数理解析研究所講究録 1375 巻 2004 年 78-84

On generating instances for MAX2SAT with optimal solutions

Masaki Yamamoto (山本真基)
masaki.yamamoto@is.titech.ac.jp

Dept. of Mathematical and Computing Sciences (数理・計算科学),
Tokyo Institute of Technology (東京工業大学)

March 30, 2004

Abstract

A *test instance generator* (an *instance generator* for short) for MAX2SAT is a procedure that produces, given a number n of variables, a 2-CNF formula F of n variables (randomly chosen from some reasonably large domain), and simultaneously provides one of the optimal solutions for F. We propose an outline to design an instance generator using an expanding graph of a certain type, called here an "exact 1/2-enlarger". We first show a simple algorithm for constructing an exact 1/2-enlarger, thereby deriving one concrete polynomial-time instance generator GEN. We also show that an exact 1/2-enlarger can be obtained with high probability from graphs *randomly* constructed. From this fact, we propose another type of instance generator RGEN; it produces a 2-CNF formula with a solution which is optimal for the formula with high probability. However, RGEN produces less structured formulas and much larger class of formulas than GEN's. In fact, we prove the NP-hardness of MAX2SAT over the set of 2-CNF formulas produced by RGEN.

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 a procedure that generates typical instances with their solutions. In this paper, we propose instance generators for MAX2SAT, which is a problem of obtaining a maximum assignment, an assignment satisfying the maximum number of clauses of a given a 2-CNF formula.

An *instance generator* for MAX2SAT is a randomized procedure that takes n, the number of variables, and produces a pair of a 2-CNF formula of n variables and its maximum assignment. For a "reasonable" instance generator, we would like to require some hardness, e.g., NP-hardness, for solving MAX2SAT over the formulas produced. Unfortunately, however, the following proposition suggests that there doesn't seem to exist a polynomial-time generator producing such hard instances.

Proposition 1.1 Unless NP = co-NP, there is no polynomial-time instance generator G such that MAX2SAT over $\mathcal{F}(\mathtt{G})$ is NP-hard, where $\mathcal{F}(\mathtt{G})$ is a set of 2-CNF formulas produced by G.

Thus, instead of "computational hardness", we below consider "structural hardness" for formulas produced. Let F be a 2-CNF formula and T be an assignment. We denote by $\mathrm{unsat}_T(F)$ a set of unsatisfied clauses of F under T. If T is a maximum assignment, $\mathrm{unsat}_T(F)$ is called *hole* of F, and denoted by $\mathrm{hole}(F)$. Since 2SAT is solvable in polynomial time, we should produce formulas with large hole; that is, if a 2-CNF formula F is of size m and has a hole of size a constant c, a maximum satisfying assignment for F can be found by running an algorithm solving 2SAT in at most $\sum_{i=0}^{c}\binom{m}{i}$ times, which is a polynomial of m. Thus, we require our generator to produce formulas with hole of size more than any constant. On the other hand, we would like to produce formulas with small size, that is, those with few clauses because it is considered relatively easy to solve MAX2SAT on 2-CNF formulas with large size. Thus, the number of clauses of formulas produced should be kept linear to the number of variables. More specifically to say above all, for some constant $\alpha > 0$ and $\beta > 0$, and for every number n of variables, we require the following conditions for all formulas F produced:

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

Of course, there is a trivial way to produce formulas satisfying these conditions; just producing a set of four clauses $(x \vee y)$, $(x \vee \overline{y})$, $(\overline{x} \vee y)$, and $(\overline{x} \vee \overline{y})$ for pairs x and y of variables. This case bears $\beta = 0.25$. In this paper, while it is hard to formally specify "nontriviality", we aim at producing nontrivial formulas satisfying these two conditions.

Our idea is to use directed graphs with a certain "expansion" property. Here we explain an outline of producing 2-CNF formulas briefly. Let X be a variable set $\{x_1, \cdots, x_n\}$, and V be a vertex set $\{v_1, \cdots, v_n\}$. We first construct a directed graph G (that has some nice property) over V, and then use it to produce a 2-CNF formula over X as follows: First, we label every vertex of V some literal of X by a (random) permutation π over $\{1, \cdots, n\}$. Then, regarding each edge (v_i, v_j) of G as a logical expression $x_{\pi(i)} \Rightarrow x_{\pi(j)}$, we produce a clause $(\overline{x_{\pi(i)}} \vee x_{\pi(j)})$ equivalent to the logical expression. The 2-CNF formula produced is a collection of such clauses.

We *intuitively* explain how the expansion property is used. Suppose that a 2-CNF formula F over X is obtained from a directed graph $G = (V, E)$ over V. Let us further suppose that 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 of X. The vertex set is divided into two sets; namely P and N, which are sets of vertices assigned respectively true and false under the assignment T. Then under 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 a 2-CNF formula F with large hole, we need a graph with many cut edges for all cuts. More specifically, we would like to have a graph with the expansion property defined in Definition 2.2 of the next section. Furthermore, in order to keep the number of clauses linear to n, we require that the number of edges is linearly bounded by n, say cn for some constant c. We call a graph with such properties an *(n, c, δ)-enlarger*. Recall that we also provide a maximum assignment at the same time. For this end, we need to provide a cut such that the smallest ratio of cut edges is precisely known. (See Definition 2.3 of the next section.) An (n, c, δ)-enlarger having such a cut is called an *exact (n, c, δ)-enlarger*. Once an exact (n, c, δ)-enlarger is given, by following the outline explained above, we can produce a 2-CNF formula and its maximum assignment; we actually use two exact (n, c, δ)-enlargers of $\delta = 1/2$. (Below, an exact $(n, c, 1/2)$-enlarger will be simply called an exact $1/2$-enlarger.)

For presenting a concrete generator following the outline above, we first show an algorithm that, for a given n, constructs an exact $1/2$-enlarger within polynomial time of n. The algorithm is based on the explicit construction of expanders [1]. Then, we give our polynomial-time instance generator GEN that produces 2-CNF formulas satisfying the conditions (1) and (2). We also investigate the probability that an exact $1/2$-enlarger is constructed from random graphs, that is equivalent to the probability that every random graph $G = (V, E)$ has some expansion property (see Lemma 2.1) so that those random graphs derive an exact $1/2$-enlarge. We show that under a certain edge degree, this probability is high. Using this graph G constructed from random graphs, and following the outline above, we can again obtain a 2-CNF formula and its maximum assignment if the graph G is indeed an exact $1/2$-enlarger. This gives us a slightly weaker instance generator RGEN; while RGEN produces a 2-CNF formula and its maximum assignment with high probability, there is some chance that the assignment provided with a formula is not optimal, which happens rarely. On the other hand, compared with formulas produced by GEN, those produced by RGEN have the following advantages: (i) they have less structure, and (ii) they have a better hole ratio β. We also show that RGEN produces formulas with reasonable hardness, by demonstrating that MAX2SAT is NP-hard on the set $\mathcal{F}(\text{RGEN})$, the set of formulas produced by RGEN. From Proposition 1.1, this result about hardness would be impossible for $\mathcal{F}(\text{G})$ such that G *always* produces a pair of formulas and its maximum assignment.

A full version of this paper is available online [2].

2 Explicit Construction of Exact Enlarger

In this section, we first review an *expander* used for producing instances, and obtain a precise value about the edge degree needed to construct a desired graph. Let $G = (U, V, E)$ be an undirected bipartite graph such that $U = \{u_1, \cdots, u_n\}$ and $V = \{v_1, \cdots, v_n\}$. Throughout this paper, a graph may have multiple edges, but doesn't have any self-loops. We denote by $\Delta(S)$ for $S \subset U$ the number of edges from S to vertices of V having the different index numbers from S; that is, $\Delta(S) = \{(u_i, v_j) \in E : u_i \in S \text{ and } v_j \in V \text{ and } i \neq j\}$.

Definition 2.1 An (n, c, δ)-*expander* is an undirected bipartite graph $G = (U, V, E)$ such that $|U| = |V| = n$, $|E| \leq cn$, and for any subset S of U, $|\Delta(S)| \geq \delta \cdot (1 - |S|/n) \cdot |S|$.

Our definition of an expander is a little different from [1]'s; The definition above is w.r.t. $|\Delta(S)|$, i.e., the number of *edges* emanated from S while an expander of [1] is defined w.r.t. the number of *neighbors*, which are defined as *vertices* of V connected to vertices of S. But for our definition, we can also derive the same theorem as [1]'s: that says for any n such that $n = h^2$ for some integer h, an $(n, 7, \delta)$-expander can be explicitly constructed where $\delta = (2 - \sqrt{3})/2$.

Below, 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 seen below, our generators need δ-expanders such that $\delta \geq 2/3$. The graph G constructed by [1]'s method is not such one because the ratio $\delta' = (2 - \sqrt{3})/2$ is less than $2/3$. But, note that our definition of an expander is w.r.t. the number of edges (*not* vertices). Thus, by using random permutations on vertices, we construct l isomorphic δ'-expanders and then merge them to a single graph, thereby yielding a $2/3$-expander. Note that this resulting expander has $l \cdot 7n$ $(= 35n)$ edges; that is, it is an $(n, 35, 2/3)$-expander. In the following sections, we denote by (n, c_0, δ_0)-expander such an expander, where $c_0 = 35$ and $\delta_0 = 2/3$.

Next, we introduce our new notions called an "enlarger" and an "exact enlarger", and show their explicit constructions.

Definition 2.2 An (n, c, δ)-*enlarger* $G = (V, E)$ is a directed graph such that $|V| = n$, $|E| \leq cn$, and for any subset S of V such that $|S| \leq |V|/2$, $|S \times (V - S) \cap E| \geq \delta|S|$, and $|(V - S) \times S \cap E| \geq \delta|S|$.

Again, we simply call an (n, c, δ)-enlarger a δ-enlarger in the case that the values n and c are given in the context. Observe that a directed graph $G = (V, E)$ of $|V| = n$ constructed from two isomorphic (n, c_0, δ_0)-expanders $G_1 = (U, W, E_1)$ and $G_2 = (U, W, E_2)$ such a way of $E = \{(v_i, v_j) : (u_i, w_j) \in E_1\} \cup \{(v_j, v_i) : (u_i, w_j) \in E_2\}$, is an $(n, 2c_0, \delta_0)$-enlarger.

Lemma 2.1 For any graph $G = (V, E)$ constructed in the way above, we have that for any subset S of V

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

By definition of enlarger, for any subsets S and T of V such that $S \cup T = V$, $S \cap T = \phi$, and $|S| = |T|$, the following inequalities hold: $|S \times T \cap E| \geq \lceil \delta n/2 \rceil$ and $|T \times S \cap E| \geq \lceil \delta n/2 \rceil$. For our generators, among all those partitions of (S, T), we need a δ-enlarger having an "optimal" partition for which the inequalities above hold at equality, which we call an "exact enlarger".

Definition 2.3 An *exact* (n, c, δ)-*enlarger* $G = (V, E)$ is an (n, c, δ)-enlarger which has a partition (U, W) of V with $|U| = |W|$ such that

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

Here, we show the construction of an exact $(n, 2c_0 + 1/2, 1/2)$-enlarger from two $(n/2, 2c_0, \delta_0)$-enlargers. We first construct two $(n/2, 2c_0, \delta_0)$-enlargers $L = (U, E_l)$ and $R = (W, E_r)$, where $U = \{u_1, \cdots, u_{n/2}\}$ and $W = \{w_1, \cdots, w_{n/2}\}$. Then, we construct a graph $G = (V, E)$ defined as $V = U \cup W$ and $E = E_l \cup E_r \cup A \cup B$, where $A = \{(u_i, w_i) : 1 \leq i \leq n/4\}$, and $B = \{(w_i, u_i) : n/4 + 1 \leq i \leq n/2\}$. In what follows, we call a correspondence between U and W defined by A and B a *one-to-one balanced correspondence*.

Lemma 2.2 For any n, the graph $G = (V, E)$ of $|V| = n$ constructed in the way above is an $(n, 2c_0 + 1/2, 1/2)$-enlarger.

Proof. Consider a graph $G = (V, E)$ constructed by the way 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 $2 \cdot (2c_0 \cdot n/2) + n/2$, which is $(2c_0 + 1/2)|V|$, where $|V| = n$.

Next, we show that G is a 1/2-enlarger. Consider an arbitrary $S \subset V$ of size at most $n/2$. Let $S = S_U \cup S_W$, where $S_U \subset U$ and $S_W \subset W$. 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 prove for the following two cases:

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

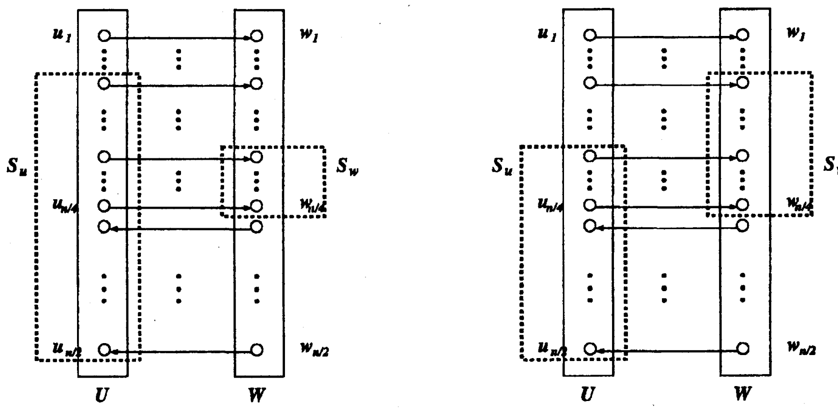
It is obvious for the first case from the definition of δ_0-enlarger. In the second case, by Lemma 2.1, we have that

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

By combining the two inequalities above, we have that

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

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



Case (i)　　Case (ii)

Figure 1: Two Cases

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

$$\begin{aligned} \text{Case (i)}: & \quad \forall (u, w) \in A \big[w \in S_W \Rightarrow u \in S_U \big], \\ \text{Case (ii)}: & \quad \forall (u, w) \in A \big[u \in S_U \Rightarrow w \in S_W \big]. \end{aligned}$$

Note that Case (i) is for $x \geq y$ and Case (ii) is for $x \leq y$. We show that G is a $1/2$-enlarger for any x, y such that $0 \leq x, y \leq 1$ in both cases. Since the proofs of those two cases are similar, we only show it for Case (i). It is easy to see that $|S \times (V-S) \cap A| = (x-y)\cdot n/4$. By combining this equality and inequality (1), we have that

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

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

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

Applying $\delta_0 = 2/3$, the inequality above is equivalent to the following (*): $(x-3/8)^2+(y+1/8)^2 \leq 13/32$. That means G is a $1/2$-enlarger if (x,y) is within the circle defined by (*). On the other hand, in Case (i), x and y must satisfy the following bounds: $y \geq 0$, $y \leq x$, and $y \leq -x+1$. (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 three inequalities. Note that this triangle is within the circle defined by (*) since its three vertices $(0,0)$, $(1,0)$, $(1/2,1/2)$ all satisfy (*). Thus, we conclude any pair of x and y in the domain satisfies (*). ■

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 2.3 For any n, the graph $G=(V,E)$ of $|V|=n$ constructed in the way explained above, is an exact $(n, 2c_0+1/2, 1/2)$-enlarger.

3 Generating Algorithms

In this section, we first show an outline for instance generators by presenting a concrete generator GEN. For producing a 2-CNF formula of n variables, we use exact $1/2$-enlargers having n vertices as follows:

Algorithm 1 GEN (input n: a number of variables)

- **step 1:** Let $X = \{x_1, \cdots, x_n\}$ be a set of variables, and $V = \{v_1, \cdots, v_n\}$ be a vertex set. Divide V into two sets $U = \{v_1, \cdots, v_{n/2}\}$ and $W = \{v_{n/2+1}, \cdots, v_n\}$.
- **step 2:** Construct independently two exact $(n, 2c_0+1/2, 1/2)$-enlargers $G_k = (V, E_k)$ for $k = 1, 2$ on U and W.
- **step 3:** Choose a random assignment $t: t_1, \cdots, t_n \in \{0,1\}$ to X and a random permutation π over X. Then, label each vertex of the two graphs as follows: $l_1(v_i) = x_{\pi(i)}^{t_i}$ for $v_i \in V$ of G_1, and $l_2(v_i) = x_{\pi(i)}^{t_i}$ for $v_i \in U$ of G_2 and $l_2(v_i) = x_{\pi(i)}^{t_i \oplus 1}$ for $v_i \in W$ of G_2, where $x_{\pi(i)}^{t_i}$ is $x_{\pi(i)}$ if $t_i = 1$, and otherwise $\overline{x_{\pi(i)}}$.
- **step 4:** Define a set F of clauses as follows: $F = \{\, (\overline{l_k(u)} \vee l_k(v)) : (u,v) \in E_k, k = 1,2\}$. Set an assignment T_0 to be $T_0(x_1) = t_1, \cdots$, and $T_0(x_n) = t_n$.

Theorem 3.1 For any pair of a 2-CNF formula F and an assignment T_0 produced by GEN, we have that $|F| = 141n$, $\text{hole}(F) = n/4$, and T_0 is a maximum assignment for F.

As is mentioned in Introduction, we require the ratio of hole to the size of F as large as possible. We calculate the precise value of the ratio. By the theorem above, the ratio of hole is

$$\frac{\text{hole}(F))}{|F|} = \frac{n/4}{141n} \approx 0.0017730496.$$

3.1 A generator based on random graphs

The generator GEN uses exact 1/2-enlargers based on expanders given by the explicit construction of [1]'s method. Even though we use randomness several times in GEN, it might be still possible that the formulas produced by GEN have a certain structure. On the other hand, it has been known that random graphs mostly have the expansion property. Here, we'll discuss the possibility of producing instances from random graphs, which leads to a generator RGEN.

Recall the explicit construction of an exact 1/2-enlarger; By the proof of Lemma 2.2, we can conclude that the procedure outputs an exact 1/2-enlarger *if* for each graph $G = (V, E)$ composing the exact 1/2-enlarger, it satisfies the two conditions presented in Lemma 2.1 for every subset of V. Note that due to the symmetry, if G satisfies the one condition for all subsets of V, so does G the other condition for all subsets of V. That is, in what follows, we will consider only the first condition: for every subset of V,

$$|S \times (V - S) \cap E| \geq 2\delta_0 (1 - |S|/|V|)\,|S|. \tag{2}$$

We first estimate the probability that the random graph satisfies (2) where $\delta = 1/2$ (*not* $\delta = 2/3$, this is for an easier calculation) for any subset of V of size up to $n/2$ (*not* n). After that, using the value, we obtain the probability for $\delta = 2/3$ and size up to n.

Here, we define the probability space for a random graph $G = (V, E)$ of $|V| = n$; For each $u \in V$, we choose d edges emanating from u by sampling *with replacement* d vertices from V independently and uniformly. The sampling with replacement means that there can be multi-edges emanating from a vertex, or even exist a vertex having no edges from itself.

Lemma 3.2 The random graph $G = (V, E)$ described above satisfies (2) where $\delta = 1/2$ for any subset $S \subset V$ of size $\leq n/2$, with probability at least p, where $p = (1 - 2r)/(1 - r)$, and $r = de^2 \cdot 2^{5/2-d}$.

Next, we extend the upper bound on the size of S to n; it is easily done by generating two random graphs $G_1 = (V, E_1)$ and $G_2 = (V, E_2)$, and construct $G = (V, E)$ defined as; $E = \{(u, v) : (u, v) \in E_1 \text{ or } (v, u) \in E_2\}$. We claim that with probability at least p^2, the graph G satisfies (2) for every subset of V.

Finally, we extend the value of δ to 2/3; it is also done in the same way. We generate another random graph and simply merge them. Then, we have that with probability at least p^4, the random graph satisfies (2) with $\delta = 2/3$ for every subset of V. Therefore, we conclude that the probability we obtain one exact 1/2-enlarger from random graphs is at least $(p^4)^2 = p^8$.

Now, we estimate the specific degree of edges such that the probability that two exact 1/2-enlargers are obtained from random graphs, which is at least $(p^8)^2$, is high. For this end, by taking $d = 15$, for example, we have $p^{16} > 0.992911$; that is, the success probability, the probability of obtaining two exact 1/2-enlargers, is more than 99%. Thus, under this choice of parameter d, we can conclude that, for more than 99% of 2-CNF formulas produced by RGEN, the assignments provided with formulas are indeed maximum assignments.

Notice also that the number m of edges (or clauses) is $m = 2 \cdot (15 + 0.5) \cdot n = 31n$ when $d = 15$, which is about one fifth of the number of clauses produced by GEN. Thus, the ratio of unsatisfied clauses to m becomes five times in compensation for small chance of providing non-maximum assignments.

3.2 Hardness of instances by RGEN

We have proposed two types of generators; producing formulas and also providing their maximum assignments although for RGEN it is with high probability. This is because RGEN uses random graphs to produce formulas. On the other hand, it can be considered that the randomness makes RGEN to produce hard formulas to solve. Here, we define the problem of those instances and discuss about the hardness to solve such formulas.

<u>Balanced MIN-ASSIGNCUT</u>

Instance: Two vertex sets U and W with $|U| = |W|$, and two directed graphs $G_i = (V_i, E_i)$ for $i = 1, 2$ as follows: We define V_i as $U \cup W$. Moreover, for each graph G_i, there exists a one-to-one balanced correspondence between U and W for $i = 1, 2$.

Question: Let $S = (T \cap U) \cup (\overline{T} \cap W)$, and for any subset T of V, let cost(T) be $\sum_{i=1,2} c_i$, where $c_1 = |T \times \overline{T} \cap E_1|$, and $c_2 = |S \times \overline{S} \cap E_2|$. Then, what is a subset T of V such that cost(T) is minimized?

It is easy to see that the problem above is MAX2SAT where instances for the problem is restricted to a subset of 2-CNF formulas. Moreover, all of those instances can be produced by `RGEN`. This follows from the following observation. Let X be a set of variables and t be an assignment to X. In the problem above, we consider X as V, and correspond t to cut T_t of V such that for each edge e and its corresponding clause C_e, cost(T_t) is increased by e if and only if C_e is not satisfied under the assignment t.

Thereby, if we prove the hardness of the problem above, we can conclude that `RGEN` surely produces hard formulas as well. Thus, the theorem to prove here is as follows:

Theorem 3.3 Balanced MIN-ASSIGNCUT is NP-hard.

This theorem can be proven by a sequence of reductions starting with a variant of NAE3SAT, where the number of occurrences of variables is a constant.

References

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

[2] M. Yamamoto, "On Generating Instances for MAX2SAT with Optimal Solutions", http://www.is.titech.ac.jp/research/research-report/C/C-191.ps.gz