Approximation Hardness of Bounded Degree MIN-CSP and MIN-BISECTION

Piotr Berman *   Marek Karpinski †

Abstract

We consider bounded occurrence (degree) instances of a minimum constraint satisfaction problem MIN-LIN2 and a MIN-BISECTION problem for graphs. MIN-LIN2 is an optimization problem for a given system of linear equations mod 2 to construct a solution that satisfies the minimum number of them. E3-OCC-MIN-E3-LIN2 is the bounded occurrence (degree) problem restricted as follows: each equation has exactly 3 variables and each variable occurs in exactly 3 equations. Clearly, MIN-LIN2 is equivalent to another well known problem, the Nearest Codeword problem, and E3-OCC-MIN-E3-LIN2 to its bounded occurrence version. MIN-BISECTION is a problem of finding a minimum bisection of a graph, while 3-MIN-BISECTION is the MIN-BISECTION problem restricted to 3-regular graphs only. We show that, somewhat surprisingly, these two restricted problems are exactly as hard to approximate as their general versions. In particular, an approximation ratio lower bound for E3-OCC-MIN-E3-LIN2 (bounded 3-occurrence 3-ary Nearest Codeword problem) is equal to MIN-LIN2 (Nearest Codeword problem) lower bound $n^{\Omega(1)/\log\log n}$. Moreover, an existence of a constant factor approximation ratio (or a PTAS) for 3-MIN-BISECTION entails existence of a constant approximation ratio (or a PTAS) for the general MIN-BISECTION.

*Dept. of Computer Science, University of Bonn, 53117 Bonn, visiting from Pensylvania State University. Partially supported by DFG grant Bo 56/157-1 and NSF grant CCR-9700053. Email berman@cse.psu.edu

†Dept. of Computer Science, University of Bonn, 53117 Bonn. Research partially done while visiting Dept. of Computer Science, Yale University. Supported in part by DFG grants KA 673/4-1 and Bo 56/157, DIMACS, and IST grant 14036 (RAND-APX). Email marek@cs.uni-bonn.de

1 Introduction

In this note we study the approximation hardness of bounded occurence minimum constraint satisfaction problem E3-OCC-MIN-E3-LIN2 and 3-MIN-BISECTION. MIN-LIN2 problem has as its input a system of linear equations over $GF[2]$ and the minimized objective function is the number of equations satisfied by a solution; -Ek- denotes its restriction to equations with exactly k variables, and Ek-OCC- denotes another restriction, namely that each variable occurs exactly k times in a system. (Infix and prefix k denote more loose restrictions, to equations with at most k variables and to at most k occurrences of each variable respectively.) The MIN-LIN2 problem or equivalently Nearest Codeword problem is known to be exceedingly hard to approximate. It is known to be NP-hard to approximate to within a factor $n^{\Omega(1)/\log\log n}$ (cf. [DKS98], [DKRS00]). Recently, also the first nonlinear approximation ratio $O(n/logn)$ algorithm has been designed for that problem [BK01], see also [BFK00]. MIN-BISECTION problem has as its input an undirected graph, say with $2n$ nodes, a solution is a subset S of nodes with n elements, and the size of a cut $|Cut(S)|$ (i.e. the number of edges from S to its complement) is its minimizing objective function; prefix k- denotes its restriction to k-regular graphs.

Our results are tight in the following sense: 2-OCC-MIN-LIN2 can be solved in polynomial time, and MIN-2-LIN2 is much easier to approximate than MIN-LIN2 (cf. [KST97], [DKS98], [DKRS00]), while E3-OCC-MIN-E3-LIN2 is as hard to approximate as MIN-LIN-2 itself, i.e. it is NP-hard to approximate to within $n^{\Omega(1)/\log\log n}$. Similarly, 2-MIN-BISECTION can be solved in polynomial time while 3-MIN-BISECTION is as hard to approximate as the general MIN-BISECTION.

In what follows we assume some basic familiarity with [BK99].

2 MIN-LIN2 Problem

2.1 Our terminology

For a system of linear equations S over $GF[2]$, we denote the set of variables with $V(S)$, the total number of occurrences of variables in S (equivalently, the number of non-zero coefficients) with $size(S)$, the set of assignments of values to variables of S with $AV(S)$. For $a \in AV(S)$ the number of equations of S that a satisfies is denoted with $sat(a,S)$. As mentioned before, $sat(a,S)$ is the minimized objective function.

A reduction of MIN-LIN-2 to 3-OCC-MIN-E3-LIN-2 will be described via the following triple of functions:

- an instance transformation f such that $f(S)$ is a system in which each variable occurs exactly 3 times and $size(f(S)) = O(size(S))$;

- a solution normalization g such that for $a \in AV(S)$ we have $g(a) \in AV(S)$ and $|sat(g(a), f(S))| \leq |sat(a, f(S))|$,
- a bijection h between $AV(S)$ and $g(AV(f(S)))$ such that $|sat(a, S)| = |sat(h(a), f(S))|$.

The above description of a reduction directly relates to the standard definition of aproximation preserving reductions. The implied solution transformation is $h^{-1} \circ g$, its desired properties follow immediately from the properties of a normalization and "equivalence bijection".

2.2 Consistency gadgets

In approximation preserving reductions there exist a number of ways to replace a variable with a set of variables and auxiliary equations so that each element of the set occurs exactly three times. We have explored one of such constructions in [BK99] in a context of the maximization problems. In the reductions described in that paper the objective function is diluted, because part of the score assures that we can normalize each solution in such a way that the new set of variables correctly replaces a single variable. The same construction yields no such dilution in our minimization problems, because a normalized solution satisfies none of the auxiliary equations.

However, we describe here a different construction, which can be computed deterministically and the size of the new formula is a linear function of the size of the original formula. The construction of our reduction depends on the existence of special graphs which we call consistency gadgets.

In an undirected graph (V, E), we define $Cut(U) = \{e \in E : \ e \cap U \neq \emptyset \text{ and } e \cap (V - U) \neq \emptyset\}$. A consistency gadget $CG(K)$ is a graph with the following properties:

- $K \subset V_K$ where V_K is the set of nodes of $CG(K)$;
- if $A \subset V_K$ and $|A \cap K| \leq |K|/2$, then $|Cut(A) \geq |A \cap K|$;
- each node in K has 2 neighbors and each node in $V_K - K$ has 3 neighbors. of size $k = |K|$.

As described in Arora and Lund [AL95] (cf. also [PY91]), we could use a family of graphs that we may call *strong expanders*. The graph (V, E) is a strong expander if $|U| \leq |V|/2$ implies $|Cut(U) \geq |U|$ for each $U \subset V$. Lubotzky *et al.* [LPS88] showed that a family of 14-regular strong expanders is constructible in polynomial time.

Arora and Lund suggest that one can obtain $CG(K)$ by constructing a 14-regular strong expander with node set K, and then by replacing each node with a cycle of 15 nodes: one node being the element of K and the other node being the terminal of the 14 connections to the other cycles.

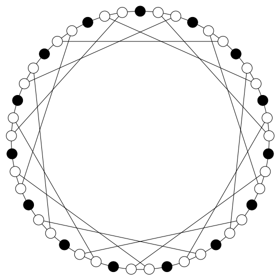

Figure 1: An example of $CG(S_{15})$; element of S_{15} are black.

One can see that this construction is not quite sufficient for our purposes, because it is conceivable that the new graph contains a set that contradicts the $CG(K)$ property, and which properly intersects some of the cycles. Aurora and Lund use a weaker notion of consistency, but in this paper we wish to establish an exact relationship between the tresholds of approximability.

To remedy this problem, we can modify the construction slightly. In particular, let S_{15} be a set of 15 elements, we will replace each node of the 14-regular strong expander with a copy of a $CG(S_{15})$ which is shown in Fig. 1.

2.3 Reduction to E3-OCC-MIN-3-LIN2

Recall that in E3-OCC-MIN-3-LIN2 we also allow equations in which exactly 2 variables appear.

Consider a system of equations S. Without loss of generality, we assume that each variable of S occurs at least 2 times, otherwise we reduce the problem by removing the equations in which a variable has its sole occurence.

Suppose that S contains an equation with more than 3 variables, $\xi + \zeta = b$, where ξ is a sum of 2 variables and ζ is a sum of at least 2 variables. We obtain $f(S)$ by replacing in S this equation with two equations, $\xi + x = 1$ and $x + \zeta = b$, where x is a new variable. We define the normalizing transformation $g(a)$ as follows: $g(a)(y) = a(y)$ for every variable y other than x, and $g(a)(x) = a(\xi)$ where $a(\xi)$ is the value of the linear expression ξ under assignment of values a. This assures that $g(a)$ does not satisfy equation $\xi + x = 1$. In turn, $g(a)$ satisfies $\xi + x = b$ if and only if a satisfies $\xi + \zeta = b$. The bijection h is defined by the same formula as g.

Now we assume that each equation consists of at most 3 variables. Consider a variable x that occurs in $k > 3$ equations. Let K be the set of these equations. We form the graph $CG_3(K)$ with node set V_K, for each node in V_K we introduce a new variable with the same name (so now V_K becomes a set of variables) and for each edge $\{u, g\}$ we introduce an equation $u+v = 1$. Then we replace an occurrence of x in equation e with the variable that replaced e (as a node of V_K). This is the instance transformation (obviously, we may need a sequence of such transformations to achieve our goal).

To define $h(a)$ for an assignment of values a of the new instance, we find value b that is assumed by at least half of the variables in K; then for every $x \in V_K$ we set $h(a)(x) = b$. Suppose that this normalization changed the values of l variables from K . Then up to l "old" equations may become satisfied. However, none of the equations that replaced the edges of $CG_3(K)$ is satisfied now. Because $2l < |K|$, we have l edge disjoint paths from the l variable/nodes that change the value to nodes that did not, in turn, on each path we have at least one edge corresponding to an equation that ceased to be satisfied, thus at least l "new" equations ceased to be satisfied. consequently, $sat(a, f(S))$ did not increase.

The bijection transformation simply $h(a)$ assigns $a(x)$ to each variable in V_K.

To summarize the reasoning of this section we introduce the following definition.

We call an approximation algorithm A for an optimization problem P, an $(r(n), t(n))$-approximation algorithm if A approximates P within an approximation ratio $r(n)$ and A works in $O(t(n))$ time with n the size of the problem instance.

We can formulate the following lemma.

Lemma 1 *There exists a constant c such that if there exists an $(r(n), t(n))$-approximation algorithm for E3-OCC-MIN-3-LIN2 then there exists an $(r(cn), t(cn))$-approximation algorithm for MIN-LIN2.*

2.4 Reducing MIN-LIN2 to E3-OCC-MIN-E3-LIN2

An existing method of converting equations with 2 variables into equations with 3 equations, described by Khanna, Sudan and Trevisan [KST97] cannot be applied here because it increases the number of occurences of variables. Therefore we need to provide a different technique.

Instead, we will modify the reduction of the previous sections. First, we can make sure that the variables in a resulting instance of E3-OCC-MIN-3-LIN2 can be colored with two colors blue and red, so that the following holds true: in an equation of three variables all three must be blue, in an equation with two variables, one must be blue and the other must be red. If we accomplish that, we can replace each red variable with a sum of two new variables.

This simple idea has some minor complications, so we describe it in more detail. The reduction again consists of functions f, g and h — transformation, normalization and bijection — except that now $|sat(a, S)| = 3|sat(h(a), f(S)|$, *i.e.* the normalized solutions for a transformed instance will satisfy 3 times as many equations as the equivalent solutions of the original instance.

We first color occurences of variables as described above: in an equation with 3 variables all occurences are blue, in an equation with 2 variables, one is blue, the other is red. Then we replicate the occurences 3 times, and each replicated occurence is a new variable. Each original equation is replaced with 3 new equations in an obvious manner.

Now, let K be the set of occurences of a variable. Note that the numbers of blue and red elements of K are divisible by 3. We create a gadget $CG(K)$ very similarly as before, so we will only mention the differences in a construction:

- we double the lenth of all cycles that replace nodes of a strong expander, the colors on the cycle alternate, if a contact belongs to such a cycle, it keeps its original color;
- an edge of the strong expander is replaced by two edges between the respective cycles, one being {blue, red} and the other {red, blue};
- at the moment, each cycle that contains a node that is connected only to its neighbors on the cycle and that has a color different than the color of the K element of this cycle, within $CG(K)$ we contact, we connect each 3 such blue nodes with a new red node, and each 3 reds with a new blue.

Now, each equation with 2 variables has one blue and one red variable, so we can replace each red variable with a sum of two new variables, and as a result each equation has 3 variables.

Because we increase the size of the transformed instance only by a constant factor, we can restate our lemma as the following theorem:

Theorem 1 *There exists a constant c such that if there exists an $(r(n), t(n))$-approximation algorithm for E3-OCC-MIN-E3-LIN2 then there exists an $(r(cn), t(cn))$-approximation algorithm for MIN-LIN2.*

3 MIN-BISECTION

3.1 Notation

For a graph G, $V(G)$ is a set of nodes, $E(G)$ is a set of edges, $\overline{S} = V - S$, $B(G)$ is the set of *bisections*, e.g. sets $S \subset V(G)$ such that $|S| = |\overline{S}|$ and $Cut(S) = \{e \in E(V) :\ e \cap S \neq \emptyset \text{ and } e \cap \overline{S} \neq \emptyset\}$ for $S \subset V(G)$.

The MIN-BISECTION problem is to find $S \in B(G)$ such that$|Cut(S)|$ is minimum.

Our reduction is described using the following three functions:

- an instance transformation f such that if G is a graph with n nodes then $f(G)$ is a graph of degree 3 with $O(n^3)$ nodes;

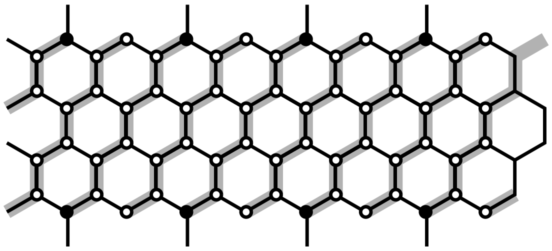

Figure 2: Gadget for $d = 8$; black circles are the contact nodes.

- a solution normalization g such that for $S \in B(f(G))$ we have $g(S) \in B(f(G))$ and $|Cut(g(S))| \leq |Cut(S)|$;
- a bijection $h : B(G) \to g(B(f(G)))$ such that $|Cut(S)| = |Cut(h(S))|$.

3.2 The reduction

We assume that the maximum node degree in G is bounded by some even d. Our instance transformation replaces every node v with a gadget A_v in which d nodes are the contacts (black circles in the Fig. 2). The diagram depicts a gadget for $d = 8$. The gadget is a cylindrical hexagonal mesh that can be alternatively decomposed into $d/2$ *horizontal cycles* and into d *diagonal paths*. Each diagonal path contains d nodes, for the total of d^2 nodes. Every other diagonal path has two *contact nodes* at its ends.

An edge $\{u, v\}$ is replaced with an edge between the contacts of A_u and A_v, these replacement edges are disjoint.

Solution normalization will assure that for each u either $A_u \subset g(S)$ or $A_u \subset \overline{g(S)}$. We will compute $g(S)$ in two stages. In the first stage we decide whether to place A_u within $g(S)$ or within $\overline{g(S)}$ based solely on $S \cap A_u$. As a result, k_u nodes that belong to $\overline{S}$ will become members of $g(S)$, or $-k_u$ nodes that belong to S will become members of $\overline{g(S)}$.

After the first stage, $|g(S)| - |S| = \sum_{u \in V(f(G))} k_u = s$. If $s \neq 0$, $g(S)$ is not a bisection. Therefore in the second stage we remove s/d^2 gadgets from $g(S)$ (if $s < 0$, we insert $-s/d^2$ gadgets).

The normalization causes some edges that did not belong to $Cut(S)$ to become members of $Cut(g(S))$. In the first stage we move some contacts from S to $\overline{g(S)}$, or from $\overline{S}$ to $g(S)$; for each such contact the incident replacement edge may become a member of $Cut(g(S))$. In the second stage, we move s/d^2 gadgets, thus s/d contacts, and again, the replacement edges incident to the contacts that moved may become members of $Cut(g(S))$. This allows us to estimate the consequences of the decision made during the first stage about each gadget A_u.

Let E_u be the set of edges inside A_u. A decision moves $k = |k_u|$ nodes, among them, i contacts. To offset the increases of $Cut(g(S))$ it suffices to gain $i + k/d$ edges, and this happens if $|Cut(S) \cap E_u| \geq i + k/d$.

Suppose first that $|Cut(S) \cap E_u| \geq d$. Let j be the number of contacts in $S \cap A_u$ and $k = |S \cap A_u|$. If $j + k/d \leq d$, we can place A_u in $g(S)$; otherwise $(d - j) + (d^2 - k)/d \leq d$ and we can place A_u in $\overline{g(S)}$.

Now we can assume that $|Cut(S) \cap E_u| < d$. We will make an observation. Our gadget contains $d/2$ vertical cycles, if one of them contains an edge of $Cut(S)$, it must contain two such edges; we can conclude that at least one of the vertical vertical cycles has no edges in $Cut(S)$, and thus is contained in S or in $\overline{S}$. For similar reasons, one of the diagonal paths must be contained in Because every horizontal cycle overlaps every diagonal path, it is true for exactly one $\mathbf{S} \in \{S, \overline{S}\}$ that $\mathbf{S}$ contains a horizontal cycle and a diagonal path. We place A_u is $\mathbf{S}$.

Let U be a connected component of $A_u - \mathbf{S}$, and let $C(U) = Cut(U) \cap E_v$. Assume that U contains i_U contacts. To finish the proof of correctness of our normalization, it suffices to show that $i + |U|/d \leq |C(U)|$, or, equivalently, that $\beta(U) = |C(U)| - i_U - |U|/d \geq 0$.

Suppose, by the way of contradiction, that for some U that is disjoint with a horizontal cycle and with a diagonal path we have $\beta(U) < 0$. We choose such U with the smallest possible $|C(U)|$, and, with this constraint, the smallest possible $\beta(U)$.

Observation 1. A node $v \in U$ cannot have two neighbors in $A_u - U$. Otherwise if v has no neighbors in U, then $U = \{v\}$ and $\beta(U) \geq 2 - 1 - 1/d$. If v has a neighbor in U, we could remove v from U and decrease both $C(U)$ and $\beta(U)$.

Observation 2. A node $v \in A_u - U$ must have at least two neighbors in $A_u - U$. otherwise we could insert v to U and decrease both $|C(U)$ and $\beta(U)$.

Our gadget A_u can be covered with a collection of cycles of length 6 that we will call *hexagons*.

Observation 3. Set U cannot contain exactly 1,4, or 5 nodes of a hexagon. If U contains 1 or 5 nodes, it contradicts Observation 1 or Observation 2. If U contains 4 nodes, then the other two nodes, if they do not contradict Observation 2, form an edge with exactly two neighbors in U and two neighbors in $A_u - U$, thus we can insert this pair to U without increaseing $|C|$ while increasing $\beta(U)$.

Observation 4. Assume that D_0, D_1 are two adjacent diagonal paths, $U \cap D_0 = \emptyset$ and $B = U \cap D_1 \neq \emptyset$. Then B forms a path that has a contact node at one of its ends.

To show it, consider D_2, the other diagonal path adjacent to D_1, and A', a connected component of A. A brief and easy case analysis shows that A' has more neighbors in $D_0 \cup (D_1 - U)$ than in D_2. Because at least one horizontal cycle is disjoint with U, $|A| < d$. Therefore removing A' from U decreases $|C|$ and $|C(U)| - |U|/d$. Because $\beta(U)$ cannot be decreased in this fashion, A' must contain a contact node. Because U is disjoint with a horizontal cycle, there is only one contact node for which it is possible.

Another brief and easy case analysis shows that U must form a "trapesoid", with one basis being a fragment of a horizontal cycle that forms a path between two contact nodes, the other basis being a path that is a fragment of another horizontal cycle and the sides are the initial fragments of two diagonal paths.

Assume that such a set U contains a contact nodes and overlaps b horizontal cycles. U contains $4a-3$ nodes in the basis that contains the contact nodes, then it the consecutive horizontal cycles it contains $4a-5$, $4a-7, \ldots$ down to $4a-3-2(b-1) = 4a-2b-1$ nodes. Thus

$$|U| \;=\; b\,\frac{4a-3+4a-2b-1}{2} \;=\; b\,(4a-b-2).$$

In turn, $C(U)$ contains 2 edges in each of the b horizontal cycles, plus the edges extending to the horizontal cycle that is disjoint with U and adjacent to its smaller basis. This basis has $4a-2b-1$ nodes and the nodes with such edges alternate with the nodes without them, so we have

$$|C(U)| = 2b + \lfloor (4a-2b-1)/2 \rfloor = 2b+2a-b-1 = 2a+b-1.$$

Thus $\beta(U) = (2a+b-1) - a - b(4a-b-2)/d = a+b-1-b(4a-b-2)/d$, while we have the following constraint: $|C| < d$, *i.e.* $2a+b-1 < d$. If we decrease d, $\beta(U)$ will also decrease, so we assume $d = 2a+b$ and thus $b = d-2a$. Then we have

$$\begin{aligned}
d\beta(U) = d(a+d-2a-1) - (d-2a)(4a-d+2a-2) = \\
d(d-a-1) - (d-2a)(6a-d-2) = \\
d^2 - da - d - 6da + d^2 + 2d + 12a^2 - 2da - 4a = \\
2d^2 + 12a^2 - 9da + d - 2a = \\
2(d-2.25a)^2 + 1.875a^2 + d - 2a.
\end{aligned}$$

This concludes the proof of the following

Theorem 2 *If there exists an $(r(n), t(n))$-approximation algorithm for 3-MIN-BISECTION then there exists an $(r(n^3), t(n^3))$-approximation algorithm for MIN-BISECTION.*

□

Acknowledgments

We thank Lars Engebretsen, Ravi Kannan, Mario Szegedy, and Ran Raz for stimulating discussions.

References

[AL95] S. Arora, C. Lund, *Hardness of Approximations*, in *Approximation Algorithms for NP-Hard Problems*, D. S. Hochbaum (ed.), PWS Publishing, Boston 1995, 399-446.

[BFK00] C. Bazgan, W. Fernandez de la Vega and M. Karpinski, Approximability of dense instances of NEAREST CODEWORD problem, ECCC Tech. Report TR00-091, 2000.

[BK99] P. Berman and M. Karpinski, *On some tighter inapproximability results*, Proc. of 26th ICALP, LNCS 1644, Springer-Verlag, Berlin, 1999, 200-209.

[BK01] P. Berman and M. Karpinski, *Approximating Minimum Unsatisfiability of Linear Equations*, ECCC Technical Report, TR01-025, 2001

[GG81] O. Gabber and Z. Galil, *Explicit construction of linear size superconcentrators,* JCSS 22(1981), 407-420.

[DKS98] I. Dinur, G. Kindler and S. Safra, *Approximating CVP to within almost polynomial factors is NP-hard*, Proc. of 39th IEEE FOCS, 1998, 99-109.

[DKRS00] I. Dinur, G. Kindler, R. Raz and S. Safra, *An improved lower bound for approximating CVP*, 2000, submitted.

[KST97] S. Khanna, M. Sudan and L. Trevisan, *Constraint Satisfaction: the approximability of minimization problems*, Proc. of 12th IEEE Computational Complexity 1997, 282-296.

[LPS88] A. Lubotzky, R. Phillips and P. Sarnak, *Ramanujan graphs*, Combinatorica, 8 (1988), 261–277.

[PY91] C. Papadimitriou and M. Yannakakis, *Optimization, approximation and complexity classes*, JCSS **43**, 1991, pp. 425–440.