP's with the number of constraints linear in the number of variables, whereas we require $\Omega(n^2 \log n)$ constraints for n variables. At densities this high, it seems far less likely that a random instance will be hard. Thus, it is not clear that it would be reasonable to conjecture hardness of approximation for random unique 3-CSP's, at least based on our work.

1.4.4 Unique Games as a Complexity Class

In light of the difficulty of resolving the Unique Games Conjecture, it has been suggested by some (e.g., [29]) that UG may be a problem of intermediate complexity. For problems not known to be in P or to be NP-complete, there is always the possibility that these problems lie in some intermediate complexity class, the existence of which is guaranteed by Ladner's Theorem [32], given that P $\neq$ NP. This would still imply the intractability of these problems, since they would not be in P, but it would preclude the possibility of any proof of NP-hardness, which has become the standard measure of intractability. The proof of Ladner's Theorem generates incredibly unnatural languages, so there is not yet any reason to suppose that these intermediate classes will necessarily contain any problems that we care about. To date, there has been no evidence that any important problems are of intermediate complexity, other than lack of progress in finding algorithms or proofs of hardness for them.

For this reason, and also to simplify the categorizations that our current knowledge yields, it seems reasonable to define a new complexity class of problems which are at least as hard as UG. The most natural way to do this is by analogy with the class PCP, which we can define as follows:

Definition 1.16 ($\mathsf{PCP}_s^c(\Sigma, r(n), q(n))$ [3])**.** Let $\mathcal{L}$ be a language, $q, r : \mathbb{N} \to \mathbb{N}$, and $c, s \in [0, 1]$. We say that $\mathcal{L}$ has a $(\Sigma, r(n), q(n), c, s)$-*verifier* if there is a polynomial-time probabilistic algorithm V satisfying:

- (Efficiency) On input a string $x \in \{0, 1\}^n$, and given access to a proof string $\pi \in \Sigma^*$, V uses at most $r(n)$ random coins and makes at most $q(n)$ non-adaptive queries to locations of π. It outputs "1" when it accepts π as a proof that $x \in \mathcal{L}$, and "0" when it rejects the proof. We will use $V^\pi(x)$ to denote the random variable that is V's output when it is given input x and proof π, for all possible choices of its $r(n)$ random bits.

- (Completeness) If $x \in \mathcal{L}$ then there exists a proof $\pi \in \Sigma^*$ such that $\Pr[V^\pi(x) = 1] \geq c$. The number c is called the *completeness*.

- (Soundness) If $x \notin \mathcal{L}$ then for every proof $\pi \in \Sigma^*$, $\Pr[V^\pi(x) = 1] \leq s$. The number s is called the *soundness*.

We will say that $\mathcal{L} \in \mathsf{PCP}_s^c(\Sigma, r(n), q(n))$ if it has a $(\Sigma, O(r(n)), O(q(n)), c, s)$-verifier.

We can then adapt the definition to only allow proof systems corresponding to unique games:

Definition 1.17 ($\mathsf{UGP}_s^c(\Sigma, r(n))$)**.** Let $\mathcal{L}$ be a language, $q, r : \mathbb{N} \to \mathbb{N}$, and $c, s \in [0, 1]$. We say that $\mathcal{L}$ has a $(\Sigma, r(n), q(n), c, s)$-*UG-verifier* if there is a polynomial-time probabilistic algorithm V satisfying:

- (Efficiency) On input a string $x \in \{0, 1\}^n$, and given access to a proof string $\pi \in \Sigma^*$, V uses at most $r(n)$ random coins and makes *exactly two* non-adaptive queries to locations of π. It outputs "1" when it accepts π as a proof that $x \in \mathcal{L}$, and "0" when it rejects the proof. We will use $V^\pi(x)$ to denote the random variable that is V's output when it is given input x and proof π, for all possible choices of its $r(n)$ random bits.

- (Uniqueness) When V queries locations i and j, for every $s \in \Sigma$, if V reads s at location i, there must be a unique value $t \in \Sigma$ that V can read at location j that will cause V to accept (and vice versa, for every $t \in \Sigma$ that V reads at location j, there must be a unique value that V will accept at location i).
- (Completeness) If $x \in \mathcal{L}$ then there exists a proof $\pi \in \Sigma^*$ such that $\Pr[V^\pi(x) = 1] \geq c$. The number c is called the *completeness*.
- (Soundness) If $x \notin \mathcal{L}$ then for every proof $\pi \in \Sigma^*$, $\Pr[V^\pi(x) = 1] \leq s$. The number s is called the *soundness*.

We will say that $\mathcal{L} \in \mathsf{UGP}_s^c(\Sigma, r(n))$ if it has a $(\Sigma, O(r(n)), O(q(n)), c, s)$-UG-verifier.

We can then define $\mathsf{UGP}_\delta^{1-\epsilon}(r(n))$ to be the union of $\mathsf{UGP}_\delta^{1-\epsilon}([N], r(n))$ over all $N \in \mathbb{N}$, so that we are allowed to use any constant-sized alphabet. The Unique Games Conjecture would then be the conjecture that $\mathsf{NP} \subseteq \mathsf{UGP}_\delta^{1-\epsilon}(\log n)$ for all $\delta, \epsilon > 0$. By Theorem 1.5, this is equivalent to the statement that there is some constant $\mu < 1$ such that $\mathsf{NP} \subseteq \mathsf{UGP}_\mu^{1-\epsilon}(\log n)$ for all $\epsilon > 0$. This is analogous to the PCP Theorem, which says that $\mathsf{NP} \subseteq \mathsf{PCP}_{1/2}^1(\{0,1\}, \log n, O(1))$. As is the case with the PCP Theorem, the reverse containment is trivial, so we would really be proving equality of the two classes.

In this view, all of the UGC-based hardness results can be thought of as proofs that certain gap problems are hard for the class $\mathsf{UGP}_\delta^{1-\epsilon}(\log n)$ for sufficiently small δ, ϵ: if any problem in $\mathsf{UGP}_\delta^{1-\epsilon}(\log n)$ is NP-hard, they are also. Moreover, even if they are not NP-hard, there are polynomial-time algorithms for some of them if and only if there are polynomial-time algorithms for all of them.

Our main result, then, shows that the gap version of $\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$ would be an example (possibly the first) of a problem in this class that is not known to be in P, besides UG itself. The reverse implication establishes that $\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$ is actually complete for the class $\mathsf{UGP}_\delta^{1-\epsilon}(\log n)$. Theorem 1.13 can then be interpreted as saying:

Theorem 1.18. $\mathsf{Gap}_\mu^{1-\epsilon}\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$$\in \mathsf{UGP}_{\mu'}^{1-\epsilon}(\log n)$*, where* $\mu' = \max\{\mu + 8\alpha, 1 - \alpha\beta\}$*. Moreover,* $\mathsf{Gap}_\delta^{1-\epsilon}\textsc{Exp}_{\alpha,\beta}$-U3-$\textsc{Csp}(L)$ *is hard for the class* $\mathsf{UGP}_{\delta'}^{1-\epsilon}(\log n)$*, for* $\delta' = \delta - \frac{1}{L-2}$ *when* L *is prime.*

This means that $\mathsf{UGP}_\delta^{1-\epsilon}(\log n)$ is not *such* an unnatural complexity class, and having another complete problem for the class means that we have another way to think about it.

1.5 Organization of the Paper

The rest of the paper is organized as follows: In Section 2, we give basic definitions related to constraint satisfaction problems, and prove a few of their basic properties. In Section 3, we define our notion of hypergraph expansion, and discuss some of its properties. We also prove that a random hypergraph of the appropriate edge density satisfies this definition, and give some algebraic constructions of expander hypergraphs. In Section 4, we define the pair construction, our main technical tool, and we prove a number of lemmas about the approximation behavior of this construction. This section contains the main technical work. In Section 5, we give a simpler reduction, from UG to expanding unique 3-CSP's, which establishes the reverse implication of our main result (Theorem 1.13), that the Unique Games Conjecture implies the Expanding Unique 3-CSP's Conjecture. In Section 6, we give our algorithm for approximating expanding unique 3-CSP's, and prove that it works.

2 k-CSP's and Unique k-CSP's

In this section, we mainly give preliminaries that we will use in the rest of the paper. Section 2.1 defines CSP's, and gives some notation and terminology that will be used later. In section 2.2 we explain the notion of gap problems, which are the main formalism used in hardness of approximation results. Section 2.3 gives a few basic properties of unique k-CSP's.

2.1 Constraint Satisfaction Problems

Definition 2.1 (k-CSP)**.** A *k-ary constraint satisfaction problem*, which we will abbreviate k-CSP, is a triple $P = (V, C, \Sigma)$, where V is a set of *variables*, C is a set of *constraints*, and Σ is an *alphabet*. Each variable will take *values* from Σ, An *assignment* to the variables will be a function $\sigma : V \to \Sigma$, where $\sigma(v)$ is the value assigned to the variable v.

Each constraint will be a predicate defined on k variables $v_1, \ldots, v_k \in V$, so that each $c \in C$ will be a function from Σ^k to $\{\text{true, false}\}$. We will say that c is satisfied by an assignment $\sigma : V \to \Sigma$ when $c(\sigma(v_1), \ldots, \sigma(v_k)) = \text{true}$.

We will say that c is *incident* on the variables $v_1, \ldots, v_k$, and write $c \ni v_1, \ldots v_k$. (This is a slight abuse of notation, since a constraint contains more information than just the incident variables. But it would be correct for the underlying hypergraph, where c is just a hyperedge, i.e., a set of vertices of size k.)

Definition 2.2 (Weighted k-CSP)**.** A weighted k-CSP is a CSP $P = (V, C, \Sigma)$ together with a weight function defined on the constraints $wt : C \to (0, \infty)$. Constraints with weight 0 would be irrelevant, so we do not allow them in our definition. We will sometimes write wt_P to distinguish our weight function from other weight functions when it is not clear from context which is meant.

For $S \subseteq C$, we will set $wt(S) = \sum_{c \in S} wt(c)$.

We will think of $wt(\cdot)$ as a probability distribution on C, where each $c \in C$ is picked with probability $wt(c)/wt(C)$. We will call this distribution the *weight distribution*, and write $c \sim wt(\cdot)$ when we wish to pick a constraint c at random according to it.

Definition 2.3 ($sat(\sigma, P)$)**.** Let $P = (V, C, \Sigma)$ be a CSP. Let $\sigma : V \to \Sigma$ be an assignment for P. We define the *satisfaction* of P by σ, denoted $sat(\sigma, P)$, to be the weighted fraction of P's constraints satisfied by σ, i.e.,

$$sat(\sigma, P) = wt(\{c \in C \mid c \text{ satisfied by } \sigma\})/wt(C)$$

Equivalently, $sat(\sigma, P)$ is the probability that a randomly constraint, selected according to the weight distribution, has been satisfied by σ:

$$sat(\sigma, P) = \Pr_{c \sim wt(\cdot)}[c \text{ satisfied by } \sigma]$$

The *satisfiability* of P is the largest fraction of constraints that can be satisfied simultaneously by an assignment:

Definition 2.4 ($sat(P)$)**.** Let P be a CSP. We define the *satisfiability* of P, denoted $sat(P)$, to be the largest weighted fraction of constraints satisfied by any assignment σ, i.e.,

$$sat(P) = \max_{\sigma : V \to \Sigma} sat(\sigma, P)$$

Note that satisfaction and satisfiability are always numbers between 0 and 1, regardless of the total weight of the constraints in our CSP.

To simplify definitions and the statements of theorems, we will often treat unweighted CSP's as if they were weighted. When we do this, it should be assumed that we are using the uniform weight function $wt(c) = 1$ for all $c \in C$. When we are studying approximability, it is trivial to pass back and forth between weighted and unweighted CSP's, as long as the weights are not too large. To get a weighted CSP from an unweighted CSP, we use the uniform weighting. To go the other way, we have the following lemma:

Lemma 2.5. *Let $\epsilon > 0$. For every weighted* CSP *$P = (V, C, \Sigma)$, with weight function $wt(\cdot)$, which has polynomially bounded weights*

$$\frac{1}{\text{poly}(|V|, |C|)} \leq wt(c) \leq \text{poly}(|V|, |C|)$$

there is a corresponding unweighted CSP *$P' = (V, C', \Sigma)$ such that $|sat(P) - sat(P')| < \epsilon$, which can be constructed in time polynomial in $|V|, |C|$, and $1/\epsilon$.*

Proof. To get an unweighted CSP from a weighted CSP, first scale all the weights, so that $wt(c) \geq 1$ for all $c \in C$. Then, for all $c \in C$, we have

$$1 \leq wt(c) \leq \text{poly}(|V|, |C|).$$

Then we use our error parameter ϵ, and choose D large enough ($\lceil 1/\epsilon \rceil$ should work) that all of the weights $wt(c)$ have some integer n_c such that $\left|wt(c) - \frac{n_c}{D}\right| < \epsilon < \epsilon wt(c)$. Then replace every weighted constraint c with n_c copies of the constraint, each with weight $1/D$. The weight of any constraint will change by at most $\epsilon wt(c)$, so the weight of any set S of constraints will change by at most $\epsilon wt(S)$. Then, since

$$sat(\sigma, P) = wt(\{c \in C \mid c \text{ satisfied by } \sigma\})/wt(C)$$

the weight of any set of satisfied constraints will change by at most $\epsilon wt(S)$, and the satisfaction will change by at most $\epsilon wt(S)/wt(C) < \epsilon$. Thus the new CSP will have satisfiability that is off by at most ϵ, and will be larger by at most a factor of $D \cdot \operatorname{poly}(|V|, |C|)$, since each constraint gets replaced by $n_c \leq D \cdot \operatorname{poly}(|V|, |C|)$ constraints. Finally, since all constraints now have the same weight, we can treat the CSP as an unweighted CSP. $\square$

For most contexts, we can choose ϵ to be $1/\operatorname{poly}(n)$ without changing the approximation behavior (in fact usually ϵ can be a small constant), so this is only a polynomial blow-up.

For 2-CSP's, we are interested in permutation constraints, the subject of the Unique Games Conjecture:

Definition 2.6. Let x, y be two CSP variables, taking values from an alphabet Σ. A permutation constraint c is a 2-variable constraint which, for each value of x, is satisfied by one and only one value of y, and for each value of y, is satisfied by one and only one value of x. We can associate a permutation $\pi : \Sigma \to \Sigma$ with c, so that c is satisfied if and only if $y = \pi(x)$, or, equivalently, $x = \pi^{-1}(y)$.

In this paper, we examine the more general notion of unique constraints, which can take more than two variables.

Definition 2.7 (Unique Constraints)**.** A constraint c on k variables $v_1, \ldots, v_k$ is *unique* if, for any assignment of values to any $k-1$ of its variables, there is exactly one value which can be assigned to the last variable which will satisfy c.

A unique k-CSP is a k-CSP, all of whose constraints are unique.

A simple but useful example of a unique constraint is linear equations modulo 2, where $\Sigma = \{0, 1\}$ and each constraint is of the form $x + y + z = 0 \text{ or } 1 \mod 2$.

2.2 Gap Problems

When studying optimization problems, it often simplifies the analysis greatly to consider only decision problems, which have a yes-or-no answer. This is usually done by asking the question "Does the problem have a solution of value greater than k?" for different values of k. (Here we are assuming that our problem is a maximization problem. For a minimization problem, we would change "greater than k" to "less than k".) This is enough for the purpose of studying the hardness of solving optimization problems, since any polynomial-time algorithm for producing an optimum solution can also answer questions about the value of that solution. Furthermore, by using binary search, we can adapt any algorithm for the decision version of the optimization problem to get an algorithm that tells us the value of the optimum solution.

When studying the approximability of optimization problems, the corresponding notion of a decision version of the problem is that of a *gap problem*. In a gap problem, we are given an optimization problem (in this paper it will be a maximization problem, but minimization problems are very similar), together with two thresholds α and β (where $\alpha < \beta$), and we are supposed to decide if the maximum solution has value less than α (a NO instance) or at least β (a YES instance). In the case where it falls in between (a meh instance), we are allowed to say whatever we want. This situation is summarized in Figure 1.

Suppose that we have an algorithm $\mathcal{A}$ for the gap version of the maximization problem P. When $\mathcal{A}$ answers NO ("less than α"), we need to be certain that $sat(P) < \beta$ (but $sat(P)$ is allowed to fall between α and β), and when $\mathcal{A}$ answers YES ("at least β"), we need to be certain that $sat(P) > \alpha$ (but, again, $sat(P)$ is allowed to fall between α and β). Here, to emphasize the parameters, we will call this the (α, β) gap problem:

Definition 2.8. Let $\alpha < \beta$. Let P be a maximization problem. We define $\mathsf{Gap}_{\alpha}^{\beta}P$, the (α, β) gap problem of P, to be the problem of deciding which of two cases P falls into, given the promise that it falls into one of them:

- (YES instances) There exists a solution for P with value $\geq \beta$.
- (NO instances) Every solution for P has value $< \alpha$.

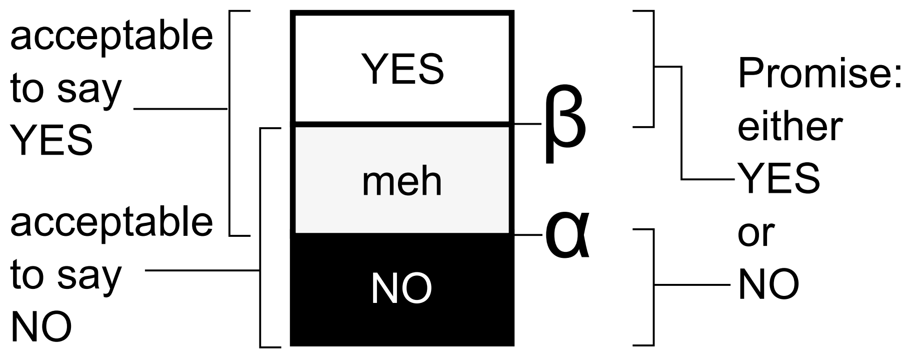


Figure 1: The (α, β) gap problem, with vertical height indicating satisfiability. On the left we indicate which answers are acceptable in which range - in the intermediate range, either answer is allowed. On the right, we give the promise problem formulation, as in Definition 2.8.

This definition does a good job of capturing the essential difficulty of approximation:

Lemma 2.9. *If there exists an efficient β/α-approximation algorithm $\mathcal{A}$, then there is an efficient algorithm for the (α, β) gap problem.*

Proof. To solve the gap problem, we run $\mathcal{A}$ on our maximization problem, and get a solution with value c. If $c \geq \alpha$, then we know that the best solution has value at least α, so we can safely answer YES ("at least β"), knowing that the true maximum is either at least β, or lies in the no man's land between the two thresholds. If $c < \alpha$, we use our approximation guarantee to assure us that the true maximum is at most $(\beta/\alpha) \cdot c$, and thus strictly less than β. In this case, we can safely answer NO ("less than α"), because we have a proof that the maximum is either less than α or lies between the two thresholds. □

The converse of this lemma is not true in general; a decision algorithm for the gap problem does not automatically yield an approximation algorithm. For one thing, each approximation ratio corresponds to many gap problems - for example, an approximation ratio of 2 corresponds to the $(\alpha, 2\alpha)$ gap problem for every α. However, in practice, algorithms for gap problems often yield approximate solutions, so that the gap problem is usually a reasonable proxy for the approximate optimization problem, and it is always useful in proving inapproximability results.

2.3 Basic Properties of Unique k-CSP's

Lemma 2.10. *For any unique k-*CSP *on an alphabet Σ of size L, there exists an assignment of value $1/L$.*

Proof. Consider giving each variable a value chosen uniformly at random from Σ. For any constraint c on variables $v_1, \dots, v_k$, regardless of what values have been assigned to $v_1, \dots, v_{k-1}$, there is a $1/L$ chance that v_k will receive the unique value that will satisfy the constraint. By linearity of expectation, a random assignment will satisfy this fraction in expectation, therefore there exists an assignment that satisfies a $1/L$ fraction of the constraints. □

Lemma 2.11. Ek-Lin-2 *is the only unique k-*CSP *over a binary alphabet*

Proof. By induction on k: for $k = 2$, there are only two unique constraints, namely $u = v$ and $u \neq v$, which can be written as $u + v = 0$ and $u + v = 1$.

Now, suppose true for k. Pick any unique $(k+1)$-ary constraint c on variables $v_1, \dots, v_{k+1}$, and plug in 0 and 1 for v_1. Call the two constraints you get c_0 and c_1; these constraints will be unique. By the inductive hypothesis, each c_i have the form $v_2 + \cdots + v_{k+1} = b_i$, where $b_i = 0$ or 1. If the two values of b_i are the same, then c_0 and c_1 can be satisfied by the same assignment to $v_2, \dots, v_{k+1}$ (call it σ), so σ cannot be uniquely extended to get a satisfying value for v_1. This can never happen for a unique constraint, so the values for b_i must disagree. Either $b_i = v_1 + 0$ or $b_i = v_1 + 1$. Thus the constraint is $v_2 + \cdots + v_{k+1} = b + v_1$, which is equivalent to $v_1 + \cdots + v_{k+1} = b$. □

We also have the following, noted by Khot in his original paper:

Observation 2.12 ([28])**.** Uniqueness is preserved by parallel repetition.

2.4 Unique Constraints have Algebraic Structure

Unique constraints in two variables correspond to permutations of the alphabet. In the case of unique constraints in 3 variables, it turns out that they also have a well-known structure: they correspond to Latin squares.

Definition 2.13 (Latin Square)**.** A Latin square is an $n \times n$ table filled with n different symbols in such a way that each symbol occurs exactly once in each row and column. We will take these symbols to be the numbers 1 to n. We will write $L[i, j]$ to denote the symbol in the i-th row and j-th column of L.

(Latin squares are often defined such that the first row consists of the numbers 1 to n in order, since we can always put them in this form by swapping the symbols. For our purposes, this would be slightly less convenient, so we will break with this convention.)

Lemma 2.14. *Unique ternary constraints on an alphabet of size n are in one-to-one correspondence with $n \times n$ Latin squares. For each Latin square $L[\cdot, \cdot]$, the corresponding constraint is satisfied with $u = i, v = j, w = k$ if and only if $L[i, j] = k$.*

Proof. Let c be a constraint incident on three variables u, v, w. We associate u with the rows of a Latin square, v with the columns, and w with the symbols. We then fill in our Latin square by placing in row i, column j the unique value that w should take to satisfy c when u takes value i and v takes value j. Every symbol occurs exactly once in every column, because once we pick a column (and thus a value for v) and a symbol (and thus a value for w), there is a unique choice of value for u, and thus a unique row which contains the correct symbol. A similar argument shows that every symbol occurs exactly once in every row. $\square$

We also know that the multiplication table of a group of order n will always be an $n \times n$ Latin square. In general, Latin squares can have a much weaker algebraic structure, called a quasigroup.

Definition 2.15 (Quasigroup)**.** A quasigroup is a set Q, together with a binary operation $* : Q \times Q \to Q$, such that, for all elements $a, b \in Q$, there exist unique elements $x, y \in Q$ such that

$$a * x = b$$

and

$$y * a = b.$$

Lemma 2.16. *Every Latin square is the multiplication table of a quasigroup. Conversely, the multiplication table of every quasigroup is a Latin square.*

Proof. Suppose we have a Latin square, with symbols $a_1, \ldots, a_n$. We define a quasigroup on the a_i by first choosing an arbitrary association between the a_i's and the rows of our Latin square, and another arbitrary association between the a_i's and the columns of our Latin square. Without loss of generality, let a_i

be associated to the i-th row and i-th column. Then we define $a_i * a_j$ to be the symbol found in the i-th row and j-th column of our Latin square. We need to check that for all a_i, a_j, there exist unique a_k, a_ℓ such that $a_i * a_k = a_\ell * a_i = a_j$. Then k will be the index of the column in which the i-th row has the symbol a_j, and ℓ will be the index of the row in which the i-th column has the symbol a_j. Both k and ℓ exist and are unique, so a_k and a_ℓ exist and are unique. Therefore, this construction gives a quasigroup of which our Latin square is the multiplication table.

The converse is also clear: let Q be a quasigroup with elements $a_1, \ldots, a_n$, and consider its multiplication table. Every a_k occurs exactly once in the i-th row of the multiplication table of Q, since $a_i * x = a_k$ has a unique solution, and occurs exactly once in the j-th column of the multiplication table, since $y * a_j = a_k$ has a unique solution. □

This tells us that the most general unique constraint is given by quasigroup multiplication. Quasigroups do not have much structure (in particular, they are not associative), so this is probably not all that useful. We point out, however, that we can use the multiplication table of a group as a unique constraint. We can also have constraints of the form $uvw = g$, where our variables u, v, w take values in a group G, and $g \in G$. This gives some examples of unique constraints that are not the multiplication tables of groups.

Definition 2.17. Suppose that σ takes values in some group $(G, *)$. A *G-constraint* on three variables u, v, w is a constraint that is satisfied iff $\sigma(u) * \sigma(v) = \sigma(w)$. A *$G$-product constraint* on three variables u, v, w is a constraint that is satisfied iff $\sigma(u) * \sigma(v) * \sigma(w) = g$ for some fixed $g \in G$.

If our constraints are all G-constraints or G-product constraints for $G = \mathbb{Z}_p$, we can find an assignment satisfying all equations, if one exists, using Gaussian elimination. In general, it seems that we cannot adapt this method. For non-abelian groups, it is often not possible to simplify equations, so that we cannot make progress by substituting one equation into another. We could even define two different constraints in terms of two different groups, or define two different constraints in terms of the same group, but with different elements of the alphabet playing the role of different elements of the group. We could define all the constraints in terms of a single non-associative quasigroup; without associativity, we cannot do substitution. It does not seem like the method can be adapted.

If we have two unique constraints, one on variables u, v, w, the other on variables u, x, y, then we can put them together to get a unique constraint on v, w, x, y: for every value for v, w, x, there is a unique value for u that will satisfy the first constraint. This give us values for u, v, w, x, and thus a unique value for y. We can do elimination this way, but we will not necessarily be able to do it in polynomial time. Our constraints could start to involve a large number of variables. There does not seem to be any compact representation of a large unique constraint, other than a truth table or a Latin-square-like representation. This will have exponential size when it involves a linear number of variables.

Therefore, for unique k-CSP's, $k \geq 3$, there does not seem to be a polynomial time algorithm for finding exact solutions when they exist.

This is in contrast to UG, where there is such an algorithm: fix one root variable v, guess a value for v, and then, while some constraint is between two variables x and y, and x has been given a value and y has not, set $v = \pi_x y(x)$. This process will eventually either cause a contradiction, or fill in all the values in v's connected component with a consistent assignment. (If there is more than one connected component, then we really just have multiple problems to solve which are unrelated, so we can apply this method to each component.) If we try all possible values for v, and there is a solution, we are guaranteed to find it. This algorithm works in polynomial time, since we are performing a breadth-first search on the constraint graph in each round, and there are $|\Sigma|$ rounds. ($|\Sigma|$ is a constant for purposes of evaluating the running time.)

3 Expansion on Hypergraphs

In the rest of this paper, we will be relating the approximability of k-CSP's to their combinatorial structure. When we do this, we will be thinking of each k-CSP as having an underlying hypergraph, as follows:

Definition 3.1 (Underlying Hypergraph)**.** The *underlying hypergraph* of a k-CSP $P = (V, C, \Sigma)$ is the k-uniform hypergraph $G = (V, E)$ with vertex set equal to V, P's set of variables, and edge set $E = \{\{v_1, v_2, \ldots, v_k\} \mid \exists c \in C$ s.t. $c \ni v_1, \ldots, v_k\}$. If P is a weighted k-CSP, then we will get a weighted hypergraph, where $wt_G(\{v_1, v_2, \ldots, v_k\}) = \sum_{c \in C, c \ni v_1, \ldots, v_k} wt_P(c)$

Our main result, Theorem 1.14, depends on a notion of hypergraph expansion. Several reasonable but not equivalent definitions of hypergraph expansion are possible. One such definition is given by Friedman and Wigderson in [20], which generalizes the second eigenvalue characterization of expander graphs to hypergraphs in a linear-algebraic way. They are able to show that their hypergraphs satisfy some isoperimetric inequalities, in much the same way that expander graphs do. Our definition is more combinatorial than theirs, and is implied by theirs.

The rest of the section is organized as follows: In Section 3.1, we give some preliminary definitions that allow us to discuss hypergraphs. In Section 3.2, we give the definition of hypergraph expansion that we will need later, which is Definition 3.4.

After that, we prove some theorems about local expanders which are not central to the main theorems of this paper, but which do help to justify the naturalness of our definition. Specifically, we prove in Section 3.3 that random hypergraphs with an appropriate edge density satisfy our definition with high probability, and in Sections 3.4 and 3.5 we give one explicit and one semi-explicit construction of local expanders, both using techniques from algebra.

3.1 Hypergraph Preliminaries

We will assume that our hypergraphs are undirected, i.e., that all the different orderings of a triple u, v, w describe the same edge. We will usually assume that our hypergraph has non-degenerate edges, i.e., that every edge $\{u, v, w\}$ is between distinct u,v, and w. In particular, we will make this assumption everywhere in Sections 4, 5 and 6. We will call such hypergraphs *simple*. A non-simple hypergraph will have degenerate edges of the form $\{u, u, v\}$ and $\{u, u, u\}$. (These are multisets: $\{u, u, v\}$ and $\{u, v, v\}$ are different edges.)

We will, however, allow multiple copies of the same edge, even in simple hypergraphs, to accommodate k-CSP's which have two different constraints on the same k variables.

We will use the notation $E(A, B, C)$ to denote the set of hyperedges with one vertex in A, one vertex in B, and one vertex in C. We will sometimes abuse notation slightly by writing, for instance, $E(a, B, C)$ instead of $E(\{a\}, B, C)$ when a is a single vertex. When we have multiple edge sets, say E and F, we will write $E(A, B, C)$ and $F(A, B, C)$ to distinguish between the set of such edges drawn from E and the set of such edges drawn from F.

We will let the weight of a set of edges $F \subset E$ be $wt(F) = \sum_{f \in F} wt(F)$.

We will define the *weighted degree* of a vertex to be $d(u) = \sum_{\{v,w\},v,w \neq u} wt(\{u, v, w\})+ 2\sum_{v \neq u} wt(\{u, u, v\})+3wt(\{u, u, u\})$. (Note that we will have $d(u) = wt(E(u, V, V))$ when our hypergraph is simple, but not otherwise.) We want to count edges of the form $\{u, u, v\}$ twice towards the degree of u, and edges of the form $\{u, u, u\}$ three times. We are doing this because it is convenient to think of the degrees as a probability distribution over the variables:

Definition 3.2 (Degree Distribution)**.** Consider the following method for selecting a random vertex:

1. Select a hyperedge e with probability $wt(e)/wt(E)$.
2. Select one of its three incident vertices uniformly at random.

Then the probability of selecting u is $d(u)/(3wt(E))$, since the weight of each edge e involving u gets added with weight $\frac{k}{3}$, where k is the number of times u appears in e (and thus $\frac{k}{3}$ is the probability that we pick u in Step 2 of our selection procedure). We will call this distribution the *degree distribution*. We will let

$$\Delta(u) = \frac{\frac{1}{3}d(u)}{wt(E)}$$

be the weight of u in this distribution, and we will write $u \sim \Delta$ when we wish to pick u from this distribution.

We can check that $\sum_u \Delta(u) = 1$: each simple edge $\{u, v, w\}$ with u, v, w distinct gets counted with a third of its normalized weight $wt(\{u, v, w\})/wt(E)$ in each of $\Delta(u), \Delta(v), \Delta(w)$, so each simple edge contributes exactly its normalized weight. Degenerate edges of the form $\{u, u, v\}$ count three times, twice in $\Delta(u)$ and once in $\Delta(v)$, and degenerate edges of the form $\{u, u, u\}$ count three times in $\Delta(u)$, so these edges also contribute exactly their normalized weight. Therefore, the sum of $\Delta(u)$ is one.

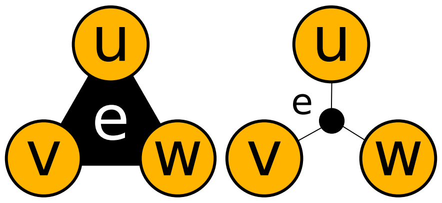


Figure 2: Two ways of drawing hypergraphs. Both pictures depict a hyperedge e between vertices u, v, and w.

Finally, a word about drawing hypergraphs. In this paper, we will use one of two conventions to represent edges in a hypergraph: (shown in Figure 2) (1) a filled-in triangle whose points are the incident vertices, or (2) a smaller black circle, which is connected by a line to each of the three incident vertices. This second method was suggested by Max Shron.

3.2 Local Expansion

In this section, we define the concept of local expansion, which is used in section 4 in the proof of our main technical lemma, Theorem 1.12. The general idea is that the hypergraph does not look like this at any vertex:

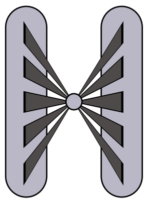


Figure 3: This graph is very far from being a local expander. We can divide the neighbors of the central vertex into two classes such that every incident edge lies entirely on the right side or entirely on the left side.

That is, we do not want it to be the case that any vertex u has two disjoint sets of neighbors S and T, with few hyperedges involving both sets. Throughout this section, let $G = (V, E)$ be a 3-uniform hypergraph with vertices V and edges E.

Definition 3.3 (Neighborhood Graph)**.** Let $u \in V$ be a fixed vertex of G. The *neighborhood graph of G at u*, which we denote $G_u = (V_u, E_u)$, is a graph with vertex set $V_u = \{v \mid \{u, v, w\} \in E \text{ for some } w \in V\}$, and edge set $E_u = \{\{v, w\} \mid \{u, v, w\} \in E\}$. If G is weighted, then G_u is weighted with weight function $wt_u(\{v, w\}) = wt(\{u, v, w\})$.

When G is non-simple, we do the most natural thing, which is to map edges of the form $\{u, v, v\}$ to loops $\{v, v\}$ in G_u, edges of the form $\{u, u, v\}$ to edges $\{u, v\}$ in G_u, and edges of the form $\{u, u, u\}$ to loops $\{u, u\}$ in G_u.

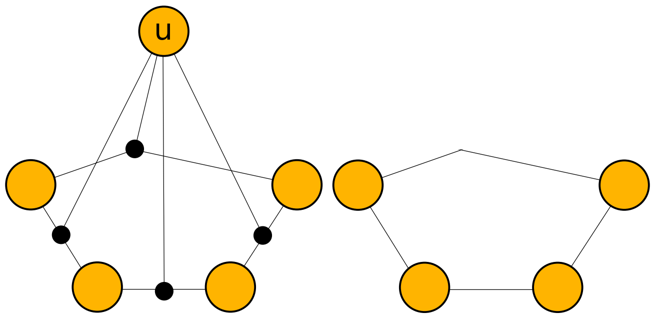


Figure 4: Left: a hypergraph G. Right: the neighborhood graph G_u. Every edge in G_u arises from a hyperedge in G containing u.

We define $d_u(v) = \sum_{w \in V_u \setminus \{v\}} wt(\{u, v, w\}) + 2wt(\{u, v, v\})$, which is just the weighted degree of v in G_u (with loops counted twice). For $S \subseteq V_u$, we will let $d_u(S) = \sum_{v \in S} d_u(v)$.

We would like to define a marginal degree distribution $\Delta_u(v)$, which is the probability that we get v if we randomly select an edge $\{v, w\}$ in E_u with probability $wt_u(\{v, w\})/wt_u(E_u)$, and then randomly select one of its two endpoints with probability $\frac{1}{2}$. Equivalently, we could say that $\Delta_u(v)$ is the probability that we generate v in the following generation procedure:

1. Pick a random edge e containing u, according to the weight distribution.
2. Remove one copy of u from e, and call the resulting edge e'. If e is of the form $\{u, u, v\}$, then e' is $\{u, v\}$. If e is of the form $\{u, u, u\}$, then e' is $\{u, u\}$.
3. Pick one of the two vertices in e', each with probability $\frac{1}{2}$.

From this, it is clear that $\Delta_u(v) = d_u(v)/2wt(E_u)$, since each simple edge $\{u, v, w\}$ contributes half its weight (normalized in E_u) to each of $\Delta_u(v)$ and $\Delta_u(w)$, and each edge $\{u, v, v\}$ contributes all its weight (normalized in E_u) to $\Delta_u(v)$.

Now we are ready to give one of the main technical definitions in this paper, which is the notion of expansion that we need in the proof of our main technical lemma.

Definition 3.4 ((α, β)-Local Expansion)**.** Fix $u \in V$, and let $G_u = (V_u, E_u)$ be the neighborhood graph of G at u. If, for all $S \subseteq V_u$ with $\alpha d_u(V_u) \leq d_u(S) \leq$

$\frac{1}{2}d_u(V_u)$ (or equivalently $\alpha \leq \Delta_u(S) \leq \frac{1}{2}$), we have

$$\begin{aligned} wt(E(u, S, V_u \setminus S)) &\geq \beta \cdot \frac{1}{2} d_u(V_u) \\ &= \beta \cdot wt(E_u) \end{aligned}$$

we say that G is an (α, β)-local expander at u. We say that G is an (α, β)-local expander if it is an (α, β) local expander at all $u \in V$.

We will call a pair $(S, \overline{S})$ a "cut" on the neighborhood graph, where $\overline{S} = V_u \setminus S$. Here we are thinking of trying to separate V_u into two sets S and $\overline{S}$, so that not many edges are cut, i.e., go between S and $\overline{S}$. On a local expander, this is not possible unless one of S or $\overline{S}$ is very small. We will follow the convention that S is the smaller of the two sets in $(S, \overline{S})$, i.e., that $d_u(S) \leq d_u(\overline{S})$.

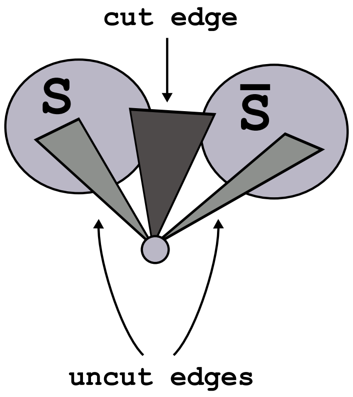


Figure 5: If we are considering the neighborhood graph of the bottom vertex, with cut $(S, \overline{S})$, then only edges going between S and $\overline{S}$ are cut.

Note that this definition only makes sense for $\alpha \leq \frac{1}{2}$, and that an (α, β)-local expander is also an (α', β)-local expander for $\alpha' > \alpha$ and an (α, β')-local expander for $\beta' < \beta$. This is true because moving from α to α' strengthens the assumptions in our definition, and moving from β to β' weakens the conclusion in our definition.

A rather trivial example of a local expander is provided by the complete 3-uniform hypergraph.

Example 3.5 (Complete uniform hypergraph)**.** Let $K^{(3)}_{n+1}$ be the complete 3-uniform hypergraph on $n+1$ vertices, with all possible edges $\{u, v, w\}$ with no two of u, v, w equal, and where all edges have weight 1. Then every neighborhood graph $(K^{(3)}_{n+1})_u$ is just the complete graph K_n on $V_u = V \setminus \{u\}$. Let $(S, \overline{S})$ be a cut (with $|S| = s \leq \frac{1}{2}n$). Then $|E(u, S, V_u \setminus S)| = s(n-s)$, and $|E(u, V_u, V_u)| = \binom{n}{2}$. If $d_u(S) = |S|(n-1)$ is greater than $\alpha \cdot d_u(V_u) = \alpha \cdot 2\binom{n}{2} = \alpha n(n-1)$ then $|S| \geq \alpha n$, and $|S| \leq \frac{1}{2}n$. Then we have $s(n-s) \geq \alpha n(1-\alpha)n$, since $x(n-x)$ is greatest at $x = n/2$ and is increasing in x for $x < n/2$.

Then $wt(E(u, S, V_u \setminus S)) = |E(u, S, V_u \setminus S)| \geq \alpha(1-\alpha)n^2 > \alpha(1-\alpha)2\frac{n(n-1)}{2}$, which is equal to $2\alpha(1-\alpha)wt(E(u, V_u, V_u))$. Therefore, the complete hypergraph $K_{n+1}^{(3)}$ is an $(\alpha, 2\alpha(1-\alpha))$-local expander for all α with $0 < \alpha \leq \frac{1}{2}$. In particular, it is an (α, α)-local expander, since $2(1 - \alpha) \geq 1$ for $\alpha \leq \frac{1}{2}$.

Local expansion is not completely analogous to expansion in graphs, since local expanders do not have to be connected. Local expansion is, as the name suggests, a local phenomenon, not a global one.

Remark 3.6. Consider any hypergraph G, and let $G_2 = G \amalg G$ be the union of two disjoint copies of G. Then G_2 is disconnected. But, if G were an (α, β)-local expander, G_2 is also an (α, β)-local expander. To see this, note that any neighborhood graph $(G_2)_u$ lies entirely inside one of the copies of G, and thus G_2 inherits the local expansion of G.

3.3 Random Hypergraphs are Local Expanders

We now wish to prove that a random hypergraph of sufficiently high edge density is almost surely a local expander. We can compare this result to a similar result in [20]: there, for a different definition of hypergraph expansion, it is proved that hypergraphs of a comparable density (in a slightly different random model) are almost surely expanders. Our result is slightly stronger (theirs requires some extra log factors in the edge density), since our notion of hypergraph expansion is weaker.

We wish to model hypergraphs with edge density d, where we mean that the hypergraph on n vertices has about dn^2 edges. For example, the complete hypergraph would have edge density $\Theta(n)$, and a Steiner triple system[1] would have constant density. As is usually the case with random graph theory, it is much simpler to consider the model of independent edges, each included with probability p. We will call p the *edge probability*. If we want to have roughly dn^2 edges, then we should set $p = d/n$.

We only want to deal with simple hypergraphs in this section. Therefore, we will include each of the $\binom{n}{3}$ possible undirected, non-degenerate edges ($\{u, v, w\}$ with u, v, w distinct) with probability $p = \frac{d}{n}$. Then we will have $\Theta(dn^2)$ edges in expectation, and by Chernoff bounds, we have $\Theta(dn^2)$ edges with all but exponentially small probability.

We will prove:

Theorem 3.7. *There exist constants c (sufficiently large) and C (sufficiently small) such that the following holds: Let G be a random hypergraph, chosen according to the independent edge model with probability $p = \frac{d}{n}$, for any $d \geq c \log n$. Then, for every $\alpha \geq \alpha_{min} = \frac{1}{\log n}$, G is an $(\alpha, C\alpha)$-local expander with probability at least $1 - e^{-\Omega(n \log n)}$.*

First, we need a particular form of the Chernoff bound:

[1] A Steiner triple system is a set system $\mathcal{S}$ on $[n]$ consisting of triples $\{i, j, k\}$ such that, for every $i, j \in [n]$, $i \neq j$, there is a unique k such that $\{i, j, k\} \in \mathcal{S}$.

Theorem 3.8 ([36], Theorem 4.4, p. 64)**.** *Let $X_1, \dots X_n$ be independent $0-1$ random variables. Let $X = \sum_{i=1}^{N} X_i$ and $\mu = \mathbb{E}[X]$. Then, for $0 < \delta \leq 1$,*

$$\Pr[X \geq (1+\delta)\mu] \leq e^{-\mu\delta^2/3}$$

Theorem 3.9 ([36], Theorem 4.5, p. 66)**.** *Let $X_1, \dots X_n$ be independent $0-1$ random variables. Let $X = \sum_{i=1}^{N} X_i$ and $\mu = \mathbb{E}[X]$. Then, for $0 < \delta < 1$,*

$$\Pr[X \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$$

Let $G = (V, E)$ be our hypergraph.

To simplify matters, we will take our cuts to be $(S, V \backslash S)$ instead of $(S, V_u \backslash S)$ (even when $V_u \neq V$) for the rest of this section.

Recall that, for G to be an (α, β)-local expander, we must have $|E(u, S, V \setminus S)| \geq \beta |E(u, V, V)|$ for every u, and for every subset $S \subset V$ with $\alpha d_u(V_u) \leq d_u(S) \leq \frac{1}{2} d_u(V_u)$. Let us call a cut $(S, \overline{S})$ that fails this test a $(\geq \alpha, < \beta)$ cut: S has at least an α fraction of G_u's vertices (weighted by degree), but a less than β fraction of G_u's edge cross the cut.

To prove Theorem 3.7, we wish to bound the probability that for any $u \in V$, the neighborhood graph G_u has a $(\geq \alpha, < \beta)$ cut on it. We will use the union bound over possible choices of u and S, so let us fix a vertex $u \in V$, and a cut $(S, \overline{S})$ on G_u. Let $s = |S|$. We want to estimate the probability that $(S, \overline{S})$ is a $(\geq \alpha, < \beta)$ cut. We first need to know how many edges G_u contains. We can get a fairly good estimate of this using Chernoff bounds:

Lemma 3.10. *Let $u \in V$ be any vertex. With probability at least $1 - e^{-\Omega(dn)}$, both of the following hold:*

1. *$E(u, V, V) \leq C_3 dn$*
2. *$E(u, V, V) \geq C_4 dn$*

Proof. We will have an indicator variable X_i for every potential edge in G_u. Let $X = \sum X_i = |E(u, V, V)|$. Let $\mu = \mathbb{E}[X]$. Then $\mu = p\binom{n-1}{2} = \frac{d}{n}\frac{(n-1)(n-2)}{2} \leq \frac{1}{2} dn$. We also have that $\mu \geq \frac{49}{100} dn$ for sufficiently large n, since $\binom{n-1}{2} \sim \frac{1}{2} n^2$. Then we apply the Chernoff bounds given by Theorems 3.8 and 3.9.

$$\Pr[|E(u, V, V)| \geq (1+\delta)\mu] \leq e^{-\mu\delta^2/3}$$
$$\Pr[|E(u, V, V)| \leq (1-\delta)\mu] \leq e^{-\mu\delta^2/2}$$

Since, for sufficiently large n, $\frac{49}{100} dn \leq \mu \leq \frac{1}{2} dn$, this implies

$$\Pr[|E(u, V, V)| \geq (1+\delta)\frac{1}{2} dn] \leq e^{-\mu\delta^2/3}$$
$$\Pr[|E(u, V, V)| \leq (1-\delta)\frac{49}{100} dn] \leq e^{-\mu\delta^2/2}$$

We will choose $\delta = \frac{1}{2}$, so that $C_3 = \frac{3}{4}$ and $C_4 = \frac{49}{200}$. The probability that at least one of our two inequalities fails to hold is thus at most $2e^{-\delta^2\mu/3} = e^{-\frac{1}{4}\frac{1}{2}\frac{1}{3}dn} = e^{-\Omega(dn)}$. □

Lemma 3.11. *Let $u \in V$ be any vertex, and let $(S, \overline{S})$ be any cut on G_u, with $s = |S|$. With probability at least $1 - e^{-\Omega(ds)}$, $d_u(S) \le C_1 ds$.*

Proof. Again, we can make an indicator variable X_i for each potential edge $\{u, v, w\}$, where u is our fixed vertex, $v \in S$, and w is any third distinct vertex, except that we need to only count edges with both v and $w \in S$ once. Let $X = \sum X_i = d_u(S)$. There are $s(n-2) - \binom{s}{2}$ potential edges, so $\mu = \mathbb{E}[d_u(S)] = p\left[s(n-2) - \binom{s}{2}\right] = \frac{d}{n}s(n-2) = \frac{d}{n}\left[s(n-2) - \binom{s}{2}\right]$. Then we have $\mu < ds$, and $\mu = \frac{d}{n}\left[s(n-2) - \binom{s}{2}\right] = \left[\frac{n-2}{n} - \frac{1}{2}\frac{s-1}{n}\right] ds \ge \left[\frac{99}{100} - \frac{1}{4}\right] ds = \frac{74}{100}ds$ for n sufficiently large, so we can say by Theorem 3.8 that

$$\begin{aligned} \Pr[X \ge (1+\delta)\mu] &\le e^{-\mu\delta^2/3} \\ \Pr[X \ge (\frac{3}{2})ds] &\le e^{-\frac{74}{100}ds\frac{1}{4}/3} \\ &= e^{-\frac{111}{200}}ds \end{aligned}$$

where we have chosen $\delta = \frac{3}{2}$, made the event whose probability we are measuring on the left less likely by increasing the bound, and we have made the number on the right bigger by making the number in the exponent less negative.

The probability that this inequality is not satisfied is at most $e^{-\frac{111}{200}ds} = e^{-\Omega(ds)}$. □

Unfortunately, the probability bound that we get in the previous lemma depends on s, so when S is too small this probability is not as small as we would like. This is why we need α_{min} - once S gets this large, the previous lemma give us a good bound on the probability.

Lemma 3.12. *Let $u \in V$ be any vertex. Let $(S, \overline{S})$ be a cut satisfying $|S| \ge \alpha n$, for $\alpha \ge \alpha_{min}$. Then, with probability at least $1 - e^{-\Omega(\alpha_{min} dn)}$, $E(u, S, \overline{S}) \ge C_2 \alpha dn$.*

Proof. Again, we apply the Chernoff bound. We will have a set of indicator variables $X_1, \ldots, X_N$, for each potential edge that crosses $(S, \overline{S})$, where X_i is 1 if the corresponding edge is present, and 0 otherwise. So we will have $N = s(n-1-s)$. Let X be the sum of the X_i. What is $\mathbb{E}[X]$? By linearity of expectation, $\mathbb{E}[X]$ is just $\mu = ps(n-1-s)$. If we assume that $\alpha n \le s \le \frac{1}{2}n$, then we have $\mu = p \cdot s \cdot (n-1-s) \ge p \cdot \alpha n \cdot (\frac{1}{2}n - 1) \ge \frac{49}{100}\alpha p n^2$ for n sufficiently large. This is then equal to $\frac{49}{100}\alpha n d$.

Then we have

$$\begin{aligned} \Pr[|E(u, S, \overline{S})| \le (1-\delta)\mu] &\le e^{-\mu\delta^2/2} \\ \Pr[|E(u, S, \overline{S})| \le (1-\delta)\frac{49}{100}\alpha n d] &\le e^{-\frac{49}{100}\alpha n d\delta^2/2} \\ &\le e^{-\frac{49}{100}\alpha_{min} n d\delta^2/2} \end{aligned}$$

In the second step, we have used the fact that $\mu \geq \frac{1}{2}\alpha nd$ to make the event whose probability is measured on the left-hand side less likely, by tightening the bound on $E(u, S, \overline{S})$, and we have made the number on the right-hand side larger. We have also made the number of the right-hand side larger by passing from α to α_{min}. Therefore, this inequality is, if anything, even more unequal.

If we plug in $\delta = \frac{1}{2}$, we get

$$\Pr[|E(u, S, \overline{S})| \leq \frac{49}{200}\alpha nd] \leq e^{-\frac{49}{800}\alpha nd}.$$

If we let $C_2 = \frac{49}{200}$, we get that the desired inequality holds with probability at least $1 - e^{-\frac{49}{800}} = 1 - e^{-\Omega(\alpha_{min} dn)}$. □

This bound tells us that with high probability, a large number of edges cross the cut. The previous lemmas told us that with high probability, the number of edges with at least one vertex in S was not too large, and that the number of edges in the neighborhood graph is about what we expect it to be. The next lemma says that, if all these conditions hold simultaneously, the cut we are examining cannot be a bad cut.

Lemma 3.13. *Let $u \in V$ be any vertex, and let $(S, \overline{S})$ be any cut on G_u. Let $s = |S|$. If G satisfies the following conditions:*

1. $d_u(S) \leq C_1 ds$
2. $E(u, S, \overline{S}) \geq C_2 \frac{s}{n} dn.$
3. $E(u, V, V) \leq C_3 dn$
4. $E(u, V, V) \geq C_4 dn$

then $(S, \overline{S})$ is not a $(\geq \alpha, < \frac{C_2 C_4}{C_1 C_3}\alpha)$ cut for any value of α.

Proof. Suppose $(S, \overline{S})$ is a $(\geq \alpha, < \frac{C_2 C_4}{C_1 C_3}\alpha)$ cut. Then $d_u(S) \geq \alpha d_u(V_u)$, so we know that $d_u(S) \geq \alpha C_4 dn$ by (4). We also know that $d_u(S) \leq C_1 ds$ by (1). Putting these together, we have $\alpha C_4 dn \leq d_u(S) \leq C_1 ds$, so $s \geq \frac{C_4}{C_1}\alpha n$. Let $\alpha' = \frac{C_4}{C_1}\alpha$, so that $s = |S| \geq \alpha' n$.

Then apply (2) to get $|E(u, S, \overline{S})| \geq C_2 \alpha' dn = \frac{C_2}{C_3}\alpha' C_3 dn \geq \frac{C_2}{C_3}\alpha' |E(u, V, V)|$ by (3), and then $|E(u, S, \overline{S})| \geq \frac{C_2 C_4}{C_1 C_3}\alpha |E(u, V, V)|$, which is what we wanted. □

Proof of Theorem 3.7. Let $u \in V$, and let $S \subset V_u = V \setminus \{u\}$ be a cut on G_u. Let $\alpha_S = d_u(S)/d_u(V_u)$. We claim that it is enough to show that $(S, \overline{S})$ is not a $(\geq \alpha_S, < C\alpha_S)$ cut. Clearly $(S, \overline{S})$ is not a $(\geq \alpha, < C\alpha)$ cut for $\alpha > \alpha_S$, since then it would not satisfy $d_u(S) = \alpha_S d_u(V) \geq \alpha d_u(V_u)$.

Suppose that $\alpha < \alpha_S$. Then

$$wt(E(u, S, V_u \setminus S)) < C\alpha \frac{1}{2} d_u(V_u)$$

implies that

$$wt(E(u, S, V_u \setminus S)) < C\alpha_S \frac{1}{2} d_u(V_u).$$

So, if $(S, \overline{S})$ were a $(\geq \alpha, < C\alpha)$ cut, it would also be a $(\geq \alpha_S, < C\alpha_S)$ cut. So, if $(S, \overline{S})$ is a problematic cut of any sort, it is going to be a $(\geq \alpha_S, < C\alpha_S)$ cut.

Therefore, if $(S, \overline{S})$ is a $(\geq \alpha, < C\alpha)$ cut, then one of the statements in Lemma 3.13 is false for S. We know that this happens with probability at most $e^{-\Omega(dn)} + e^{-\Omega(ds)} + e^{-\Omega(\alpha_{min} dn)}$ for any S satisfying $\alpha_{min} n \leq |S| \leq \frac{1}{2} n$, and we can use the union bound to show that with high probability it does not happen for any choice S.

There are fewer than 2^n possible cuts S for each neighborhood graph, and there are n neighborhood graphs, so we can use the union bound to say that

$$\Pr[\exists (\geq \alpha, < C\alpha) \text{ cut}] < n2^n [e^{-\Omega(dn)} + e^{-\Omega(ds)} + e^{-\Omega(\alpha_{min} dn)}].$$

For $d \geq c \log n$, $\alpha_{min} = \frac{1}{\log n}$, and $s \geq \alpha_{min} n$, we get that this is at most

$$\begin{aligned} & n2^n [e^{-\Omega(c \log n \cdot n)} + e^{-\Omega(c \log n \cdot \frac{n}{\log n})} + e^{-\Omega(\frac{1}{\log n} \cdot c \log n \cdot n)}] \\ =& e^{\ln n} e^{(\ln 2) n} [e^{-\Omega(cn \log n)} + e^{-\Omega(cn)} + e^{-\Omega(cn)}] \\ \leq& e^{\ln n} e^{(\ln 2) n} [3e^{-\Omega(cn)}] \\ \leq& 3e^{\ln n + (\ln 2) n - cc' n} \end{aligned}$$

for some constant c'. This in turn is less than

$$\leq e^{-c'' n}$$

for a suitable choice of c'', and sufficiently large n. Thus, the probability that our graph is not a $(\alpha, C\alpha)$-local expander is at most $e^{-\Omega(n)}$. $\square$

3.4 Explicit Constructions from Gowers's Theorem

For explicit examples of graphs with good expansion, the most powerful techniques seem to come from algebra. Here we use a theorem about the combinatorial structure of finite groups to provide explicit constructions of hypergraphs with good local expansion. We note that [19] has some similar constructions based on the Cayley sum hypergraph, which achieve a different kind of hypergraph expansion.

Definition 3.14 (Cayley sum hypergraph)**.** Fix a finite group K, and a subset $H \subseteq K$. The Cayley sum hypergraph $Cay(K, H)$ is a 3-uniform hypergraph $G = (K, E)$. Our vertex set will be K, and we will make $\{u, v, w\}$ an edge if and only if $uvw \in H$.

In the usual notion of a Cayley graph, H would be a set of generators; here, we do not require that the elements of H generate K. When H is larger than the largest proper subgroup of K, it will generate K, regardless of what H is.

This hypergraph will be non-simple: if $uuv \in H$, then $\{u, u, v\}$ is an edge, and if $uuu \in H$, then $\{u, u, u\}$ is an edge.

For a vertex u, there are three types of edges, corresponding to the three different ways that u can be in a triple product:

$$ust = h$$
$$sut = h$$
$$stu = h$$

Since, in this section, we will be interested exclusively in nonabelian groups (and in particular groups that are "far from abelian", in the sense that they do not have nontrivial representations of low dimension), we will consider these three edges to be distinct, so that they are counted multiple times if, for example, ust and sut are equal, or both are in H. We will also double count edges of the form $uut = h$, and triple count edges of the form $uuu = h$.

We will later need to know how many edges there are in G_u. We can generate a random edge in G_u by first fixing u, then picking one of the three edge types above, and then picking $s \in K$ and $h \in H$ uniformly at random, and solving for t. For any fixed $s \in K, h \in H$, each of the three edge types gives a unique value for $t \in K$. Since we are counting different orderings separately, this gives us $3|K||H|$ total edges in G_u. We can also see that these hypergraphs have the nice property that the neighborhood graphs are all regular.

We can give fairly strong bounds on expansion because of the following theorem of Gowers:

Theorem 3.15 (Gowers, [23], Theorem 3.3)**.** *Let $A, B, C \subseteq K$ be nonempty subsets of a group K. Let m be the minimum dimension of a nontrivial representation*[2] *of K. Let $N(A, B, C)$ denote the number of solutions to the equation $xy = z$ where $x \in A, y \in B, z \in C$.*

If $\gamma > 0$ and $|A||B||C| \geq \gamma n^3/m$ then

$$N(A, B, C) > \left(1 - \frac{1}{\sqrt{\gamma}}\right) \frac{|A||B||C|}{n}.$$

Gowers calls groups with m large "quasirandom", since the number of solutions $N(A, B, C)$ is close to what it would be for a random choice of A, B, and C, simultaneously for all sufficiently large A, B, and C.

To put this in context, for $K = SL_2(q)$, we have $m \sim n^{1/3}/2$.

Lemma 3.16. *If $G = Cay(K, H)$ is the Cayley sum hypergraph of a group K, and H is any subset of K satisfying $|H| \geq cn/m$, then G is an $(\alpha, \frac{1}{4}\alpha)$-local expander for all $\alpha \geq 8/c$.*

[2]A representation of a group K is a homomorphism $\rho : K \to GL_m(\mathbb{C})$ from K to the general linear group of invertible $m \times m$ matrices with entries in $\mathbb{C}$. The *dimension* of such a representation is the number m. A representation is nontrivial if the homomorphism ρ maps at least one element of K to a matrix in $GL_m(\mathbb{C})$ other than the identity matrix I_m. Every group has a trivial representation of dimension 1 that arises from the map $\rho : k \mapsto (1) \in GL_1(\mathbb{C})$. Every finite group K has a nontrivial representation of dimension $|K|$, so the number m will in general be somewhere in between 1 and $|K|$, and, in fact, the smallest such m for a nontrivial representation will always be significantly smaller than $|K|$. For (nontrivial) abelian groups, this number will be 1, so when this number is large, it is reasonable to say that the group is "far from abelian".

Proof. Let $n = |K|$.

Let $u \in K$, and consider the neighborhood graph $G_u = (K, E_u)$. Let $S \subseteq K$, and let $T = K \setminus S$.

We need to estimate two quantities: how many edges are cut by (S, T), i.e., go from S to T, and how many edges there are in total in G_u.

We already know the second quantity - G_u contains $3n|H|$ edges.

Now, to estimate $|E(u, S, T)|$, we use Theorem 3.15. For edges of types $ust = h, sut = h, stu = h$, we can group these equations as $(us)t = h$, $(su)t = h$, and $s(tu) = h$, so that we have, respectively $N(uS, T, H), N(Su, T, H), N(S, Tu, H)$ edges in G_u that go from S to T. (Note that we are using the double and triple counting of degenerate edges here, to divide $E(u, S, T)$ into three types of edges depending on the position of u, knowing that we will count each edge the correct number of time.) We then have the bound $|E(u, S, T)| = N(uS, T, H) + N(Su, T, H) + N(S, Tu, H) \geq 3\left(1 - \frac{1}{\sqrt{\gamma}}\right)\frac{|S||T||H|}{n}$, as long as we have $|S||T||H| \geq \gamma n^3/m$, and can apply Theorem 3.15. (Since $|uS| = |Su| = |S|$ and $|Tu| = |T|$.)

Now, suppose that $|S| = \theta n$. Then $T = (1-\theta)n$. G is pairwise-$3|H|$-regular (i.e., for any pair of vertices $u, v \in K$, including pairs with $u = v$, there are exactly $3|H|$ edges of the form $\{u, v, w\}$), so this is equivalent to saying that $d_u(S) = \theta d_u(K)$.

Then we have

$$\begin{aligned}
|E(u, S, T)| &> 3\left(1 - \frac{1}{\sqrt{\gamma}}\right)\frac{\theta n(1-\theta)n|H|}{n} \\
&= \left(1 - \frac{1}{\sqrt{\gamma}}\right)\theta(1-\theta)\cdot 3n|H| \\
&= \left(1 - \frac{1}{\sqrt{\gamma}}\right)\theta(1-\theta)\cdot |E(u, K, K)|
\end{aligned}$$

Therefore, if $|S| = \theta n$, and $|S||T||H| = \theta n(1-\theta)n|H| \geq \gamma n^3/m$ (or equivalently, $|H| \geq \frac{\gamma n}{\theta(1-\theta)m}$) then $|E(u, S, T)| \geq (1 - \frac{1}{\sqrt{\gamma}})\theta(1-\theta)|E(u, K, K)|$.

If we let $\gamma = c\theta(1-\theta)$, then as long as $|H| \geq cn/m$, we have

$$\begin{aligned}
|E(u, S, T)| &\geq \left(1 - \frac{1}{\sqrt{c\theta(1-\theta)}}\right)\theta(1-\theta)|E(u, K, K)| \\
&= (1-\theta)\left(1 - \frac{1}{\sqrt{c\theta(1-\theta)}}\right)\theta|E(u, K, K)| \\
&\geq \frac{1}{4}\theta|E(u, K, K)|
\end{aligned}$$

as long as we have

$$(1-\theta)\left(1 - \frac{1}{\sqrt{c\theta(1-\theta)}}\right) \geq \frac{1}{4},$$

which we do, since $(1-\theta) \geq \frac{1}{2}$ for $\theta \leq \frac{1}{2}$, and $\theta \geq 8/c \implies \theta(1-\theta) \geq 4/c \implies c\theta(1-\theta) \geq 4 \implies \sqrt{c\theta(1-\theta)} \geq 2 \implies \frac{1}{\sqrt{c\theta(1-\theta)}} \leq \frac{1}{2} \implies \left(1 - \frac{1}{\sqrt{c\theta(1-\theta)}}\right) \geq \frac{1}{2}$.

Thus, for all $\alpha \geq 8/c$, if $|S| = \theta n \geq \alpha n$, our hypergraph satisfies the condition to be a $(\alpha, \frac{1}{4}\alpha)$-local expander. □

We should remark that the bound derived in the preceding lemma only makes sense when the cut set S is of a certain minimum size. Otherwise the factor $1 - \frac{1}{\sqrt{c\theta(1-\theta)}}$ could become zero or negative. This factor is positive exactly when $\theta(1-\theta) > \frac{1}{c}$. Thus, for a fixed H, the lemma does not tell us anything at all about sets that are too sparse.

For $K = SL(2, q)$, we get that, as long as $H \geq (1 + o(1))2cn^{2/3}$, $G = Cay(K, H)$ is an $(\alpha, \frac{1}{4}\alpha)$-local expander for all $\alpha \geq 8/c$. Our edge density is $\Theta(|H|)$, so we see that we require a much higher edge density than we did for random hypergraphs in Theorem 3.7.

3.5 Semi-explicit Constructions from Fourier Analysis

In this section, we give a partially random, partially algebraic construction of local expanders. We will now be working with abelian groups, so we will write them additively. We will also now consider the three edges

$$u + s + t = h$$
$$s + u + t = h$$
$$s + t + u = h$$

to be the same, so that we now have $|K||H|$ edges instead of $3|K||H|$ edges.

In order to apply the machinery we wish to use, we first give a brief overview of the Fourier analysis of finite abelian groups. A less abbreviated treatment of this material can be found in [9], which we are following closely.

Definition 3.17 (Group Character)**.** A *group character* χ of a finite abelian group G is a homomorphism $\chi : G \to \mathbb{C}^\times$ from G to the multiplicative group of nonzero complex numbers. Since G is a finite group, the order of each element of G must divide $|G|$, and the same must be true of all values taken on by $\chi(a)$ for $a \in G$. Therefore, all values taken on by $\chi(a)$ are $|G|$-th roots of unity.

Definition 3.18 (Principal Character, χ_0)**.** The *principal character* of G is the trivial homomorphism $\chi_0(a) = 1$ for all $a \in G$.

Definition 3.19 (The Dual Group $\widehat{G}$)**.** The *dual group of* G, denoted $\widehat{G}$, is the group of characters of G, with the operation of pointwise multiplication:

$$(\chi_1\chi_2)(a) = \chi_1(a)\chi_2(a).$$

Clearly, the principal character χ_0 is the identity element of $\widehat{G}$.

It can be shown with a small amount of work that $\widehat{G}$ is isomorphic to G, so that $\widehat{G}$ has $|G|$ elements.

Definition 3.20 (Fourier Coefficients, $\widehat{f_A}(\chi)$)**.** Let $f : G \to \mathbb{C}$.

For a character $\chi \in \widehat{G}$, we define the Fourier coefficient of f with respect to χ to be

$$\widehat{f}(\chi) = \sum_{a \in G} \chi(a) f(a).$$

For a set $A \subseteq G$, we can define the Fourier coefficients of A by examining the Fourier coefficients of the indicator function

$$f_A(a) = \begin{cases} 1 & a \in A \\ 0 & a \notin A \end{cases}$$

We define the following measure of pseudorandomness of a set $A \subseteq G$:

Definition 3.21 ($\Phi(A)$)**.**

$$\Phi(A) = \max\{|\widehat{f_A}(\chi)| : \chi \in \widehat{G}, \chi \neq \chi_0\}$$

Intuitively speaking, the Fourier coefficients of f_A measure the correlation of f_A with different characters. When $\Phi(A)$ is low, this correlation is low for every non-principal χ, so our set does not have a structure that correlates well with the group structure of G. It is important that we omit χ_0 from the definition of Φ_A; the correlation of f_A with the principal character χ_0 just tells us the size of A, which we would like to regard as a parameter, not as a quantity which needs to be pseudorandom.

As in the previous section, we are interested in the number of solutions to the equation $a_1 + a_2 + a_3 = h$ where a_1 is fixed, $a_2 \in S$, $a_3 \in \overline{S}$, and $h \in H$, our set of generators. This is equivalent to $a_2 + a_3 - h = -a_1$.

Theorem 3.22 ([9],Theorem 4.1)**.** *Let $A_1, A_2, A_3 \subseteq G$, $a \in G$, and let N denote the number of solutions of the equation*

$$x_1 + x_2 + x_3 = a \quad (x_i \in A_i, i = 1, 2, 3).$$

Then

$$\left| N - \frac{|A_1||A_2||A_3|}{n} \right| < \Phi(A_3)\sqrt{|A_1||A_2|}.$$

Note the somewhat surprising fact that this is true for all a simultaneously.

A straightforward argument transforms this into:

Lemma 3.23. *Let K be a group, H a set of generators. Let $\alpha \in (0, \frac{1}{2}]$. The Cayley-sum hypergraph $G = Cay(K, H)$ is an (α, α)-local expander if $\Phi(H) \leq \frac{1}{2\sqrt{2}}\sqrt{\alpha}|H|$.*

Proof. First, we note that, for $\alpha \leq \frac{1}{2}$, $1 - \alpha \geq \frac{1}{2}$, so

$$\begin{aligned} \Phi(H) &\leq \frac{1}{2\sqrt{2}}\sqrt{\alpha}|H| \\ &\leq \frac{1}{2}\sqrt{\alpha(1-\alpha)}|H|. \end{aligned}$$

The second inequality is actually the condition we need; we have substituted the first inequality (which is more difficult to satisfy) to simplify the statement of the lemma.

Let $n = |K|$. Let $u \in$, and consider the neighborhood graph $G_u = (K, E_u)$. Let $S \subseteq K$, and let $T = K \setminus S$.

We need to estimate two quantities: how many edges are cut by S, i.e., go from S to T, and how many total edges there are in G_u.

The second quantity is easy. $\{u, v, w\}$ is an edge iff $u + v + w \in H$, so we need to know how many solution there are to $v + w - h = u$, where $v, w \in K$, $h \in H$. For any v, h, there is a unique such value for w, so we have exactly $n|H|$ solutions.

Now, to estimate $|E(u, S, T)|$, we need to know how many solutions there are to the equation $u + s + t = h$ for $s \in S, t \in T, h \in H$. This is the same as having $s + t - h = -u$. By the previous theorem, the number of solutions to this equation is at least

$$\frac{|S||T||H|}{n} - \Phi(H)\sqrt{|S||T|}.$$

Now, suppose that $|S| = \theta n, |T| = (1 - \theta)n$, where $\theta \geq \alpha$. The ratio we are interested in is $wt(E(u, S, T))/wt(E(u, V, V))$, which by the above is at least

$$\frac{\frac{\theta n(1-\theta)n|H|}{n} - \Phi(H)\sqrt{\theta n(1-\theta)n}}{n|H|},$$

which is the same as

$$\theta(1 - \theta) - \sqrt{\theta(1 - \theta)}\frac{\Phi(H)}{|H|}.$$

This ratio will be at least $\frac{1}{2}\theta(1 - \theta)$ as long as we have

$$\sqrt{\theta(1 - \theta)}\frac{\Phi(H)}{|H|} \leq \frac{1}{2}\theta(1 - \theta),$$

which happens iff

$$\Phi(H) \leq \frac{1}{2}\sqrt{\theta(1 - \theta)}|H|.$$

Thus, if we assume that $\Phi(H)$ meets this criterion, the ratio of cut edges to total edges is at least $\frac{1}{2}\theta(1 - \theta)$. This bound holds for all $\theta \leq \frac{1}{2}$, and is weakest when θ is smallest. Therefore, if $|S| \geq \alpha n$, the ratio of cut edges to total edges is at least $\frac{1}{2}\alpha(1 - \alpha)$, which is at least α (since $\alpha \leq \frac{1}{2}$). Thus, under the hypotheses of the lemma, our graph is an (α, α) expander. □

We also have the following theorem, due to Tom Hayes:

Theorem 3.24 (Hayes, [26, 9])**.** *Let $\epsilon > 0$. Let $t \leq n/2$. For all but a $O(n^{-\epsilon})$ fraction of subsets $H \subseteq G$ such that $|H| = t$,*

$$\Phi(H) < 4\sqrt{(1 + \epsilon)|H|\ln(n)}.$$

Thus, if we randomly pick a set H of size $t = c\ln(n)$, we will have, with probability at least $1 - O(1/\sqrt{n})$, $\Phi(H) < 4\sqrt{\frac{3}{2}c\ln^2(n)} = 4\sqrt{\frac{3}{2c}}c\ln(n)$. Then, if we have $4\sqrt{\frac{3}{2c}} \leq \frac{1}{2\sqrt{2}}\sqrt{\alpha}$, we will satisfy the conditions of Lemma 3.23 with high probability. This happens if $\alpha c \geq 192$. This proves:

Theorem 3.25. *If we form the Cayley sum hypergraph $Cay(K, H)$ for an abelian group K and a random subset H, chosen uniformly from sets of size $c\ln(|K|)$, and $\alpha \geq \frac{192}{c}$, then, with probability at least $1 - O(1/\sqrt{n})$, this hypergraph is an (α, α)-local expander.*

Thus, we can choose c to be $\Theta\left(\frac{1}{\log n}\right)$, and we will get, with high probability, a hypergraph which is an (α, α)-local expander for all $\alpha \geq \Omega\left(\frac{1}{\log n}\right)$. This hypergraph will have edge density $\Theta(\log^2 n)$. This is comparable to the edge density and expansion we achieved for random hypergraphs in Theorem 3.7.

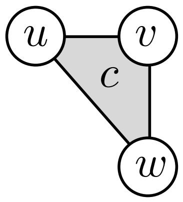
4 The Pair Construction

In this section, we define our main technical tool, the Pair Construction, and give our main technical lemmas, which prove that the Pair Construction is a valid, approximation-preserving reduction when it is applied to a locally expanding 3-CSP. We define the Pair Construction in Section 4.1. In Sections 4.2 and 4.3, we prove the lemmas needed. In Section 4.4, we put some of these lemmas together to show that a hardness result for expanding unique 3-CSP's such as that posited in the Expanding Unique 3-CSP's Conjecture (Conjecture 1.10) would imply the Unique Games Conjecture. In the next section, Section 5, we show the reverse implication, so that the Unique Games Conjecture is equivalent to Conjecture 1.10.

Since (as we discussed in the introduction) the Unique Games Conjecture is equivalent to the same conjecture for E2-Lin-p [29, 37], and because of the specific form of the construction in the next section, we can use E3-Lin-p, which shows that both conjectures are equivalent to the Expanding Linear Equations Conjecture.

4.1 Definitions

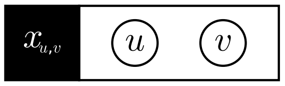
Definition 4.1 (Pair Construction)**.** Let $P = (V, C, \Sigma, wt)$ be a 3-CSP, where V is the set of variables, C is the set of constraints, Σ is the alphabet, and $wt : C \to \mathbb{R}$ is the weight function. From this 3-CSP, we will produce a 2-CSP $P_{\text{pair}} = (V_{pair}, C_{pair}, \Sigma \times \Sigma, wt_p)$, which is called the pair construction of P. Let c be a constraint dealing with variables u, v, w, which we represent pictorially:



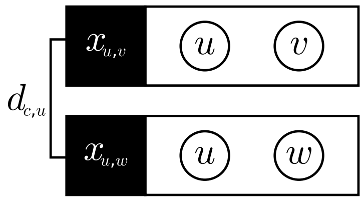
V_{pair}, the set of variables of P_{pair}, will consist of variables $x_{\{u,v\}}$ corresponding to unordered pairs $\{u, v\}$ of variables in V that appear together in some constraint in C.

We interpret the value given to a variable $x_{\{u,v\}}$ as a pair of values, one for u and one for v. Let us fix some linear ordering of the variables, so that every unordered pair $\{u, v\}$ has a canonical order. Say that $u < v$. For an assignment $\sigma : V_{pair} \to \Sigma \times \Sigma$, we will let $\sigma(x_{\{u,v\}})(u)$ denote the first coordinate of $\sigma(x_{\{u,v\}})$, and $\sigma(x_{\{u,v\}})(v)$ the second, so as to emphasize this interpretation.

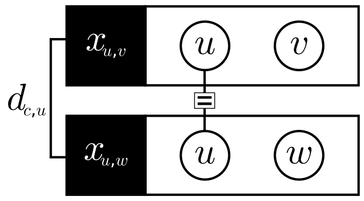
From now on, we will abuse notation and refer to $x_{\{v,u\}}$ even when $u < v$ and $x_{\{u,v\}}$ is the canonical ordering of the unordered pair, with the understanding that $\sigma(x_{\{v,u\}}) = (\sigma(x_{\{u,v\}})(v), \sigma(x_{\{u,v\}})(u))$, i.e., we care about the *interpretation* rather than the *order*. In particular, when we look at $x_{\{u,v\}}$ and let v run over all variables, we will want to get all the pair variables, not just those with $u < v$, and we will want $\sigma(x_{u,v})(u)$ to be the first coordinate of $\sigma(x_{\{u,v\}})$ when $u < v$, and the second coordinate when $u > v$. We will represent $x_{\{u,v\}}$ thusly:



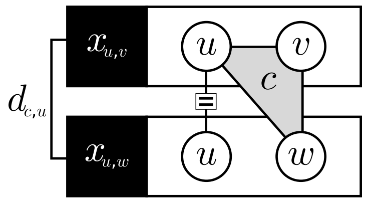
For each constraint $c \in C$, we then make a binary constraint as follows: Let $\sigma : V_{pair} \to \Sigma \times \Sigma$ be an assignment for P_{pair}. If $c \in C$ is incident on the variables $u, v, w \in V$, we make a constraint $d_{c,u}$ between $x_{\{u,v\}}$ and $x_{\{u,w\}}$



that is satisfied if $\sigma(x_{\{u,v\}})(u) = \sigma(x_{\{u,w\}})(u)$



and if $\sigma(x_{\{u,v\}})(u)$,$\sigma(x_{\{u,v\}})(v)$,$\sigma(x_{\{u,w\}})(w)$ satisfy c.



We will call this constraint $d_{c,u}$. We index it by u because the construction depends on which variable we choose to "pivot" around, in this case u. We also make the other two constraints that come from pivoting around v and w, so that each constraint $c \in C$ generates three constraints in C_{pair}.

Finally, we can perform the construction on weighted 3-CSP's. In this case, we will simply give each constraint $d_{c,u} \in C_{pair}$ one-third of the weight of the constraint $c \in C$ that generated it, $wt_p(d_{c,u}) = \frac{1}{3}wt(c)$. Since each constraint $c \in C$ gives rise to three constraints, one for each choice of pivot, the total weight of the constraints remains the same, $wt_p(C_{pair}) = wt(C)$.

This is a useful definition for dealing with unique constraints (see Definition 2.7), because of the following:

Lemma 4.2 (The Pair Construction Preserves Uniqueness)**.** *If a 3-*CSP *P has unique constraints, then P_{pair} has permutation constraints.*

Proof. Suppose we want to satisfy a constraint $d_{c,u}$ between variables $x_{\{u,v\}}$ and $x_{\{u,w\}}$. If we fix $\sigma(x_{\{u,v\}})$, then $\sigma(x_{\{u,w\}})(u)$ has to be equal to $\sigma(x_{\{u,v\}})(u)$, and $\sigma(x_{\{u,w\}})(w)$ has to be the unique value for w satisfying c, given that u gets the value $\sigma(x_{\{u,v\}})(u)$ and v gets the value $\sigma(x_{\{u,v\}})(v)$. $\square$

4.2 The Honest Case

We would like it to be the case that the assignments to V_{pair} are all consistent with a single assignment to V, so that $\sigma(x_{\{u,v\}})(u)$ has the same value for all v. We will call such assignments *honest*, and we will define the honest encoding of an assignment to P:

Definition 4.3 (Honest Encoding)**.** Let $\sigma : V \to \Sigma$ be an assignment to the variables of P. The *honest encoding* of σ is the assignment $\sigma_{hon}(x_{\{u,v\}}) = (\sigma(u), \sigma(v))$.

Since P_{pair}'s constraints encode the constraints of P, honest assignments will do about as well as assignments to P: when σ is a good assignment for P, the honest encoding of σ is a good assignment for P_{pair} (Lemma 4.5), and when there is no good assignment for P, assignments that are close to honest do not do much better in P_{pair} (Lemma 4.8). Unfortunately, there is in general no guarantee that far-from-honest assignments will not do well, so we will be forced to deal with them separately, in the next section.

Definition 4.4 (Popular assignment)**.** Let $\sigma : V_{pair} \to \Sigma \times \Sigma$ be an assignment for P_{pair}. We define the *popular assignment of* σ, denoted σ_p , to be the assignment for P that, for each $u \in V$, takes the most common value of $\sigma(x_{\{u,v\}})(u)$, where v runs over V.

When P is a weighted CSP, we will define $\sigma_p(u)$ to be the value of $\sigma(x_{\{u,v\}})(u)$ that is most popular by weight, where the vote of each v is counted as the sum of the weights of the constraints in which both appear. Thus

$$\sigma_p(u) = \operatorname*{argmax}_{s \in \Sigma} \sum \sum_{c \ni u,v} wt(c),$$

where the first sum runs only over those v such that $\sigma(x_{\{u,v\}})(u) = s$ or $\sigma(x_{\{v,u\}})(u) = s$.

The honest encoding of the popular assignment is in some sense the nearest honest assignment.

Lemma 4.5 (Completeness)**.** *$sat(P_{pair}) \geq sat(P)$*

Proof. Let $\sigma : V \to \Sigma$ be an optimal assignment for P, so that $sat(\sigma, P) = sat(P)$. Then we claim that σ_{hon}, the honest encoding of σ, does just as well in P_{pair}, i.e., $sat(\sigma_{hon}, P_{pair}) = sat(\sigma, P)$. If $d_{c,v}$ is some constraint in C_{pair}, dealing with $x_{\{u,v\}}$ and $x_{\{v,w\}}$, we have $\sigma_{hon}(x_{\{u,v\}})(v) = \sigma_{hon}(x_{\{v,w\}})(v)$ because σ_{hon} is honest. Therefore, $d_{c,v}$ will be satisfied if and only if the values $\sigma_{hon}(x_{\{u,v\}})(u)$,$\sigma_{hon}(x_{\{u,v\}})(v)$, and $\sigma_{hon}(x_{\{v,w\}})(w)$ satisfy c. These values are just $\sigma(u), \sigma(v), \sigma(w)$ by definition, so $d_{c,v}$ is satisfied by σ_{hon} if and only if c is satisfied by σ. Since the weight of each constraint $c \in C$ is distributed evenly among the three constraints $d_{c,u}, d_{c,v}, d_{c,w}$, the weighted fraction of constraints satisfied by σ_{hon} will be the same as the weighted fraction satisfied by σ.

Therefore, this assignment will satisfy a $sat(P)$ fraction of the constraints in P_{pair}, so $sat(P_{pair}) \geq sat(\sigma_{hon}, P_{pair}) = sat(P)$. ☐

Now we would like to prove the soundness part of the reduction, so that $sat(P_{pair})$ cannot be too close to 1 unless $sat(P)$ was also. We will split this analysis into two cases, according to how closely an assignment σ for P_{pair} corresponds to the honest encoding of some assignment for P. To this end, we make several definitions:

Definition 4.6 ($dh(\sigma)$)**.** Let $\sigma : V_{pair} \to \Sigma \times \Sigma$. The *dishonesty of* σ, denoted $dh(\sigma)$, is the fraction of variables in V_{pair} (weighted by degree) that do not agree with the popular assignment in one or both coordinates. If we let Δ be the degree distribution $\Delta(P_{pair})$,

$$dh(\sigma) = \Pr_{x_{\{u,v\}} \sim \Delta} [\sigma(x_{\{u,v\}}) \neq (\sigma_p(u), \sigma_p(v))].$$

Now, let us set an honesty threshold $\zeta \in (0, 1)$.

Definition 4.7 (ζ-Honesty)**.** We will call an assignment to the variables in X ζ-*honest* if $dh(\sigma) \leq \zeta$. We will call an assignment ζ-*dishonest* if it is not ζ-honest. We will denote by $sat_{dh \leq \zeta}(P_{pair})$ the greatest value of $sat(\sigma, P_{pair})$ that can be achieved by a ζ-honest assignment σ, and $sat_{dh > \zeta}(P_{pair})$ the greatest value that can be achieved by a ζ-dishonest assignment.

Lemma 4.8 (Soundness, Honest Case)**.** *$sat(\sigma, P_{pair}) \leq sat(\sigma_p, P) + 2dh(\sigma)$. Therefore, $sat_{dh \leq \zeta}(P_{pair}) \leq sat(P) + 2\zeta$.*

Proof. Let $\sigma : V_{pair} \to \Sigma \times \Sigma$ be a ζ-honest assignment. Then, by definition,

$$sat(\sigma, P_{pair}) = \Pr_{d_{c,v} \sim wt_p(\cdot)}[d_{c,v} \text{ satisfied by } \sigma].$$

Let $u, v, w \in V$ be the variables which c deals with. We can break this probability into two cases, when σ agrees with σ_p on both relevant values, and when it disagrees on at least one of them:

$$\begin{aligned}
&= \Pr_{d_{c,v} \sim wt_p}[d_{c,v} \text{ satisfied by } \sigma \text{ and} \\
&\qquad (\sigma(x_{\{u,v\}}) = (\sigma_p(u), \sigma_p(v)) \wedge \sigma(x_{\{v,w\}}) = (\sigma_p(v), \sigma_p(w)))] \\
&+ \Pr_{d_{c,v} \sim wt_p}[d_{c,v} \text{ satisfied by } \sigma \text{ and} \\
&\qquad (\sigma(x_{\{u,v\}}) \neq (\sigma_p(u), \sigma_p(v)) \vee \sigma(x_{\{v,w\}}) \neq (\sigma_p(v), \sigma_p(w)))] \\
&\leq \Pr_{c \sim wt}[c \text{ satisfied by } \sigma_p] \\
&+ \Pr_{d \sim wt_p}[\sigma(x_{\{u,v\}}) \neq (\sigma_p(u), \sigma_p(v)) \vee \sigma(x_{\{v,w\}}) \neq (\sigma_p(v), \sigma_p(w))]
\end{aligned}$$

since, if σ agrees with σ_p in the relevant locations, c and $d_{c,v}$ are equivalent, and we picked one of the three constraints $d_{c,u}$ with probability $wt(c)/wt(C)$. By the union bound and symmetry, we have

$$\begin{aligned}
&\leq sat(P) + \Pr_{d_{c,v} \sim wt_p}[\sigma(x_{\{u,v\}}) \neq (\sigma_p(u), \sigma_p(v))] \\
&\qquad + \Pr_{d_{c,v} \sim wt_p}[\sigma(x_{\{v,w\}}) \neq (\sigma_p(v), \sigma_p(w))] \\
&= sat(P) + 2 \Pr_{d_{c,v} \sim wt_p}[\sigma(x_{\{u,v\}}) \neq (\sigma_p(u), \sigma_p(v))]
\end{aligned}$$

We are picking $d_{c,v}$ according to the weight distribution wt_p, and then picking one of its two variables $x_{\{u,v\}}$. So, we are picking $x_{\{u,v\}}$ according to the degree distribution $\Delta(P_{pair})$. Therefore, the above

$$= sat(P) + 2dh(\sigma)$$

Since this holds for any ζ-honest σ, $sat_{dh \leq \zeta}(P_{pair}) \leq sat(P) + 2\zeta$. ☐

4.3 The Dishonest Case

The pair construction does not work in general because there can exist good assignments for P_{pair} which are far from honest, and thus do not correspond to any assignment for P. As an example, consider the following system of linear equations over $\mathbb{Z}_2$:

$$\begin{aligned} t+u+v&=0\\ t+u+w&=0\\ t+v+w&=0\\ u+v+w&=0\\ w+x+y&=1\\ w+x+z&=1\\ w+y+z&=1\\ x+y+z&=0 \end{aligned}$$

No assignment to the variables can satisfy more than seven of these eight constraints. We can see this by observing that the subproblem

$$\begin{aligned} t+u+v&=0\\ t+u+w&=0\\ t+v+w&=0\\ u+v+w&=0 \end{aligned}$$

has the unique solution $t=u=v=w=0$, and the subproblem

$$\begin{aligned} w+x+y&=1\\ w+x+z&=1\\ w+y+z&=1\\ x+y+z&=0 \end{aligned}$$

has the unique solution $w=1, x=y=z=0$. We have drawn the underlying hypergraph of this problem in Figure 6.

After the pair construction, the graph we get is disconnected. This is also shown in Figure 6. In particular, all the values associated with w are now associated with only one of the two subproblems, so there is no connection between the different appearances of w. We are thus free to use the assignment $t=u=v=w=0$ on the first subproblem, and $w=1, x=y=z=0$ on the second subproblem, so the resulting 2-CSP has satisfiability 1, even though the original had satisfiability $\frac{7}{8}$.

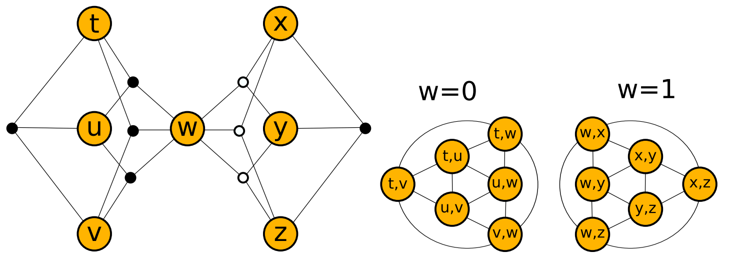


Figure 6: Left: The underlying hypergraph of our CSP. Small black circles represent equations of the form $a+b+c=0$, and small white circles represent equations of the form $a+b+c=1$. Right: After the pair construction, the underlying graph is disconnected, allowing us to use dishonest assignments. As a result, the satisfiability goes up from $\frac{7}{8}$ to 1.

This behavior resulted from the underlying hypergraph of our problem not being well-enough connected. In this section, we show that if the underlying hypergraph of P is a local expander (see section 3) assignments which are far from honest will be penalized.

Lemma 4.9 (Soundness, Dishonest Case)**.** *Let $P=(V,C,\Sigma)$ be a 3-CSP with weight function $wt: C\to\mathbb{R}$, and suppose that the underlying hypergraph of P is a $(\frac{1}{4}\zeta,\beta)$-local expander. Then $sat_{dh>\zeta}(P_{pair})\le 1-\frac{1}{4}\beta\zeta$.*

Proof. Let $\sigma: V_{pair}\to\Sigma\times\Sigma$ be a ζ-dishonest assignment for P_{pair}. Let Δ be the degree distribution on P_{pair}. We define the *disparity of* σ to be

$$Disp = \Pr_{x_{\{u,v\}}\sim\Delta}[\sigma(x_{\{u,v\}})(u)\neq\sigma_p(u)],$$

where we have chosen a random order for $x_{\{u,v\}}$, i.e., checked each of its two coordinates with probability $\frac{1}{2}$. Then

$$\begin{aligned} dh(\sigma) &= \Pr_{x_{\{u,v\}}\sim\Delta}[\sigma(x_{\{u,v\}})\neq(\sigma_p(u),\sigma_p(v))] \\ &\le 2\Pr_{x_{\{u,v\}}\sim\Delta}[\sigma(x_{\{u,v\}})(u)\neq\sigma_p(u)] \\ &= 2Disp. \end{aligned}$$

Therefore, since $dh(\sigma)$ is greater than ζ, $Disp>\frac{1}{2}\zeta$. Next we define the *disparity of* σ *at* u to be

$$\begin{aligned} Disp(u) &= \Pr_{v\sim\Delta_u}[\sigma(x_{\{u,v\}})(u)\neq\sigma_p(u)] \\ &= \Pr_{x_{\{w,v\}}\sim\Delta}[\sigma(x_{\{w,v\}})(w)\neq\sigma_p(w)\mid w=u]. \end{aligned}$$

Then $0\le Disp(u)\le 1$ and $Disp=\mathbb{E}_{u\sim\Delta}Disp(u)$ (the expectation of conditional expectations is the original expectation).

We want to divide the variables into two classes, those with high disparity and those with low disparity. Let t_c be some threshold to be picked later, $0 < t_c < 1$. Let $Cont = \{u | Disp(u) > t_c\}$; we will call these the *controversial variables*. Let $p_c = \Pr_{u \sim \Delta}[u \in Cont]$. Then

$$\begin{aligned} \frac{1}{2}\zeta &\leq \mathbb{E}_u Disp(u) && (1)\\ &= p_c \mathbb{E}_{u\sim\Delta}[Disp(u) | u \in Cont] + (1 - p_c)\mathbb{E}_{u\sim\Delta}[Disp(u) | u \notin Cont] && (2)\\ &\leq p_c 1 + (1 - p_c)t_c && (3) \end{aligned}$$

Thus, $p_c \geq \frac{\frac{1}{2}\zeta - t_c}{1 - t_c} > \frac{1}{2}\zeta - t_c$. So, if we set $t_c = \frac{1}{4}\zeta$, we have $p_c > \frac{1}{4}\zeta$. Therefore, for at least a $\frac{1}{4}\zeta$ fraction of the variables $u \in V$, $Disp(u) \geq \frac{1}{4}\zeta$.

Now, suppose $u \in Cont$. Let V_u be the set of variables which appear in constraints with u. We can partition V_u into sets $V_{u,s}$, one for each $s \in \Sigma$, where $V_{u,s} = \{v \in V_u \mid \sigma(x_{\{u,v\}})(u) = s\}$. In English, $V_{u,s}$ is the set of u's neighbors who want u's value to be s. We know that the largest of these is $V_{u,\sigma_p(u)}$, and that $d_u(V_{u,\sigma_p(u)}) \leq (1 - \frac{1}{4}\zeta d_u(v))$ since $u \in Cont$. We want to divide V_u into two disjoint sets A and B such that $d_u(A) \geq \frac{1}{2}d_u(V_u) \geq d_u(B) \geq \frac{1}{4}\zeta d_u(V_u)$. To do this, we will let B be the union of some of the $V_{u,s}$. There are three cases to consider:

- $d_u(V_{u,\sigma_p(u)}) \geq \frac{1}{2}d_u(V_u)$, in which case we can take $A = V_{u,\sigma_p(u)}$. $Disp(u)$ is the combined weight of all the unpopular assignments, and, since $u \in Cont$, this is at least $\frac{1}{4}\zeta$. Therefore, we can let B be the union of $V_{u,s}$ for $s \neq \sigma_p(u)$.
- $\frac{1}{2}d_u(V_u) > d_u(V_{u,\sigma_p(u)}) \geq \frac{1}{4}\zeta d_u(V_u)$, in which case we can take $B = V_{u,\sigma_p(u)}$,
- $\frac{1}{4}\zeta d_u(V_u) > d_u(V_{u,\sigma_p(u)})$.

In this last case, since $\sigma_p(u)$ is the most popular value for u, we know that $d_u(V_{u,s})$ is smaller than $\frac{1}{4}\zeta d_u(V_u)$ for all $s \in \Sigma$. Therefore, we can build B by starting with $B_0 = V_{u,\sigma_p(u)}$, and iteratively setting $B_{k+1} := B_k \cup V_{u,s}$ for some arbitrary $s \in \Sigma$. If we continued this process for all $s \in \Sigma$, we would have $B = V_u$, so there must be some point at which $d_u(B_k) < \frac{1}{4}\zeta d_u(V_u)$, and $d_u(B_{k+1}) \geq \frac{1}{4}\zeta d_u(V_u)$. But the difference between the two is $d_u(V_{u,s})$ which is smaller than $\frac{1}{4}\zeta d_u(V_u)$, so $d_u(B_{k+1}) < 2 \cdot \frac{1}{4}\zeta < \frac{1}{2}\zeta < \frac{1}{2}$, so $V_u \setminus B_{k+1}$ is still the larger half of V_u. Therefore, we can take $B = B_{k+1}$ in this case.

Now we have that $B \subseteq V_u$ and $\frac{1}{4}\zeta d_u(V_U) \leq d_u(B) \leq \frac{1}{2}d_u(V_u)$, and we know that the underlying hypergraph of P is a $(\frac{1}{4}\zeta, \beta)$-local expander. Thus we have $wt(E(u, B, A)) \geq \beta wt(E(u, V_u, V_u))$ in G.

In P, this means that

$$wt(\{c \in C \mid c \ni u, v, w \text{ for some } v \in B, w \in A\} \geq \beta wt(\{c \in C \mid c \ni u, v, w\}).$$

If $c \ni u, v, w$, $u \in Cont, v \in B, w \in A$, then the constraint $d_{c,u}$ must be violated in P_{pair}, since $x_{\{u,v\}}$ and $x_{\{u,w\}}$ live in different $V_{u,s}$, and thus disagree on the value of u, the pivot variable. Thus

$$wt(\{d_{c,u} \in C_{pair} \mid d_{c,u} \text{ not satisfied by } \sigma\}) \geq \beta wt(\{d_{c,u}\}).$$

Summing over $u \in Cont$,

$$\begin{aligned} wt(\{d_{c,u} \in C_{pair} \mid d_{c,u} \text{ not satisfied by } \sigma\}) &\geq \beta wt(E(Cont, V, V)) \\ &= \beta p_c wt(C_{pair}) \\ &> \beta \left(\frac{1}{4}\zeta\right) wt(C_{pair}). \end{aligned}$$

Thus $sat_{dh>\zeta}(P_{pair}) \leq 1 - \frac{1}{4}\beta\zeta$. □

4.4 The Expanding Unique 3-CSP's Conjecture implies the Unique Games Conjecture

In this section, we put together the lemmas of the previous few sections in order to get a reduction from Unique 3-CSP's to Unique Games that is gap-preserving when it is applied to an expanding instance of Unique 3-CSP's. This will prove Theorem 1.12.

Theorem 4.10 (Theorem 1.12)**.** *Let $\mu, \epsilon \in (0,1)$. There exists some function $c(\mu)$ (with $c(\mu) < 1$ for $\mu \in (0,1)$), and a polynomial time reduction F from unique 3-CSP's to UG such that, if P is a unique 3-CSP:*

- *if P has an assignment satisfying at least a $1-\epsilon$ fraction of constraints, $F(P)$ does also.*
- *if no assignment for P satisfies more than a μ fraction of P's constraints, and the underlying hypergraph of P was a $(\frac{1}{16}(1+\mu), \frac{1}{64}(1+\mu))$-local expander, then no assignment for $F(P)$ can satisfy more than a $c(\mu) < 1$ fraction of $F(P)$'s constraints.*

Proof. The reduction F is the pair construction. The first part of the theorem is given by Lemma 4.5, since $sat(F(P)) \geq sat(P)$.

By Lemmas 4.8 and 4.9, in the case that the underlying hypergraph of P is a $(\frac{1}{4}\zeta, \beta)$-local expander, we have the two inequalities

$$sat_{dh>\zeta}(F(P)) \leq 1 - \frac{1}{4}\beta\zeta$$

$$sat_{dh\leq\zeta}(F(P)) \leq sat(P) + 2\zeta \leq \mu + 2\zeta.$$

$sat(F(P))$ is clearly the maximum of $sat_{dh>\zeta}(F(P))$ and $sat_{dh\leq\zeta}(F(P))$, for any choice of ζ such that the underlying hypergraph of P is a $(\frac{1}{4}\zeta, \beta)$-local expander. If we choose $\zeta = \frac{1-\mu}{4}$, then $sat(F(P))\mu + 2\zeta = \frac{1+\mu}{2}$. If we further know that P was a $(\frac{1}{4}\zeta, \frac{1}{16}\zeta)$ expander for this choice of ζ, then $sat_{dh>\zeta}(F(P)) \leq 1 - \frac{1}{4} \cdot \frac{1}{16}\zeta \cdot \zeta$. Thus

$$sat(F(P)) \leq \max\left\{\frac{1+\mu}{2}, 1 - \frac{1}{1024}(1-\mu)^2\right\}.$$

For $\mu \in (0,1)$, the second term is larger, so we can simplify this to

$$sat(F(P)) \leq 1 - \frac{1}{1024}(1-\mu)^2,$$

given that the underlying hypergraph of P is a $(\frac{1}{16}(1+\mu), \frac{1}{64}(1+\mu))$ local expander. We can thus let $c(\mu) = 1 - \frac{1}{1024}(1-\mu)^2$, and $c(\mu)$ is clearly less than one for $\mu \in (0,1)$. □

In particular, we can show that the Expanding Unique 3-CSP's Conjecture implies the Unique Games Conjecture. We note that, by Theorem 1.5, it is enough to deal with a slightly weaker version of the conjecture, where we only ask that the soundness parameter be constant, not that it go to zero.

Conjecture 4.11 (Unique Games Conjecture with constant soundness). *There exists some constant μ, such that, for all $\epsilon > 0$ and sufficiently large $L = L(\epsilon)$, given an instance of $UG(L)$ it is NP-hard to distinguish between the cases:*

- *There exists an assignment for $v_1, \ldots, v_n$ that satisfies at least a $1-\epsilon$ fraction of the c_i.*
- *No assignment satisfies more than a μ fraction of the c_i.*

Theorem 4.12. *If $\mathsf{Gap}_{\mu}^{1-\epsilon}\text{EXP}_{\alpha,\beta}\text{-U3-CSP}(L)$ is NP-hard for some $\alpha, \beta, \epsilon, \mu$ such that $\mu + 8\alpha < 1$, and $\beta, \epsilon > 0$, then $\mathsf{Gap}_{\alpha'}^{1-\epsilon}UG$ is NP-hard, for some α' depending on μ, α, and β. If, for some fixed μ, $\mathsf{Gap}_{\mu}^{1-\epsilon}\text{EXP}_{\alpha,\beta}\text{-U3-CSP}(L)$ is NP-hard for every $\epsilon > 0$ (with the same condition on α, and for any $\beta > 0$), then the Unique Games Conjecture (Conjecture 1.3) is true.*

Proof. Same as previous theorem. □

5 Reducing Unique Games to Expanding Unique 3-CSP's

Theorem 4.12 of the previous section says that, if we had a hardness result for unique 3-CSP's on locally expanding hypergraphs, we would be able to prove the Unique Games Conjecture, Conjecture 1.3. Therefore, it is worth formulating such a hardness result in the form of a conjecture:

Conjecture 5.1 (Expanding Unique 3-CSP's Conjecture). *For some fixed $\mu < 1$, and for every $\epsilon > 0$, there exist L, α satisfying $\alpha < \frac{1-\mu}{8}$, and $\beta > 0$ such that $\mathsf{Gap}_{\mu}^{1-\epsilon}\text{EXP}_{\alpha,\beta}\text{-U3-CSP}(L)$ is NP-hard.*

If, for some fixed ϵ, this conjecture is true for any L, it will be true for all sufficiently large L.

We can then restate Theorem 4.12 as follows:

Theorem 5.2. *The Expanding Unique 3-CSP's Conjecture (Conjecture 1.10) implies the Unique Games Conjecture (Conjecture 1.3).*

In this section, we prove the reverse implication, that the Unique Games Conjecture implies the Expanding Unique 3-CSP's Conjecture. Thus we have:

Theorem 5.3. *The Expanding Unique 3-*CSP*'s Conjecture (Conjecture 1.10) is equivalent to the Unique Games Conjecture (Conjecture 1.3).*

The converse implication follows from a somewhat simpler reduction, which we call the Reverse Construction. This reduction has the nice property that it maps instances of E2-Lin-p to instances of E3-Lin-p, so that the Expanding Unique 3-CSP's Conjecture is still equivalent to the Unique Games Conjecture in the special case of linear equations modulo p.

Theorem 5.4. *The Expanding Linear Equations Conjecture (Conjecture 1.11) is equivalent to the Unique Games Conjecture (Conjecture 1.3).*

5.1 The Reverse Construction

In this section, we describe the reduction that goes in the opposite direction to our main result. Together with Theorem 1.12, this shows that the Unique Games Conjecture is equivalent to the Expanding Unique 3-CSP's Conjecture. We will call this construction the Reverse Construction.

In [29, 37], it is shown that the Unique Games Conjecture is equivalent to hardness of approximation of the special case of E2-Lin-p. Therefore, we will assume that we are reducing from an instance of E2-Lin-p. The Reverse Construction works for any alphabet of prime size (and it can be made to work slightly less well for alphabets of non-prime size). In fact, there is a straightforward (almost) approximation-preserving reduction from $UG(L)$ to $UG(L+L')$ for any $L' > L$, so it is fairly simple to reduce to the case of an alphabet of prime size without significantly changing the approximation behavior.

Let $P = (V, C, \Sigma)$ be a unique 2-CSP, with $\Sigma = \{0, \dots, p-1\}$, p prime. For each $u \in V$, we will have two variables u_1 and u_2. For each constraint π_{uv} between variables u and v, which is satisfied by an assignment σ iff $\pi_{uv}(\sigma(u)) = \sigma(v)$, we will have $4(p-2)$ constraints. For $i, j \in \{1, 2\}, a \in \mathbb{Z}_p, a \neq 0, 1$, we have the constraint

$$\sigma(v_i) - \sigma(v_{3-i}) = a \cdot [\sigma(v_i) - \pi_{uv}(\sigma(u_j))] \tag{4}$$

or equivalently

$$(a-1)\sigma(v_i) + \sigma(v_{3-i}) - a\pi_{uv}(\sigma(u_j)) = 0$$

When it is written in the second form, we can see that it is a unique constraint, since $(a-1), 1, -a$ are all invertible in $\mathbb{Z}_p$ when $a \neq 0, 1$.

Where do these constraints come from? Intuitively, we want $\sigma(v_i)$ to be the same for $i = 1, 2$, and we want to use this value for $\sigma(v)$ in P. If $v_1 = v_2$, then the left hand side of our first equation is zero, and the right hand side is zero if $\pi_{uv}(\sigma(v_i)) = \sigma(u_j)$, i.e. the constraint is satisfied iff $\sigma(v_i)$ and $\sigma(u_j)$ satisfy the constraint π_{uv}. On the other hand, when $\pi_{uv}(\sigma(v_i)) \neq \sigma(u_j)$,

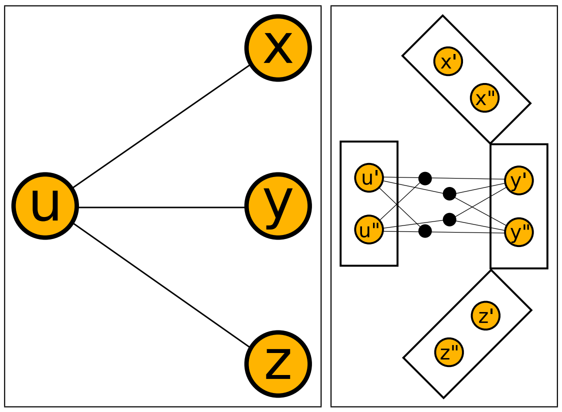


Figure 7: The Reverse Construction. Left: In our UG instance, u is in constraints with x, y, and z. Right: Each variable has become a pair of variables. (The rectangles around each pair are intended to highlight this fact; they do not have any meaning for the variables or constraints of the 3-CSP.) Between each pair, we have four kinds of constraints. We have left out the u-x and u-z edges to make the diagram easier to understand.

$[\sigma(v_i) - \pi_{uv}(\sigma(u_j))]$ is non-zero and thus invertible, so there is a unique value of a that satisfies the equation once we fix values for $\sigma(v_i)$ and $\sigma(v_{3-i})$. Therefore, when $\pi_{uv}(\sigma(v_i)) \neq \sigma(u_j)$, we cannot satisfy more than one of the $p-2$ different constraints that arise from the $p-2$ different permissible values of a.

When we perform this construction on instances of E2-Lin-p, we will be creating instances of E3-Lin-p. When P is an instance of E2-Lin-p, and every constraint c is an equation of the form $bu+cv = d$, $b, c \neq 0$, we have $\pi_{uv}(\sigma(u)) = c^{-1}(d - b\sigma(u))$, so equation 4 becomes

$$\sigma(v_i) - \sigma(v_{3-i}) = a \cdot [\sigma(v_i) - c^{-1}(d - b\sigma(u_j))] \tag{5}$$

or equivalently

$$(a-1)\sigma(v_i) + \sigma(v_{3-i}) + ac^{-1}b\sigma(u_j) = ac^{-1}d$$

which is a non-degenerate linear equation in three variables, since $(a-1), 1, ac^{-1}b$ are all invertible in $\mathbb{Z}_p$ when $a \neq 0, 1$ and $b, c \neq 0$. Since the Unique Games Conjecture is equivalent to the case of E2-Lin-p, our arguments in this section will establish that the Expanding Unique 3-CSP's Conjecture is equivalent to the Expanding Linear Equations Conjecture.

We now have three lemmas to prove in order to establish that the Unique Games Conjecture implies the Expanding Linear Equations Conjecture: we need to know that when P has high satisfiability, P_{rev} does also (Lemma 5.5); that when P has low satisfiability, P_{rev} does also (Lemma 5.6); and that P_{rev} is a local expander (Lemma 5.7).

Lemma 5.5. *Let σ be an assignment for P, and let σ_{rev} be the assignment which assigns every u_i the value $\sigma(u)$. Then $sat(\sigma_{rev}, P_{rev}) \geq sat(\sigma, P)$. Therefore, $sat(P_{rev}) \geq sat(P)$.*

Proof. This is clear: given an assignment σ for P, there is an assignment σ_{rev} for P_{rev} which encodes σ, and thus does at least as well. Specifically, $\sigma_{rev}(v_i) = \sigma(v)$. Each constraint in P_{rev} arises from a constraint in P, and the total weight of constraints arising from a specific constraint π adds up to the weight of π.

Moreover, as stated above, for $\sigma_{rev}(u_i) = \sigma(u)$, we satisfy all the constraints arising from π_{uv} exactly when $\sigma(u)$ and $\sigma(v)$ satisfy π_{uv}. If $\sigma(u)$ and $\sigma(v)$ satisfy π_{uv}, and $\sigma_{rev}(v_1) = \sigma(v_2)$, then both sides of (4) are zero, regardless of the value of a. (When $\sigma(u)$ and $\sigma(v)$ do not satisfy π_{uv}, we will still satisfy some of these constraints, when a takes on a particular value.) □

Now, suppose that $sat(P_{rev}) > c$. Given an assignment σ_{rev} for P_{rev} with high satisfaction, we wish to extract an assignment σ for P which has high satisfaction. It is enough to extract a distribution over possible σ and then argue that it has high (expected) satisfaction, since then some σ must attain at least this value. Our distribution is simple: for each variable u of P, we pick a random $i = \{1, 2\}$ uniformly at random, and let $\sigma(u) = \sigma_{rev}(u_i)$.

Lemma 5.6. *Let σ be an assignment for P_{rev}, and let τ be chosen randomly according to the distribution above. Then $sat(\sigma, P_{rev}) \leq \mathbb{E}[sat(\tau, P)] + \frac{1}{p-2} \leq sat(P) + \frac{1}{p-2}$. Therefore, $sat(P_{rev}) \leq sat(P) + \frac{1}{p-2}$.*

Proof. As usual, when we want to know the fraction of constraints satisfied, we can pick a random constraint and then see what the probability is that it is satisfied. If we simultaneously calculate the probability of acceptance of a random constraint with a random choice of assignment σ, this will give us the expected fraction of constraints satisfied, by linearity of expectation. So, if we let $wt_{rev}()$ be the weight distribution on P_{rev}, and c be the constraint $\sigma(v_i) - \sigma(v_{3-i}) = a \cdot [\sigma(v_i) - \pi_{uv}(\sigma(u_j))]$:

$$\begin{aligned}
\Pr_{c \sim wt_{rev}()}[c \text{ satisfied by } \sigma] = \Pr[(\sigma(v_i) = \pi_{uv}(\sigma(u_j)) \wedge v_i = v_{3-i}) \vee \\
(\sigma(v_i) \neq \pi_{uv}(\sigma(u_j)), a = \frac{\sigma(v_i) - \sigma(v_{3-i})}{\sigma(v_i) - \pi_{uv}(\sigma(u_j))}] \\
\leq \Pr[(\sigma(v_i) = \pi_{uv}(\sigma(u_j)) \wedge v_i = v_{3-i})] + \\
\Pr\left[(\sigma(v_i) \neq \pi_{uv}(\sigma(u_j)), a = \frac{\sigma(v_i) - \sigma(v_{3-i})}{\sigma(v_i) - \pi_{uv}(\sigma(u_j))}\right] \\
\leq \Pr[(\sigma(v_i) = \pi_{uv}(\sigma(u_j)))] + \\
\Pr\left[a = \frac{\sigma(v_i) - \sigma(v_{3-i})}{\sigma(v_i) - \pi_{uv}(\sigma(u_j))} \mid (\sigma(v_i) \neq \pi_{uv}(\sigma(u_j))\right]
\end{aligned}$$

When $\sigma(v_i) \neq \pi_{uv}(\sigma(u_j))$, this formula gives a unique satisfying value for a among the $p-2$ values we are choosing from, except that the solution might be $a = 0$ or $a = 1$.

$$\begin{aligned} &\leq \Pr_{\pi_{uv}\sim wt()}[(\sigma(v_i) = \pi_{uv}(\sigma(u_j))] + \frac{1}{p-2} \\ &= \Pr_{\pi_{uv}\sim wt(),\tau}[\tau(v) = \pi_{uv}(\tau(u))] + \frac{1}{p-2}, \end{aligned}$$

since τ is defined by picking random i and j and using the value of σ there

$$\begin{aligned} &= \Pr_{\pi_{uv}\sim wt(),\tau}[\pi \text{ satisfied by } \tau] + \frac{1}{p-2} \\ &= sat(\tau, P) + \frac{1}{p-2} \end{aligned}$$

□

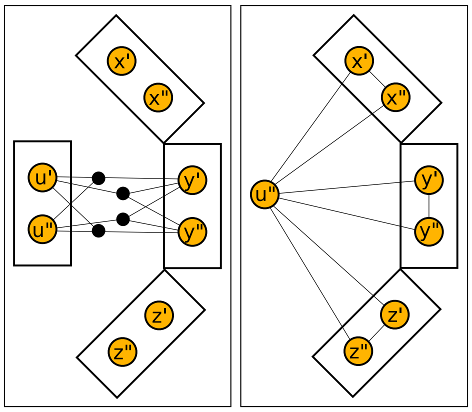


Figure 8: Left: The reverse construction. Right: the neighborhood graph at u'. We prove in Lemma 5.7 that this hypergraph is a local expander, because every vertex in the neighborhood graph is connected to u''.

Lemma 5.7. *P_{rev} is an $(\alpha, \frac{1}{2}\alpha)$-local expander for all $\alpha > 0$.*

Proof. Fix $u \in V$, and let $v^1, \ldots, v^d$ be its neighbors in P, so that d is the degree of u.

Consider the neighborhood graph G_{u_1}, shown in Figure 8. Each edge $\{u, v^j\}$ gives rise to four edges (we can ignore the choice of a, since we are just looking at the underlying hypergraph), three of which are incident on u_1: $\{u_1, u_2, v_1^j\}$, $\{u_1, u_2, v_2^j\}$, and $\{u_1, v_1^j, v_2^j\}$. Therefore, G_{u_1} has $3d$ edges, consisting of d triangles of the form u_2, v_1^j, v_2^j, as shown in Figure 8. We then have that $d_{u_1}(u_2) = 2d$, and $d_{u_1}(v_1^j) = d_{u_1}(v_2^j) = 2$. Let $S \subseteq V_{u_1}$ be a potential cut, with $d_{u_1}(S) \leq \frac{1}{2}d_{u_1}$. There are two cases to consider: either $u_2 \in S$, or $u_2 \notin S$.

If $u_2 \in S$, then since $d_{u_1}(u_2) = 2d$, and $d_{u_1}(S) \leq \frac{1}{2}6d = 3d$, S must include at most $\frac{d}{2}$ of the $2d$ v_i^j. Therefore, at least $2d - \frac{1}{2}d = \frac{3}{2}d$ of the v_i^j are not in S, and every edge between them and u_2 is cut. Therefore, we cut at least $\frac{3}{2}d$ edges, which is $\frac{1}{4}6d = \frac{1}{4}d_{u_1}(V_{u_1})$. Therefore, in the case $u_2 \in S$, we cut at least a fourth of the edges in E_{u_1}, regardless of the size of S. Let $\alpha = d_{u_1}(S)/d_{u_1}(V_{u_1})$. Since $d_{u_1}(S) \leq \frac{1}{2}d_{u_1}(V_{u_1})$, $\alpha \leq \frac{1}{2}$. Therefore, $wt(E(u_1, S, V_{u_1} \setminus S)) \geq \frac{1}{4}wt(E_{u_1}) \geq \frac{1}{2}\alpha wt(E_{u_1})$.

If $u_2 \notin S$, then all elements of S are v_i^j, which have $d_{u_1}(\cdot) = 2$, so we have exactly $d_{u_1}(S)/2$ of them in S. Now, we will cut all of the edges between these v_i^j and u_2, so we will cut at least $\frac{1}{2}d_{u_1}(S)$ edges. Therefore, if $d_{u_1}(S) \geq \alpha d_{u_1}(V_{u_1}) = \alpha \cdot 6d$, we will cut at least $\frac{1}{2}d_{u_1}(S) \geq \alpha \cdot 3d = \alpha|E_u|$ edges.

Therefore, the underlying hypergraph of P_{rev} is an $(\alpha, \frac{1}{2}\alpha)$-local expander for all $\alpha > 0$. $\square$

Proof of Theorem 1.13. We have already established that the Expanding Linear Equations Conjecture implies the Unique Games Conjecture, in Section 4.4. Therefore, all that remains is the reverse implication.

Suppose that the Unique Games Conjecture is true. Therefore, for all $\delta, \epsilon > 0$, there is some prime p such that it is NP-hard to tell the difference between instances of E2-Lin-p with satisfiability greater than $1 - \epsilon$ and instances with satisfiability less than δ. If we take an instance of E2-Lin-p that falls into one of these two cases and perform the Reverse Construction on it, we get an instance of E3-Lin-p which has satisfiability which is either greater than $1 - \epsilon$ (by Lemma 5.5), or less than $\delta + \frac{1}{p-2}$ (by Lemma 5.6). We can make $\delta + \frac{1}{p-2}$ arbitrarily small by picking δ small enough and p large enough, so this will be an instance of $\mathsf{Gap}_{\delta'}^{1-\epsilon}$E3-Lin-$p$ for some δ'. Moreover, our instance will be an $(\alpha, \frac{1}{2}\alpha)$-local expander (by Lemma 5.7) so we will actually have an instance of $\mathsf{Gap}_{\delta'}^{1-\epsilon}\text{Exp}_{\alpha,\beta}$-E3-Lin-$p$ satisfying the conditions of the Expanding Linear Equations Conjecture.

The Reverse Construction is clearly a polynomial-time reduction, so if it was NP-hard to classify our original E2-Lin-p instance, it must be NP-hard to classify this instance. Therefore we have the desired hardness result. $\square$

6 Approximating unique 3-CSP's

In this section, we prove Theorem 1.14, and give an algorithm which performs well on unique 3-CSP's when the underlying hypergraph is an expander, as long as the alphabet size is not too large.

Charikar, Makarychev, and Makarychev give an algorithm in [11] that does well on instances of UG that have high satisfiability. Their algorithm is based on rounding a semidefinite program.

Theorem 6.1 (Charikar, Makarychev, and Makarychev, [11])**.** *Let P be an instance of $UG(L)$. There is a polynomial time algorithm that finds an assignment of variables which satisfies a $1 - O(\sqrt{\epsilon \log L})$ fraction of P's constraints if $sat(P) = (1 - \epsilon)$.*

Using the pair construction, we can reduce unique 3-CSP's to UG:

Algorithm 6.2. Input: *A U3*CSP *P with an alphabet of size L such that $sat(P) > 1 - \epsilon$*
Output: *An assignment for P*

1. *Let P_{pair} be the 2-*CSP *resulting from applying the pair construction to P.*

2. *Apply the CMM algorithm to P_{pair} to get a solution σ for P_{pair}.*

3. *Output σ_p, the popular assignment for σ.*

Proof of Theorem 1.14. First, note that since P_{pair} has alphabet $\Sigma \times \Sigma$ when P has alphabet Σ, the alphabet size after the pair construction will be L^2. By Lemma 4.5, we know that when P has satisfiability greater than $1 - \epsilon$, P_{pair} does also. Therefore, the algorithm of [11] will output an assignment σ for P_{pair} satisfying at least a $1 - C_{CMM}\sqrt{\epsilon \log(L^2)}$ fraction of constraints, where C_{CMM} is the constant implicit in Theorem 6.1. We can rewrite this as $1 - C_{CMM}\sqrt{2\epsilon \log L} = 1 - C'_{CMM}\sqrt{\epsilon \log L}$ for some other constant C'_{CMM}.

By Lemma 4.8, we know that $sat(\sigma_p, P) \geq sat(\sigma, P_{pair}) - 2dh(\sigma)$. This gives us the desired result, as long as we have a bound on $dh(\sigma)$.

Putting together the guarantee of the CCM algorithm and Lemma 4.9, we have

$$1 - C'_{CMM}\sqrt{\epsilon \log L} \leq sat(\sigma, P_{pair}) \leq 1 - \frac{1}{4}\beta \cdot dh(\sigma)$$

whenever P is a $(\frac{1}{4}dh(\sigma), \beta)$-local expander. In particular, if P is an $(\alpha, \frac{1}{4}\alpha)$-local expander for $\alpha > \alpha_{min}$, this gives us

$$\frac{1}{4} \cdot \frac{1}{16} dh(\sigma) \cdot dh(\sigma) \leq C'_{CMM}\sqrt{\epsilon \log L}$$

for $\frac{1}{4}dh(\sigma) > \alpha_{min}$. Solving for $dh(\sigma)$, we get

$$dh(\sigma) \leq \sqrt{64 C'_{CMM}}\sqrt[4]{\epsilon \log L}$$

if $dh(\sigma) > 4\alpha_{min}$, or

$$dh(\sigma) \leq \max\{C\sqrt[4]{\epsilon \log L}, 4\alpha_{min}\}.$$

This gives us the guarantee

$$sat(\sigma, P) \geq 1 - C'_{CMM}\sqrt{\epsilon \log(L)} - 2\max\{C\sqrt[4]{\epsilon \log(L)}, 4\alpha_{min}\}.$$

If $\epsilon \log(L) < 1$, then the $\sqrt[4]{\epsilon \log(L)}$ term dominates the $C'_{CMM}\sqrt{\epsilon \log(L)}$ term. If $\epsilon \log(L)$ is unbounded, then these bounds are meaningless. If $\epsilon \log(L)$ is bounded and greater than 1 then both of these terms are essentially constant. So, in all cases where these bounds make sense, the $\sqrt[4]{\epsilon \log(L)}$ term dominates, and we can eliminate the other term by increasing the constant C to some other constant C'.

$$\begin{aligned} sat(\sigma, P) &\geq 1 - 2\max\{C' \sqrt[4]{\epsilon \log(L)}, 4\alpha_{min}\} \\ &= 1 - \max\{C'' \sqrt[4]{\epsilon \log(L)}, 8\alpha_{min}\}. \end{aligned}$$

which is our desired bound. □

7 Acknowledgments

I would like to thank the following people:

- Laci Babai, for much sage advice, and for suffering through a number of much rougher drafts
- Julia Chuzhoy and Prahladh Harsha, for being on my committee, and for numerous helpful suggestions about this paper
- Sourav Chakraborty, Bill Fefferman, and Prahladh Harsha, for helpful discussions
- Andrew Cone and Max Shron, for helpful advice on the design of my figures
- Kelly Kennedy
- My parents

References

[1] *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008.* ACM, 2008.

[2] N. Alon, K. Makarychev, Y. Makarychev, and A. Naor, "Quadratic forms on graphs," in *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing.* New York, NY, USA: ACM, 2005, pp. 486–493.

[3] S. Arora and B. Barak, "Computational complexity: A modern approach." [Online]. Available: http://www.cs.princeton.edu/theory/complexity/

[4] S. Arora, S. A. Khot, A. Kolla, D. Steurer, M. Tulsiani, and N. K. Vishnoi, "Unique games on expanding constraint graphs are easy," in *STOC.* ACM, 2008.

[5] S. Arora, J. R. Lee, and A. Naor, "Euclidean distortion and the sparsest cut," in *STOC*, H. N. Gabow and R. Fagin, Eds. ACM, 2005, pp. 553–562.

[6] Y. Aumann and Y. Rabani, "An $O(\log k)$ approximate min-cut max-flow theorem and approximation algorithm," *SIAM J. Comput.*, vol. 27, no. 1, pp. 291–301, 1998.

[7] P. Austrin, "Balanced max 2-sat might not be the hardest," in *STOC*, D. S. Johnson and U. Feige, Eds. ACM, 2007, pp. 189–197.

[8] ——, "Towards sharp inapproximability for any 2-CSP," in *FOCS*. IEEE Computer Society, 2007, pp. 307–317.

[9] L. Babai, "The Fourier transform and equations over finite abelian groups: An introduction to the method of trigonometric sums." [Online]. Available: http://people.cs.uchicago.edu/ laci/reu02/fourier.pdf

[10] E. Ben-Sasson, P. Harsha, O. Lachish, and A. Matsliah, "Sound 3-query PCPPs are long," ECCC, Tech. Rep. 127, 2007.

[11] M. Charikar, K. Makarychev, and Y. Makarychev, "Near-optimal algorithms for unique games," in *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM, 2006, pp. 205–214.

[12] S. Chawla, R. Krauthgamer, R. Kumar, Y. Rabani, and D. Sivakumar, "On the hardness of approximating multicut and sparsest-cut," in *IEEE Conference on Computational Complexity*. IEEE Computer Society, 2005, pp. 144–153.

[13] J. Chuzhoy and S. Khanna, "Polynomial flow-cut gaps and hardness of directed cut problems," July 2007, manuscript. [Online]. Available: http://ttic.uchicago.edu/ cjulia/papers/dir-multicut2-full.pdf

[14] ——, "Polynomial flow-cut gaps and hardness of directed cut problems," in *STOC*, D. S. Johnson and U. Feige, Eds. ACM, 2007, pp. 179–188.

[15] E. Dahlhaus, D. S. Johnson, C. H. Papadimitriou, P. D. Seymour, and M. Yannakakis, "The complexity of multiterminal cuts," *SIAM J. Comput.*, vol. 23, no. 4, pp. 864–894, 1994.

[16] I. Dinur, V. Guruswami, S. Khot, and O. Regev, "A new multilayered PCP and the hardness of hypergraph vertex cover," in *STOC*. ACM, 2003, pp. 595–601.

[17] U. Feige, "Relations between average case complexity and approximation complexity," in *Proc. of STOC 2002, Montreal.*, 2002. [Online]. Available: citeseer.ist.psu.edu/feige02relations.html

[18] U. Feige and D. Reichman, "On systems of linear equations with two variables per equation," in *APPROX-RANDOM*, 2004, pp. 117–127.

[19] J. Friedman, “Some graphs with small second eigenvalue,” *Combinatorica*, vol. 15, no. 1, pp. 31–42, 1995.

[20] J. Friedman and A. Wigderson, “On the second eigenvalue of hypergraphs,” *Combinatorica*, vol. 15, no. 1, pp. 43–65, 1995.

[21] N. Garg, V. V. Vazirani, and M. Yannakakis, “Approximate max-flow min-(multi)cut theorems and their applications,” *SIAM J. Comput.*, vol. 25, no. 2, pp. 235–251, 1996.

[22] M. X. Goemans and D. P. Williamson, “Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming,” *J. ACM*, vol. 42, no. 6, pp. 1115–1145, 1995.

[23] W. T. Gowers, “Quasirandom groups,” *Combinatorics, Probability & Computing*, vol. 17, no. 3, pp. 363–387, May 2008.

[24] J. Håstad, “On linear equations and satisfiability.” [Online]. Available: http://www.cs.ioc.ee/yik/schools/win2003/jhshort.ps

[25] ——, “Some optimal inapproximability results,” *J. ACM*, vol. 48, no. 4, pp. 798–859, 2001.

[26] T. P. Hayes, “A large-deviation inequality for vector-valued martingales,” February 2003, manuscript. [Online]. Available: http://people.cs.uchicago.edu/∼hayest/papers/VectorAzuma

[27] D. S. Johnson and U. Feige, Eds., *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007.* ACM, 2007.

[28] S. Khot, “On the power of unique 2-prover 1-round games,” in *IEEE Conference on Computational Complexity*, 2002, p. 25.

[29] S. Khot, G. Kindler, E. Mossel, and R. O’Donnell, “Optimal inapproximability results for max-cut and other 2-variable CSPs?” in *FOCS.* IEEE Computer Society, 2004, pp. 146–154.

[30] S. Khot and A. Naor, “Linear equations modulo 2 and the L1 diameter of convex bodies,” in *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science (Focs’07)*, 2007, pp. 318–328.

[31] S. Khot and O. Regev, “Vertex cover might be hard to approximate to within $2 - \varepsilon$,” in *IEEE Conference on Computational Complexity.* IEEE Computer Society, 2003, pp. 379–.

[32] R. E. Ladner, “On the structure of polynomial time reducibility,” *J. ACM*, vol. 22, no. 1, pp. 155–171, 1975.

[33] F. T. Leighton and S. Rao, "An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms," in *FOCS*. IEEE, 1988, pp. 422–431.

[34] M. Lewin, D. Livnat, and U. Zwick, "Improved rounding techniques for the MAX 2-SAT and MAX DI-CUT problems," in *IPCO*, ser. Lecture Notes in Computer Science, W. Cook and A. S. Schulz, Eds., vol. 2337. Springer, 2002, pp. 67–82.

[35] N. Linial, E. London, and Y. Rabinovich, "The geometry of graphs and some of its algorithmic applications," *Combinatorica*, vol. 15, no. 2, pp. 215–245, 1995.

[36] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. New York, NY, USA: Cambridge University Press, 2005.

[37] E. Mossel, R. O'Donnell, and K. Oleszkiewicz, "Noise stability of functions with low influences: invariance and optimality," in *FOCS*. IEEE Computer Society, 2005, pp. 21–30.

[38] P. Raghavendra, "Optimal algorithms and inapproximability results for every CSP?" in *STOC*. ACM, 2008.

[39] A. Rao, "Parallel repetition in projection games and a concentration bound," in *STOC*. ACM, 2008.

[40] ——, "Parallel repetition in projection games and a concentration bound," ECCC, Tech. Rep. 13, February 2008, manuscript. [Online]. Available: http://eccc.hpi-web.de/eccc-reports/2008/TR08-013/index.html

[41] L. Trevisan, G. B. Sorkin, M. Sudan, and D. P. Williamson, "Gadgets, approximation, and linear programming," *SIAM J. Comput.*, vol. 29, no. 6, pp. 2074–2097, 2000.