Technical Reports on Mathematical and Computing Sciences: TR-CXXX
title: Instance generating algorithms for MAX2SAT with optimal solutions
author: Masaki Yamamoto
affiliation:

Dept. of Mathematical and Computing Sciences, Tokyo Institute of Technology
Meguro-ku Ookayama, Tokyo 152-8552, Japan
(`masaki.yamamoto@is.titech.ac.jp`)

Abstract: Instance generating algorithms for MAX2SAT are proposed. The algorithms produce instances with their optimal solutions. We propose two types of algorithms: a constructive one and a randomized one. In the former, nontrivial 2-CNF instances are produced using expander graphs explicitly constructed. On the other hand, in the latter, more complicated instances are produced although it is with high probability that produced solutions are optimal ones.

1 Introduction

For testing the performance of an algorithm for a given problem, it would be useful if we could generate typical instances of the problem with their solutions. A "test instance generator", an instance generator for short, is an algorithm that generates typical instances with their solutions. In this paper, we propose instance generators for MAX2SAT problem, one of the well-known NP-hard optimization problems [1].

We first recall this problem. Given a set X of Boolean *variables* $x_1, \cdots, x_n$, which can take either 1 (meaning *true*) or 0 (meaning *false*), a *literal* over X is a variable $x_i \in X$ or a negated variable $\overline{x_i}$ of $x_i \in X$, and a *clause* is a disjunction of literals, e.g., $(x_1 \vee \overline{x_2} \vee x_3)$. The size of a clause is the number of literals in the clause. A *CNF* (abbrev. for *Conjunctive Normal Form*) formula is a conjunction of clauses, e.g., $C_1 \wedge C_2 \wedge \cdots$, where each C_i, $i = 1, 2, \cdots$, is a clause. A *k-CNF* formula is a CNF formula such that the size of each clause is at most k. In this paper, we will focus on 2-CNF formulas.

A *truth assignment* (or simply, an *assignment*) over X is a map from X to $\{0, 1\}$. We say that an assignment satisfies a clause if it is evaluated 1 under the assignment. If an assignment satisfies all clauses in a formula, the formula is called *satisfiable*. Let F be a 2-CNF formula. An assignment for F is called a *maximum assignment* if it satisfies the largest number of clauses among all assignments. Then, we consider *MAX2SAT* problem, which asks the following question: Given a 2-CNF formula, obtain a maximum assignment. We call a maximum assignment an *optimal solution*, and the number of clauses of F satisfied under the assignment an *optimal value* denoted by $\mathrm{OPT}(F)$.

Throughout this paper, We will use n and m to denote the number of variables and clauses, respectively. (Depending on the context, symbols n and m are also used to denote the number of vertices and edges of a graph, respectively.)

We next describe a test instance generator that we demand. In general, an *instance generator* for MAX2SAT is an algorithm that takes n, the number of variables, and produces a pair of a 2-CNF formula and its maximum assignment. This is a basic description indeed, and for a "reasonable instance generator", we would like to require several properties for produced 2-CNF formulas.

It would be nice if we could produce all MAX2SAT instances (i.e., 2-CNF formulas) with their maximum assignments. As the following simple proposition suggests, however, there doesn't seem to exist a polynomial time algorithm satisfying this requirement.

Proposition 1.1 Unless NP = co-NP, no polynomial time instance generator `A` exists that satisfies the following for all n: $\mathtt{A}(n)$ produces every 2-CNF formula of n variables with some positive probability. (Note that we can bound the number of clauses by $2n(n-1)$, by suppressing redundant clauses; hence, the set of 2-CNF formulas of n variables is finite.)

For any 2-CNF formula F and for any assignment T, let $\mathrm{unsat}_T(F)$ be a set of unsatisfied clauses of F under T. We denote by $\mathrm{hsize}(F)$ the *hole size* of F, that is, the number of unsatisfied clauses by the maximum assignment. In this paper, we try to produce formulas with many unsatisfiable clauses while keeping the total number of clauses linear to n. More specifically, for some constant $\alpha > 0$, $\beta > 0$, we require the following conditions for all produced formulas F:

$$(1)\ |F| \le \alpha n, \quad \text{and} \quad (2)\ \mathrm{hsize}(F) \ge \beta |F|.$$

Since $\mathrm{hsize}(F) \le 0.25|F|$ for any F such that F has only clauses of size 2, β must be at most 0.25. There sure is a trivial generator satisfying these conditions; choose randomly two variables x and y, and produce the following four clauses: $(x \vee y)(\bar{x} \vee y)(x \vee \bar{y})(\bar{x} \vee \bar{y})$. But any formula produced by such a trivial generator has a simple structure; in fact, any assignment is one of the maximum assignments. Here we try to produce formulas with no such easy structures.

Our idea is to use graphs with a certain "expansion" property. Here we explain the idea briefly. We first construct a directed graph G (with some nice property), and then use it to produce a 2-CNF instance as follows. First label every vertex of G by some literal (randomly). Then regard every edge (u, v) of G as a logical expression $l(u) \Rightarrow l(v)$ (where $l(u)$ and $l(v)$ are the literals associated with u and v), which is equivalent to a clause $(\overline{l(u)} \vee l(v))$. The 2-CNF instance is a collection of such clauses.

Suppose that an instance F with n variables $X = \{x_1, ..., x_n\}$ is obtained from a directed graph $G = (V, E)$ with $|V| = n$. For simplicity, let us further suppose[1] all vertices $v_i \in V$ are simply labeled with a positive literal $x_i \in X$. Now consider any assignment T to the variables in X. The vertex set is divided to two sets; namely P and N, which are vertices assigned true and false, respectively. Then under this assignment T, a clause produced from an edge (v_i, v_j) is unsatisfied if and only if $v_i \in P$ and $v_j \in N$. Hence, the number of unsatisfied clauses under T is the number of edges from P to N, i.e., the number of cut edges.

This observation suggests that in order to have an instance F with large $\mathrm{hsize}(F)$, we need a graph with enough number of cut edges for all partitions. More specifically, we would like to have a graph with the following property for some constant δ.

$$\forall S \subset V \Big[\ |S \times (V - S) \cap E| \ge \delta \cdot \min\{|S|, |V - S|\}\ \Big].$$

Furthermore, in order to keep the number of clauses linear to n, we require that the number of edges is linearly bounded by n. We call a graph with such properties an δ*-enlarger*. Recall that we also need to provide a solution, i.e., a maximum assignment; for this, we also need the

[1]This is just for our explanation. In fact, this labeling produces a trivial instance; that is, under this labeling, all produced clauses can be satisfied by simply assigning true to all variables.

partition that achieves the above bound. An enlarger graph with such a partition is called an *exact δ-enlarger.*

In this paper, based on the explicit construction of expanders [3], we propose an instance generator that constructs exact 1/2-enlargers and then produces a 2-CNF instance (with its maximum assignment). We show that it is also possible to construct exact 1/2-enlargers *randomly.* By using such random graphs, we can produce 2-CNF instances with the following two advantages: (i) they are "random" 2-CNF instances with no structure, and (ii) they have a better ratio β, because less number of clauses is required. On the other hand, we can only guarantee that the produced assignment is a maximum assignment *if* all random graphs used to produce the 2-CNF formula are exact 1/2-enlargers; but we can show that, under a certain edge degree parameter of random graphs, this occurs with high probability.

2 Expander graph

As explained in Introduction, we would like to produce instances with many unsatisfiable clauses while keeping the size linear to n. For producing such instances we use a graph called an *expander.* In the definition below, for a bipartite graph $G = (U, V, E)$ such that $U = \{u_1, \cdots, u_n\}, V = \{v_1, \cdots, v_n\}$, we use $\Delta(S)$ as follows:

$$\Delta(S) = \{(u_i, v_j) \in E | u_i \in S \text{ and } i \neq j \text{ where } u_j \in S\}.$$

Definition 2.1 An (n, c, δ)-*expander* is a (an undirected) bipartite graph $G = (U, V, E)$ such that $|U| = |V| = n$, $|E| \leq kn$, and

$$\forall S \subset U \left[|\Delta(S)| \geq \delta \cdot \left(1 - \frac{|S|}{n}\right) |S| \right].$$

An expander may have multiple edges.

The definition above is for the number of edges from $S \subset V$ to vertices in V having the different index numbers from S while an expander defined in [3] is for the number of neighbors. But we can also obtain for our definition the same theorem as [3].

According to [3], Margulis [2] first found a way to explicitly construct a family of expanders for $k = 5$, although the constant value δ was not known. On the other hand, Gabber and Galil [3] proposed similar constructions for some specified value of δ. Here we explain one of their constructions, which we will use for our instance generator.

Let $n = h^2$ for any integer $h \geq 1$, and A be $\{0, 1, \cdots, h-1\} \times \{0, 1, \cdots, h-1\}$. Consider the following seven permutations $\sigma_0, \cdots, \sigma_6$ on A: for each $(x, y) \in A$,

$$\begin{aligned}
&\sigma_0(x, y) = (x, y), \\
&\sigma_1(x, y) = (x, y + 2x), \quad \sigma_2(x, y) = (x, y + 2x + 1), \quad \sigma_3(x, y) = (x, y + 2x + 2), \\
&\sigma_4(x, y) = (x + 2y, y), \quad \sigma_5(x, y) = (x + 2y + 1, y), \quad \sigma_6(x, y) = (x + 2y + 2, y),
\end{aligned}$$

where the $+$ is modulo h. Then define a bipartite graph $G = (U, V, E)$ as follows: $U = V = A$, and $E = \{((x, y), (x', y')) | (x', y') = \sigma_i(x, y) \text{ for some } i,\ 0 \leq i \leq 6\}$. Here we call this construction `GG` *method.* Note that G may have multiple edges between some pairs of vertices $(x, y) \in U$ and $(x', y') \in V$. Clearly, it is computationally easy to construct such a graph G from a given n. Gabber and Galil showed that G is indeed an expander in the following sense.

Theorem 2.1 (Gabber and Galil [3]) For any n, the graph G defined above is an $(n, 7, \delta)$-expander with $\delta = (2 - \sqrt{3})/2$.

From now on, we simply call an (n, c, δ)-expander a δ-expander in the case that the values n and c are given in the context.

As is shown in the next section, our generator needs an δ-expander such that $\delta \geq 2/3$. The graph G given by `GG` method is not such one because $(2 - \sqrt{3})/2 < 2/3$; but we can easily construct a 2/3-expander by merging l Gabber-Galil expanders, where $l = \lceil (2/3) \div ((2 - \sqrt{3})/2) \rceil = 5$. More specifically, by using random permutations on vertices, we construct l isomorphic Gabber-Galil expanders and merge them to a single graph, thereby yielding a 2/3-expander. Note that this constructed expander has $l \cdot 7n$ $(= 35n)$ edges; that is, it is an $(n, 35, 2/3)$-expander. In the following sections, we denote such an expander (n, c_0, δ_0)-expander, where $c_0 = 35$ and $\delta_0 = 2/3$.

3 Explicit Construction of Exact Enlarger

In this section, we introduce our new notions called an "enlarger" and an "exact enlarger" and show their constructions.

As defined in the preceding section, an expander $G = (U, V, E)$ is an undirected bipartite graph with a property that every subset S of U has enough number of edges between S and the opposite side of S (i.e., V). Intuitively, an "enlarger" is a directed graph having such a property.

Definition 3.1 An (n, c, δ)-*enlarger* $G = (V, E)$ is a directed graph such that $|V| = n$, $|E| \leq cn$, and

$$\forall S \subset V : \ |S| \leq \frac{n}{2} \ \left[\ |S \times (V - S) \cap E| \geq \delta |S|, \ \text{ and } \ |(V - S) \times S \cap E| \geq \delta |S| \ \right].$$

An enlarger may have multiple edges, but doesn't have any self-loops.

Again, we simply call an (n, c, δ)-enlarger a δ-enlarger in case that the values n and c are given in the context.

We construct an $(n, 2c_0, \delta_0)$-enlarger from two isomorphic (n, c_0, δ_0)-expanders as follows. Recall that for constructing any (n, c, δ)-expander by `GG` method the number n of vertices must be h^2 for some integer h. In the following construction, we assume that an input number n satisfies $n = h^2$ for some integer h.

Algorithm 1 Construction of an enlarger

Input: A number n of vertices.

step 1: By `GG` method, construct an (n, c_0, δ_0) - expander $G_{out} = (U, W, E_{out})$, where $U = \{u_1, \cdots, u_n\}$ and $W = \{w_1, \cdots, w_n\}$.

step 2: Choose a random permutation π over W, and construct an (n, c_0, δ_0) - expander $G_{in} = (U, W_\pi, E_{in})$, where $U = \{u_1, \cdots, u_n\}$ and $W_\pi = \{w_{\pi(1)}, \cdots, w_{\pi(n)}\}$.

step 3: Construct a directed graph $G = (V, E)$ such that $V = \{v_1, \cdots, v_n\}$ and $E = E' \cup E''$, where

$$\begin{aligned} E' &= \{(v_i, v_j) \ : \ (u_i, w_j) \in E_{out} \text{ and } i \neq j\}, \quad \text{and} \\ E'' &= \left\{(v_{\pi(j)}, v_i) \ : \ (u_i, w_{\pi(j)}) \in E_{in} \text{ and } i \neq \pi(j)\right\}. \end{aligned}$$

In this algorithm, we used a permutated W_π in order to prevent a produced formula from being trivial. We could use as G_{in} a graph identical to G_{out}. But an enlarger G constructed in this way has a very simple structure; that is, the graph G has an edge (u, v) if and only if it has an edge (v, u). Such an instance generator would result in producing trivial instances.

Lemma 3.1 Any graph $G = (V, E)$ constructed by **Algorithm 1** for some n, we have that for any subset S of V:

$$|S \times (V - S) \cap E| \geq 2\delta_0 \left(1 - \frac{|S|}{n}\right) |S|, \quad \text{and} \quad |(V - S) \times S \cap E| \geq 2\delta_0 \left(1 - \frac{|S|}{n}\right) |S|.$$

Proof. Since $G_{out} = (U, W, E_{out})$ is a δ_0-expander, by definition, for any subset S of V, we have

$$|S \times (V - S) \cap E| \geq \delta_0 \left(1 - \frac{|S|}{n}\right) |S|.$$

Similarly, since $G_{in} = (U, W_\pi, E_{in})$ is also a δ_0-expander, by definition and the construction, for any subset $V - S$ of V, we have

$$|S \times (V - S) \cap E| \geq \delta_0 \left(1 - \frac{|V - S|}{n}\right) |V - S|.$$

Therefore, by combining two inequalities, we have

$$\begin{aligned}
|S \times (V - S) \cap E| &\geq \delta_0 \left(1 - \frac{|S|}{n}\right) |S| + \delta_0 \left(1 - \frac{|V - S|}{n}\right) |V - S| \\
&= \delta_0 \left(|S| - \frac{|S|^2}{n} + (n - |S|) - \frac{(n - |S|)^2}{n}\right) \\
&= 2\delta_0 \left(1 - \frac{|S|}{n}\right) |S|.
\end{aligned}$$

A similar argument holds for $|(V - S) \times S \cap E|$. ■

Since from the above lemma for any subset S of V such that $0 \leq |S| \leq n/2$, $2\delta_0(1 - S|/n)|S| \geq \delta_0|S|$ and $|E| \leq 2c_0 n$, we have that any graph $G = (V, E)$ constructed by **Algorithm 1** is an $(n, 2c_0, \delta_0)$-enlarger.

By the definition of enlarger, for any δ)-enlarger $G = (V, E)$, and for any partitions $S, T \subset V$ (i.e., $S \cup T = V$ and $S \cap T = \phi$) such that $|S| = |T|$, the following inequalities hold.

$$|S \times T \cap E| \geq \left\lceil \delta \frac{n}{2} \right\rceil, \quad \text{and} \quad |T \times S \cap E| \geq \left\lceil \delta \frac{n}{2} \right\rceil.$$

For our generator, among those partitions (S, T) we need a δ-enlarger having an "optimal" partition for which the equalities hold in the above inequalities, which we call an "exact enlarger".

Definition 3.2 An *exact* (n, c, δ)*-enlarger* $G = (V, E)$ is an (n, c, δ)-enlarger having a partition (U, W) (i.e., $U \cup W = V$ and $U \cap W = \phi$) with $|U| = |V|$ such that

$$|U \times W \cap E| = \left\lceil \delta \frac{n}{2} \right\rceil, \quad \text{and} \quad |W \times U \cap E| = \left\lceil \delta \frac{n}{2} \right\rceil.$$

We construct an exact $(n, 2c_0{+}1/2, 1/2)$-enlarger from two $(n/2, 2c_0, \delta_0)$-enlargers as follows. Recall that for constructing any $(n/2, c, \delta)$-enlarger by **Algorithm 1** the number $n/2$ of vertices must be h^2 for some integer h. In addition, in the following construction we assume an input number n satisfies $n = 2h^2$ for some *even* h.

Algorithm 2 Construction of an exact enlarger

Input: A number n of vertices.

step 1: By **Algorithm 1** construct two $(n/2, 2c_0, \delta_0)$-enlargers $L = (U, E_l)$ and $R = (W, E_r)$, where $U = \{u_1, \cdots, u_{n/2}\}$, $W = \{w_1, \cdots, w_{n/2}\}$.

step 2: Construct a graph $G = (V, E)$ such that $V = U \cup W$ and $E = E_l \cup E_r \cup A \cup B$, where

$$A = \{(u_i, w_i) : 1 \le i \le n/4\}, \text{ and } B = \{(w_i, u_i) : n/4 + 1 \le i \le n/2\}.$$

Lemma 3.2 Any graph $G = (V, E)$ constructed by **Algorithm 2** for some n is an $(n, 2c_0 + 1/2, 1/2)$-enlarger.

Proof. Consider a graph $G = (V, E)$ constructed with $|V| = n$. We first count the number of edges of G. Since G is composed of two $(n/2, 2c_0, \delta_0)$-enlargers and additional $n/2$ edges, the number of edges is $2 \cdot 2c_0 n/2 + n/2$, which is $(2c_0 + 1/2)|V|$, where $|V| = n$.

Next we show that G is a $1/2$-enlarger. Consider any $S \subset V$ of size at most $n/2$. Let $S = S_U \cup S_W$, where $S_U \subset U$, $S_W \subset W$. We count the number of edges from S to $V - S$. (A similar argument holds for $(V - S) \times S \cap E$.) Since $|S| \le n/2$, and the construction of G is symmetric, it suffices to consider the following two cases:

$$(I)\ |S_U|, |S_W| \le \frac{n}{4}, \quad \text{and } (II)\ |S_U| \ge \frac{n}{4} \quad \text{and} \quad |S_W| \le \frac{n}{4}.$$

In the first case, by the definition of δ_0-expander, it follows that

$$|S_U \times (U - S_U) \cap E_l| \ge \delta_0 |S_U|, \text{ and } |S_W \times (W - S_W) \cap E_r| \ge \delta_0 |S_W|.$$

By combining the two inequalities, we have

$$|S \times (V - S) \cap (E_l \cup E_r)| \ge \delta_0(|S_U| + |S_W|).$$

Therefore, the number of edges is,

$$\begin{aligned} |S \times (V - S) \cap E| &= |S \times (V - S) \cap (E_l \cup E_r \cup A \cup B)| \\ &\ge |S \times (V - S) \cap (E_l \cup E_r)| \\ &\ge \delta_0(|S_U| + |S_W|) \ge |S|/2 \quad (\because \delta_0 = 2/3)\,. \end{aligned}$$

In the second case, by **Lemma 3.1** we have

$$\begin{aligned} &|S_U \times (U - S_U) \cap E_l| \ge 2\delta_0 \left(|S_U| - \frac{|S_U|^2}{n/2} \right), \quad \text{and} \\ &|S_W \times (W - S_W) \cap E_r| \ge 2\delta_0 \left(|S_W| - \frac{|S_W|^2}{n/2} \right). \end{aligned}$$

By combining the two inequalities, we have

$$|S \times (V - S) \cap (E_l \cup E_r)| \ge 2\delta_0 \left(|S_U| + |S_W| - \frac{|S_U|^2}{n/2} - \frac{|S_W|^2}{n/2} \right). \tag{1}$$

This bound is for the number of edges from S to $V - S$ *within* the subgraphs L and R. There are some more edges as a whole; that is, those of $A \cup B$. For estimating the lower bound of the number of the additional edges, we let for some x and y, $0 \le x, y \le 1$, $|S_U| = n/4 + nx/4$, and $|S_W| = ny/4$, and consider the minimum number of those edges for fixed values of x and y. From the construction of A and B, it is easy to see the following claim holds. (See Fig.1.)

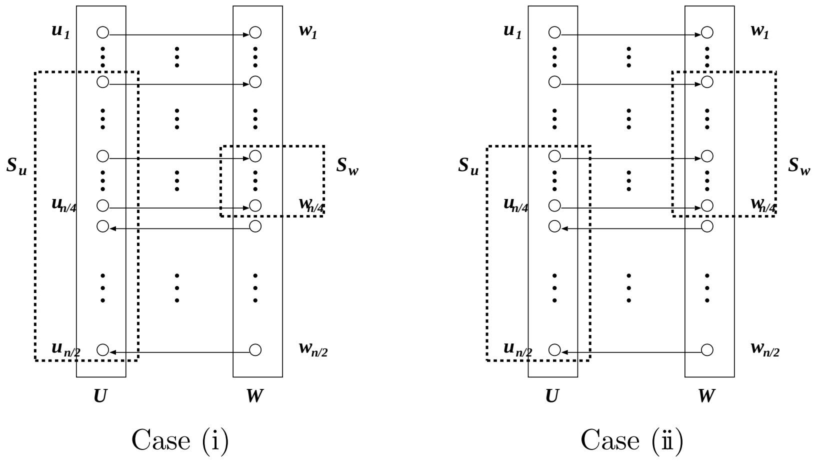

Figure 1: Two Cases

Claim 1 For fixed values of x and y, $|S \times (V-S) \cap (A \cup B)|$ is minimized in case that $S_U \supset \{u_{n/4+1}, \cdots, u_{n/2}\}$, $S_W \subset \{w_1, \cdots, w_{n/4}\}$, and one of the followings holds.

$$\begin{array}{ll} \text{Case (i) (i.e., } x \geq y): & \forall (u,v) \in A\big[v \in S_W \Longrightarrow u \in S'_U\big], \\ \text{Case (ii) (i.e., } x \leq y): & \forall (u,v) \in A\big[u \in S'_U \Longrightarrow v \in S_W\big], \end{array}$$

where $S'_U = S_U - \{u_{n/4+1}, \cdots, u_n\}$.

We show that G is a 1/2-enlarger for any x, y, $0 \leq x, y \leq 1$ in both cases. In Case (i), it is easy to see from Fig.1 that the following holds: $|S \times (V-S) \cap A| = (x-y)n/4$. By combining this equality and (1), we have

$$\begin{aligned} |S \times (V-S) \cap E| &\geq 2\delta_0 \left(|S_U| + |S_W| - \frac{|S_U|^2}{n/2} - \frac{|S_W|^2}{n/2} \right) + \frac{x-y}{4} n \\ &= \frac{n}{4} \left(\delta_0 (1 - x^2 - y^2 + 2y) + x - y \right). \end{aligned}$$

Recall $|S| = n/4 + (n/4)x + (n/4)y$; thus, for proving that G is a 1/2-enlarger, it suffices to show

$$\frac{n}{4} \left(\delta_0 (1 - x^2 - y^2 + 2y) + x - y \right) \geq \frac{1}{2} \left(\frac{n}{4} + \frac{n}{4}x + \frac{n}{4}y \right).$$

This inequality is equivalent to the following one by applying $\delta_0 = 2/3$.

$$\left(x - \frac{3}{8} \right)^2 + \left(y + \frac{1}{8} \right)^2 \leq \frac{13}{32} \tag{2}$$

That means G is a 1/2-enlarger if (x,y) is within the circle defined by the above.

On the other hand, in Case (i) x and y must satisfy the following bounds: $y \geq 0$, $y \leq x$, $y \leq -x+1$. (The last bound follows from $|S| = |S_U| + |S_W| \leq n/2$.) In other words, the domain of (x,y) is within the triangle defined by these inequalities. Note that this triangle is within the circle defined by (2) since its three vertices $(0,0)$, $(1,0)$, $(1/2,1/2)$ satisfy (2). Thus, any pair of x and y in the domain satisfies (2).

In Case (ii), it is easy to see from Fig.1 that the following holds: $|S \times (V-S) \cap A| = 0$. By combining this equality and (1), we have

$$|S \times (V-S) \cap E| = \frac{n}{4} \delta_0 (1 - x^2 - y^2 + 2y).$$

Likewise to Case (i), for proving that G is a 1/2-enlarger, it suffices to show

$$\frac{n}{4}\delta_0(1-x^2-y^2+2y) \geq \frac{1}{2}\left(\frac{n}{4}+\frac{n}{4}x+\frac{n}{4}y\right),$$

that is equivalent to

$$\left(x+\frac{3}{8}\right)^2+\left(y-\frac{5}{8}\right)^2 \leq \frac{25}{32}. \tag{3}$$

On the other hand, in Case (ii) x and y must satisfy the following bounds: $x \geq 0$, $y \geq x$, $y \leq -x+1$. Again, any pair of x and y in the domain defined by these inequality satisfies (3).

Thus, $G = (V, E)$ is a 1/2-enlarger with n vertices and $(2c + 1/2)|V|$ edges, that is, a $(n, 2c + 1/2, 1/2)$-enlarger. ■

From the construction, it is clear that (U, W) is a partition for which the number of edges between U and W is exactly $1/2 \cdot n/2$ in both directions. This proves the following main theorem of this section.

Theorem 3.3 For any $G = (V, E)$ constructed by **Algorithm 2** for some n, G is an exact $(n, 2c_0 + 1/2, 1/2)$-enlarger.

4 Generating algorithms

In this section, we show two instance generators for MAX2SAT. First we show the one using an exact 1/2-enlarger constructed in the previous section. Then we discuss the possibility of using random graphs.

4.1 Constructive generator

For generating an instance of n variables, we will use exact 1/2-enlargers of n vertices. Recall that for our construction of an exact enlarger we needed n to be h^2 for some even h. Thus, we will also assume this condition on n below.

Algorithm 3 Constructive instance generator – `GEN` –

Input: A number n of variables.

step 1: Divide a vertices set $V = \{v_1, \cdots, v_n\}$ into two sets $U = \{v_1, \cdots, v_{n/2}\}$ and $W = \{v_{n/2+1}, \cdots, v_n\}$.

step 2: By **Algorithm 2** construct independently six exact $(n, 2c_0 + 1/2, 1/2)$-enlargers $G_k = (V, E_k)$ for $k = 1, \cdots, 6$.

step 3: Choose a random permutation π over a variable set X ($|X| = n$), and label each vertex of the six graphs as follows.

$$\begin{array}{ll}
l_1(v_i) = x_{\pi(i)} & \text{for } v_i \in V \text{ of } G_1, \\
l_2(v_i) = x_{\pi(i)} & \text{for } v_i \in U \text{ of } G_2, \\
l_2(v_i) = \overline{x_{\pi(i)}} & \text{for } v_i \in W \text{ of } G_2, \\
l_3(v_i) = \overline{x_{\pi(i)}} & \text{for } v_i \in U \text{ of } G_3, \\
l_3(v_i) = x_{\pi(i)} & \text{for } v_i \in W \text{ of } G_3, \\
l_4(v_i) = \overline{x_{\pi(i)}} & \text{for } v_i \in V \text{ of } G_4, \\
l_5(v_i) = x_{\pi(i)} & \text{for } v_i \in U \text{ of } G_5, \\
l_5(v_i) = \overline{x_{\pi'(\pi(i-n/2))}} & \text{for } v_i \in W \text{ of } G_5, \\
l_6(v_i) = x_{\pi''(\pi(i+n/2))} & \text{for } v_i \in U \text{ of } G_6, \\
l_6(v_i) = \overline{x_{\pi(i)}} & \text{for } v_i \in W \text{ of } G_6,
\end{array}$$

where l_k is a label function for G_k, and π' and π'' are permutations from $\{\pi(i) : 1 \le i \le n/2\}$ to $\{\pi(i) : 1 \le i \le n/2\}$ and from $\{\pi(i) : n/2+1 \le i \le n\}$ to $\{\pi(i) : n/2+1 \le i \le n\}$, respectively.

step 4: Define a set F of clauses as follows.

$$F = \left\{ \left(\overline{l_k(u)} \vee l_k(v) \right) \ : \ (u,v) \in E_k, k = 1, \cdots, 6 \right\}.$$

Also define an assignment T_0 as $T_0(x_1) = 1, \cdots, T_0(x_n) = 1$.

step 5: Output F as an instance and T_0 as a maximum assignment.

Theorem 4.1 For any pair of set F and assignment T_0 produced by GEN, we have $\mathrm{OPT}(F) = m - n$, and T_0 is a maximum assignment for F.

Proof. W.l.o.g. we may fix the random permutation π used by GEN to the identity permutation, that is, $\pi(x_1) = x_1, \cdots, \pi(x_n) = x_n$. Let T an arbitrary assignment over X. By the assignment T, we assume U and W are divided into sets A, B, C, D of vertices as follows.

$$\begin{array}{lllclll}
A & = & \{v_i \in U \ : \ T(x_i) = \text{true}\}, & & B & = & \{v_i \in W \ : \ T(x_i) = \text{true}\} \\
C & = & \{v_i \in U \ : \ T(x_i) = \text{false}\}, & \text{and} & D & = & \{v_i \in W \ : \ T(x_i) = \text{false}\}.
\end{array}$$

We denote by a, b, c, d the size of A, B, C, D, respectively.

We estimate $|\mathrm{unsat}_T(F)|$, the number of F's clauses unsatisfied by T. Recall that each clause of F is produced from an edge (u,v) of some G_k. For example, consider the case that (u,v) is an edge of G_2. Then we have that a clause produced from (u,v) is unsatisfied if and only if $l_2(u) = 1$ and $l_2(v) = 0$ under T, and further iff $u \in U_1 \cup W_0$ and $v \in U_0 \cup W_1$, where U_0 and U_1 is a subset of U containing all vertices which are labeled literals assigned under T false and true, respectively. (W_0 and W_1 are similarly defined.) That is, (u,v) is a cut edge from $N = U_0 \cup W_1$ to $P = U_1 \cup W_0$. Since G_2 is an exact 1/2-enlarger, the total number of such edges of G_2 is at least

$$\frac{1}{2} \min\{|U_0| + |W_1|, |U_1| + |W_0|\} = \frac{1}{2} \min\{a+d, b+c\}.$$

In other words, there are at least this number of clauses (associated with edges of G_2) that are unsatisfied under T. By similar arguments, we have

$$\mathrm{unsat}_T(F_{G_1}) \ge \frac{1}{2} \min\{a+b, c+d\}, \quad \mathrm{unsat}_T(F_{G_3}) \ge \frac{1}{2} \min\{a+d, b+c\},$$
$$\mathrm{unsat}_T(F_{G_4}) \ge \frac{1}{2} \min\{a+b, c+d\},$$

where F_{G_k} is a set of clauses associated with edges of G_k. On the other hand, the number of unsatisfied clauses produced from G_5, G_6 is $(a+c)/2$ and $(b+d)/2$ respectively. Thus, for the total number of unsatisfied clauses under T, we have

$$\begin{aligned}
|\mathrm{unsat}_T(F)| &= \sum_k |\mathrm{unsat}_T(F_{G_k})| \\
&\geq \min\{a+b, c+d\} + \min\{a+d, b+c\} + (a+c)/2 + (b+d)/2 \\
&= \min\{2a+b+d, 2d+a+c, 2b+a+c, 2c+b+d\} + n \\
&\geq \min\{2a+n/2, 2d+n/2, 2b+n/2, 2c+n/2\} + n \ \geq \ n.
\end{aligned}$$

The transformations of the above inequalities come from $a+c = b+d = n/2$. Thus, for the upper bound of the optimal value of F we have

$$\mathrm{OPT}(F) \leq m - |\mathrm{unsat}_T(F)| \leq m - n. \tag{4}$$

Since each G_k is an exact 1/2-enlarger, that is, the number of edges between U and W is exactly $1/2 \cdot n/2$ in each direction, assignment T_0 produced by `GEN` that assigns true to all variables of X gives the exact n unsatisfied clauses. (Those are produced from edges of G_2, G_3, G_5 and G_6.) The number of satisfied clauses by T_0 attains the upper bound (4) for $\mathrm{OPT}(F)$. Hence, we conclude $\mathrm{OPT}(F) = m - n$ and T_0 is a maximum assignment. ■

Now that we give an algorithm to produce a family of 2-CNF formulas, we evaluate the ratio of the number of unsatisfied clauses to the number of all clauses under the maximum assignment.

By **Lemma 3.2**, an exact 1/2-enlarger has $(2c_0 + 1/2)n$ edges for n vertices, where $c_0 = 35$. Since we used six exact 1/2-enlargers, and produced one clause from one edge of those graphs, the number of produced clauses is equal to the total number of edges of six graphs, that is,

$$m = 6 \cdot \left(2 \cdot 35 + \frac{1}{2}\right) n = 423n.$$

The ratio of the number of unsatisfied clauses to the number of clauses is

$$\frac{m - \mathrm{OPT}(F)}{m} = \frac{n}{423n} \approx 0.0023640622.$$

4.2 Randomized generator

The generator `GEN` uses exact 1/2-enlargers based on expanders given by the explicit construction of `GG`method. Though in the construction we used randomness several times, it may be still possible that the produced 2-CNF instances have some structure. On the other hand, it has been known that "expanders" can be obtained *randomly*. Here we discuss the possibility of producing instances from randomly constructed enlargers. That is, we use the same outline of the instance generator `GEN`. Only the difference is to construct exact 1/2-enlargers based on random graphs.

Let n be an integer given as an input for specifying the number of variables. (We will assume below that n is even.) In the algorithm `GEN`, six exact 1/2-enlargers of n vertices are used to produce one 2-CNF instance. Note that optimal partitions are also needed; but it seems difficult to construct exact 1/2-enlargers randomly and provide their optimal partitions. Thus, we will construct those graphs based on randomly generated 2/3-enlargers.

Recall that, for constructing an exact 1/2-enlarger, we needed 2/3-enlarger graphs $G = (V, E)$ satisfying the following, a bit stronger condition: for any subset S of V ($|V| = n$),

$$|S \times (V-S) \cap E| \;\geq\; 2\delta\left(1 - \frac{|S|}{n}\right)|S|, \text{ and} \tag{5}$$

$$|(V-S) \times S \cap E| \;\geq\; 2\delta\left(1 - \frac{|S|}{n}\right)|S|, \tag{6}$$

where $\delta = 2/3$. Notice here that, due to the symmetry, if G satisfies (5) for all subsets of V, so does G (6) for all subsets of V. Thus, below we will consider only the first condition.

We first estimate the probability that the random graph satisfies (5) where $\delta = 1/2$ (*not* $\delta = 2/3$) for any subset of V with size up to $n/2$ (*not* n).

Lemma 4.2 Consider a graph $G = (V, E)$ such that $|V| = n$, and that is randomly generated by choosing for edges from u a vertex from $V \backslash \{u\}$ equiprobably and independently d times. Then $G = (V, E)$ satisfies (5) where $\delta = 1/2$ for any subset $S \subset V$ of size $\leq n/2$, with probability at least p, where

$$p \;=\; \frac{1-2r}{1-r}, \quad \text{and } r = d^3 e^4 \cdot \left(\frac{1}{2}\right)^d,$$

Proof. Consider a subset S of V and let $|S| = s$. Here we say that S is *bad* if $|S \times (V-S) \cap E| < c(s)$, where $c(s) = 2\delta(1 - s/n)s$. Since d edges emanate from each vertex of S, S is bad if and only if at least $ds - \lceil c(s) \rceil - 1$ edges are introverted; that is, there are at least $ds - \lceil c(s) \rceil - 1$ edges whose sources are in S. Thus, for any subset S of size s,

$$\begin{aligned}
\Pr\{S \text{ is bad}\} &\leq \binom{ds}{ds - \lceil c(s) \rceil - 1}\left(\frac{s-1}{n-1}\right)^{ds - \lceil c(s) \rceil - 1} \\
&\leq \binom{ds}{\lceil c(s) \rceil + 1}\left(\frac{s}{n}\right)^{ds - \lceil c(s) \rceil - 1}.
\end{aligned}$$

Thus, we have the probability that there exists a bad subset of size s,

$$\begin{aligned}
\Pr\left\{\bigvee_{S:|S|=s} \{S \text{ is bad}\}\right\} &\leq \sum_{S:|S|=s} \Pr\{S \text{ is bad}\} \\
&\leq \binom{n}{s}\binom{ds}{\lceil c(s) \rceil + 1}\left(\frac{s}{n}\right)^{ds - \lceil c(s) \rceil - 1} \\
&\leq \left(\frac{ne}{s}\right)^s \left(\frac{dse}{\lceil c(s) \rceil + 1}\right)^{\lceil c(s) \rceil + 1} \left(\frac{s}{n}\right)^{ds - (\lceil c(s) \rceil + 1)}.
\end{aligned}$$

Here, applying $\delta = 1/2$, we have

$$\lceil c(s) \rceil + 1 \;=\; \left\lceil \left(1 - \frac{s}{n}\right) s \right\rceil + 1 \;\leq\; \left(3 - \frac{s}{n}\right) s.$$

The last transformation comes from $s \geq 1$. Since a function $(dse/x)^x$ on x is increasing for

$ex \le dse$ and $\lceil c(s) \rceil + 1 \le ds$, we have,

$$
\begin{aligned}
\Pr\left\{ \bigvee_{S:|S|=s} \{S \text{ is bad}\} \right\} &\le \left(\frac{ne}{s}\right)^s \left(\frac{dse}{(3-s/n)s}\right)^{(3-s/n)s} \left(\frac{s}{n}\right)^{ds-(3-s/n)s} \\
&= \left(\frac{ne}{s}\left(\frac{de}{3-s/n}\right)^{3-s/n}\left(\frac{s}{n}\right)^{d-(3-s/n)}\right)^s \\
&\le \left(d^3e^4 \cdot \left(\frac{s}{n}\right)^{d-4}\left(\frac{1}{3-s/n}\right)^{3-s/n}\left(\frac{s}{n}\right)^{s/n}\right)^s \qquad (7) \\
&\le \left(d^3e^4\left(\frac{1}{2}\right)^{d-4}\left(\frac{1}{3-s/n}\right)^{3-s/n}\left(\frac{s}{n}\right)^{s/n}\right)^s \qquad (8) \\
&\le \left(d^3e^4 \cdot \left(\frac{1}{2}\right)^{d-1}\right)^s. \qquad (9)
\end{aligned}
$$

The transformation (7), (8) comes from $(de)^3 \ge (de)^{3-s/n}$, and $s/n \le 1/2$. We can obtain the last transformation (9) from the following fact: $x^x \le 1$ for $0 \le x \le 1/2$, and $x^{-x} \le (1/2)^3$ for $5/2 \le x \le 3$. Thus, we have,

$$\Pr\{S \subset V \text{ of size } s \text{ is bad}\} \le r^s, \quad \text{where } r = d^3e^4 \cdot \left(\frac{1}{2}\right)^{d-1}.$$

Finally, to obtain the probability that some subset of size $\le n/2$ is bad, we sum up this probability for $1 \le s \le n/2$,

$$\Pr\left\{ \bigvee_{s=1}^{n/2} \{S \subset V \text{ of size } s \text{ is bad}\} \right\} \le \sum_{s=1}^{n/2} r^s \le \frac{r - r^{n/2+1}}{1-r} \le \frac{r}{1-r}.$$

Thus, we have the probability that every subset of size $\le n/2$ is not bad; that is, the probability p that G satisfies (5) where $\delta = 1/2$ for any $S \subset V$ of size up to $n/2$ as follows:

$$p \ \ge\ 1 - \frac{r}{1-r} \ =\ \frac{1-2r}{1-r}.$$

■

To obtain the desired graphs for constructing an exact 1/2-enlarger we first consider relaxing the constraint $|S| \le n/2$ to $|S| \le n$; we generate another random graph and merge them. More specifically, we generate two random graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ such that $V_1 = V_2$, and construct $G = (V, E)$ on V ($V = V_1 = V_2$) and E defined as, $E = \{(u,v) : (u,v) \in E_1 \text{ or } (v,u) \in E_2\}$. We then show the graph $G = (V, E)$ satisfies (5) for all subsets of V with probability p^2. For subsets of size $\le n/2$, G satisfies (5) with probability p since it is satisfied with G_1. For subsets of size $> n/2$, G also satisfies (5) with probability p since G_2 satisfies (6) for subsets of size $\le n/2$, that is, for any subset S of size $\ge n/2$

$$
\begin{aligned}
|S \times (V-S) \cap E| &\ge |(V-S) \times S \cap E_2| \\
&\ge 2\delta\left(1 - \frac{|V-S|}{|V|}\right)|V-S| \ =\ 2\delta\left(1 - \frac{|S|}{|V|}\right)|S|.
\end{aligned}
$$

Hence, $G = (V, E)$ constructed from G_1 and G_2 satisfies (5) where $\delta = 1/2$ for all subsets of V with probability at least p^2.

Note also that for extending the condition $\delta = 1/2$ to $\delta = 2/3$ we only generate another random graph and merge them while the probability becomes p^4. Thus, we conclude the probability that we obtain an exact 1/2-enlarger from those random graphs is p^8.

We next estimate the degree of edges such that all graphs used to construct six exact 1/2-enlargers satisfy the desired property with high probability. Since we need eight random graphs for constructing one exact 1/2-enlarger, the probability that all random graphs used to construct six exact 1/2-enlargers are desired 2/3-enlargers is at least $(p^8)^6$. For making this probability high, by taking $d = 35$ for example, we can show that $p^{48} > 0.993480$; that is, the success probability, the probability of obtaining six exact 1/2-enlargers, is more than 99%. Thus, under this choice of parameter d, we can guarantee that, for more than 99% of produced 2-CNF instances, the assignments that our generator produces are indeed maximum assignments. Notice also that the number m of edges (or clauses) is $m = (35 + 0.5) \cdot 6 \cdot n = 213n$ when $d = 35$, which is about a half of the number of clauses produced by `GEN`. Thus, the ratio of unsatisfied clauses to m becomes double in compensation for small chance of producing non-optimal solutions.

Acknowledgement

I thank Professor Osamu Watanabe for all his supports: giving valuable comments, discussing, and improving this report, In particular, in random generator, the technique for relaxing the condition $|S| \leq n/2$ to $|S| \leq n$ is his idea.

References

[1] M. R. Garey and D. S. Johnson, "Computers and Intractability, A Guide to the Theory of NP-Completeness", W. H. Freeman, 1979.

[2] G. A. Margulis, "Explicit constructions of concentrators", Problemy Peredachi Informatsii, 71-80, 1973. English translation in Problems of Information Transmission 9, 325-332, 1975.

[3] O. Gabber and Z. Galil, "Explicit constructions of linear-sized superconcentrators", Journal of Computer and System Sciences 22, 407-420, 1981.