Approximability of Maximum Splitting of k-sets and some other APX-complete Problems

Viggo Kann* Jens Lagergren* Alessandro Panconesi†

February 27, 1996

Key words: Approximation; Algorithms; Combinatorial problems; Computational complexity

1 Introduction

Recent breakthroughs in the theory of approximation algorithms introduced new powerful tools to obtain both hardness results and better approximation algorithms for many combinatorial problems (see [4, 7]). In particular, the lower bound techniques have been considerably fine tuned, and for several problems it is now possible to show lower bounds for the approximation which are close to the best known upper bounds [4, 5, 12]. On the algorithmic front, Goemans and Williamson, building on earlier work, have recently introduced a new approach based on probabilistic algorithms and mathematical programming. This paradigm has delivered the first substantial improvements for several important combinatorial problems after many years of essentially unsuccessful efforts [7, 10].

In this paper we shrink the gap for the MAX 3-SET SPLITTING problem and its generalization MAX k-SET SPLITTING from both ends, finding new lower bounds and new upper bounds. In the MAX k-SET SPLITTING problem, the instance consists of subsets of size k of a finite set of elements. The problem is to partition the elements into two parts, such that as many subsets as possible will be split, i.e. contain elements from both parts. The problem was shown to be NP-complete by Lovász [11]. Recently it was shown to be APX-complete [15]. We also consider another generalization of MAX 3-SET SPLITTING, namely MAX NAE 3SAT, that was shown to be APX-complete in [14].

First, we give an approximation algorithm for MAX 3-SET SPLITTING using semi-definite programming. Our algorithm is guaranteed to compute (roughly) a 1.14 approximation.

*Department of Numerical Analysis and Computing Science, Royal Institute of Technology, S-100 44 Stockholm, email viggo@nada.kth.se, jensl@nada.kth.se. Supported by TFR and NFR.

†Fachbereich Informatik, Freie Universität, Takustr. 9, 14195 Berlin; this work was done when the author was visiting the Royal Institute of Technology with funds kindly provided by SICS in Stockholm. The author gratefully acknowledges the help, hospitality and freedom provided by both institutions.

Second, we use a technique of [4] to prove a lower bound 1.0132 on the approximability of Max 3-Set Splitting. The technique consists of implementing certain tests (performed by a verifier in an interactive proof) by so called gadgets. Finally, we give a lower-bound for Max k-Set Splitting using approximation preserving reductions. This bound is not too far from an upper bound given by a well known heuristic.

2 Definitions

Solving a maximization problem F given the input instance x means finding a solution y such that the value of the objective function $m_F(x,y)$ is maximum. Let $opt_F(x)$ denote this maximum value of m_F.

Approximating a maximization problem F given the input x means finding some feasible solution y. How good the approximation is depends on the relation between $m_F(x,y)$ and $opt_F(x)$. There are two equivalent measures of this relation: the performance ratio and the relative error. The *performance ratio* of a solution y with respect to the optimum of a maximization problem F is defined as $R_F(x,y) = opt_F(x)/m_F(x,y)$. The *relative error* is defined as

$$\mathcal{E}_F(x,y) = \frac{opt_F(x) - m_F(x,y)}{opt_F(x)} = 1 - \frac{1}{R_F(x,y)}.$$

An optimization problem F *is approximable within* c for a constant c if there exists a polynomial time algorithm A such that for all input instances x, $A(x)$ is a feasible solution and $R_F(x, A(x)) \le c$. A problem is *approximable* if it is approximable within some constant. Apx is the important complexity class consisting of all approximable NP optimization problems.

Although various reductions preserving approximability within constants have been proposed (see [6]), the L-reduction defined in [14] is perhaps the easiest to use. Given two NP optimization problems F and G and a polynomial time transformation f from instances of F to instances of G, we say that f is an *L-reduction* if there are positive constants α and β such that for every instance x of F

i) $opt_G(f(x)) \le \alpha \cdot opt_F(x)$,

ii) for every feasible solution y of $f(x)$ with objective value $m_G(f(x),y) = c_2$ we can in polynomial time find a solution y' of x with $m_F(x,y') = c_1$ such that $|opt_F(x) - c_1| \le \beta\,|opt_G(f(x)) - c_2|$.

The composition of L-reductions is an L-reduction. If F L-reduces to G with constants α and β and there is a polynomial time approximation algorithm for G with worst-case relative error ε, then there is a polynomial time approximation algorithm for F with worst-case relative error $\alpha\beta\varepsilon$ [14]. Conversely, if δ is a lower-bound for the approximation of F then, $\delta(\alpha\beta)^{-1}$ is a lower-bound for G. An L-reduction from F to G therefore can be used both to show hardness of approximability (for G) and to find a new approximation algorithm (for F).

Using L-reductions (or reductions similar to them) one can show that a problem F is Apx-complete, i.e., F is approximable within c for some c and every approximable problem can be L-reduced to F. In this paper we will consider the following APX-complete problems.

- MAX k-SET SPLITTING: Given a set of subsets where each subset consists of *exactly* k elements, find a partition of the elements into two parts so that a maximum number of subsets are split, i.e., contain elements from both parts.
- MAX k-SAT: Given a set of CNF clauses where each clause consists of *exactly* k literals, find a truth assignment that satisfies a maximum number of clauses.
- MAX NAE 3SAT: Given a set of CNF clauses where each clause consists of at most three literals, find a truth assignment that makes a maximum number of clauses not-all-equal, i.e., maximizes the number of clauses that contain both true and false literals.

3 Approximation of MAX 3-SET SPLITTING

3.1 Upper bound of approximating MAX 3-SET SPLITTING

We use semi-definite programming to approximate MAX 3-SET SPLITTING. The algorithm is very similar to the MAX CUT approximation algorithm by Goemans and Williamson [7, 8]. We refer to this article for the details of the method.

Given 3-sets $S_1 \dots S_n$ where $S_i = \{x_{i_1}, x_{i_2}, x_{i_3}\}$. For every element x_j, we introduce a variable y_j taking values over $\{-1, 1\}$. The values of the y_j variables corresponds to the partition. The problem of finding the maximum split can be expressed as

$$\begin{cases} \max \sum_{i=1}^{n} \frac{3 - y_{i_1} \cdot y_{i_2} - y_{i_1} \cdot y_{i_3} - y_{i_2} \cdot y_{i_3}}{4} \\ y_i \in \{-1, 1\} \end{cases}$$

because in every 3-set that is *not* split we have $y_{i_1} = y_{i_2} = y_{i_3}$ and therefore $3 - y_{i_1} \cdot y_{i_2} - y_{i_1} \cdot y_{i_3} - y_{i_2} \cdot y_{i_3} = 0$, and in every 3-set that is split we have $3 - y_{i_1} \cdot y_{i_2} - y_{i_1} \cdot y_{i_3} - y_{i_2} \cdot y_{i_3} = 4$. The objective function can also be written

$$\max \sum_{i=1}^{n} \frac{1 - y_{i_1} \cdot y_{i_2}}{4} + \frac{1 - y_{i_1} \cdot y_{i_3}}{4} + \frac{1 - y_{i_2} \cdot y_{i_3}}{4}$$

which means that the problem is of a general form that is always approximable within 1.14 (see Section 2.1 in [7]).

For completeness we will describe how the approximation result can be obtained in this particular case.

We relax the problem to a semidefinite programming problem by relaxing the $\{-1, 1\}$ variables y_j to m-dimensional unit vectors v_j (and the multiplications to scalar products), where m is the number of variables in the problem. An optimum solution of the relaxed problem can in polynomial time be found to within any prespecified accuracy [2]. We then find a solution of the original problem by taking a random hyperplane through the origin (defined by a random vector r) and let all the elements on one side of the hyperplane be on one side in the partition. Thus a set is split if it contains elements corresponding to vectors on both sides of the hyperplane.

The probability that a 3-set is split depends on the angles between the v-vectors of the included elements. Let $\theta_{i,j}$ be the smallest angle between the vectors v_i and v_j. Consider the 3-set $\{v_1, v_2, v_3\}$, and observe that it is split if and only if exactly two of the sets $\{v_1, v_2\}$, $\{v_1, v_3\}$ and $\{v_2, v_3\}$ are split. The probability that a 2-set $\{v_i, v_j\}$ is split is equal to $\theta_{i,j}/\pi$. A simple counting argument thus gives us

$$\begin{aligned}
&\Pr[\{v_1, v_2, v_3\} \text{ is split}] = \\
&= \frac{\Pr[\{v_1, v_2\} \text{ is split}] + \Pr[\{v_1, v_3\} \text{ is split}] + \Pr[\{v_2, v_3\} \text{ is split}]}{2} = \\
&= \frac{\theta_{1,2} + \theta_{1,3} + \theta_{2,3}}{2\pi}.
\end{aligned}$$

Since $\cos\theta_{i,j} = v_i \cdot v_j$ it is not hard to verify, using calculus, that the inequality

$$\begin{aligned}
\frac{\theta_{1,2} + \theta_{1,3} + \theta_{2,3}}{2\pi} &\geq \alpha \frac{3 - \cos\theta_{1,2} - \cos\theta_{1,3} - \cos\theta_{2,3}}{4} \\
&= \alpha \frac{3 - v_1 \cdot v_2 - v_1 \cdot v_3 - v_2 \cdot v_3}{4}
\end{aligned}$$

is satisfied for $\alpha = 0.878$ for any allowed combination of $\theta_{1,2}$, $\theta_{1,3}$ and $\theta_{2,3}$. This is the same value of α as obtained for Max Cut in [7].

Due to the linearity of expectation and the fact that the optimum solution of the relaxed problem must be at least as large as the optimum solution of Max 3-Set Splitting, we have that the expected number of split sets in our algorithm is

$$\begin{aligned}
\text{E[number of split sets]} &= \sum_{i=1}^{n} \Pr[\text{set } S_i \text{ is split}] \\
&\geq \alpha \cdot \sum_{i,j} \frac{3 - v_1 \cdot v_2 - v_1 \cdot v_3 - v_2 \cdot v_3}{4} \\
&\geq \alpha \cdot opt_{\text{Max 3-Set Splitting}}.
\end{aligned}$$

The algorithm can be derandomized using the methods in [13]. Hence Max 3-Set Splitting is approximable within $1/0.878 \approx 1.14$.

The algorithm can also be used if the input consists both of 2-sets and 3-sets. We just have to change each 2-set's contribution to the objective function of the semi-definite programming problem to

$$\frac{1 - v_{i_1} \cdot v_{i_2}}{2}.$$

The approximability is the same for this case. If there are only 2-sets the problem is the same as Max Cut.

In fact we can use the algorithm also to approximate Max NAE 3Sat. In this case, for each variable in the input we have one vector v_i, and if the variable occurs negated in the clause we use $-v_i$ instead of v_i in the objective function.

3.2 Lower bound of approximating Max 3-Set Splitting

Bellare, Goldreich and Sudan have developed an interactive proof system suitable for finding lower bounds of the approximability of Apx-complete problems like Max 3-Sat [4]. In order to find a lower bound for a problem F it is suffices to encode the bits in the proof in F in some way, and then construct gadgets that perform two *atomic tests* on subsets of these bits. The outcome of the tests must correspond to the objective value of the instance of F.

The first test is the *parity check* Pc, which returns the parity of three boolean variables, or equivalently, checks if

$$a + b = c \pmod 2.$$

The second test is called the *respecting monomial basis check* Rmbc, which checks if

$$ab = c + d \pmod 2.$$

Following [4] we say that a gadget for one of the tests is an (α, β)-gadget if the gadget contributes α to the objective value when the test succeeds and contributes at most $\alpha - \beta$ when the test fails. If the atomic tests can be expressed in the maximization problem F with a Pc(α_1, β_1)-gadget and an Rmbc(α_2, β_2)-gadget, then, unless P=NP, F is not approximable within

$$1 + \frac{1 - S}{Q\alpha_1/\beta_1 + (1 - Q)\alpha_2/\beta_2 - (1 - S)}$$

where $S = 0.86337$ and $Q = 0.63566$. See the paper by Bellare, Goldreich and Sudan for the proof of this result and more information about the method [4].

We will now use this method to construct gadgets for Max 3-Set Splitting. The ground set of the problem instance will consist of one element for each bit in the proof, one special element T, and one auxiliary element for each gadget. In a solution to the set splitting problem the elements will be partitioned into two parts. If the element corresponding to a bit b_i is in the same part as T we will set $b_i = 1$, and if it is in the other part we will set $b_i = 0$. In this way we have encoded the values of all bits in the proof.

In the parity test gadget we use three elements a, b and c corresponding to bits in the proof, the special element T, and an auxiliary element x that does not occur in any other gadget. The gadget will consist of all 3-sets over the elements a, b, c, and T (abc, abT, acT, bcT), together with 4 copies of each of the 3-sets containing x $(abx, acx, aTx, bcx, bTx, cTx)$. (Equivalently one can think of each set having weight 4.) Let S denote the set of all elements and $\{T_i\}$ the set of all (3-element) sets so introduced.

In a partition $(A, S-A)$ of S where three out of $\{a, b, c, T\}$ are put in A (this corresponds precisely to all cases where the parity test is successful), we will split 27 of the sets T_i by putting x in the second part. In all other partitions it is not possible to split more than 24 sets. Therefore this is a Pc$(27, 3)$-gadget.

In the Rmbc gadget we use four elements a, b, c and d corresponding to bits, the special element T and an auxiliary element x that does not occur in any other gadget. The gadget will consist of one copy of the 3-sets $abx, acd, axT, bcd, bxT, cdT$, two copies of the set abT, three copies of the set cdx, and five copies of the sets $acx, adx, bcx, bdx, cxT, dxT$. It is easy to verify that in every case where the test is successful (i.e. when $ab = c + d \pmod 2$) it is possible to split 39 sets. In the other cases it is never possible to split more than 36 sets. Thus we have an Rmbc$(39, 3)$-gadget.

The formula above now gives a lower bound of 1.0132 for the performance ratio of Max 3-Set Splitting.

Remark: by using L-reductions we can map Max 3-Sat into Max 3-Set Splitting. This approach yields a lower-bound for Max 3-Set Splitting of 1.003, not too far from the upper bound of 1.14 and the above lower-bound.

4 Approximation of Max k-Set Splitting

4.1 Upper bound of approximating Max k-Set Splitting

Assume the input consists of n sets of exactly k elements for some $k \geq 4$. A straightforward randomized heuristics finds a partition $(A, S-A)$ of the element set S that splits $(1 - 2/2^k)n$ sets: each element is put in A or $S - A$ with probability 1/2 uniformly and independently at random. This algorithm can be derandomized using the method of conditional probabilities, see [3]. Since the optimum number of split sets is at most n the algorithm approximates Max k-Set Splitting within $1/(1 - 2^{1-k})$. The algorithm also works if the input consists of sets of size *at least* k. The same upper bound was obtained by Alimonti via non-oblivious local search [1].

4.2 Lower bound of approximating Max k-Set Splitting

In this section we will show that the upper-bound given by the naive randomized heuristic cannot be improved by too much (unless P=NP). We will show a lower-bound for the problem by L-reducing Max 3-Set Splitting to Max k-Set Splitting. Start with n sets of size 3. We add qk new elements $z_1, \ldots, z_{qk}$ for some even integer q to be specified later. For each 3-set S_i, we construct K k-sets $S_{i,1}, \ldots, S_{i,K}$ where

$$K = \binom{qk}{k-3}.$$

Each k-set is the union of S_i and $k - 3$ of the z-elements. For each i, every combination of $k - 3$ z-elements appears in one of the sets $S_{i,j}$.

The only case where a k-set is *not* split in a solution is when all the elements in S_i are on the same side as all z-elements that appear in the set.

For each i, the number of such sets is $\binom{a}{k-3}$ where a is the number of z-variables on the same side as the elements in S_i.

We also add n copies of all possible k-sets of z-variables. In the optimum solution to the MAX k-SET SPLITTING instance, the z-elements will be split into two equal size parts. The reason for this is that if the z elements were split into parts of size $a+\delta$ and a, where $\delta \geq 2$, and we move one z element from the first part to the second, we would gain $n\binom{a+\delta-1}{k-1}$ new split k-sets (among the sets consisting of only z-elements), while losing at most

$$n\binom{a}{k-1} + n\binom{a}{k-4} < n\binom{a+1}{k-1} \leq n\binom{a+\delta-1}{k-1}.$$

If the z-elements are split into two equal parts of size $kq/2$, the number of split sets in the optimum solution will be

$$n\left[\binom{qk}{k} - 2\binom{qk/2}{k}\right] + n\binom{qk}{k-3} + (s-n)\binom{qk/2}{k-3}$$

where s is the number of split sets in the MAX 3-SET SPLITTING problem instance. This means that

$$\alpha = \frac{opt_{\text{MAX } k\text{-SET SPLITTING}}}{opt_{\text{MAX 3-SET SPLITTING}}} \leq \frac{2n}{(1-2^{1-3})n}\binom{qk}{k} = \frac{8}{3}\binom{qk}{k}$$

and that

$$\beta = \frac{1}{\binom{qk/2}{k-3}}.$$

Setting $q = k$ (or $k+1$ if k is odd) the product $\alpha\beta$ asymptotically approaches $(\sqrt{e} + O(1/k))k^3 2^{k-3}$. We conclude that for sufficiently large k MAX k-SET SPLITTING is not approximable within $1 + 0.063 \cdot k^{-3}2^{-k}$.

The same method can be used to show a lower bound of MAX k-SAT, by reducing from MAX 3-SAT. We extend each 3-clause by $k-3$ new literals in all 2^{k-3} possible ways. The only way for a clause *not* to be satisfied is when the original 3-clause is not satisfied *at the same time* as all the added literals are false. The number of not satisfied k-clauses will be same as the number of not satisfied 3-clauses in the original problem.

This means that

$$opt_{\text{MAX } k\text{-SAT}} = n(2^{k-3}-1) + opt_{\text{MAX 3-SAT}} < n2^{k-3} \leq \frac{8}{7}2^{k-3} opt_{\text{MAX 3-SAT}}$$

where n is the number of clauses in the MAX 3-SAT instance. This is an L-reduction with $\beta = 1$ and $\alpha = \frac{8}{7}2^{k-3}$. The resulting lower bound of the relative error is $7 \cdot 2^{-k}$ times the lower bound of the relative error of MAX 3-SAT. This should be compared to the approximation algorithm of Johnson [9], that obtains a relative error of about 2^{-k}.

5 Summary

The following table summarizes the upper and lower bounds of the approximability of the problems treated in the paper.

	upper bound of relative error	lower bound of relative error
Max 3-Set Splitting	0.122	0.013
Max k-Set Splitting	$2 \cdot 2^{-k}$	$0.063 \cdot k^{-3} 2^{-k}$
Max NAE 3Sat	0.122	0.013
Max k-Sat	2^{-k} [9]	$0.18 \cdot 2^{-k}$

References

[1] P. Alimonti. Non-oblivious local search for graph and hypergraph coloring problems. In *Proc. of 21st International Workshop on Graph-Theoretic Concepts in Comp. Sci.*, pages 167–180. Springer-Verlag, 1995. Lecture Notes in Comput. Sci. 1017.

[2] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optimization*, 5:13–51, 1995.

[3] N. Alon and J. H. Spencer. *The Probabilistic Method.* Interscience series in discrete mathematics and optimization. Wiley, New York, 1991.

[4] M. Bellare, O. Goldreich, and M. Sudan. Free bits, PCPs and non-approximability – towards tight results. In *Proc. of 36th Ann. IEEE Symp. on Foundations of Comput. Sci.*, pages 422–431. IEEE Computer Society, 1995. Full version available as ECCC Report TR95-024, `http://www.eccc.uni-trier.de/pub/eccc/reports/1995/TR95-024/index.html`.

[5] V. Chvátal. A greedy heuristic for the set covering problem. *Math. Oper. Res.*, 4:233–235, 1979.

[6] P. Crescenzi, V. Kann, R. Silvestri, and L. Trevisan. Structure in approximation classes. In *Proc. of 1st Ann. Int. Conf. on Computing and Combinatorics*, pages 539–548. Springer-Verlag, 1995. Lecture Notes in Comput. Sci. 959.

[7] M. X. Goemans and D. P. Williamson. .878-approximation algorithms for MAX CUT and MAX 2SAT. In *Proc. Twenty sixth Ann. ACM Symp. on Theory of Comp.*, pages 422–431. ACM, 1994.

[8] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 1996. To appear.

[9] D. S. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.*, 9:256–278, 1974.

[10] D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. In *Proc. of 35th Ann. IEEE Symp. on Foundations of Comput. Sci.*, pages 2–13. IEEE Computer Society, 1994.

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

[12] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41:960–981, 1994.

[13] S. Mahajan and H. Ramesh. Derandomizing semidefinite programming based approximation algorithms. In *Proc. of 36th Ann. IEEE Symp. on Foundations of Comput. Sci.*, pages 162–169. IEEE Computer Society, 1995.

[14] C. H. Papadimitriou and M. Yannakakis. Optimization, approximation, and complexity classes. *J. Comput. System Sci.*, 43:425–440, 1991.

[15] E. Petrank. The hardness of approximation: gap location. *Computational Complexity*, 4:133–157, 1994.