Electronic Colloquium on Computational Complexity, Report No. 49 (2003)

Approximation Hardness of Short Symmetric Instances of MAX-3SAT

Piotr Berman [*] Marek Karpinski [†] Alexander D. Scott [‡]

Abstract

We prove approximation hardness of *short symmetric* instances of MAX-3SAT in which each literal occurs *exactly twice*, and each clause is exactly of size 3. We display also an explicit approximation lower bound for that problem. The bound *two* on the number of occurrences of literals in symmetric MAX-3SAT is thus the *smallest* possible one which makes the instances hard to approximate.

1 Introduction

We define a *symmetric (balanced) MAX-(3,Bk)-SAT* instance of the *maximizing* MAX-3SAT problem as a set of clauses of size exactly 3, in which every literal occurs exactly k times. MAX-(3,k)-SAT stands for the set of *relaxed* (possibly unbalanced) instances of MAX-3SAT in which every variable occurs exactly k times and each clause is of size exactly 3. We will also denote by (3,Bk)-SAT and (3,k)-SAT the corresponding sets of formulas.

It was proven in [BKS03] that MAX-(3,4)-SAT is hard to approximate to within a certain constant. It was also shown that the *balanced* MAX-(3,B3)-SAT is hard to approximate [F98], [FLT02]. It remained an open question on whether, in fact, the balanced class MAX-(3,B2)-SAT remains hard to approximate. Because MAX-(3,4)-SAT is the smallest, with respect to the occurrence number, class of instances which are still inapproximable, the *balanced bound* 2 (B2), would be then the best possible.

In this paper we answer this question, and prove somewhat surprisingly that MAX-(3,B2)-SAT is, in fact, hard to approximate to within a certain constant. We

[*]Dept. of Computer Science and Engineering, Pennsylvania State University. Research done in part while visiting Dept. of Computer Science, University of Bonn. Work partially supported by NSF grant CCR-9700053 and NIH grant 9R01HG02238-12. E-mail berman@cse.psu.edu.

[†]Dept. of Computer Science, University of Bonn. Research done in part while visiting the Institute for Advanced Study, Princeton. Work partially supported by DFG grants, Max-Planck Research Prize, DIMACS, and IST grant 14036 (RAND-APX). E-mail marek@cs.uni-bonn.de.

[‡]Dept. of Mathematics, University College London, Gower Street, London WC1E 6BT. E-mail scott@math.ucl.ac.uk

ISSN 1433-8092

display also an explicit factor for the approximation hardness of that problem. The bound 2 for the number of occurrences of literals is thus the smallest bound for symmetric MAX-3SAT for which the approximation gap property is still NP-hard (see for the applications of regular and symmetric 3SAT gap properties towards other lower approximation bounds in [ALMSS98], [F98], and [FLT02]).

We note also, that, interestingly, a dual version of this balanced satisfiability problem leads to a certain natural problem studied in graph theory. Let C be the set of clauses and $V = \{v_1, \ldots, v_n\}$ the set of boolean variables. For each $v_i \in V$, let e_i be the pair of clauses in which v_i occurs without negation, and let f_i be the pair of clauses in which v occurs negated. Thus if we set v_i true then we satisfy both clauses in e_i and if we set v_i false we satisfy both clauses in f_i. Now consider the graph $G_{V,C}$ with vertex set C and edges $e_1, f_1, \ldots, e_n, f_n$. Finding a satisfying assignment for (V, C) is equivalent to choosing one edge from each pair $\{e_i, f_i\}$ such that the resulting subgraph of $G_{V,C}$ has no isolated vertices (or, equivalently, finding a spanning forest of $G_{V,C}$ with no isolated vertices and at most one edge from each pair). We remark that a problem of similar type, where the edges come in pairs but we instead attempt to choose one edge from each pair without creating a giant component, has been considered by Bohman, Frieze and Wormald [BFW03] (see also [BF01]).

We follow in this paper a line of [BKS03] of constructing efficient enforcers for the boolean variables; however, in our present setting we have to produce resulting *balanced* unsatisfiable (3,B2)-SAT formulas. In fact, at that time the existence of such balanced and unsatisfiable formulas was an open question in the area.

In Section 2 we give the first construction of balanced *enforces*, and a resulting balanced unsatifiable (3,B2)-SAT formula. In Section 3 we show how to transform the existence of balanced enforcers and unsatisfiable (3,B2)-SAT formulas into the NP-hardness result in exact setting. In Section 4 we prove our main result on approximation hardness of MAX-(3,2B)-SAT and give an explicit approximation lower bound and a gap property.

2 Balanced Enforcers

We refer to [BKS03] for a general background on a concept of small enforces of boolean variables. The constructions given in [BKS03] however do not give *balanced* enforcers of boolean variables, and existence of such balanced enforcers for (3,B2)-SAT formulas was at the time an open problem.

Here we give a construction of a balanced enforcer of literals, and display also the first unsatisfiable (3,B2)-SAT formula.

To make our constructions easier to follow, we denote a CNF formula by a matrix with each row listing literals of one clause. We use columns to help count the number of occurrences of variables.

For given literals l_0, l_1 and l_2, we construct the set $\mathcal{S}(l_0, l_1, l_2)$ of 5 clauses (or more precisely the conjunction of 5 clauses) as follows:

$$\mathcal{S}(l_0, l_1, l_2) \equiv \begin{array}{cccc} l_0 & \neg a & b & \\ l_1 & & \neg b & c \\ l_2 & a & & \neg c \\ & a & b & c \\ & \neg a & \neg b & \neg c \end{array}$$

for a, b, c new variables. We call $\mathcal{S}(l_0, l_1, l_2)$ the enforcer for $(l_0 \vee l_1 \vee l_2)$. It is easy to see that there are boolean values for a,b and c such that $\mathcal{S}(l_0, l_1, l_2)$ is *equivalent* to the clause $(l_0 \vee l_1 \vee l_2)$. The advantage of an enforcer $\mathcal{S}(l_0, l_1, l_2)$ over the clause $(l_0 \vee l_1 \vee l_2)$ lies in the fact each literal occurs in $\mathcal{S}(l_0, l_1, l_2)$ in a different clause, and can be used more than once independent on other literals. For a given boolean variable x we define the enforcer $x^{(3)}$ for x to be

$$x^{(3)} = \mathcal{S}(x, x, x),$$

and another enforcer

$$x^{(2)} = \mathcal{S}(x, y, y) \cup \mathcal{S}(x, \neg y, \neg y)$$

for a new variable y. In either case we need to create 5 clauses per one forced occurrence of x.

We now construct an *unsatisfiable* (3,B2)-SAT-formula f of 20 clauses and 15 variables as

$$f \equiv x^{(2)} \cup \neg x^{(2)}.$$

In this formula each variable occurs exactly twice in its negated and unnegated form. This yields the following lemma.

Lemma 1. *There exists an unsatisfiable (3,B2)-SAT formula f with 20 clauses and 15 variables.*

We notice also that if we relax the property of being balanced for a formula f, and construct it as

$$f \equiv x_0^{(3)} \cup x_1^{(3)} \cup x_2^{(3)} \cup \{\neg x_0 \vee \neg x_1 \vee \neg x_2\},$$

we get an unsatisfiable formula f with 16 clauses and 14 variables (an improvement over an unbalanced formula with 20 clauses and 15 variables of [BKS03]).

Lemma 2. *There exists an unsatisfiable (3,4)-SAT formula f with 16 clauses and 12 variables.*

We are going to use the constructions of the explicit enforcers of this section to obtain NP-hardness and approximation hardness results on balanced (3,B2)-SAT formulas.

3 NP-Hardness in Exact Setting

The technique of turning explicit balanced enforcers into NP-hardness result for (3,B2)-SAT formula is more evolved than the corresponding technique for unbalanced (3,4)-SAT formulas, cf. [BKS03]. We start here with the NAE-3SAT problem, of deciding for a given set of clauses C of size exactly equal to 3 whether there is an assignment making in each clause at least one literal true and at least one literal false. The problem NAE-3SAT is known to be NP-complete (c.f. [S78]).

We construct now in five stages a (3,2B)-SAT formula g.

In the first stage we replace each variable occurrence with a different variable. In the second stage, we replace a NOT-ALL-EQUAL clause NAE (l_1, l_2, l_3) by $(l_0 \vee l_1 \vee l_2) \wedge (\neg l_0 \vee \neg l_1 \vee \neg l_2)$; note that now each variable x occurs exactly twice, *once* as a literal x, and *once* as a literal $\neg x$.

In the third stage, we create a "wheel of implications": $x_0 \rightarrow x_1, x_1 \rightarrow x_2, \cdots, x_k \rightarrow x_0$ for the variables $x_0, \cdots, x_k$ which has replaced one original variable. In the fourth stage, we replace each implication $x \rightarrow y$ with a 3-clause $\neg x \vee y \vee \neg a$ for a a new variable. Notice that we use each new variable exactly twice. Finally, for each new variable a we add the enforcer $a^{(2)}$. The resulting (3,2B)-SAT formula g is satisfiable iff NAE-formula f is satisfiable. This proves the NP-completeness of (3,2B)-SAT.

Theorem 1. *The problem (3,B2)-SAT is NP-complete.*

□

4 Approximation Hardness

We prove now approximation hardness result on MAX-(3,2B)-SAT problem. The resulting lower approximation bound improves also on a lower bound of [BKS03] proven for more general problem of MAX-(3,4)-SAT.

Theorem 2. *For every $0 < \varepsilon < 1$, it is NP-hard to approximate MAX-(3,2B)-SAT to within an approximation ratio smaller than $(1016 - \varepsilon)/1015$.*

Proof. We will use Håstad's theorem that for every $0 < \varepsilon < 1/4$ it is NP-hard to distinguish between E3-LIN-2 instances in which one can satisfy at least $1 - \varepsilon$ of all equations from those when one can satisfy at most $1/2 + \varepsilon$ of all equations [H97]. We will refer also a general reader to the bounded occurrence techniques of [BK01] and [BK03].

The general strategy will be to translate a system $\mathcal{S}$ of n equations over $\mathbf{Z}_2$, each with 3 variables, in two stages. In the first, we replicate each equation k times, where k is "sufficiently large" ($k = n$ is sufficiently large, and the exact minimal sufficient value of k is not important). Afterwards we may assume that each variable in $\mathcal{S}$ occurs at least k times. Next, we will define a (3,2B)-SAT formula $B(\mathcal{S})$ with $508n$ clauses such that

1. Let $\mathcal{T}$ be the set of truth assignments for $B(\mathcal{S})$; we have a *normalization function* $\nu : \mathcal{T} \rightarrow \mathcal{T}$ such that $\nu(\vec{x})$ satisfies at least as many clauses as $\vec{x}$ for every truth assignment $\vec{x}$.
2. Let $\mathcal{V}$ be the set of value assignments to variables of $\mathcal{S}$, we have a bijection $\beta : \mathcal{V} \rightarrow \nu(\mathcal{T})$ such that if i is the number of equation that $\vec{x}$ does not satisfy, then $\beta(\vec{x})$ does not satisfy i clauses.

We construct $B(\mathcal{S})$ in stages. First, we replace each equation of $\mathcal{S}$ with the equivalent formula in disjunctive normal form, and in that formula, we replace occurrences of a variable, say x, with two new variables. E.g. $x + y + z = 1 \ \mathbf{mod}\ 2$ is transformed into

$$(x_1 \vee y_1 \vee z_1) \wedge (x_2 \vee \neg y_1 \vee \neg z_1) \wedge (\neg x_1 \vee y_2 \vee \neg z_1) \wedge (\neg x_2 \vee \neg y_2 \vee z_1).$$

Note that each new variable occurs once negated and once non-negated.

Secondly, for each pair of variables that replace the same occurrence of a variable in $\mathcal{S}$, we introduce 13 new variables and we "connect" them with the implication into a gadget described in Figure 1.

Before describing the way these gadgets are connected, we may show how a truth assignment can be normalized without decreasing the number of satisfied clauses. Suppose that a pair of occurrences of a variable is assigned different values. Then in its gadget we can change this assignment to either of the two values without changing the number of satisfied implications. Thus, if we have three pairs of occurrences of a variable from the same formula that replaced an equation, if even one of the pairs has two values assigned, we can change the assignment so each pair is assigned one value, the equation is satisfied and the number of satisfied implication is not smaller. From now on, we can view this pair of variables as one.

As a result, in a formula that replaced an equation we have all 4 clauses satisfied if the formula is satisfied, or 3 clauses are satisfied.

Now suppose that not all variables in a gadget are assigned the same value. We convert all variables to the majority value of the "solid dot" variables—we count 7 of them. If the minority consisted of 1 variable, we gain at least 1 implication among these variables, while loosing at most 1 clause outside the gadget. A short inspection shows that if the minority consisted of 2 variables, we gain at least 2 implications, and if it consisted of 3 variables, we gain at least 3 implications, so in no case we decrease the number of the satisfied clauses.

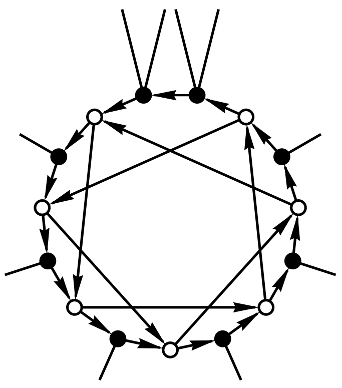

Figure 1:
Gadget for the copies of a variable. Arrows describe implications. Non-arrows correspond to the occurrences in the clauses that are not the part of this gadget: the two copies have two occurrence each in the replacement of an equation, six other gadget variables participate in one equivalence each—a pair of implications—with a similar variable of another gadget.

Next, given a variable of $\mathcal{S}$ with m occurrences, we get m gadgets with $6m$ variables that should be connected with others using equivalences (pairs of implications); we connect them using a random matching; by Bóllobas [B88] the resulting multigraph of gadgets has, with high probability, isoperimetric number equal at least 1 (this probability grows with the number of nodes, thus we assured that we have at least k nodes/gadgets). This in turn means that if a minority of i gadgets has a truth value different from the majority, we can convert all these gadgets to the majority value and at least i of the equivalences will become true. Thus this is a *good* normalization even if all i equations connected with these gadgets become false.

To finish, we group implications into "consecutive pairs", i.e. pairs of the form $x \to y \wedge y \to z$. In such a pair at least one implication is true. Equivalences clearly form such pairs, and inside a gadget, implications form an Eulerian graph, so they can form a single chain of 22 implications. A consecutive pair of implications can be converted to a pair of 3-clauses by adding a new variable: $x \to y \wedge y \to z$ converts to $(\neg x \vee y \vee \neg a) \wedge (\neg y \vee z \vee \neg a)$; to this we add the enforcer $a^{(2)}$. If one or more clause in the enforcer is false, we normalize the assignment so that all of them are true, and as a result we loose the truth of only one implication.

One can see that an equation of $\mathcal{S}$ was replaced with 4 clauses of its normal form, these clauses have occurrences of 3 pairs of variables, and for each pair we have build a gadget. Inside a gadget we have 22 implications, and a gadget participates in 6 equivalences with other gadgets, so we can say that each gadget has 28 implications.

For each implication we use 6 clauses (the implication itself and a half of an enforcer). Thus $B(\mathcal{S})$ replaces an equation with $4 + 3(28 \times 6) = 508$ clauses. In the process, a variable is replaced with a set of variables, but the normalization makes all these variables equivalent and this defines an obvious bijection between value assignments for $\mathcal{S}$, and the normalized truth assignments for $B(\mathcal{S})$. For a normalized truth assignment all clauses in the enforcers and gadgets are true; if an equation of $\mathcal{S}$ is satisfied, all 4 clauses of its normal form are true, otherwise all but 1 are true.

Now by result of Håstad [H97], it is NP-hard to distinguish between the *good* systems of $2n$ equations in which at least $(2-\epsilon)n$ equations can be satisfied, and the *bad* systems in which at most $(1+\epsilon)n$ equations can be satisfied. For a system $\mathcal{S}$ of $2n$ equation, $B(\mathcal{S})$, is a $(3, B2)$-SAT formula with $2 \times 508n$ clauses. If $\mathcal{S}$ is a *good* system, we can satisfy at least $(1016 - \epsilon)n$ clauses of $B(\mathcal{S})$, and if $\mathcal{S}$ is a *bad* system we can satisfy at most $(1015+\epsilon)n$ clauses of $B(\mathcal{S})$. This completes the proof.

$\square$

Acknowledgments. We thank Sanjeev Arora, Alan Frieze and Mark Jerrum for stimulating remarks and discussions.

References

[ALMSS98] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, *Proof Verification and the Hardness of Approximation Problems*, Journal of the ACM **45(3)** (1998) pp. 501 - 555.

[BF01] T. Bohman and A. M. Frieze, *Avoiding a Giant Component*, Random Structures and Algorithms **19** (2001), pp. 75–85.

[BFW03] T. Bohman, A. M. Frieze, and N. C. Wormald, *Avoiding a Giant Component II*, to appear.

[B88] B. Bollobás, *The Isoperimetric Number of Random Graphs*, Europ. J. Combinatorics **9** (1988), pp. 241 - 244.

[BK03] P. Berman and M. Karpinski, *Improved Approximation Lower Bounds on Small Occurrence Optimization*, Elec. Coll. on Comp. Compl., ECCC TR03-008 (2003).

[BK01] P. Berman and M. Karpinski, *Efficient Amplifiers and Bounded Degree Optimization*, Elec. Coll. on Comp. Compl., ECCC TR01-053 (2001).

[BKS03] P. Berman, M. Karpinski and A. D. Scott, *Approximation Hardness and Satisfiability of Bounded Occurrence Instances of SAT*, Elec. Coll. on Comp. Compl., ECCC TR03-022 (2003).

[F98] U. Feige, *A Threshold of* $\ln n$ *for Approximating Set Cover*, *Journal of ACM*, **45(4)** (1998), pp.634–652.

[FLT02] U. Feige, L. Lovász and P. Tetali, *Approximating MIN-SUM Set Cover*, Proc. 5th APPROX (2002), LNCS Vol. 2462 (2002), pp. 94-107.

[H97] J. Håstad, *Some Optimal Inapproximability Results*, Proc. 29th ACM STOC, 1-10, 1997.

[T84] C. A. Tovey, *A Simplified Satisfiability Problem*, Discr. Appl. Math. **8** (1984), pp. 85 - 89.

[S78] T. J. Schaefer, *The Complexity of Satisfiability Problems*, Proc. 10th ACM STOC (1978), pp. 216 - 226.