Inapproximability Results for Set Splitting and Satisfiability Problems with no Mixed Clauses*

Venkatesan Guruswami

University of California at Berkeley
Computer Science Division
Berkeley, CA 94720.
Email: venkat@lcs.mit.edu

Abstract

We prove hardness results for approximating set splitting problems and also instances of satisfiability problems which have no "mixed" clauses, i.e., every clause has either all its literals unnegated or all of them negated. Results of Håstad [5] imply tight hardness results for set splitting when all sets have size exactly $k \geq 4$ elements and also for non-mixed satisfiability problems with exactly k literals in each clause for $k \geq 4$. We consider the case $k = 3$. For the Max E3-Set Splitting problem in which all sets have size exactly 3, we prove an NP-hardness result for approximating within any factor better than 19/20. This result holds even for satisfiable instances of Max E3-Set Splitting, and is based on a PCP construction due to Håstad [5]. For "non-mixed Max 3Sat", we give a PCP construction which is a slight variant of the one given by Håstad for Max 3Sat, and use it to prove the NP-hardness of approximating within a factor better than 11/12.

Keywords: Hardness of approximations, Set splitting, PCP, Gadgets.

*A preliminary version of this paper [4] appears in the Proceedings of the 3rd International Workshop on Approximations Algorithms for Combinatorial Optimization Problems (APPROX), September 2000.

1 Introduction

We study the approximability of set splitting problems and satisfiability problems whose clauses are restricted to have either all literals unnegated or all of them negated. The latter seems to be a natural variant of the fundamental satisfiability problem.

1.1 Set Splitting Problems

We first discuss the set splitting problems we consider and the prior work on them. In the general MAX SET SPLITTING problem, we are given a universe U and a family $\mathcal{F}$ of subsets of U, and the goal is to find a partition of U into two disjoint (not necessarily equal sized) sets as $U = U_1 \cup U_2$ that maximizes the number of subsets in $\mathcal{F}$ that are *split* (where a set $S \subseteq U$ is said to be split by the partition $U = U_1 \cup U_2$ if $S \cap U_1 \neq \emptyset$ and $S \cap U_2 \neq \emptyset$). The version when all subsets in the family $\mathcal{F}$ are of size *exactly* k is referred to as MAX Ek-SET SPLITTING. For any fixed $k \geq 2$, MAX Ek-SET SPLITTING was shown to be NP-hard by Lovász [10]. Obviously, MAX E2-SET SPLITTING is exactly the extensively studied MAX CUT problem. MAX CUT is known to be NP-hard to approximate within $16/17 + \varepsilon$, for any $\varepsilon > 0$ [5, 14], and Goemans and Williamson, in a major breakthrough, used semidefinite programming to give a factor 0.878-approximation algorithm for MAX CUT [3].[1] Here we investigate the approximability of MAX Ek-SET SPLITTING for $k \geq 3$.

The MAX SET SPLITTING problem is related to the constraint satisfaction problem MAX NAE SAT, which is a variant of MAX SAT, but where the goal is to maximize the total weight of the clauses that contain both true and false literals. MAX SET SPLITTING is simply a special case of MAX NAE SAT where all literals appear unnegated (i.e., MAX SET SPLITTING is the same problem as monotone MAX NAE SAT). Similarly MAX Ek-SET SPLITTING is just the monotone version of MAX NAE-Ek-SAT, and in fact is the same problem as MAX 2-COLORABLE HYPERGRAPH on k-uniform hypergraphs (i.e., given a k-uniform hypergraph, find a 2-coloring of its vertices that maximizes the number of hyperedges which are not monochromatic).

Prior Work. We discuss below the status of the MAX Ek-SET SPLITTING and MAX NAE-Ek-SAT problems for $k \geq 3$. These problems are all NP-hard and MAX SNP-hard. Moreover, it was shown that for each $k \geq 3$, there is a constant $\varepsilon_k > 0$, such that is is NP-hard to distinguish between MAX Ek-SET SPLITTING instances where *all* sets can be split by some partition (which we call *satisfiable* instances in the sequel), and those where no partition splits more than a fraction $(1 - \varepsilon_k)$ of the sets [11].

Following the striking inapproximability results of Håstad [5], it has become possible to prove reasonable explicit bounds on the inapproximability ratios of MAX Ek-SET SPLITTING and MAX NAE-Ek-SAT by construction of appropriate *gadgets* (see Section 2.1 for a discussion about gadgets). In particular, the result for $k = 2$ (i.e., MAX CUT) mentioned above follows this approach, and so does the $11/12+\varepsilon$ hardness result for MAX NAE-E2-SAT [5]. In the same paper [5], Håstad proved a tight inapproximability bound of $1-2^{-k}+\varepsilon$, for an arbitrary constant $\varepsilon > 0$, for satisfiable instances of MAX k-SAT, for $k \geq 3$. It follows by a simple reduction that even satisfiable instances of MAX NAE-Ek-SAT, for $k \geq 4$, are hard to approximate within $1 - 2^{-k+1} + \varepsilon$, for an arbitrary constant $\varepsilon > 0$. Note that this result is *tight*, since a random truth assignment will "satisfy" a fraction $1 - 2^{-k+1}$ of the clauses of a MAX NAE-Ek-SAT instance. This leaves out only the case

[1]Throughout this paper we deal only with maximization problems and by an α factor approximation we mean a solution whose value is at least α times that of an optimum solution. Consequently, all factors of approximation we discuss will be less than 1.

$k = 3$, and this turns out to be an intriguing case where a tight result is not yet in sight. We now review the results that are known for $k = 3$.

For Max NAE-E3-Sat, a hardness of approximation within $15/16 + \varepsilon$, even for satisfiable instances, follows from Håstad's inapproximability result for Max 3-Sat and an easy 2-gadget from Max 3-Sat to Max NAE-E3-Sat [11, 15]. On the algorithmic side, the best known approximation algorithm, due to Zwick [16], achieves a ratio of 0.908. (This bound is as yet only based on numerical evidence, the best proven bound is 0.87868 [8, 15], which is slightly better than the Goemans and Williamson approximation guarantee for Max Cut [3].) For satisfiable instances of Max NAE-E3-Sat, an approximation ratio of 0.91226 can be achieved in polynomial time [15].

Turning again to set splitting problems, Håstad [5] has shown the tight result that it is NP-hard to approximate (even satisfiable instances of) Max E4-Set Splitting within $7/8 + \varepsilon$, for any $\varepsilon > 0$. For Max E3-Set Splitting, by the results of Zwick [15, 16] mentioned above, there exist approximation algorithms achieving a ratio of 0.908 (resp. 0.912) for general (resp. satisfiable) instances. As regards hardness results for Max E3-Set Splitting, no explicit bound in the literature appears to be correct. Inapproximability within a factor of (approximately) 0.987 is claimed in [7], and it is based among other things on a 9-gadget reducing a PC_0 constraint to 3-Set Splitting constraints (a PC_0 constraint is of the form $x \oplus y \oplus z = 0$ where x, y, z are *unnegated* variables). This claim suffers from the "well-known" flaw in early gadget results that use PC_0 gadgets without giving explicit PC_1 gadgets (a PC_1 constraint checks if $x \oplus y \oplus z = 1$) to conclude hardness results for approximating monotone constraint satisfaction problems (like Max Cut, Max E3-Set Splitting, etc). The problem is that when the target problem is monotone, one cannot "convert" a PC_1 constraint to a PC_0 constraint by simply negating a variable, and one has to pay an explicit cost in the gadget for negating a variable (eg., to "simulate" $\bar{x}$, one has to introduce an auxiliary variable, say t, that will play the role of $\bar{x}$, and then one has to introduce suitable constraints in the gadget at additional cost to enforce that $t = \bar{x}$). This error occurs in early versions of [1] and in [7]. For the case of Max Cut, the error can be (and was) fixed in [1] who construct a PC_1 gadget from a PC_0 gadget by negating a variable at a unit extra cost. One can similarly fix the error for Max E3-Set Splitting by incurring an extra cost of 4 for the PC_1 gadget. These gadgets together with Håstad's optimal inapproximability result for 3-parity constraints will give an inapproximability bound of $21/22 + \varepsilon$ for Max E3-Set Splitting. In this paper, by reducing from a different constraint satisfaction problem, we will improve this bound further to $19/20+\varepsilon$. Moreover, the hardness result also applies to satisfiable instances; more details appear in Section 1.3.

1.2 Satisfiability with no Mixed Clauses

As mentioned earlier, the set splitting problem is just a special case of Nae-Sat where one adds the restriction that in every clause all literals appear unnegated (or, equivalently, all appear negated), i.e., no clause is "mixed". This leads us to consider the corresponding "non-mixed" version of the even more fundamental problem of satisfiability where one adds the restriction that none of the clauses in the instance are "mixed". We refer to the version of Max Sat where all clauses have at most k literals and none of the clauses have both negated and unnegated literals as Max k-NM-Sat (here NM-Sat stands for non-mixed satisfiability). The version where all the clauses have exactly k literals will be referred to as Max Ek-NM-Sat. This problem appears to be a fairly natural variant of Sat, and does not appear to have been explicitly considered in the literature.

Known results on approximating Max k-NM-Sat: Clearly, any algorithm that approximates Max k-Sat within factor α_k also approximates Max k-NM-Sat within the same factor; in particular approximation factors of $\alpha_2 = 0.931$ and $\alpha_3 = 7/8$ can be achieved in this way [2, 8]. For MAX Ek-NM-SAT, an approximation factor of $1 - 2^{-k}$ can be achieved trivially for all $k \geq 3$, by simply picking a random truth assignment. There are no algorithms known which perform any better on non-mixed clauses than on general satisfiability instances. For $k \geq 4$, a recent result of Håstad [5] shows that MAX Ek-SET SPLITTING is hard to approximate within a factor of $1 - 2^{-k+1} + \varepsilon$ for any $\varepsilon > 0$. [2] Since there is a trivial 2-gadget reducing MAX Ek-SET SPLITTING to MAX Ek-NM-SAT (namely, replace the constraint split$(x_1, x_2, \ldots, x_k)$ by the two clauses $(x_1 \vee x_2 \vee \cdots \vee x_k)$ and $(\bar{x}_1 \vee \bar{x}_2 \vee \cdots \vee \bar{x}_k)$), this implies that MAX Ek-NM-SAT (and hence also MAX k-NM-SAT) is NP-hard to approximate within a factor better than $(1 - 2^{-k})$. Hence, for $k \geq 4$, the naive algorithms that work for the more general MAX Ek-SAT are really the best possible for MAX Ek-NM-SAT as well. As in the case of set splitting problems, our focus, therefore, is on the case $k = 3$.

1.3 Our Main Results

For MAX E3-SET SPLITTING, we prove that for every $\varepsilon > 0$, it is NP-hard to find a partition that splits more than a fraction $(19/20 + \varepsilon)$ of the 3-sets of a *satisfiable* MAX E3-SET SPLITTING instance. This result is proved by demonstrating a simple "5-gadget" reducing the predicate $\mathsf{SNE}_4(x, y, z, w) = (x \neq y) \vee (z \neq w)$ to 3-set splitting constraints. The gadget is in fact optimal, and hence this is the best hardness one can obtain for MAX E3-SET SPLITTING by a reduction from SNE_4. We also consider the MAX 2-2-SET SPLITTING problem, which is a variant of MAX E4-SET SPLITTING where the goal is to find a partition that maximizes the number of 4-sets that are split evenly (i.e., have a 2-2-split), and prove an inapproximability ratio of $11/12 + \varepsilon$ for satisfiable instances, and $8/9 + \varepsilon$ for near-satisfiable instances.

For MAX 3-NM-SAT, one can prove an inapproximability result by starting with a hard to approximate instance of MAX 3-SAT, and use a "2-gadget" to replace each *mixed* clause with clauses that only have either all negated or all unnegated literals (for example, replace a clause $(a \vee b \vee \bar{c})$ with $(a \vee b \vee t)$ and $(\bar{t} \vee \bar{c})$ and a clause $(a \vee \bar{b} \vee \bar{c})$ with $(a \vee t)$ and $(\bar{t} \vee \bar{b} \vee \bar{c})$). Since MAX 3-SAT is NP-hard to approximate within $(7/8 + \varepsilon)$ for arbitrary constant $\varepsilon > 0$ [5], this gives the hardness of approximating MAX 3-NM-SAT within a factor better than 15/16.

We improve this bound by giving a new PCP construction which is essentially the same as the one Håstad's construction for MAX E3-SAT [5] — we simply observe that one of the two proof tables the verifier reads in Håstad's construction need not be *folded* (folding is a technical requirement in PCP constructions which will be elaborated later in the paper). This modified PCP construction enables us to prove a hardness of $11/12 + \varepsilon$ for MAX 3-NM-SAT and a hardness of $15/16 + \varepsilon$ for MAX E3-NM-SAT. Note that no polynomial time algorithm with better performance ratio than 7/8 is known for either of these problems, and progress in closing this gap remains an open problem.

2 Hardness of Approximating MAX E3-SET SPLITTING

2.1 Gadgets: A brief discussion

The hardness results of this section are proven by giving appropriate "gadgets" reducing constraint satisfaction problems already known to be hard to approximate, to MAX E3-SET SPLITTING. We use the definitions of gadgets following [1, 9, 14].

Definition 1 *An* α-gadget *reducing a Boolean function f on variables $x_1, x_2, \ldots, x_k$ to a constraint family $\mathcal{F}$ is a finite collection of (rational) weights w_j and constraints C_j from $\mathcal{F}$ over $x_1, x_2, \ldots, x_k$*

[2] Actually this result is shown only for $k = 4$ in [5]; we extend it for larger k in Section 3.1.

and auxiliary variables $y_1, y_2, \ldots, y_p$ such that for each assignment $\vec{a} = a_1, a_2, \ldots, a_k$ to the x_i's that satisfies f, there is an assignment $\vec{b}$ to the y_i's such that a total weight α of the constraints C_j are satisfied by $(\vec{a}, \vec{b})$, and if $\vec{a}$ does not satisfy f, then for every assignment $\vec{b}$ to the y_i's, the weight of the constraints C_j satisfied by $(\vec{a}, \vec{b})$ is at most $\alpha - 1$. When the total weight $\sum_j w_j$ of the constraints in the gadget equals α, we call the gadget a sound *α-gadget.*

The quantity α is a measure of the quality of the reduction, a smaller value of α implies a better approximation preserving reduction.

A gadget reducing a constraint satisfaction problem A to another problem B is simply a gadget that reduces each function (constraint) allowed under A to a collection of constraints allowed under B. The utility of gadgets comes from the following (by now, folklore) lemmas which state how a gadget reducing A to B enables one to translate an inapproximability result for A into one for B.

Lemma 1 *Suppose that it is NP-hard to distinguish satisfiable instances of some problem A from instances where at most a fraction c of the constraints are satisfiable. Suppose further that there exists a* sound *α-gadget reducing problem A to B. Then, it is NP-hard to distinguish between satisfiable instances of B from instances where at most a fraction $c + (1-c)(1-1/\alpha)$ of its constraints are satisfiable.*

The proof of the above lemma is straightforward. Since we will be using it later, we also state the following result concerning gadgets reducing from parity constraints PC_0 and PC_1 (recall that $\mathrm{PC}_i(a,b,c) = 1$ iff $a \oplus b \oplus c = i$). The result follows from Håstad's powerful optimal inapproximability result for 3-parity constraints [5] and it is stated formally in [14].

Lemma 2 *Suppose for some constraint satisfaction problem A, there exists a* sound *α_0-gadget reducing PC_0 to A and a* sound *α_1-gadget reducing PC_1 to A. Then, for every constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of A where a fraction $(1-\varepsilon)$ of the constraints can be satisfied and those where no assignment satisfies more than a fraction $(1 - \frac{1}{\alpha_0+\alpha_1} + \varepsilon)$ of the constraints.*

2.2 Result for Max E3-Set Splitting

Theorem 3 *For every constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of* Max E3-Set Splitting *where all the sets can be split by some partition and those where any partition splits at most a fraction $(\frac{19}{20} + \varepsilon)$ of the sets.*

The starting point for proving the above is the following powerful hardness result which follows from a PCP construction by Håstad [5] (namely the one he used to show the hardness result for 4-set splitting).

Definition 2 (Max SNE$_4$) *An instance of* Max SNE$_4$ *consists on Boolean variables $\{x_1, x_2, \ldots, x_n\}$ and a collection $\mathcal{C}$ of constraints of the form*

$$\mathsf{SNE}_4(x_{i_1}, x_{i_2}, x_{i_3}, x_{i_4}) = (x_{i_1} \neq x_{i_2}) \vee (x_{i_3} \neq x_{i_4})$$

(no negations allowed) defined on certain 4-element subsets of these variables.

Theorem 4 ([5]) *For any constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of* Max SNE$_4$ *where all the constraints can be satisfied by some (Boolean) assignment to the variables and where no assignment satisfies more than a $3/4 + \varepsilon$ fraction of the constraints. In other words, even satisfiable instances of* Max SNE$_4$ *are NP-hard to approximate within a factor better than $3/4$.*

Proof of Theorem 3: The result will be proven by construction of an appropriate gadget that reduces SNE_4 constraints to 3-Set Splitting constraints. Indeed we claim the following is a 5-*gadget* reducing $\mathsf{SNE}_4(a, b, c, d)$ to 3-set splitting constraints: Introduce auxiliary variables x, y, z (specific just to this constraint) and replace $\mathsf{SNE}_4(a, b, c, d)$ with the five constraints $\mathsf{SPLIT}(a, b, x)$, $\mathsf{SPLIT}(a, b, y)$, $\mathsf{SPLIT}(c, d, y)$, $\mathsf{SPLIT}(c, d, z)$, and $\mathsf{SPLIT}(x, y, z)$ (the constraint $\mathsf{SPLIT}(p, q, r)$ represents the condition that not all of p, q, r are equal). Assuming for the moment that this is a 5-gadget, the result claimed in Theorem 3 follows immediately by appealing to Lemma 1 and Theorem 4.

It remains to prove that the above is indeed a 5-gadget. Firstly, when $\mathsf{SNE}_4(a, b, c, d)$ is satisfied by some assignment to a, b, c, d; we wish to verify that all five constraints in the gadget can be satisfied by an appropriate assignment to the auxiliary variables. Indeed, let us assume that, say, $(a \neq b)$ without loss of generality (since the gadget is symmetric in $\{a, b\}$ and $\{c, d\}$); then the assignment $y = z = \bar{c}$ and $x = c$ satisfies all the five 3-set splitting constraints. On the other hand, if $\mathsf{SNE}_4(a, b, c, d)$ is not satisfied by some assignment to a, b, c, d, then we have $a = b$ *and* $c = d$. Now two cases arise: (i) $(a = b) \neq (c = d)$, in which case one of the two constraints $\mathsf{SPLIT}(a, b, y)$ and $\mathsf{SPLIT}(c, d, y)$ is violated irrespective of the assignment to y, and (ii) $a = b = c = d$, in which case the only way to satisfy all the four constraints $\mathsf{SPLIT}(a, b, x)$, $\mathsf{SPLIT}(a, b, y)$, $\mathsf{SPLIT}(c, d, y)$ and $\mathsf{SPLIT}(c, d, z)$ is to set $x = y = z = \bar{a}$ and in this case $\mathsf{SPLIT}(x, y, z)$ is violated. Thus at most four of the five constraints can be satisfied for any assignment to x, y, z if $\mathsf{SNE}_4(a, b, c, d)$ is not satisfied. In other words, the gadget above is a 5-gadget, and this completes the proof. $\Box$

Optimality of the Gadget. Using some extension of the "technology" from [14], Sorkin [13] has shown that there is no α-gadget reducing an SNE_4 constraint to CUT constraints for $\alpha < 5$. Since there is a 1-gadget reducing a 3-set splitting constraint to CUT constraints, this also shows that there is no α-gadget reducing an SNE_4 constraint to 3-set splitting constraints for any $\alpha < 5$, and thus the gadget construction we used in the proof of Theorem 3 is in fact optimal.

3 k-Set Splitting for $k \geq 4$

3.1 Hardness of MAX Ek-SET SPLITTING for $k \geq 4$

In [5], Håstad proves that even satisfiable instances of MAX E4-SET SPLITTING are hard to approximate within any factor better than 7/8. Note that this result is tight since simply picking a partition at random gives an algorithm which in expectation splits a fraction 7/8 of the sets. In this section, we prove a similar result for MAX Ek-SET SPLITTING for $k \geq 5$, namely that beating the performance achieved by a random partition is NP-hard. Specifically, we prove:

Theorem 5 *For every integer $k \geq 4$, for any constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of* MAX Ek-SET SPLITTING *where all the sets can be split by some partition and those where any partition splits at most a fraction $(1 - 2^{-k+1} + \varepsilon)$ of the sets.*

Our strategy to prove the above result is to "observe" that the PCP construction for 4-set splitting in Håstad's paper [5], together with some new arguments, gives Theorem 5. Though we will not delve into any deep details about PCPs, it will still be useful to quickly recap here the notion of a Probabilistically Checkable Proof (PCP). Moreover, this will also be useful for the hardness result for MAX 3-NM-SAT in Section 4.

The definition of PCP below is not the most general, but will suffice for our purposes.

Definition 3 (PCP) *A* verifier *for a q-query* PCP *for a language L is a probabilistic polynomial time algorithm V which aims to verify membership claims in L with the aid of oracle access to a* *binary string π. On input x and oracle access to the "proof" π, the verifier tosses $O(\log |x|)$ random coins and uses it to read q locations from π. It then accepts the proof if and only if these q bits satisfy a certain q-ary constraint C. The verifier has the property that if $x \in L$, then there is some proof π that it accepts with probability 1 (over its random coin tosses). The verifier is said to have* soundness *s if for $x \notin L$, it does not accept any proof with probability greater than s.*

The following connection between PCP and hardness of approximation is well-known.

Lemma 6 (Folklore) *Let $\mathcal{F}$ be a family of q-ary constraints. Suppose that there exists a PCP verifier with soundness s for all of NP that reads q bits and always checks if they satisfy come constraint in $\mathcal{F}$. Then, it is NP-hard to distinguish between satisfiable instances of* MAX $\mathcal{F}$ *and those where at most a fraction s of the constraints can be satisfied.*

For example, in order to prove Theorem 5, it suffices to construct a PCP verifier for NP languages that reads k locations from the proof and accepts whenever the bits read from those locations are not all equal, and has a soundness of $(1 - 2^{-k+1} + \varepsilon)$.

Let us now get back to proving Theorem 5. The case $k = 4$ of Theorem 5 is exactly Theorem 7.6 in [5]. In order to prove the result for $k \geq 5$, we could directly modify Håstad's PCP in a simple way and obtain the result. The basic idea is to read an extra bit from the proof (this is for the case $k = 5$) in addition to the four bits read by Håstad's verifier, and check that the five bits read are not all equal. Clearly this verifier accepts whenever the previous verifier accepts. One can also show that the soundness of the new verifier increases from $7/8 + \varepsilon$ to only $15/16 + \varepsilon$ as a result of this "slackening" of the test, and this gives the $15/16 + \varepsilon$ hardness result for MAX E5-SET SPLITTING. A similar approach works for larger k as well. However, we prefer and feel it is more natural to prove this result in a more "combinatorial" way, by a reduction from Håstad's result for MAX E4-SET SPLITTING. Unfortunately, we do not see any way of doing this directly from Håstad's $7/8 + \varepsilon$ hardness result for MAX E4-SET SPLITTING — instead, we need the following stronger result on the hardness of MAX E4-SET SPLITTING, which is interesting in its own right. Let us call a partition $\Pi : U \to \{0, 1\}$ of a universe U to be a ρ-partition if it maps $\rho|U|$ elements of U to 1 and the remaining $(1 - \rho)|U|$ elements to 0.

Lemma 7 *For any constant $\varepsilon > 0$, it is NP-hard to distinguish between satisfiable instances of* MAX E4-SET SPLITTING *and those where, for each ρ, $0 < \rho < 1$, every ρ-partition leaves a fraction $\rho^4 - \varepsilon$ of the sets with all 1s and a fraction $(1 - \rho)^4 - \varepsilon$ of the sets with all 0s (note that this also means that every ρ-partition splits at most a fraction $1 - \rho^4 - (1 - \rho)^4 + 2\varepsilon$ of the sets).*

Proof (Sketch): A random ρ-partition splits an expected fraction $(1 - \rho^4 - (1 - \rho)^4)$ of the sets, so the above result is tight. The proof follows from the same PCP construction that Håstad used to prove Theorem 5 for $k = 4$ [5, Sec. 7]. However, we need a different analysis that takes into account the fraction, say ρ, of the bits in the proof that are 1. Such an analysis has been recently performed by Holmerin [6]. However, his analysis applied to a slight modification of Håstad's PCP where the verifier, instead of checking if the four read bits are not-all-equal, only checks if not all of them are set to 0. Hence, the proof which sets all bits to 1 is accepted by the verifier with probability 1.

Holmerin [6] modified Håstad's analysis to show that, for every ρ, $0 < \rho < 1$, if only a fraction ρ of the bits in the proof can be set to be 1, then the soundness becomes $1 - (1 - \rho)^4 + \varepsilon$, or in other words a fraction $(1 - \rho)^4 - \varepsilon$ of the 4-sets have all their elements set to 0. We claim that this also implies that in any such proof, a fraction $\rho^4 - \varepsilon$ of the sets have all their elements set to 1.

Indeed, if not, one can flip all bits in this proof, and we will get a proof with fraction $(1-\rho)$ of 1s which has fewer than $\rho^4 - \varepsilon$ of the sets with all their elements set to 0. This contradicts Holmerin's analysis. Therefore, we conclude that, for every ρ, $0 < \rho < 1$, any proof with fraction ρ of 1s leaves a fraction $\rho^4 - \varepsilon$ of the sets with all 1s and a fraction $(1-\rho)^4 - \varepsilon$ of the sets with all 0s. (As a corollary, we also know that if fails to split at least a fraction $\rho^4 + (1-\rho)^4 - 2\varepsilon$ of the sets.) This proves the result claimed in Lemma 7. □

We are now ready to prove the following result for Max Ek-Set Splitting for $k \geq 5$ using a simple reduction from the $k = 4$ case. We note that the result below immediately implies Theorem 5, our main result of this section.

Theorem 8 *For any constant $\varepsilon > 0$ and every integer $k \geq 4$, it is NP-hard to distinguish between satisfiable instances of* Max Ek-Set Splitting *and those where, for each ρ, $0 < \rho < 1$, every ρ-partition splits at most a fraction $1 - (\rho^k + (1-\rho)^k) + \varepsilon$ of the sets.*

Proof: The case $k = 4$ is handled by Lemma 7. For larger k, we give a reduction from the hard instances of Max E4-Set Splitting guaranteed by Lemma 7. Given an instance $\mathcal{I} = (U, \mathcal{F})$ of Max E4-Set Splitting with $|U| = n$ and $|\mathcal{F}| = M$ where $\mathcal{F}$ is a collection of 4-element subsets of U, we define an instance $\mathcal{I}'$ of Max Ek-Set Splitting as follows. The universe U remains the same. For each 4-set $S \in \mathcal{F}$, we add all the k-sets $\{T : |T| = k \text{ and } S \subset T\}$ to the instance $\mathcal{I}'$.

Clearly, if the instance $\mathcal{I}$ is satisfiable, then the same partition also splits all the sets in the instance $\mathcal{I}'$. Now assume that $\mathcal{I}$ is not satisfiable. Consider any ρ-partition Π of U. We know by the "promise" on $\mathcal{I}$ from Lemma 7 that at least a fraction $(\rho^4 - \varepsilon)$ of the sets in $\mathcal{F}$ have all their elements set to 1 by Π; let $\mathcal{F}_1$ be the collection of these sets. Similarly if $\mathcal{F}_0$ is the collection of all sets in $\mathcal{F}$ which have all their elements set to 0 by Π, then $|\mathcal{F}_0| \geq ((1-\rho)^4 - \varepsilon)M$. It is now easy to see that for each $S \in \mathcal{F}_1$, a fraction $\rho^{k-4} - o(1)$ of the k-sets containing it (which are all part of the instance $\mathcal{I}'$) have all their elements set to 1, and a similar statement holds for the sets in $\mathcal{F}_0$. Hence, it follows that at least a fraction $(\rho^k + (1-\rho)^k - 2\varepsilon - o(1))$ of the sets in the instance $\mathcal{I}'$ are not split by the partition Π. Since this holds for every ρ-partition and $\varepsilon > 0$ was arbitrary, the result claimed in Theorem 8 follows. □

3.2 "Restricted" 4-set splitting

For Max E4-Set Splitting, if the goal is only to split as many sets as possible, then Håstad's result shows that the best one can do is a factor of 7/8, and this is tight. One can be more specific about requirements on the partition, and for example require that that we maximize the number of 4-sets which have a 1-3 split under the partition, or similarly require that we maximize the number of sets that have a 2-2 split under the partition. Thus we put restrictions on the way in which a set can be split for it to count towards the solution. We call these "restricted" 4-set splitting problems Max 1-3-Set Splitting and Max 2-2-Set Splitting respectively. Other than being natural questions, one motivation to consider these "restricted" set splitting questions is their potential applicability to improving the hardness result for approximating Max Cut; we will discuss this possibility in more detail at the end of this section.

It is easy to see that a Max 1-3-Set Splitting instance can be cast as a system of linear equations over GF(2) each of the form $x_{i_1} \oplus x_{i_2} \oplus x_{i_3} \oplus x_{i_4} = 1$ (no negations involved in the literals), and by yet another powerful result in [5] (specifically, Theorem 5.6 of that paper), this problem is hard to approximate within any factor better than 1/2. (Of course, picking a random partition will satisfy half of these linear constraints, so this result is again tight.) We focus, therefore, on Max

2-2-SET SPLITTING problem. Note that this problem is NP-hard by a simple reduction from 3-set splitting: simply replace each 3-set (a, b, c) by (a, b, c, t) where t is a new variable that occurs only in this constraint. We now prove the following inapproximability result:

Theorem 9 *For any constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of* MAX 2-2-SET SPLITTING *which are satisfiable (i.e., all the sets can be 2-2-split by some partition) and those where any partition 2-2-splits at most a fraction $(\frac{11}{12} + \varepsilon)$ of the sets.*

Proof: The proof proceeds along similar lines to that of Theorem 3, namely via a gadget construction from SNE_4. Specifically we will give a 3-gadget reducing an SNE_4 constraint to 2-2-set splitting constraints. Using this gadget in Lemma 1 together with the hardness result for MAX SNE_4 from Theorem 4, we immediately get the claimed hardness result for MAX 2-2-SET SPLITTING.

The gadget is in fact very simple. Given a constraint $\mathsf{SNE}_4(a, b, c, d)$, introduce auxiliary variables x, y, z, w and replace $\mathsf{SNE}_4(a, b, c, d)$ with the three constraints: 2-2-SPLIT(a, b, x, y), 2-2-SPLIT(c, d, y, z) and 2-2-SPLIT(x, y, z, w). We claim that this is a 3-gadget. Indeed suppose that $\mathsf{SNE}_4(a, b, c, d)$ is satisfied. We consider two cases: (i) $(a \neq b) \wedge (c = d)$, and (ii) $(a \neq b) \wedge (c \neq d)$ (the third case $(a = b) \wedge (c \neq d)$ is similar to case (i)). In case (i), we can satisfy all three 2-2-SPLIT constraints by setting $y = z = \bar{c}$ and $x = w = c$. In case (ii), we can satisfy all three 2-2-SPLIT constraints by setting $x = z = 0$ and $y = w = 1$.

Now consider the case when $\mathsf{SNE}_4(a, b, c, d)$ is not satisfied, so that $(a = b) \wedge (c = d)$. In order to satisfy both 2-2-SPLIT(a, b, x, y) and 2-2-SPLIT(c, d, y, z), one is forced to pick $x = y = \bar{a}$ and $y = z = \bar{c}$. Now either $a \neq c$ in which case the two forced assignments to y are not consistent, or we have $x = y = z$ which violates 2-2-SPLIT(x, y, z, w). Hence at most two of the three constraints can be satisfied in this case. Thus the above gadget is indeed a 3-gadget and the proof is complete. □

If we relax the requirement that the instance of MAX 2-2-SET SPLITTING in Theorem 9 above be satisfiable, then we can prove a better hardness result.

Theorem 10 *For any constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of* MAX 2-2-SET SPLITTING *where a fraction $(1 - \varepsilon)$ of all the sets can be 2-2-split by some partition and those where any partition 2-2-splits at most a $(\frac{8}{9} + \varepsilon)$ fraction of the sets.*

To prove the above theorem, we first prove a hardness result for the MAX ONE problem, where the constraint ONE(a, b, c) is satisfied if and only if exactly one of the three Boolean variables a, b, c takes on the value 1, and furthermore we do not allow any negations in the ONE constraints.

Lemma 11 *For any constant $\varepsilon > 0$, it is NP-hard to distinguish between instances of* MAX ONE *where a $(1 - \varepsilon)$ fraction of the* ONE *constraints can be satisfied and those where no assignment satisfies more than a fraction $(\frac{8}{9} + \varepsilon)$ of the constraints.*

Proof: To prove the claimed hardness result for MAX ONE, we give gadgets from PC_0 and PC_1. Given a PC_1 constraint $a \oplus b \oplus c = 1$, the gadget transforms it into ONE constraints as follows: Introduce auxiliary variables x, y, z and the three constraints ONE(a, x, y), ONE(b, y, z) and ONE(c, z, x). It is easy to check that this is a 3-gadget.

Given a PC_0 constraint $a \oplus b \oplus c = 0$, note that this is equivalent to $\bar{a} \oplus \bar{b} \oplus \bar{c} = 1$, and thus one can use the PC_1 gadget if one can somehow force three variables t_1, t_2, t_3 to equal $\bar{a}, \bar{b}, \bar{c}$ respectively (recall that we do *not* allow negations in our constraints). To achieve this we use a global auxiliary variable F that is shared across all constraints; we think of F as being set to 0.

The gadget now consists of the six ONE constraints: $\mathsf{ONE}(a, t_1, F)$, $\mathsf{ONE}(b, t_2, F)$, $\mathsf{ONE}(c, t_3, F)$, $\mathsf{ONE}(t_1, x, y)$, $\mathsf{ONE}(t_2, y, z)$ and $\mathsf{ONE}(t_3, z, x)$ (here t_1, t_2, t_3, x, y, z are auxiliary variables specific to PC_0 constraint $a \oplus b \oplus c = 0$ and F is a global auxiliary variable). The purpose of the first three constraints is to ensure (assuming $F = 0$) that $t_1 = \bar{a}$, $t_2 = \bar{b}$ and $t_3 = \bar{c}$ and the last three check that $t_1 \oplus t_2 \oplus t_3 = 1$. If we force $F = 0$, it is therefore easy to see that the above is a 6-gadget. If $F = 1$, then the only way first three ONE constraints can be satisfied is if $t_1 = t_2 = t_3 = 0$, and in this case at most two of the last three ONE constraints can be satisfied. Hence, the above is indeed a 6-gadget.

We thus have a 3-gadget from PC_1 and a 6-gadget from PC_0. Now, applying Lemma 2 with these gadgets, we get the claimed inapproximability result for MAX ONE. □

Proof of Theorem 10: Given an instance $\mathcal{I}$ of MAX ONE consider the MAX 2-2-SET SPLITTING instance $\mathcal{I}'$ produced by adding one global element T to each ONE constraint of $\mathcal{I}$, i.e., replace $\mathsf{ONE}(a, b, c)$ with $\text{2-2-}\mathsf{SPLIT}(a, b, c, T)$. Clearly the optimum value of $\mathcal{I}$ is the same as that of $\mathcal{I}'$ and any solution to $\mathcal{I}'$ that satisfies Q constraints can be transformed efficiently into a solution to $\mathcal{I}$ that satisfies the same number Q of ONE constraints. Thus the hardness result for MAX ONE from Lemma 11 implies the result claimed in Theorem 10. □

Discussion. One motivation for studying MAX 2-2-SET SPLITTING is a quest to obtain an improved hardness result for MAX CUT. The strong hardness results for set splitting obtained recently, for example the one for MAX E4-SET SPLITTING, are not directly amenable to reduction to CUT constraints by a "good" gadget, and the reason for this is precisely the lack of stringent requirements on how the set must be split in the set splitting constraint. On the other hand, there is a simple 4-gadget reducing the constraint $\text{2-2-}\mathsf{SPLIT}(a, b, c, d)$ to CUT constraints: namely include the $\binom{4}{2}$ CUT constraints on all pairs of elements drawn from $\{a, b, c, d\}$. This raises the hope that new, strong inapproximability results for some suitable "restricted" set splitting problem (akin to Håstad's recent results [5] for MAX E4-SET SPLITTING), if they can be proved, would give us a "new" shot at improving the inapproximability results for MAX CUT (and may be even MAX E3-SET SPLITTING).

To get such strong hardness results, it seems that one would have to design and analyze special purpose PCPs whose acceptance predicate is close to the concerned set splitting question (just like Håstad's result for MAX E4-SET SPLITTING), and this appears to be an extremely challenging task. Of course, in addition to MAX 2-2-SET SPLITTING, one could also consider other "restricted" set splitting problems; for example a good candidate could be MAX 2-3-SET SPLITTING. For this problem, an inapproximability result for a factor $(\frac{7}{8} + \varepsilon)$ follows by a trivial reduction from MAX E4-SET SPLITTING. On the algorithmic side, a simple reduction to MAX E3-SET SPLITTING gives an algorithm with approximation ratio roughly 0.64, which is better than the trivial factor 5/8 achieved by a random solution. Once again there is a simple 3-gadget reducing 2-3-SPLIT constraints to CUT constraints, so a hardness result for approximating MAX 2-3-SET SPLITTING within a factor, say 0.8, would give an improvement on the currently best known factor $(\frac{16}{17} + \varepsilon)$ hardness result for MAX CUT.

4 Hardness of approximating MAX 3-NM-SAT

4.1 Main Results

We will prove the following theorems in this section.

Theorem 12 *For any $\varepsilon > 0$, it is NP-hard to distinguish between satisfiable instances of* Max 3-NM-Sat *and those where at most a fraction $11/12 + \varepsilon$ of the clauses can be satisfied.*

Theorem 13 *For any $\varepsilon > 0$, it is NP-hard to distinguish between satisfiable instances of* Max E3-NM-Sat *and those where at most a fraction $15/16 + \varepsilon$ of the clauses can be satisfied.*

Sketch of Idea: The instances of 3Sat which are proved hard to approximate within a factor of $(7/8 + \varepsilon)$ by Håstad [5] have the property that (at least) $3/4$ of the clauses are "mixed" (i.e., have both positive and negative literals). In proving a hardness result for, say, Max 3-NM-Sat, one converts these clauses into non-mixed clauses using a 2-gadget as described in Section 1.3. This method will give a factor $(13/14 + \varepsilon)$ hardness for Max 3-NM-Sat. In order to obtain an improved hardness bound, our approach will be to get a similar hardness result for 3Sat when the fraction of mixed clauses is smaller, and then use the same gadget approach to get hardness for Max 3-NM-Sat. To this end, we will prove:

Theorem 14 *For any $\varepsilon > 0$, given a* Max E3-Sat *instance in which at most a fraction $1/2$ of the clauses are mixed, it is NP-hard to distinguish between instances which are satisfiable and those where every assignment satisfies at most a fraction $(7/8 + \varepsilon)$ of the clauses.*

Let us postpone the proof of the above result to Section 4.2 and first prove the results claimed in Theorems 12 and 13 using Theorem 14.

Proof of Theorem 12: There is a simple reduction from E3Sat to non-mixed 3Sat obtained by replacing a clause $(a \vee b \vee \bar{c})$ with two non-mixed clauses $(a \vee b \vee t)$ and $(\bar{t} \vee \bar{c})$ where t is a new variable used only in these two clauses, and by similarly replacing a clause $(a \vee \bar{b} \vee \bar{c})$ with two non-mixed clauses $(a \vee t)$ and $(\bar{t} \vee \bar{b} \vee \bar{c})$. Starting from a hard instance of E3Sat (3Sat with all clauses having *exactly* three literals) as in Theorem 14, satisfiable instances of E3Sat get mapped to satisfiable instances of Max 3-NM-Sat, while instances where at most a fraction $7/8 + \varepsilon$ of the clauses are satisfiable get mapped to instances of Max 3-NM-Sat where at most a fraction $11/12 + 2\varepsilon/3$ of the clauses are satisfiable. Since $\varepsilon > 0$ is arbitrary, the result of Theorem 12 follows. $\square$ (*Theorem 12*)

Proof of Theorem 13: The proof is similar to the above, except now a clause $(a \vee b \vee \bar{c})$ is replaced by the three non-mixed clauses $(a \vee b \vee t_1)$, $(a \vee b \vee t_2)$ and $(\bar{c} \vee \bar{t_1} \vee \bar{t_2})$, and a clause $(a \vee \bar{b} \vee \bar{c})$ is replaced by the three non-mixed clauses $(\bar{b} \vee \bar{c} \vee \bar{t_1})$, $(\bar{b} \vee \bar{c} \vee \bar{t_2})$ and $(a \vee t_1 \vee t_2)$ where t_1, t_2 are new variables used only in these three clauses. Using the result of Theorem 14, a simple calculation now shows the hardness of approximating (even satisfiable instances of) Max E3-NM-Sat within any factor strictly greater than $15/16$. $\square$ (*Theorem 13*)

4.2 Proof of Theorem 14

We now turn to the proof of Theorem 14. Since we want a hardness result for (a restricted version) 3SAT, we will design a PCP verifier that reads only three bits of the proof and simply checks that some 3SAT constraint on them is satisfied. Moreover, for 50% of the random strings, the 3SAT constraint checked by the verifier will be non-mixed. Further, the soundness of the verifier will be arbitrarily close to $7/8$. One can then apply Lemma 6 to prove Theorem 14.

A PCP verifier with soundness $(7/8 + \varepsilon)$ and whose acceptance predicate is a 3SAT constraint has already been given by Håstad [5]. However, the 3SAT constraint checked by the verifier is non-mixed only with probability $1/4$. We essentially observe that with a slightly different verifier and proof access mechanism, one can make this probability 1/2. Our contribution in showing this is minor, and for the readers who are aware of the PCP literature, it suffices to say that we simply note that one can perform a tight analysis of Håstad's construction even without the folding assumption for one of the two proof tables. The PCP expert might thus want to skip the rest of this section. However, in order to make it easy for non-experts to understand how and why Håstad's PCP can be thus modified, we below sketch the PCP construction and give pointers to the specific Lemmas in [5] that have to be reworked without the "folding" assumption.

4.2.1 PCP construction methodology: The Basics

We first give an overview of the by now standard paradigm for PCP constructions. The basic construction methodology is to start with a bounded occurrence instance of 3SAT where there is a gap in the optimum (either the formula is satisfiable or at most a fraction μ of clauses are satisfied by any assignment). One then uses Raz's parallel repetition theorem [12] to obtain a *two-prover one-round* (2P1R) proof system with completeness 1 and arbitrarily low soundness. The verifier in the 2P1R system picks a set of u clauses of the 3SAT instance at random and also picks a set U of u variables, one in each of the picked clauses, at random. The first prover P_1 is expected to give an assignment to the set W of all $3u$ variables in the u clauses and the second prover P_2 is expected to give an assignment to the variables in U (both P_1 and P_2 are each unaware of the other prover's question). The verifier accepts if the answer of the P_1 satisfies the clauses and is consistent with the answer of P_2. Note that if the original 3SAT formula was satisfiable, there exist strategies for Raz showed that this protocol has soundness at most c^u for some absolute constant c.

While this gives a 2P1R system and hence a PCP construction with very low soundness, the query complexity is high (proportional to u). We would like the number of queries to be small (in fact, 3 for the application to 3SAT), and are willing to compromise a bit in the soundness. The way this is achieved is the following. The verifier now expects the proof tables of P_1 and P_2 to consist of *encodings* of their respective answers as per some error-correcting code C. The hope is that the verifier can then check with reasonable accuracy the necessary property about the answers by querying their purported encodings at very few locations. Of course, there is nothing which prevents a cheating prover from writing down something very different from a codeword of C, and the challenge is to find a code C and design a highly efficient, yet somewhat sound, verification process for it. It turns out that one such code is the *Long Code*, a great invention of [1] that has played a crucial role in almost all recent PCP constructions (including all the ones in [5]). In the sequel, it will be convenient to represent Boolean values with values in $\{1,-1\}$ with -1 standing for TRUE and 1 for FALSE.

For a set U of variables, we denote by $\{1,-1\}^U$ the set of all possible Boolean assignments to these variables. We let $\mathcal{F}_U$ be the set of all functions $f : \{1,-1\}^U \to \{1,-1\}$. We now define the Long Code formally.

Definition 4 (Long Code) *The* Long Code *of an assignment $x \in \{1,-1\}^U$ is the mapping $A_x : \mathcal{F}_U \to \{1,-1\}$ where $A_x(f) = f(x)$.*

The proof expected by the PCP verifier now consists of purported Long Codes A_U, A_W of the assignments to the variables in U and W for each possible choice U, W of the 2P1R verifier. The final verifier picks the set of variables U, W as the 2P1R verifier does, and then aims to check if A_U and A_W are Long Codes of two strings x, y such that:

(i) y satisfies all the clauses in W (i.e. whether $h_W(y) = -1$ where $h_W \in \mathcal{F}_W$ is the Boolean function which is TRUE for exactly those assignments which satisfy all the clauses corresponding to W), and

(ii) $y_{|U} = x$ (i.e., y agrees with x on the coordinates in U).

Since there is no way to guarantee that the proofs written down are actually codewords of the Long Code, soundness proofs of Long Code based verifiers depend heavily on Fourier analysis of real functions $A : \mathcal{F}_U \to \mathbb{R}$. Crucial elements here are the *Fourier coefficients* of A which are defined below.

Definition 5 *Let $A : \mathcal{F}_U \to \mathbb{R}$. Then, for each $\alpha \subseteq \{1,-1\}^U$, the* Fourier coefficient *of A w.r.t α, denoted $\hat{A}_\alpha$, is defined as:*

$$\hat{A}_\alpha = \mathop{\mathbf{E}}_{f \in \mathcal{F}_U}\left[A(f) \prod_{x \in \alpha} f(x)\right] .$$

The Fourier inversion formula gives

$$A(f) = \sum_{\alpha \subseteq \{1,-1\}^U} \hat{A}_\alpha \prod_{x \in \alpha} f(x) .$$

Fourier analysis of the tables A_U, A_W using the above expansion plays a central role in analysing Long Code based PCP verifiers.

Condition (i) above that the verifier needs to check ends up being automatically taken care of by a trick which Håstad calls "conditioning over h": namely we insist that $A_W(g \wedge h_W) = A_W(g)$ for every $g \in \mathcal{F}_W$. We may assume that A_W is "conditioned upon h" for any desired h since in the test we can work with a table $A_{W,h}$ defined by $A_{W,h}(g) = A_W(g \wedge h)$ (note that an access to the oracle $A_{W,h}$ can be "simulated" using one access to A_W and thus query complexity of the test is unaltered whether we work with A_W or $A_{W,h}$). If $B = A_{W,h_W}$, then for every non-zero Fourier coefficient $\hat{B}_\beta$ (here $\beta \subseteq \{1,-1\}^W$), each $y \in \beta$ satisfies $h_W(y) = -1$ [5, Lemma 2.34]. This ends up giving the verification that y satisfies the relevant clauses for free. We refer the reader to [5] (eg. the proof of Lemma 5.2 therein) for further details on this aspect.

4.2.2 Folding of Long Codes

It remains to check the Condition (ii), namely the "projection" requirement $y_{|U} = x$. For the 3SAT result in [5], the verifier does this by picking functions $f \in \mathcal{F}_U$ and $g_1, g_2 \in \mathcal{F}_W$ from a "suitable" distribution and checking if at least one of the three bits $A_U(f), A_{W,h_W}(g_1), A_{W,h_W}(g_2)$ equals -1. Note that this test is as such no good, since the proof which sets all entries in each of the A_U and A_W tables to -1 will make the verifier accept with probability 1. This predicament is handled by another elegant idea called *"folding"*. A function $A : \mathcal{F}_U \to \{1,-1\}$ is said to be *folded* if $A(-f) = -A(f)$ for every $f \in \mathcal{F}_U$. Note that the Long Code of a string $x \in \{1,-1\}^U$ is always folded. In order to avoid the all -1's table as a potential proof, the construction in [5] assumes that the A_U and A_W tables are all folded. One is allowed to assume this since given access to a (potentially unfolded) table $A : \mathcal{F}_U \to \{1,-1\}$, we can work with its "folded version", denoted A^{fold}, defined by

$$A^{\text{fold}}(f) = \begin{cases} A(f) & \text{if } f(a_0) = 1 \\ -A(-f) & \text{otherwise} \end{cases}$$

where a_0 is some fixed element in $\{1,-1\}^D$. Now, $A^{\text{fold}}(-f) = -A^{\text{fold}}(f)$ is clearly satisfied for a table thus defined, and moreover, an oracle call to A^{fold} can be "simulated" using one oracle call to A (so the query complexity of a verifier is unaltered with the addition of the folding assumption).

4.2.3 Håstad's PCP construction for 3SAT

Håstad's PCP verifier for 3SAT (see [5, Theorem 6.5]) works with these folded tables, i.e. it picks three functions f, g_1, g_2 and then checks if at least one of $A_U^{\text{fold}}(f)$, $A_{W,h_W}^{\text{fold}}(g_1)$ and $A_{W,h_W}^{\text{fold}}(g_2)$ equals -1. Note that if f, g_1, g_2 were picked uniformly at random and independently of each other, among the constraints checked by the verifier, there would be a fraction 3/4 of them that are mixed. (This is because, by the definition of folding, for exactly a fraction 1/2 of the functions f, $A_U^{\text{fold}}(f)$ equals the complement of the bit $A_U(-f)$, and thus is a "negated literal", and similarly for the A_{W,h_W}^{fold} tables.) For Håstad's verifier, the individual distribution of each of f, g_1, g_2 is uniform over the respective function space. However, the joint distribution of (f, g_1, g_2) as picked by Håstad's verifier is more complicated. Nevertheless, by digging deeper into how these functions are actually picked, one can arrange things so that at most a fraction $(3/4 + \delta)$ of the clauses are mixed.

4.2.4 Our modified PCP

We reduce the fraction of mixed clauses in a hard instance of 3SAT (to 1/2 from $3/4 + \delta$) by observing that in the above PCP, there is no need to fold the A_W tables. That is, our verifier picks U, W, $f \in \mathcal{F}_U$, and $g_1, g_2 \in \mathcal{F}_W$ exactly as in Test $\mathbf{F3S}^\delta(u)$ in the proof of Theorem 6.5 in [5], and it checks the constraint

$$A_U^{\text{fold}}(f) \vee (A_{W,h_W}(g_1) \neq A_{W,h_W}(g_2)) \ , \tag{1}$$

which is the same as simultaneously checking the two 3SAT constraints

$$(A_U^{\text{fold}}(f) \vee A_{W,h_W}(g_1) \vee A_{W,h_W}(g_2)) \quad \text{and} \quad (A_U^{\text{fold}}(f) \vee \overline{A_{W,h_W}(g_1)} \vee \overline{A_{W,h_W}(g_2)}) \ . \tag{2}$$

We remark here, as is also made explicit in Theorem 6.18 of [5], that the functions f, g_1, g_2 picked by Håstad's verifier are such that for every $y \in \{1,-1\}^W$, the triple $(f(y_{|U}), g_1(y), g_2(y))$ never takes the values $(1,1,1)$ or $(1,-1,-1)$ (even though for the application to 3SAT one only needs to avoid the triple $(1,1,1)$). This implies that the verifier is "allowed" to make the stronger test (1). That is, for YES instances of the language being verified by the PCP, there is a proof, namely one consisting of Long Codes of restrictions of a satisfying assignment to subsets of variables, which makes the verifier accept with probability 1.

Note that of the two 3SAT clauses (2) checked by the verifier, one of them is non-mixed and other is mixed. Therefore, the 3SAT instance produced by listing all the constraints tested by the verifier (over all its random coin tosses) will have exactly a fraction 1/2 of mixed clauses. Let us label the above verifier as $V_\delta^{\text{3SAT}}(u)$ (u, δ are parameters of the verifier and are the same ones as in [5, Test $\mathbf{F3S}^\delta(u)$]).

To prove Theorem 14, we now claim that it suffices to prove the following:

Lemma 15 *For every small enough $\delta > 0$, for all sufficiently large u, the soundness of the verifier $V_\delta^{\text{3SAT}}(u)$ is at most $(3 + 5\delta)/4$.*

Indeed, Theorem 14 follows from the above lemma and Lemma 1 with the choice $c = (3+5\delta)/4$ and $\alpha = 2$, since the transformation (2) of the constraint (1) gives a sound 2-gadget reducing the predicate tested by the verifier to 3SAT. We therefore turn to proving Lemma 15 next.

4.2.5 Soundness analysis: Proof of Lemma 15

To estimate the soundness, we begin by noting that the expression

$$X = X_{U,W,f,g_1,g_2} \ = \ 1 - \ \frac{(1 + A_U^{\text{fold}}(f))(1 + A_{W,h_W}(g_1))(1 + A_{W,h_W}(f))}{8}$$

$$- \frac{(1+A_U^{\text{fold}}(f))(1-A_{W,h_W}(g_1))(1-A_{W,h_W}(f))}{8}$$
$$= \frac{3}{4} - \frac{1}{4}\Big(A(f) + B(g_1)B(g_2) + A(f)B(g_1)B(g_2)\Big) \quad (3)$$

(where in the second step we have used the shorthand $A = A_U^{\text{fold}}$ and $B = A_{W,h_W}$) equals 1 whenever the verifier accepts the triple $(A(f), B(g_1), B(g_2))$ and equals 0 otherwise.

The acceptance probability of $V_\delta^{3\text{SAT}}(u)$ is thus exactly the expectation of X over the choice of U, W, f, g_1, g_2. We want to prove this expectation is at most $(3/4 + 5\delta/4)$ for No instances of the language being verified. The way this is done is by showing that if this quantity is greater than $(3/4 + 5\delta/4)$, then there exist strategies for the two provers in the 2P1R game which makes the verifier accept with probability h_δ for some constant depending only on δ. By picking the parameter u (the number of "parallel repetitions") large enough, we can make the soundness of the 2P1R, which decreases exponentially in u by Raz's theorem [12], to be smaller than h_δ. This gives a contradiction, and thus the soundness of $V_\delta^{\text{SAT}}(u)$ must be at most $(3+5\delta)/4$.

By linearity of expectation, we have to estimate $\mathbf{E}[A(f)]$, $\mathbf{E}[B(g_1)B(g_2)]$ and $\mathbf{E}[A(f)B(g_1)B(g_2)]$ in order to estimate $\mathbf{E}[X]$. For the first two terms, we will able to explicitly bound them. For the last term, one shows that a large value of it implies the existence of good strategies for the provers in the Raz 2P1R game.

For any folded table $A : \mathcal{F}_U \to \{1, -1\}$, we have

$$\mathop{\mathbf{E}}_f[A(f)] = 0 \ , \quad (4)$$

since the term $A(f)$ can be "paired" with $A(-f) = -A(f)$. It remains to analyze the latter two terms. These are already analyzed by Håstad [5]. Therefore, in order to avoid repeating the details, we will be informal in what follows and give precise pointers to the relevant lemmas in [5] where the analysis appears (and also how they continue to work even without the folding assumption).

From Lemma 7.9 in [5] (which is really Lemma 6.7 proved without the folding assumption for the B tables), it follows that

$$\mathop{\mathbf{E}}_{U,W,f,g_1,g_2}[B(g_1)B(g_2)] \geq -4\delta \ . \quad (5)$$

Unfortunately, this may not be immediately apparent to the reader from the statement of Lemma 7.9 in [5], so we shed some light on why this follows. To do this, it will help to be more specific about how the functions f, g_1, g_2 are picked. The functions f and g_1 are picked uniformly at random from $\mathcal{F}_U$ and $\mathcal{F}_W$ respectively. The function $g_2 \in \mathcal{F}_W$ is picked by for each $y \in \{1,-1\}^W$ independently doing the following. If $f(y_{|U}) = 1$ then $g_2(y) = -g_1(y)$ while if $f(y_{|U}) = -1$, then $g_2(y) = g_1(y)$ with probability $1-\epsilon$ and otherwise $g_2(y) = -g_1(y)$. Here ϵ is a "bias" parameter which itself is picked from a particular distribution (which depends only on the parameter δ and is made explicit in Test $\mathbf{F3S}^\delta(u)$ in the proof of Theorem 6.5 in [5]). Lemma 7.9 in [5] calculates the expectation of $B(g_1)B(g_2)$ for a fixed bias parameter ϵ. Taking expectations over the choice of ϵ as well, as done in Lemma 6.13 or Lemma 7.14 of [5], one can ensure the bound claimed in (5).

It remains to analyze the $\mathbf{E}[A(f)B(g_1)B(g_2)]$ term. For this we use the following lemma:

Lemma 16 *If*

$$\Big| \mathop{\mathbf{E}}_{U,W,f,g_1,g_2}[A(f)B(g_1)B(g_2)] \Big| \geq \delta$$

then there is a strategy for P_1 and P_2 in the 2P1R game that makes the verifier of that protocol accept with probability at least $2^{-O(\delta^{-1}\log\delta^{-1})}$.

Before proving the above lemma, let us observe that the claim of Lemma 15 follows from Equations (3), (4) and (5) and Lemma 16. Indeed, if the acceptance probability as in Equation (3) is greater than $(3 + 5\delta)/4$, then together with Equations (4) and (5), we must have $\mathbf{E}[A(f)B(g_1)B(g_2)] \leq -\delta$. Lemma 16 then implies that there exist good strategies for the provers in the 2P1R game, which implies the instance whose membership is being verified must be an YES instance.

Proof of Lemma 16: The lemma is essentially the same as Lemma 6.11 of [5] and the only reason we cannot simply appeal to it is that we are not assuming that the B tables are folded. As it turns out, Håstad's proof of Lemma 6.11 does not really use the fact that the B table is folded in a crucial way, so our lemma really follows from his Lemma 6.11. We elaborate on and substantiate this claim below.

Håstad's proof proceeds by showing that if $|\mathbf{E}[A(f)B(g_1)B(g_2)]\ |$ is large, then the following expression

$$\mathop{\mathbf{E}}_{U,W}\Big[\sum_{\substack{\beta,\alpha\subseteq\pi(\beta)\\ |\beta|<S}} \hat{A}_\alpha^2 \hat{B}_\beta^2\Big] \tag{6}$$

is large (actually $\Omega(\delta^2)$). Here $\alpha \subseteq \{1,-1\}^U$; $\beta \subseteq \{1,-1\}^W$; $\pi \in \mathcal{F}_W$ is the projection function $\pi(y) = y_{|U}$ for $y \in \{1,-1\}^W$; $S = 2^{O(\delta^{-1}\log\delta^{-1})}$ is a constant depending only on δ; and $\hat{A}_\alpha$ (resp. $\hat{B}_\beta$) are *Fourier coefficients* of A (resp. B) w.r.t α (resp. β). Since A, B are functions with range $\{1,-1\}$, these Fourier coefficients satisfy the Parseval's identity: $\sum_\alpha \hat{A}_\alpha^2 = 1$ and $\sum_\beta \hat{B}_\beta^2 = 1$.

Using this, one gets strategies for P_1 and P_2 as follows. P_1 (resp. P_2) picks an β (resp. α) with probability $\hat{B}_\beta^2$) (resp. $\hat{A}_\alpha^2$) and then gives as answer a random $y \in \beta$ (resp. random $x \in \alpha$). The success probability of this strategy is at least

$$\mathop{\mathbf{E}}_{U,W}\Big[\sum_{\beta,\alpha\subseteq\pi(\beta)} \hat{A}_\alpha^2 \hat{B}_\beta^2 |\beta|^{-1}\Big]\ . \tag{7}$$

Since the expression (6) is large, this implies that the (7) is also large, and thus there exist strategies for P_1 and P_2 with good success probability.

The only potential glitch in applying this when the B tables are not necessarily folded is that we might have $\hat{B}_\emptyset \neq 0$ and the strategy for P_1 defined above might get "stuck" when it picks $\beta = \emptyset$. However, we have $\pi(\emptyset) = \emptyset$ and $\hat{A}_\emptyset = 0$ (since the A tables are folded), and therefore terms with $\beta = \emptyset$ do not contribute anything to the sum (7) (or for that matter to the sum (6)). Hence, if we modify P_2's strategy so that it simply declares a failure when it happens to pick $\beta = \emptyset$, the success probability of this modified strategy will still be at least (7). Thus the lack of folding of B tables poses no problems to Håstad's proof. This allows us to prove Lemma 16 using verbatim the proof of Lemma 6.11 in [5]. □ *(Lemma 16)*

5 Concluding Remarks

The exact approximability of MAX E3-SET SPLITTING remains an intriguing open question, though by now the gap between the positive and negative results is quite small (there is a 0.908 approximation algorithm, and it is NP-hard to get a better than 0.95 approximation; in fact the hardness holds for satisfiable instances as well for which a 0.912 approximation algorithm exists). A similar situation exists with MAX CUT where a 0.878 approximation algorithm exists, while approximating within a factor of 0.942 is NP-hard.

We established the tight result that approximating MAX E3-SAT where at most half the number of clauses are mixed to within any factor better than 7/8 is NP-hard. We then used this result to deduce hardness of approximating MAX 3-NM-SAT and MAX E3-NM-SAT within factors better than 11/12 and 15/16 respectively. It is not clear how to algorithmically exploit the non-mixed nature of all clauses and devise an algorithm with performance ratio better than 7/8 for either of these problems. In fact, it might very well be that these problems are hard to approximate within $7/8+\varepsilon$, and if this is the case, we believe an interesting new approach might be required to establish such a result.

Shedding further light on the approximability of MAX 2-2-SET SPLITTING (and similarly MAX 2-3-SET SPLITTING) are other interesting open questions arising from our work.

Acknowledgments

I would like to thank Johan Håstad, Greg Sorkin, Madhu Sudan and Uri Zwick for useful discussions, and Subhash Khot for useful comments on the write-up. I am grateful to Greg Sorkin for his proof that the 5-gadget from SNE_4 to 3-set splitting is in fact optimal. I would also like to thank the anonymous referees whose comments had a very positive influence on the presentation of this paper.

References

[1] M. Bellare, O. Goldreich and M. Sudan. Free bits, PCP's and non-approximability – towards tight results. *SIAM Journal on Computing*, 27(3):804-915, 1998. Preliminary version in *Proc. of FOCS'95*.

[2] U. Feige and M. Goemans. Approximating the value of two prover proof systems, with applications to MAX-2SAT and MAX-DICUT. *Proc. of the 3rd Israel Symposium on Theory and Computing Systems, Tel Aviv*, pp. 182-189, 1995.

[3] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115-1145, 1995.

[4] V. Guruswami. Inapproximability results for set splitting and satisfiability problems with no mixed clauses. *Proceedings of the 3rd International Workshop on Approximations Algorithms for Combinatorial Optimization Problems (APPROX)*, September 2000.

[5] J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, to appear, 2002. All specific references to this paper refer to the JACM version. Earlier versions appear as *ECCC Technical Report TR97-37*, 1997.

[6] J. Holmerin. Vertex Cover on 4-regular hyper-graphs is hard to approximate within $(2-\varepsilon)$. *Proceedings of the 34th ACM Symposium on Theory of Computing*, May 2002, to appear. Also appears as *ECCC Technical Report TR01-094*, 2001, available from `http://www.eccc.uni-trier.de/eccc-local/Lists/TR-2001.html`.

[7] V. Kann, J. Lagergren and A. Panconesi. Approximability of maximum splitting of k-sets and some other APX-complete problems. *Information Processing Letters*, 58:105-110, 1996.

[8] H. Karloff and U. Zwick. A $(7/8-\varepsilon)$-approximation algorithm for MAX 3SAT? In *Proceedings of the 38th FOCS*, 1997, pp. 406-415.

[9] S. Khanna, M. Sudan and D. Williamson. A complete classification of the approximability of maximization problems derived from Boolean constraint satisfaction. *Proc. of the 29th STOC*, 1997.

[10] L. Lovász. Coverings and colorings of hypergraphs. *Proc. 4th Southeastern Conf. on Combinatorics, Graph Theory, and Computing*, pp. 3-12, Utilitas Mathematica Publishing, Winnipeg, 1973.

[11] E. Petrank. The hardness of approximation: Gap location. *Computational Complexity*, 4:133-157, 1994.

[12] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, 1998. Preliminary version in *Proc. of STOC'95.*

[13] G. Sorkin. Private Communication, July 2000.

[14] L. Trevisan, G. Sorkin, M. Sudan and D. Williamson. Gadgets, approximation and linear programming. *Proceedings of the 37th FOCS*, pp. 517-626, 1996.

[15] U. Zwick. Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, 1998.

[16] U. Zwick. Outward rotations: a tool for rounding solutions of semidefinite programming relaxations, with applications to MAX CUT and other problems. In *Proceedings of STOC'99.*