DIMACS Technical Report 99-23

On Some Tighter Inapproximability Results

by

Piotr Berman[1]
Dept. of Computer Science,
Pennsylvania State University,
University Park, PA 16802.
Email: berman@cse.psu.edu

Marek Karpinski[2]
Dept. of Computer Science,
University of Bonn,
53117 Bonn.
Email: marek@cs.uni-bonn.de

[1]Supported in part by NSF grant CCR-9700053.
[2]Research supported in part by the International Computer Science Institute, Berkeley, California, by DFG grant 673/4-1, ESPRIT BR grants 7079, 21726, and EC-US 030, by DIMACS, and by the Max–Planck Research Prize.

ABSTRACT

We give a number of improved inapproximability results, including the best up to date explicit approximation thresholds for bounded occurence satisfiability problems like MAX-2SAT and E2-LIN-2, and the bounded degree graph problems, like MIS, Node Cover, and MAX CUT. We prove also for the first time inapproximability of the problem of Sorting by Reversals and display an explicit approximation threshold.

1 Introduction

The paper studies *explicit* approximation thresholds for bounded dependency, and bounded degree optimization problems. There was a dramatic progress recently in proving tight inapproximability results for a number of NP-hard optimization problems (cf. [H96], [H97], [TSSW96]). The goal of this paper is to develop a new method of reductions for attacking bounded instances of the NP-hard optimization problems and also other optimization problems. The method uses randomized reductions and applies to the number of problems including Maximum Independent Set in graphs of degree d (d-MIS), bounded degree Minimum Node Cover (d-Node Cover), bounded degree MAX CUT (d-MAX CUT) and bounded occurrence MAX-2SAT (d-OCC-MAX-2SAT), (cf. [PY91], [A94], [BS92], [BF94], [BF95], [AFWZ95]). This yields also the first explicit approximation lower bounds for the small degree graph problems, and the small dependency satisfiability. We apply also this method to prove approximation hardness of the problem of *sorting by reversals*, MIN-SBR, the problem motivated by molecular biology [HP95] (and with a long history of related research, cf., e.g., [GP79], [CB95]), only recently proven to be NP-hard [C97]. Interestingly, its signed version can be computed in polynomial time [HP95], [BH96], [KST97].

The core of the new method is the use of restricted versions of the E2-LIN-2 and E3-LIN-2 problems studied in [H97]. We denote by E2-LIN-2 the problem of maximizing the number of satisfied equations from a given set of linear equations mod 2 with exactly 2 variables per equation. E3-LIN-2 is a similar problem with three variables per equation. E2-LIN-2 can be viewed as a graph problem in the following way: each variable is a node, and an equation $x \oplus y = b$ is an edge $\{x, y\}$ with label b. (Note that the special case when all edges have label 1 constitutes MAX CUT problem.)

We denote by d-OCC-E2-LIN-2 and d-OCC-E3-LIN-2 the versions of these problems where the number of occurrences of each variable is bounded by d (note that in d-OCC-2-LIN-2 can be also viewed as restricted to graphs of degree d).

The rest of the paper proves the following main theorem:

Theorem 1. *For every $\epsilon > 0$, it is NP-hard to approximate*

(i) 3-OCC-E2-LIN-2 and 3-MAX CUT within factor $332/331 - \epsilon$*;*

(ii) 6-OCC-MAX 2SAT within factor $668/667 - \epsilon$*;*

(iii) 3-OCC-E3-LIN-2 within factor $62/61 - \epsilon$*;*

(iv) 4-MIS within factor $74/73 - \epsilon$ *and 4-Node Cover within* $79/78 - \epsilon$*;*

(v) 3-MIS within factor $140/139 - \epsilon$ *and 3-Node Cover within* $145/144 - \epsilon$*;*

(vi) MIN-SBR within factor $1237/1236 - \epsilon$*.*

Our proof can be easily extended to provide explicit inapproximability factors for many other optimizations problems that are related to bounded degree graphs. E.g. we get also 1.0149 lower bound for 5-MIS, 1.0138 lower bound for 5-Node Cover, and 1.0005 lower bound for 3-OCC-MAX 2SAT.

The technical core of all these results is the reduction to show (i), which forms structures that can be translated into many graph problems with the very small and natural gadgets. The best to our knowledge gaps between the upper and lower approximation bounds are summarized in Table 1. The upper approximation bounds are from [GW94], [BF95], [C98], and [FG95].

Problem	Approx. Upper	Approx. Lower
3-OCC-E2-LIN-2	1.1383	1.0030
3-OCC-E3-LIN-2	2	1.0163
3-MAX CUT	1.1383	1.0030
3-OCC-MAX 2SAT	1.0741	1.0005
6-OCC-MAX-2SAT	1.0741	1.0014
3-MIS	1.2	1.0071
4-MIS	1.4	1.0136
5-MIS	1.6	1.0149
3-Node Cover	1.1666	1.0069
4-Node Cover	1.2857	1.0128
5-Node Cover	1.625	1.0138
MIN-SBR	1.5	1.0008

Table 1: Gaps between known approximation bounds.

1.1 Notation

We list here some notation that we use in this paper. $\mathbf{Z}_2$ is the field with two elements, we use $\oplus$ to denote the addition operation (modulo 2). In an undirected graph $< V, E >$, for $S \subset V$ we define the characteristic function χ_S that equals 1 for elements of S and 0 for non-elements. Moreover, $Cut(S)$ as $\{\{u,v\} \in E : \ \chi_S(u) \oplus \chi_S(v) = 1\}$.

2 Sequence of reductions

We start from E2-LIN-2 problem that was most completely analyzed by Håstad [H97] who proved that it is NP-hard to approximate it within a factor $12/11 - \epsilon$. In the sequel we will use notation of this paper. In this problem we are given a (multi)set of linear equations over $\mathbf{Z}_2$ with at most two variable per equation, and we maximize the size of a consistent subset. In this paper, we prefer to interpret it as the following graph problem. Given an undirected graph $G = \langle V, E, l \rangle$ where l is a 0/1 edge labeling function. We define $Score(S, \{u, v\}) = \chi_S(u) \oplus \chi_S(v) \oplus l(\{u, v\})$. In turn, $Score(S) = \sum_{e \in E} Score(S, e)$. The objective of E2-LIN-2 is to maximize $Score(S)$.

Our first reduction will have instance transformation τ_1, and will map an instance G of E2-LIN-2 into another instance G' of the same problem that has three properties: G' is a graph of degree 3, its girth (the length of a shortest cycle) is $\Omega(\log n)$, and its set of nodes can be covered with cycles in which all edges are labeled 0. We will use τ_1(E2-LIN-2) to denote this restricted version of E2-LIN-2. The last two properties of $\tau_1(E2 - LIN - 2)$ are important in the subsequent reductions that lead to MIN SBR problem.

We alter the reduction τ_1 in two ways. The first modification results in graphs that have all edges labeled with 1, i.e. it reduces E2-LIN-2 to 3-MAX CUT and allows to complete the proof of (i). The second modification reduces E3-LIN-2 to a very special version of 3-OCC-E3-LIN-2, which we call HYBRID, because a large majority of equations have only two variables. This reduction instantaneously leads to (iii).

To show (ii), we use an obvious reduction from τ_1(E2-LIN-2): an instance of E2-LIN-2 can be viewed as a set of equivalence statements, and we can replace each equivalence with a pair of implications. On the other hand, we obtain (v) and (iv) using reductions from HYBRID.

Although HYBRID problem appears to be very "efficient", we cannot use it in the chain that leads to MIN-SBR. Instead, we use another reduction, with instance translation τ_2, that leads from τ_1(E2-LIN-2) to 4-MIS. This translation replaces each node/variable with a small gadget. The resulting instances of 4-MIS can be transformed into the next problem that we consider, which we call *breakpoint graph decomposition*, BGD. This problem is related to *maximum alternating cycle decomposition*, (e.g. see Caprara, [C97]) but has a different objective function (as with another pair of related problems, Node Cover and MIS, the choice of the objective function affects approximability). An instance of BGD is a so-called breakpoint graph, i.e. an undirected graph $G = \langle V, E, l \rangle$ where l is a 0/1 edge labeling function, which satisfies the following two properties:

(i) for $b \in \{0, 1\}$, each connected component of $\langle V, l^{-1}(b) \rangle$ is a simple path;

(ii) for each $v \in V$, the degrees of v in $\langle V, l^{-1}(0) \rangle$ and in $\langle V, l^{-1}(1) \rangle$ are the same.

An alternating cycle C is a subset of E such that $\langle V, C, l|C \rangle$ has the property (ii). A decomposition of G is a partition $\mathcal{C}$ of E into alternating cycles. The objective of BGD is to minimize $cost(\mathcal{C}) = \frac{1}{2}|E| - |\mathcal{C}|$.

By changing the node-replacing gadget of τ_2 and enforcing property (i) by "brute force", we obtain reduction τ_3 that maps τ_1(E2-LIN-2) into BGD. The last reduction, π, converts a breakpoint graph G into a permutation $\pi(G)$, an instance of sorting by reversals, MIN-SBR. We use a standard reduction, i.e. the correspondence between permutations and breakpoints graphs used in the approximation algorithms for MIN-SBR (this approach was initiated by Bafna and Pevzner, [BP96]). In general, this correspondence is not approximation preserving because of so-called *hurdles* (see [BP96, HP95]). However, the permutations in $\pi(\tau_3(\tau_1($E2-LIN-2$)))$ do not have hurdles, and consequently for these restricted version of BGD, π is an approximation preserving reducibility with ratio 1.

3 First Reduction

To simplify the first reduction, we will describe how to compute the instance translation using a randomized poly-time algorithm. In this reduction, every node (variable) is replaced with a *wheel*, a random graph that is defined below (some parts of this definition will not be used to describe the reduction, but will be used later, in the proof of correctness). The parameter κ used here is a small constant; in this paper we prove that $\kappa = 6$ is sufficiently large.

Definition 2. An *r-wheel* is a graph with $2(\kappa+1)r$ nodes $W = \mathit{Contacts} \cup \mathit{Checkers}$, that contains $2r$ *contacts* and $2\kappa r$ *checkers*, and two sets of edges, C and M. C is a Hamiltonian cycle in which with consecutive contacts are separated by chains of κ checkers, while M is a random perfect matching for the set of checkers (see Fig. 1 for an example).
For a set of nodes $A \subset W$ let a_A be the number of contacts in A, b_A the number of contiguous fragments of of A in the cycle C (i.e. $b_A = |Cut(A) \cap C|/2$) and $c_A = |Cut(A) \cap M|$.
We say that A is *bad* iff $r \geq a_A > 2b_A + c_A$. A set B is *wrong* iff for some bad set A we have $B = A \cap \mathit{Checkers}$. A set $B \subset \mathit{Checkers}$ is *isolated* iff no edges in M connect B with $\mathit{Checkers} - B$.

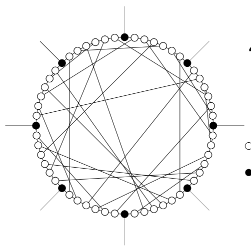

Figure 1: A very small example of a gadget used by τ_1.

Consider an instance G of E2-LIN-2 with n nodes (variables) and m edges (equations). We will describe how to transform G into $\tau_1(G)$, an instance of 3-OCC-E2-LIN-2. Let $k = \lceil n/2 \rceil$. A node v of degree d will be replaced with a kd-wheel W_v. All wheel edges are labeled 0 to indicate our preference for such a solution S that either $W_v \subset S$ or $W_v \cap S = \emptyset$.

An edge $\{v, u\}$ with label l is replaced with $2k$ edges, each of them has label l and joins a contact of W_v with a contact of W_u. In the entire construction each contact is used exactly once, so the resulting graph is 3-regular.

We need to elaborate this construction a bit to assure that $\tau_1(G)$ has a large girth. First, we will assure that no short cycle is contained inside a wheel. We can use these properties of an r-wheel W: each cycle different of length lower than $2\kappa r$ must contain at least one edge of the matching M and the expected number of nodes contained in cycles of length $0.2\log_2(\kappa r)$ or less is below $(\kappa r)^{-0.8}$ fraction). Thus we can destroy cycles of length below $0.2\log_2 n$ by deleting matching edges incident to every node on such a cycle and neglect the resulting changes in *Score*.

Later, we must prevent creation of short cycles when we introduce edges between the wheels; this can be done using a construction described by Bollobás [B78]. While Bollobás described how to build a graph of large girth from scratch, his construction can assure the following: given a graph of degree 3 with girth at least $0.5\log_2 n$ and two n-element disjoint sets of nodes of degree 2, each of size n, say A and B, one can increase the set of edges by a perfect bipartite matching of A and B without increasing the girth above $0.5\log_2 n$. Note that we are indeed replacing an edge of the original graph with a perfect matching with at least n edges, which allows us to use the construction of Bollobás.

The solution translation is simple. Suppose that we have a solution S for a translated instance. First we normalize S as follows: if the majority of contacts in a wheel W belong to S, we change S into $S \cup W$, otherwise we change S into $S - W$. A normalized solution S can be converted into a solution S' of the original problem in an obvious manner: a node belongs to S' iff its wheel is contained in S. Assuming that G has m edges/equations, we have $Score(S) = 2k((3\kappa+2)m + Score(S'))$. Håstad [H97] proved that for E2-LIN-2 instances with $16n$ equations it is NP-hard to distinguish those that have *Score* above $(12-\epsilon)n$ and those that have *Score* below $(11+\epsilon)n$, where the positive constant ϵ can be arbitrarily small. By showing that our reduction is correct for $\kappa = 6$ we will prove

Theorem 3. *For any* $\epsilon \in (0, 1/2)$*, it is NP-hard to decide whether an instance of* τ_1(E2-LIN-2) $\in$ 3-OCC-E2-LIN-2 *with* $336n$ *edges (equations) has Score above* $(332-\epsilon)n$ *or below* $(331+\epsilon)n$*.*

Recall that we have already described the instance translation and the solution translation. The latter consists of two stages. In normalization stage we transform a solution S in steps. In each step we consider a consistency wheel W that is used as a gadget that replaced a node (variable) of the original instance. Suppose that an r-wheel W is one of these; it consists of $2r$ contacts and $12r$ checkers, while its edges form two sets: cycle C and random matching M of the set of checkers. If at least r of the contacts belong to S, we replace S with $S \cup W$, and otherwise we replace S with $S - W$. The normalization step fails if $Score(S)$ decreases. If all normalization steps are successful, then together they form an approximation-preserving solution translation. In the next stage, we form a solution S' of the original instance of E2-LIN-2 in a natural manner: S' consists of nodes whose wheel gadgets are contained in S. Because we have already calculated approximation-preserving properties of this stage, it suffices to show that the first stage succeeds with a high probability. Because we can pad the wheels to a desired size, it suffices to show that the failure of a normalization step involving an r-wheel is $O(r^{-3})$. The rest of this proof is devoted to this claim.

Suppose that the normalization step has failed and A is the subset of W consisting of nodes that changed membership in S. Such a set A must satisfy a property that depends solely on the random matching M. Let a_A be the number of contacts in A, $b_A = |Cut(A) \cap C|$ and $c_A = |Cut(A) \cap M|$. It is easy to see that $Score(S,e)$ changes iff $e \in Cut(A)$. Moreover, $|Cut(A)| = a_A + b_A + c_A$, and $Score(S,e)$ increases for $e \in Cut(A) \cap (C \cup M)$. Therefore the decrease of $Score(S)$ is at most $a_A - b_A - c_A$. Consequently, A must satisfy $a_A > b_A + c_A$. We will say that such a subset of W is *wrong*.

If a set A is wrong, we say that $B = A \cap Checkers$ is *bad*. Our next goal is to characterize bad sets. For the remainder of this proof we convert W to a graph with set of nodes equal to *Checkers* by replacing each contact u with an edge (later called a contact edge) that connects the checkers that were adjacent to u. In the new W we can define b_B and c_B in the same way as b_A and c_A. One the other hand, the definition of a_B will be more complicated. Let a_B^1 (and a_B^2) be the number of contact edges that have exactly one (exactly two) endpoint in B. We consider B only if $a_B^2 \leq a_{W-B}^2$, and we define a_B is the minimum of $a_B^2 + a_B^1$ and r. With this notation, B is bad iff $a_B > b_B + c_B$. Now it suffices to show that the probability that a bad subset of W exists is low.

As a preliminary step, we must have some tools to estimate the probabilities in the random space consisting of perfect matchings. We will use the following definitions.

i) A set $A \subset W$ is M-close iff no edges of M are in $Cut(A)$.

ii) The function $\mu(n)$ denotes the number of perfect matchings in a clique with $2n$ nodes.

Lemma 4.

$$\mu(n) = \prod_{i=1}^{n}(2i-1) = \frac{(2n)!}{n!2^n}$$

Proof. By induction on n. For $n = 0$, there exists exactly one perfect matching. Now consider a clique with $2n-1$ nodes. A fix node can be matched using any of the $2n-1$ incident edges. We can complete the construction of the matching by choosing any of $\mu(n-1)$ matchings of the remaining $2n-2$ nodes, thus $\mu(n) = (2n-1)\mu(n-1)$.

Lemma 5. *The probability that a set of $2d$ checkers is M-close is $\mu(d)\mu(6r-d)/\mu(6r)$, or*

$$\prod_{i=1}^{d} \frac{2i-1}{12r-(2i-1)}$$

Proof. Straightforward consequence of Lemma 4.

Our general method of estimating the probability of a bad set existing, is to consider separatedly cases when a bad set B has a particular vector of parameters a_B, b_B and $s_B = |B|$. For each of them we will

a) estimate the numbers of candidates for a bad set, such that if a bad set exists, than one of the candidates must be bad as well;

b) find the number of subsets of a candidate B, each of size at least $s_B - a_B + b_B$, such that if B is bad, than one of these subsets must be M-close;

c) multiply the product of the results of a) and b) with the probability calculated from Lemma 5.

While discussing a candidate for a bad set, say B, we will refer to *fragments* of B, connected components of B within cycle C (note that in the modified W, the cycle C consists of checkers only). Further, we define $C_B \subset B$ as the set of nodes of B that are incident to edges of $Cut(B) \cap M$. According to our definitions, $B - C_B$ is M-closed; moreover, $|C_B| = c_B$, thus if B is bad, then $|B - C_B| = s_B - c_B > s_B - a_B + b_B$.

The following lemma limits the number of candidates for an M-closed subset.

Lemma 6. *In a bad set B of minimum size nodes of C_B are not incident to edges of $Cut(B) \cap C$.*

Proof. Suppose that a node of C_B is incident to an edge of $Cut(B) \cap C$. If we remove it from B, b_B remains unchanged, c_B is decreased by 1 and a_B is decreased by at most 1 (the later decrease occurs if the edge in question is a contact edge). Thus we have obtain a smaller bad set, a contradiction.

We will use two ways of estimating the probability that a bad set exists. The first one is appplied to sufficiently small candidates.

Lemma 7. *For $a < r/9$, the probability that there exists a bad set B such that $a_B = a$ is at most $r^{-3}0.7^a$.*

Proof. First, we will establish the relatioship between a_B, b_B, c_B and s_B. By the definition of a bad set, $a_B > b_B + c_B$. Now consider a fragment of B that is incident to, say, a_o contact edges. This fragment must contain $a_o - 1$ chains of 6 nodes, and portions (possibly empty) of two other such chains on its fringes. Thus it contains between $6(a_o - 1)$ and $6(a_o + 1)$ nodes. Because each fragment is incident to exactly two edges from $Cut(B) \cap C$, there are $b_B/2$ fragments. By adding sizes of all fragments, we obtain a result that is between $6a_B - 3b_B$ and $6a_B + 3b_B$.

One conclusion that we can draw is that $s_B < 9a_B < r$. Another is that $s_B \geq 6a_B - 3b_b \geq 6(b_B + c_B + 1) - 3b_B = 3b_B + 6c_B + 6$. The latter implies that B contains an M-closed set of size $2d$ where $d \geq 1.5b_B + 2.5c_B + 3$.

We can generate a candidate B with parameters $a_B = a$ (for some $a \leq r/9$) and $b_B = b$ (for some $b < a$) as follows. First, we select $b/2$ of the "left ends" of the fragments; this can be done in at most $\mathbf{C}(12r, b/2)$ ways, where $\mathbf{C}$ is our notation for the binomial coefficient. Next, we distribute the sizes of the fragments; because the sum of sizes is less than r, and all of them are positive, this can be done in less than $\mathbf{C}(r, b/2)$ ways. Finally, be removing a subset of $c = a-b-1$ elements (or one less, to obtain a set of even size), we will obtain an M-close sets. Altogether, we generate an M-close set in less than $\mathbf{C}(12r, b/2)\,\mathbf{C}(r, b/2)\,\mathbf{C}(r, c)$ many ways, i.e.

$$\prod_{i=1}^{b/2} \frac{12r-i+1}{i} \prod_{i=1}^{b/2} \frac{r-i}{i} \prod_{i=1}^{c} \frac{r-i}{i}.$$

By Lemma 5, each of these candidates is M-closed with probability at most

$$\prod_{i=1}^{3+\lceil (3b+5c)/2 \rceil} \frac{2i-1}{12r-(2i-1)}$$

Note that the latter product has more terms than the first three combined, and that even the largest of these terms is less than $1/11$. We will multiply each of the $b/2$ terms of $\mathbf{C}(12r, b/2)$ with two of the terms of the probability; in particular, for $i \geq 1$, the term of i with the term of $3i+1$ and one extra terms. It is easy to see that each such product is below 0.7. Next, we multiply each term of $\mathbf{C}(r, b/2)$ with one term of the probability; this time term number i with the term number $3i+2$. Except for the first two, each such product is below 0.7. Next, we multiply each of the term of $\mathbf{C}(r, c)$ with two terms of the probability, in particular, term number i with the term number $3i+3$ and one extra term. The result is below 0.49. Thus we accounted for all terms of the estimate of the number of candidates for an M-closed set and the product is below 0.7^{b+2c}. Note that the first three terms of the probability that were not accounted for yield less than r^{-3}.

Summarizing, we have at least $3+1.5a$ terms of the probability expression; the first three produce r^{-3}, $1.5b$ of them produce 0.7^b, $2c$ of them produce 0.7^{2c} and the remaining ones contribute factors below 0.1.

The significance of this lemma is the following. For larger values of a we will find $x < 1$ and a polynomial p such that the probability that there exists a bad set B with $a_B = a$ is at most $p(r)x^a$. Because we have a separate argument for the low values of a, such a result suffices to prove our theorem, regardless of the degree of the polynomial p. In practice, it allows us to introduce any fixed number of integer parameters that describe number of nodes or edges with some particular properties, and compute the probability that there exists a bad set with a particular vector of parameters. Therefore we can finish the proof with the following lemma.

Lemma 8. *There exists a polynomial p such that the probability that there exists a bad set B with $a_B = a > r/10$ is lower than $p(r)0.81^a$.*

Proof. We can fix some values of $a > r/10$, b and s and search for a bad set B such that $a_B = a$, $b_B = b$ and $s_B = s$. We can count such candidates in three ways. The first method is to count in how many ways we can select b edges of $Cut(B) \cap C$: $\mathbf{C}(12r, b)$.

To understand the second method, imagine that we label each edge of $Cut(B) \cap C$; if this edge has its right endpoint in B, we label it $<$, and otherwise (left endpoint in B) we label it $>$. Next, we move each $<$ label to the nearest contact edge to its right, and each $>$ label to its left. Finally, we move the labels back to their original positions. The positions of the labels at the time when they are all placed on the contact edges provides a lower bound on the size of B; a fragment that is incident to a_o contact edges will have its size estimated as $6(a_o - 1)$. (Note that fragments of B that do not have incident contact edges will obtain the size estimate of -6; this is because its $<$ label is at this time positioned 6 edges to the right of its $<$ label.) Therefore the sum of distances that the labels will traverse from their positions on the contact edges to their correct positions is $s - 6(a - b/2) = d$. This allows us to select any B with parameters b and d as follows: first we select the positions of b labels on $2r$ contact nodes, this can be done in $\mathbf{C}(2r + b - 1, b)$ ways; subsequently we distribute d "units of displacement" to b labels, this can be done in $\mathbf{C}(d + b - 1, d)$ ways. Summarizing, the second method is to compute $d = s - 6(a - b/2)$ and return $\mathbf{C}(2r + b - 1, b)\,\mathbf{C}(d + b - 1, d)$.

The third method is very similar, except that we move the labels in the opposite directions. The resulting formula is identical, except that we compute d differently: $d = 6(a + b/2) - s$.

Note that if we obtain negative d while using the second method, we can conclude that s is too low to be compatible with a and b, similarly, negative d in the third method implies that s is too large. If s is neither too large nor too small, we estimate the number of the candidates for a bad set using the minimum of the results of the three methods desribed here.

If we remove all endpoints of edges from $Cut(B) \cap M$ that are in B, we obtain an M-closed set. By definition, there are c_B such nodes, and if B is bad, then $c_B < a - b$. Moreover, by Lemma 6, we may assume that none of these points is adjacent in C to the complement of B, hence only $a - b$ nodes may be considered for removal. For a possible value of $c = c_B$, the probability that the removal of c nodes may turn B into an M-closed set is $\mathbf{C}(s - b, c)\mu((s - c)/2)\mu(6r - (s - c)/2)/\mu(6r)$. One can see that the latter is the increasing function of c, so it suffices to consider only its maximal value, $a - b - 1$ if $a + b + s$ is odd, and $a - b - 2$ if $a + b + s$ is even.

Our goal is to show that the probability computed according to the above principles, and raised to power $1/a$, is bounded by 0.81. We achieved this goal as follows. We define the real parameters of B as follows:

- α such that $a = \alpha r$;
- β such that $b = \beta\alpha r$; because we are looking at the parameters of bad sets, we know that $0 < \alpha \leq 1$ and $0 < \beta \leq 1$;
- ξ such that $d = \xi\beta\alpha r$, if a respective counting method (second or third) is applicable, $0 \leq \xi \leq 3$.

Using Stirling's formula, and the above estimation formula, we can compute the $1/a$ power of our probability from the parameters α, β and ξ. To consider all possible cases, we can use parameter values that are multiples of some fraction, say ϵ; then, in a subexpression that is an decreasing functions of a parameter, we use the current multiple, say $i\epsilon$, and in an subexpression that is an increasing function, we use $(i+1)\epsilon$. This covers the case of all values between $i\epsilon$ and $(i+1)\epsilon$. In our program, we used the following values for ϵ: 1/20 for ξ, 1/100 for β and 1/2000 for α. The worst case was obtained for $\alpha = 1$, $\beta = 0.77$ and $\xi = 1.15$ ($d = 4.5755r$) and it equals $e^{-0.2181} = 0.8041$.

It remains open whether the same approach may prove a similar result for wheels with 5 checkers between each pair of contacts. In our attempts we introduced several parameters, like the number of fragments that are not incident to any contacts. Even though we were not succesful, the logarithm of the target number was estimated to be 0.03. We believe that with an improved counting method this estimate can be decreased below 0.

Remark 1. One can modify reduction τ_1 as follows. We replicate the set of equations even number of times, as before, so the number of occurrences of each variable is sufficiently high. On each r-wheel the nodes are labeled with a and b, labels alternating. When we select the random matching between checkers, we choose only from perfect matchings in a full bipartite graphs formed by a-checkers and b-checkers (rather than a random perfect matching from the full graph). One can easily show that this restriction makes almost no difference in the probability calculations. Moreover, when we connect the contacts of two wheels, we do it in two ways. If the edge between the respective original variables is labeled with 0, we connect a-contacts with b-contacts, and vice versa. If this edge is labeled with 1, we connect a-contacts with a-contacts and b-contacts with b-contacts. This allows us to convert all labels in the new graph to 1, and as a result, we obtain a graph which is simultaneously an instance of E2-LIN-2 and MAX CUT (and is 3-regular). Let τ_1' be the new reduction. We obtain the following:

Theorem 9. *For any $\epsilon \in (0, 1/2)$, it is NP-hard to decide whether an instance of* τ_1'(E2-LIN-2)$\in$3-MAX CUT *with $336n$ edges has Score above $(332-\epsilon)n$ or below $(331+\epsilon)n$.*

Remark 2. We can translate MAX CUT into MAX 2SAT by replacing each edge with two clauses, i.e. and edge $\{x, y\}$ is replaced with $x \vee y$, $\bar{x} \vee \bar{y}$. This reduction allows to prove Theorem 1(ii).

Remark 3. We will also use another modification. We can start from an instance of E3-LIN-2 with $2n$ equations. (Recall that Håstad has shown that it is NP-hard to distinguish instances where $(2-\epsilon)n$ equations can be satisfied from those where we can satisfy at most $(1+\epsilon)n$.) We modify it to an instance with in which each variable occurs in at least n equations, again, by replicating the equations. Next, each variable is replaced by a r-wheel, where r is the (increased) number of occurrences. The original equations are left same as before, but occurrences of a variable are replaced with occurrences of its contacts. Now we have a new system where each variable occurs exactly three times, and consisting of $2kn$ equations with 3 variables (replicated original equations) and $(1.5\kappa+1)6kn$ equations with 2 variables (inside the wheels). We take $\kappa = 6$, so we have $60kn$ equations inside the wheels. It will be convenient to view the resulting structure as a hypergraph that has $60kn$ normal edges and $2kn$ hyperedges (of size 3), $6kn$ contact nodes and $36kn$ checker nodes.

We can modify the last reduction in a similar manner as in Remark 1. In each chain of 6 checkers (separating two contacts) we label 3 of them with a and 3 with b; then we choose a random bipartite matching between a-checkers and b-checkers. The set of resulting instances of E3-LIN-2 will be later called HYBRID (this name refers to the fact that we have a mixture of equations with 2 and 3 variables). Observe that the reduction from E3-LIN-2 to HYBRID allows to prove Theorem 1(iii).

4 Reduction of 3-MAX CUT to 3-OCC-MAX 2SAT

In order to translate an instance $G =< V, E >$ of 3-MAX CUT into a set of disjunctive two clauses, we create a separate set of 4 propositional variables for each edge $\{u, v\}$ and 4 clauses, $\sim u_0^e \vee u_1^e$, $\sim v_0^e \vee v_1^e$, $u_0^e \vee v_0^e$ and $\sim u_1^e \vee \sim v_1^e$. Moreover, for each node incident to edges e, f and g we add clauses $\sim u_1^e \vee u_0^f$, $\sim u_1^f \vee u_0^g$ and $\sim u_1^g \vee u_0^e$. Thus, if $|V| = 2n$ and $|E| = 3n$ we have $12n$ propositional variables and $18n$ clauses.

To describe a solution translation, consider a valuation of propositional variables, say I. Before we translate I into a partition of V, we will normalize I without decreasing the number of satisfied clauses. We do it in three stages.

(i) We eliminate cases when for some $e = \{u, v\}$ we have $I(u_0^e) = 0$ and $I(u_1^e) = 1$. In every such situation we change $I(u_1^e)$ to 0. Afterwards all 3 clauses where u_1^e occurs are true: two of them contain $\sim u_1^e$, and the other one contains $\sim u_0^e$. Clearly, the number of true clauses could not decrease.

(ii) We eliminate cases when for some $e = \{u, v\}$ we have $I(u_0^e) \neq I(u_1^e) = 1$. Because we performed (i)), this means $I(u_0^e) = 1$ and $I(u_1^e) = 0$. Consider $I(v_0^e)$, if it is 1, then we change $I(u_0^e)$ to 0. It results in $\sim u_0^e \vee u_1^e$ *becoming* true, $u_0^e \vee v_0^e$ and *remaining* true, so the number of true clauses cannot decrease. On the other hand, if $I(v_0^e) = 0$, then because of (i)) we have $I(v_1^e) = 0$. In this case we change $I(u_1^e)$ to 1.

(iii) We eliminate cases when for some u, e, f, i, j we have $I(u_i^e) \neq I(u_j^f)$. In such a situation, pairs of the form u_0^e, u_1^e have equal values of I, thus among 3 such pairs there must be exactly one minority pair, say the one that corresponds to edge e. We convert this pair to the majority value; as a result we gain one clause in the ring of implications of u and loose at most one clause in the gadget of e.

After the normalization, every 6-tuple of propositional variables that corresponds to a node u of G has the same valuation, which we may denote $I(u)$. We define C as the set of those nodes that have $I(u) = 0$. It is easy to see that $CUT(C) = k$ iff for our set of $18n$ clauses, I satisfies $15n - k$ (6 clauses for every of $2n$ nodes, 1 clause for each of $3n$ edges and one extra clause for every edge in $CUT(C)$).

By applying this reduction together with Theorem 9 we can show that for any $\epsilon \in (0, 1/2)$ it is NP hard to decide whether an instance of 3-OCC-MAX 2SAT with $2016n$ clauses has a truth assignment that satisfies at least $(2012 - \epsilon)n$ clauses, or it can be at most $(2011 + \epsilon)n$.

5 From HYBRID to 4-MIS and 3-MIS

Given an instance S of HYBRID, we will form graph G of degree 4, an instance of 4-MIS. Each variable/node x of S will be replace with a gadget A_x which is an induced subgraph of G. Every gadget contains a *hexagon*, i.e. a cycle of length 6 in which nodes with labels 0 and 1 alternate. Hexagons will have two types: a-hexagons, with 2 chords, and b-hexagons, with 1 chord.

If x and y are connected by an edge (equation with two variables), the hexagons of A_x and A_y will share a pair of adjacent edges; this edge of G corresponds to the equation/edge $x = y$. A checker gadget is simply a hexagon: 3 edges edges of equations connected by three other edges, and one or two diagonals. A contact gadget consists of a hexagon fused with a square; 3 such gadgets are connected by an equation gadget that contains 4 nodes that do not belong to gadgets of nodes/variables. Fig. 2 and 3 show these gadgets in detail.

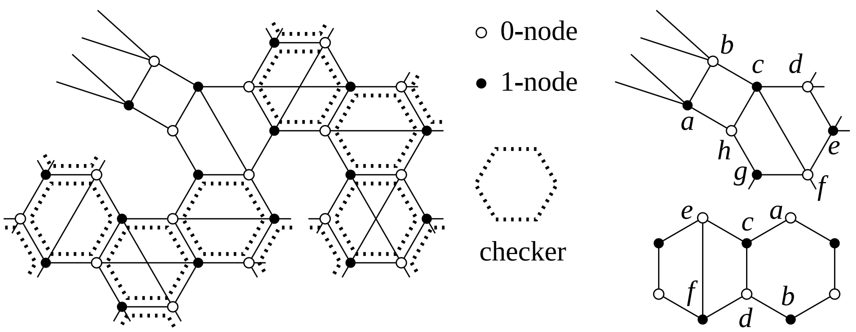

Figure 2: Consistency wheel for 4-MIS problem. The gadget used to replace a contact node is shown in the upper right corner. The lower right corner shows a way to avoid a dirty hexagon.

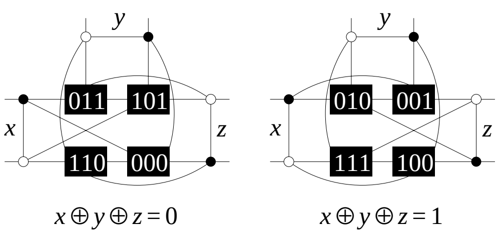

Figure 3: Equation gadgets for 4-MIS.

Given an independent set (a solution) I in graph G we form a solution of S as follows. If $A_x \cap I$ consists of one type of nodes only (i.e. only 0-nodes or only 1-nodes), we assign to x the value equal to this type. In this case, we say that A_x is *pure*. If A_x is *dirty*, can purify it without decreasing the size of I. Below we describe this *purification* in detail.

Suppose first that a hexagon H is dirty (a checker gadget or a part of a contact gadget). It is easy to see that H can be dirty in one way only: $H \cap I$ is a pair of nodes that forms a "missing diagonal" of H. In the lower right part of Fig. 2, we assume that $\{a, b\}$ is this pair. The construction of G assures that in this case there exists a quadrilateral (c, d, e, f) as in this figure, either because $\{e, f\}$ is a diagonal of an adjacent hexagon, or because hexagon H is a part of a contact gadget and this quadrilateral is the square included in this gadget. One must observe that in the cases we consider nodes adjacent to c and d are either adjacent to a or b (and consequently they cannot be in I) or belong to $\{e, f\}$. If $e \notin I$, we can purify H removing a from I and inserting c, if $f \notin I$, we do it by removing b and inserting d. One can see that one of these two cases must hold. Moreover, if the edge $\{c, d\}$ is shared with another gadget, we can always choose the replacement in such a way that we do not make the other gadget dirty when we purify H.

Once we made all hexagons pure, we can make every contact gadget pure as well. Suppose that the gadget from the upper right corner of Fig. 2 is dirty. There are two cases: if $a \in I$, then the hexagon (c, d, e, f, g, h) is 0-pure and we can replace a with h; the case when $b \in I$ is symmetric.

Now we can modify I so that each edge corresponding to an equation with two variables contains a node of I iff the respective equation is true. If such an edge contains a b-node ($b\{0, 1\}$), than both gadgets containing this edge must be b-pure; if both of them are b-pure, we can insert the b-node of this edge to I.

If we partition G into gadgets corresponding to equations, that a gadget A of an equation with three variables consists of 16 nodes: a square contained in a gadget of each participating (contact) variable and four special nodes corresponding to four legal combinations of variable values. Our goal is to assure that if the this equation is true, $A \cap I$ contains 7 nodes and 6 if the equation is false. Clearly, we can place two nodes of I in each square, so $A \cap I$ always has at least 6 nodes. We consider three cases, according to the number of special nodes in $A \cap I$. If this number is 0 and the equation is false, we are done. If it is 0 and the equation is true, then we can insert the special node corresponding to the combination of the values of the three variables. If this number is 1 and the equation is true, again, we are done. If the equation is false, than one of the special node p contained in I wrongly describes one of the variable values, and so it is connected to a node q in the respective contact gadget that has the type equal to the value of this variable; clearly we can replace p with q. Now suppose that this number is 2. Because the equation gadget is very symmetric, it suffices to consider one case, e.g. that the two special nodes in I are 000 and 011. In this case the squares of y and z contain only one node of I each, thus we can replace 000 and 011 with nodes from these two squares.

To finish our reasoning, it remains to perform the accounting. We start with an HYBRID instance with $60kn$ equations with two variables and $2kn$ equations with three variables, and the difficult question whether we can satisfy at least $(62 - \epsilon)kn$ equations, or at most $(61 + \epsilon)kn$. Each of $2kn$ gadgets corresponding to equations with three variables contributes 6 nodes to an independent set, even if they are false. Moreover, each gadget contributes a node if the respective equation is true. As a result, the new difficult question is whether the maximum independent set contains at least $(12+62-\epsilon)kn$ nodes, or at most $(12+61+\epsilon)kn$.

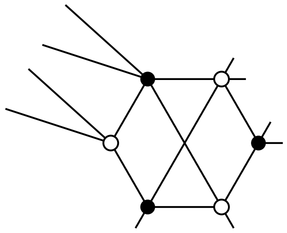

Figure 4: Contact gadget for 5-MIS.

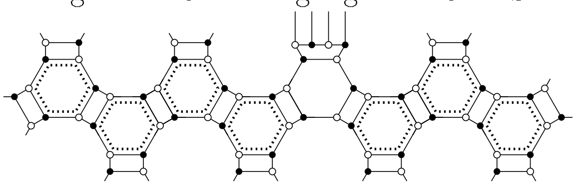

Figure 5: Consistency wheel for 3-MIS.

Theorem 10. *For any $\epsilon \in (0, 1/2)$, it is NP-hard to decide whether an instance of 4-MIS with $152n$ nodes has the maximum size of an independent set above $(74 - \epsilon)n$ or below $(73 + \epsilon)n$.*

Suppose now that we can reduce the size of the gadget corresponding to an equation with three variables so it consists of 10 nodes rather than 16, and it contributes 4 nodes to an independent set if the equation is false, and 5 if it is true. In this case the above accounting would show that for graphs with $140n$ nodes it is difficult to distinguish between those that have a maximum independent set with at least $(68 - \epsilon)n$ nodes and those that have at most $(67 + \epsilon)n$ nodes. We can achieve this by constructing the gadget for replacing contact nodes that has two nodes less than the one in Fig. 2. However, some nodes in this gadget have degree 5 (see Fig. 4 and hence the improved result, mentioned in Table 1, applies to 5-MIS (and, by extension, 5-Node Cover). Because in these instances only $12n$ nodes out of $140n$ have degree 5, we believe that this result should be easy to improve.

We can describe a similar reduction from HYBRID to 3-MIS. Given a HYBRID system of equations S, we form a graph G of degree 3. Again, each variable x of HYBRID is replaced with a gadget A_x; the gadget of a checker variable is a hexagon, and a gadget of a contact variable is a hexagon augmented with a *trapezoid*, a cycle of 6 nodes that shares one edge with the hexagon. The hexagons used here have no chords. If two variables/nodes x, y are connected by an equation/edge, $x = y$, we connect their hexagons with a pair of edges to form a rectangle in which the edges of the hexagons and the new edges alternate. The rectangle thus formed is a gadget of this equation. If three variables are connected by an equation/hyperedge, say, $x \oplus y \oplus z = 0$, the trapezoids of A_x, A_y and A_z are connected to four special nodes of the gadget of this equation. As a result, the gadget of this equation consists of 3 trapezoid and 4 special nodes, for the total of 22 nodes. The details are shown in Fig 5 and Fig. 6.

Given a solution of the new problem, and independent set I of G, we translate it into

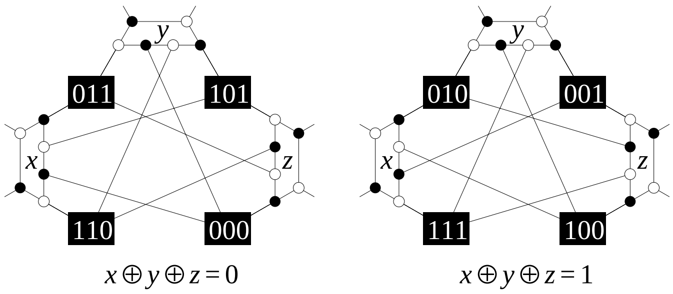

Figure 6: Equation gadgets for 3-MIS.

a solution of S in the same manner as before. Again, if some variable gadget are dirty, we need to purity them, so that this translation will be well-defined. The beginning of the purification is same as before: we purify dirty hexagon using the method illustrated in the lower right corner of Fig. 2. As a result, all checker gadgets become pure. We can also insists that if a checker variable x is connected to a contact y, I contains a node in the intersection of the gadget of this equation (a rectangle) with A_x.

Now we consider a contact gadget A_x. Of H_x is the hexagon of A_x, we will say that $A_x - H_x$ is the *front piece* of A_x and we use F_x to denote it. Before we proceed, we make the following observation:

Observation. Assume that $|F_x \cap I| = i$ and that A_u and A_v are the adjacent checker gadgets. We can modify I so that A_x becomes pure, $A_u \cap I$ and $A_v \cap I$ do not change, and the size of A_x increases by $2 - i$.

Now we can return to the gadget of $x \oplus y \oplus z = 0$. Our goal is that after all stages of the purifications, each variable gadgets is pure, each trapezoid contains 3 nodes of I and if the equation is satisfied (we can decide that once the the gadget variables are pure and the value translation is defined) its gadget contains the special node described by the triple of values of x, y and z; otherwise no special nodes belong to I. As a result, a satisfied equation corresponds to 10 nodes in I and an unsatisfied equation corresponds to 10. Moreover, a satisfied equation with two variables corresponds to 2 nodes in I. This will lead to the following accounting: the question whether we can satisfy at least $(62 - \epsilon)kn$ equations or at most $(61 + \epsilon)kn$, where $60kn$ equations have two variable translates into the question whether the maximum independent set has at least $(2 \times 60 + 20 - \epsilon)kn$ elements or at most $(2 \times 60 + 19 + \epsilon)kn$. This will lead to the following theorem:

The first case that we consider during the purification of an equation gadget is when I contains all 4 of its special nodes. In this case, $|F_v \cap I| = 0$ for $v = x, y, z$; according to the Observation, we can remove 4 special nodes form I, make all the participating contact gadgets pure and increase the size of I by at least 6-4. The second case is when I contains 3 of the special nodes; then $|F_v \cap I \leq 1$ for $v = x, y, z$; now throwing away the special nodes and purifying the contact gadgets increases the size of I be at least $3 - 3$. Lastly, when I contains two special nodes, we can remove one of them and, by Observation 1, purify one of the variable gadgets and restore the size of I (one can inspect all 6 cases to prove it). Thus at the end we need to consider only cases when I contains at most one special node (from a given equation gadget).

Theorem 11. *For any $\epsilon \in (0, 1/2)$, it is NP-hard to decide whether an instance of 3-MIS with $284n$ nodes has the maximum size of an independent set above $(140 - \epsilon)n$ or below $(139 + \epsilon)n$.*

6 From E2-LIN-2 to 4-MIS

An instance of 4-MIS can be modified to became an instance of BGD in a simple manner: each node can be replace with an alternating cycle of length 4; adjacent nodes will be replaced with a pair such cycles that have an edge (or two) in common. If we are "lucky", after the replacement we indeed obtain a breakpoint graph. Unfortunately, it is not possible to apply such transformation consistently to a graph from Fig. 3. We did not find other gadgets that can replace an equation with three variables and can later be replaced with a fragment of a breakpoint graph. Therefore we will be using a translation from τ_1(E2-LIN-2), shown in Fig 7.

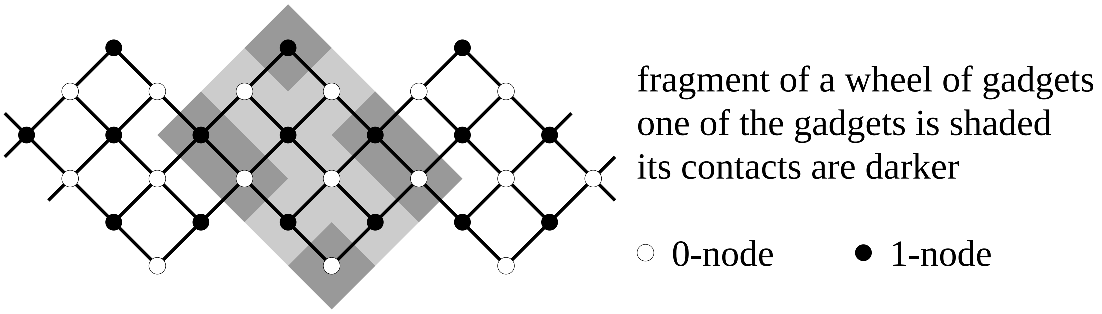

Figure 7: A part of 4-MIS instance obtained from τ_1(E2-LIN-2).

It is easy to see that the size of the resulting 4-MIS graph is $9n$, and that the correspondence between the size of the pure solution and the score in the original τ_1(E2-LIN-2) instance is $i = 3n + s$. The "purifying" normalization has to proceed somewhat different, however. We do it in two stages. The result of the first stage is that gadgets are either pure, or contain no nodes of I in their contacts.

If an impure gadget contains only 4 nodes of I (or less), we replace these nodes with the (unique) independent set of size 4 with no contact nodes (i.e. contained in the light gray area of Fig. 2b). A gadget that contains 6 nodes of the independent set is already pure. If an impure gadget contains 5 nodes of I, then it must contain one of the two "central" points (note that the non-central nodes form a cycle of length 10). Suppose that this central node has label 0. Then I cannot contain neither of the 4 adjacent 1-nodes, and the remaining 7 nodes form two isolated 0-nodes and a chain of the form 0-1-0-1-0, where the final 0-1 is a contact. If the chain contains 3 nodes of I, the gadget is pure. Otherwise we can set the intersection of I with this chain to contain two 0-nodes that do not belong to the contact; afterward the gadget becomes pure.

At this point, we have "pure" gadgets, with 0 or 1 values, and at least 5 nodes of I, and "undecided" gadgets that contain only 4 nodes of I. If an undecided gadget is adjacent to two gadgets that are either 0-pure or undecided, then we can increase I by increasing the number of nodes of I to 5, all of them 0. There is also symmetric case for 1, and one of the two cases must hold.

7 Reduction to BGD

The idea of reducing MIS problem to BGD is very simple and natural. Observe that the set E of all edges forms an alternating cycle (AC for short), a disjoint union of ACs is an AC, and a difference of two ACs, one contained in another is also an AC. Thus any disjoint collection of ACs can be extended to a decomposition of AC. Consequently, the goal of BGD is to find a collection of disjoint ACs as close in size to the maximum as possible.

Second observation is that the consequences of *not finding* an AC diminish with the size of AC. Suppose that the input has n breakpoints (edges of one color), and that we neglect to find any AC's with more than k breakpoints. The increase in the cost of the solution is smaller than n/k, while the cost is at least $n/2$. Thus if $k = \Omega(\log n)$, such oversight does not affect the approximation ratio.

The strategy suggested by these observation is to create instances of BGD in which alternating cycles that either have 2 breakpoints, or $\Omega(\log n)$. Then the task of approximating is equivalent to the one of maximizing the size of independent set in the graph $\mathcal{G}$ of all ACs of 4; we draw an edge between two ACs if they share an edge.

More to the point, we need to find a difficult family of graphs of degree 4 which can be converted into breakpoint graphs by replacing each node with an alternating cycle of size 4. To this end, we can use the results of the second reduction described in the previous section. Fig. 3 shows the result of this replacement applied to the long cycles of gadgets. The union of ACs used in the replacements is also a disjoint union of 5 ACs (in Fig. 3 these ACs are horizontal zigzags). To apply the reasoning of the previous sections, we need to establish that no cycles of length larger than 4 have to be considered. In the short version we only sketch this argument.

The cycles in question fall into three categories. The first kind of cycles are included in an adjacent pair of gadgets, identified on their diagonally placed corners. By an easy case analysis one can show that we can replace such cycles with a larger collection of cycles of size 4. The second kind traverses a collection of gadgets that is cycle-free (if each gadget is considered to be a node). Such a cycle has a defined interior; the union of the cycle with its interior can be easily decomposed into 4-cycles. The third and last kind traverses a cycle of gadgets. Then it must be at least as long as such a cycle, i.e. $\Omega(\log n)$.

At this point the translation is still not correct, as the resulting graphs MUST violated property (i) of BGD: edges of one kind form a collection of cycles: in Fig. 3 such edges form diagonal lines consisting of 5 edges each; such a line crosses to another strip of gadgets and then proceeds without end. However, these cycles induce cycles of gadgets, hence have length $\Omega(\log n)$, moreover, they are disjoint. Therefore we can remove all these cycles by breaking $O(n/\log n)$ contacts between the strips.

Given and instance G of τ_1(E2-LIN-2) with $2n$ nodes and $3n$ edges, this construction creates BGD instance G' with $20n$ breakpoints (edges of one color), and the correspondence between the cost c of a cycle decomposition in G' and s, *Score* of the corresponding solution of G is $c = 20n - 3n - s$. Together with Theorem 3 this implies

Theorem 12. *For any* $\epsilon \in (0, 1/2)$, *it is NP-hard to decide whether an instance of BGD with* $2240n$ *breakpoints has the minimum cost of an alternating cycle decomposition below* $(1236 + \epsilon)n$ *or above* $(1237 - \epsilon)n$.

8 Reduction to MIN-SBR

Our reduction from BGD to MIN-SBR is straightforward, in particular we can use the procedure GET-PERMUTATION of Caprara [C97, p.77] to obtain permutation $\pi(G)$ from a given breakpoint graph G. The number of reversals needed to sort the resulting permutation is equal to the number of black edges in G, minus the number of cycles in in the optimum cycle cover, plus the number of hurdles, plus the number of fortresses. Therefore the difference between the cost of solution for G differs from that for $\pi(G)$ by the number of hurdles and, possibly, 1. Now recall that we started from an instance of E2-LIN-2 problem with some n variables and m equations, $n < m$. Our instance of BGD has $\Theta(mn)$ nodes and edges. Below we will show that the number of hurdles not greater than n, and thus Theorem 12 applies also to MIN-SBR.

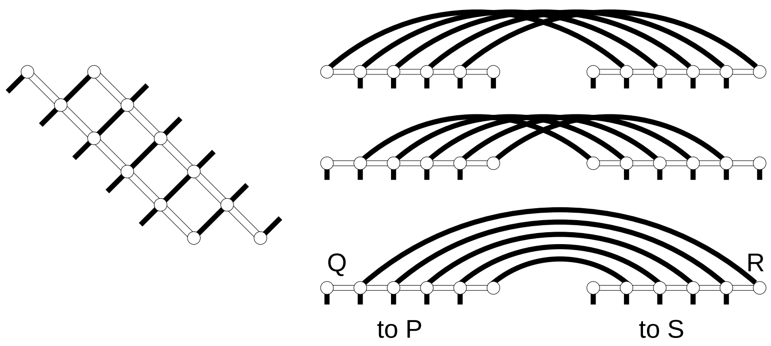

Figure 8: A slice and its possible embeddings.

The definition of hurdles is somewhat complicated, but we will use only a small part of it. GET-PERMUTATION embeds the graph of BGD in the following manner: vertices become consecutive numbers 1 to $|V|$, all white edges are between consecutive numbers and all black edges are between non-consecutive numbers, and permutation π satisfies the property that all black edges have the form $\{\pi(i), \pi(i+1)\}$. Hurdles are defined as certain equivalence classes of the transitive closure of *interleave* relation, defined on the black edges: if $i < j < k < l$, then $\{i, k\}$ and $\{j, l\}$ interleave. We will show that the number of these classes is bounded by n.

Observe that our BGD instance consists of n edge disjoint *wheels*. To show our claim, we will prove that all black edges from a single wheel must belong to the same equivalence class.

For the sake of our reasoning, we decompose a wheel (see Fig. 9) into *slices*. The left part of Fig. 8 depicts a slice; it consists of two white paths with 5 edges each, and 5 connecting black edges. Consecutive slices share a white path, but each black edge is in one slice only. The right part of Fig. 8 shows all ways in which a slice may be embedded by GET-PERMUTATION.

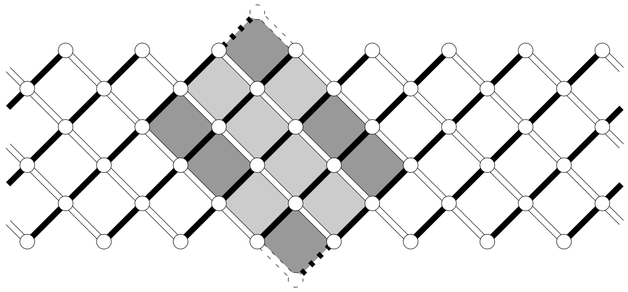

Figure 9: Gadget for breakpoint graphs.

Fist we will show that the black edges of a single slice must be in a single equivalence class. In the top two cases, it is evident, because each two of them interleave. In the remaining case, we must consider the placement of white paths of the neighboring slices, P and S, in relation to the white paths of this slice, Q and R (notation from Fig. 8, all figures are in Appendix A). If P lies between Q and R, then the "outer" black edge of PQ slice interleaves all black edges of QR slice. Similarly, if S does lie between Q and R, then the "inner" black edge of RS interleaves all black edges of QR. It remains to consider the case when P lies outside Q and R, and S inside. Then the "inner" black edge of PQ interleaves the top 3 black edges of QR, and the "outer" black edge of RS interleaves the bottom 3 black edges.

Now we can observe that black edges from consecutive slices must be in a single equivalence class. As we observe at the end of the previous reasoning, for each pair of consecutive slices, say PQ and QR, some of their black edges must interleave. Because slices of a single wheel are connected, our claim follows.

9 Further Research and Open Problems

It would very interesting to improve still huge gaps between approximation upper and lower bounds for bounded approximation problems of Table 1. The lower bound of 1.0008 for MIN-

SBR is the first inapproximability result for this problem. The especially huge gap between 1.5 and 1.0008 for the MIN-SBR problem reflects a great challenge for future improvements.

Acknowledgments

We thank Johan Håstad and Luca Trevisan for stimulating remarks on the preliminary version of this paper.

References

[AFWZ95] N. Alon, U. Feige, A. Wigderson and D. Zuckerman, *Derandomized Graph Products*, Computational Complexity **5** (1995), pp. 60–75.

[A94] S. Arora, *Probabilistic Checking of Proofs and Hardness of Approximation Problems*, Ph. D. Thesis, UC Berkeley, 1994;
available as TR94-476 at `ftp://ftp.cs.princeton.edu`

[ALMSS92] S. Arora, C. Lund, R. Motwani, M. Sudan and M. Szegedy, *Proof Verification and Hardness of Approximation Problems*, Proc. 33rd IEEE FOCS (1992), pp. 14–23.

[BP96] V. Bafna and P. Pevzner, *Genome rearrangements and sorting by reversals*, SIAM J. on Computing **25** (1996), pp. 272–289.

[BF95] P. Berman and T. Fujito, *Approximating Independent Sets in Degree 3 Graphs*, Proc. 4th Workshop on Algorithms and Data Structures, LNCS Vol. 955, Springer-Verlag, 1995, pp. 449–460.

[BF94] P. Berman and M. Fürer, *Approximating Maximum Independent Set in Bounded Degree Graphs*, Proc. 5th ACM-SIAM SODA (1994), pp. 365–371.

[BH96] P. Berman and S. Hannenhali, *Fast Sorting by Reversals*, Proc. 7th Symp. on Combinatorial Pattern Matching, 1996, pp. 168–185.

[BS92] P. Berman and G. Schnitger, *On the Complexity of Approximating the Independent Set Problem*, Information and Computation **96** (1992), pp. 77–94.

[B78] B. Bollobás, *Extremal Graph Theory*, 1978, Academic Press.

[C97] A. Caprara, *Sorting by reversals is difficult*, Proc. 1st ACM RECOMB (Int. Conf. on Computational Molecular Biology), 1997, pp. 75–83.

[C98] D.A. Christie, *A 3/2-Approximation Algorithm for Sorting by Reversals*, Proc. 9th ACM-SIAM SODA (1998), pp. 244–252.

[CB95] D. Cohen and M. Blum, *Improved Bounds for Sorting Burnt Pancakes*, Discrete Applied Mathematics **61** (1995) pp. 105-125.

[CK97] P. Crescenzi and V. Kann, *A Compendium of NP Optimization Problems*, Manuscript, 1997;
available at `http://www.nada.kth.se/theory/problemlist.html`

[FG95] U. Feige and M. Goemans, *Approximating the Value of Two Prover Proof Systems with Applications to MAX-2SAT and MAX-DICUT*, Proc. 3rd Israel Symp. on Theory of Computing and Systems, 1995, pp. 182–189.

[GP79] W.H. Gates, and C.H. Papadimitriou, *Bounds for Sorting by Prefix Reversals*, Discrete Mathematics, **27** (1979), pp. 47–57.

[GW94] M. Goemans and D. Williamson, *.878-Approximation Algorithms for MAX CUT and MAX 2SAT*, Proc. 26th ACM STOC (1994), pp. 422–431.

[H96] J. Håstad, *Clique is Hard to Approximate within* $n^{1-\epsilon}$, Proc. 37th IEEE FOCS (1996), pp. 627–636.

[H97] J. Håstad, *Some optimal Inapproximability results*, Proc. 29th ACM STOC, 1997, pp. 1–10.

[HP95] S. Hannenhali and P. Pevzner, *Transforming Cabbage into Turnip (Polynomial time algorithm for sorting by reversals)*, Proc. 27th ACM STOC (1995), pp. 178–187.

[KST97] H. Kaplan, R. Shamir and R.E. Tarjan, *Faster and simpler algorithm for sorting signed permutations by reversals*, Proc. 8th ACM-SIAM SODA, 1997, pp. 344–351.

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

[TSSW96] L. Trevisan, G. Sorkin, M. Sudan and D. Williamson, *Gadgets, Approximation and Linear Programming*, Proc. 37th IEEE FOCS (1996), pp. 617–626.