Polynomial Time Approximation Schemes for Dense Instances of the Minimum Constraint Satisfaction (Preliminary version)

Cristina Bazgan
LAMSADE
Université Paris-Dauphine
75775 Paris
bazgan@lamsade.dauphine.fr

Marek Karpinski
Dept. of Computer Science
University of Bonn
53117 Bonn
marek@cs.uni-bonn.de

January 6, 2000

Abstract

We present a polynomial time approximation scheme for dense instances of the minimal constraint satisfaction problems, MIN kCSP. This class contains minimization problems that search for boolean assignments to the variables minimizing the number of satisfied constraints depending on at most k variables. By dense instances of a problem in MIN kCSP we mean instances having $\Omega(n^{k-1})$-read boolean representations where n is the number of boolean variables.

1 Introduction

In this paper we study the approximability of dense instances of minimization versions of boolean constraint satisfaction problems. An input of a boolean constraint satisfaction problem is a collection F of boolean functions called constraints that are applied to a subset of at most k variables among n boolean variables. The problem consists in finding an assignment of the boolean variables that minimizes the number of constraints satisfied. We call the class of all such problems MIN kCSP. This class contains problems as: MIN kSAT, MIN kDNF, MIN EQUIVALENCE, MIN PAIRED BISECTION, MIN 2CNF DELETION, MIN Ek LIN 2, NEAREST CODEWORD, MIN HORN DELETION, MIN IMPLICATIVE HITTING SET-B (cf. e.g. [KST97], [BFdV99]).

Constraint satisfaction problems were studied firstly 1978 by Schaefer [S78] that gave a classification of decision problems in polynomial solvable and NP-hard. An instance of such a decision problem has in input m constraints of F. The language SAT(F) consists of all instances which have an assignment satisfying all m constraints. Schaefer described six classes of function families and he showed that if F is a subset of one of these classes then the decision problem is in P, otherwise the decision problem is NP-hard.

His work was followed by the work of Creignou [C95], Khanna, Sudan and Williamson [KSW97] and Khanna, Sudan, Trevisan [KST97] who obtained a classification of the approximability of maximization and minimization problems derived from constraint satisfaction problems. In fact [C95] and [KSW97] extend Schaefer's work to maximization problems.

Two classes of maximization problems are defined MAX CSP(F) and MAX ONES(F). An input of a problem in such a class consists of m constraints of F that are applied at a subset of at most k variables among n boolean variables. In the first case the objective is to find an assignment which maximizes the number of constraints that are satisfied. In the second case the objective is to find an assignment that satisfies all the constraints and which maximizes the number of variables set to 1. They showed the existence of a finite partition of all function families such that the approximability of such a problem is determined by the membership of F in one of the class of the partition. [KST97] considered the corresponding minimization classes: MIN CSP(F) and MIN ONES(F) and their result is similar to the result of [KSW97] for MAX CSP(F) and MAX ONES(F).

By a dense instance of a problem in MIN kCSP we mean an instance over n variables with a representation in which the number of occurrences (variable reads) of each variable is $\Omega(n^{k-1})$. An average dense instance is an instance for which the number of constraints is $\Omega(n^k)$.

The approximability theory of average dense instances of maximization problems such as Max Cut, Max kSAT has had many successes, starting with [AKK95] and [FdV96]. See [K97] for a review. In [AKK95] it was proved in particular that the average dense instances of any problem in MAX SNP have a polynomial time approximation scheme. In [GGR96], it was proved that many of these problems can be approximated in constant time with an additive error ϵn^2 where n is the size of the input in a certain probe model (implying that the dense versions have also constant-time approximation schemes). Also Frieze and Kannan [FK99], [FK96] gave constant time approximation schemes for some average dense problems in the oracle model of computation.

The case of dense instances of minimization problems seems to be harder. In [AKK95] polynomial time approximation schemes for dense Bisection and Min kCut was given. The case of dense Vertex Cover was settled in [KZ97], [CT96]: dense and average dense instances do not have approximation schemes. Also in [KZ97], a $c \ln n$ approximation algorithm was given for every constant $c > 0$ for average dense instances of Set Cover. [AFK96] gave a polynomial time approximation scheme for average dense instances of some variants of Min Linear Arrangement problem. For dense and average dense Bandwidth, [KWZ97] gave a constant approximation algorithm. In [FdVK99] it was also proved that dense and average dense Min TSP(1,2) and Longest Path have no polynomial time approximation schemes. A polynomial time approximation scheme was given in [BFdV99] for dense Min 2Sat and dense Min Equivalence problems.

In this paper we prove that some dense instances of any problem in MIN kCSP have polynomial time approximation schemes. Interestingly, contrary to its maximization version for which average dense instances have polynomial time approximation schemes [AKK95], all the average dense MIN kCSP problems are hard to approximate.

Our results imply polynomial time approximation schemes for dense instances of some optimization problems that are hard to approximate on general instances and that remains hard to approximate on average dense instances as: Min kSat, Min kDNF, Nearest Codeword, Min 2CNF Deletion. We remark that Dense MIN ONES has no polynomial time approximation scheme since Min Vertex Cover is in MIN ONES and Dense Min Vertex Cover has no polynomial time approximation scheme.

The paper is organized as follows. In Section 2 we give the necessary definitions and prove that the problems in which we are interested are NP-hard. In Section 3 we show that the instances of our problems where each clause has exactly k literals are at least as hard to approximate as the instances where the clauses have at most k literals. In Section 4 we give polynomial time approximation schemes for dense instances of our problems where each clause has exactly k literals and in Section 5 we give some conclusions.

2 Preliminaries

We begin with some basic definitions.

Approximability. Let us recall a few definitions about approximability. Given an instance x of an optimization problem A and a feasible solution y of x, we denote by $m(x, y)$ the value of the solution y, and by $opt_A(x)$ the value of an optimum solution of x. The *performance ratio* of y is

$$R(x,y) = \max\left\{\frac{m(x,y)}{opt_A(x)}, \frac{opt_A(x)}{m(x,y)}\right\}.$$

For a constant $c > 1$, an algorithm is a *c-approximation algorithm*, if for any instance x of the problem it returns a solution y such that $R(x, y) \leq c$. We say that an optimization problem is *constantly approximable* if, for some $c > 1$, there exists a polynomial time c-approximation algorithm for it. An optimization problem has a *polynomial time approximation scheme* (a PTAS, in short) if, there exists a polynomial time $(1+\varepsilon)$-approximation for it for every constant $\varepsilon > 0$.

L-reductions. Let A and B be two optimization problems. Then A is said to be *L-reducible* (cf. [PY91]) to B ($A \leq_L B$) if there are two constants $\alpha, \beta > 0$ such that

1. there exists a function, computable in polynomial time, which transforms each instance x of A into an instance x' of B such that $opt_B(x') \leq \alpha \cdot opt_A(x)$,
2. there exists a function, computable in polynomial time, which transforms each solution y' of x' into a solution y of x such that $|m(x, y) - opt_A(x)| \leq \beta \cdot |m(x', y') - opt_B(x')|$.

The important property of this reduction is that it preserves PTAS; that is, if A is L-reducible to B and B has a PTAS then A has a PTAS as well.

We introduce now some minimization constraints satisfaction problems. Some of these problems as Min kSat for $k \geq 2$ do not have PTAS on general instances [KKM94] under usual complexity theoretic assumptions but can be approximated in polynomial time within some constant factors [BTV96]. The problem Min kDNF on the general instances for $k \geq 2$ cannot be approximated even within the ratio n^c for every constant $c > 0$ [KT94].

We assume that all Sat (DNF) representations we deal with in this paper will have clauses essentially depending on all literals, i.e., if a clause C depends on k variables, there is no clause C' depending on $k-1$ variables such that C is equivalent to C'.

Min kSat
Input: A set of clauses $C_1, \ldots, C_m$ of size at most k on n variables $x_1, \ldots, x_n$.
Output: A truth assignment to the variables that minimizes the number of clauses satisfied.

Min EkSat is the variant of Min kSat problem for which each clause contains exactly k

literals.

MIN kDNF
Input: A set of conjunctions $C_1, \ldots, C_m$ of size at most k on n variables $x_1, \ldots, x_n$.
Output: A truth assignment to the variables that minimizes the number of conjunctions satisfied.

MIN EkDNF is the variant of MIN kDNF problem where each conjunction contains exactly k literals.

A constraint on n boolean variables is a non-constant boolean function $f : \{0,1\}^k \to \{0,1\}$. A constraint application is a pair $(f, (i_1, \ldots, i_k))$ (where $i_j \neq i_\ell$ if $j \neq \ell$) where the i_j, $1 \leq j \leq k$ indicate to which among the n variables the function f is applied. In the following we will not distinguish constraint applications and constraints, more exactly we will speak only about constraints. The instances of the following two problems consist of a collection of constraints, that means that a function could appear more than one time but it is applied at different sets of variables.

MIN kCSP(DNF)
Input: A collection of m constraints $f_1, \ldots, f_m$ on boolean variables $x_1, \ldots, x_n$ where each f_j depends on at most k variables and is representated by a DNF formula.
Output: An assignment to x_i that minimizes the number of satisfied constraints.

MIN kCSP(SAT)
Input: A collection of m constraints $f_1, \ldots, f_m$ on boolean variables $x_1, \ldots, x_n$ where each f_j depends on at most k variables and is representated by a SAT formula.
Output: An assignment to x_i that minimizes the number of satisfied constraints.

Remark The problems MIN kCSP(DNF) and MIN kCSP(SAT) provide an explicit syntactic representation of constraints $f_1, \ldots, f_m$ and will provide a basis for the underlying notion of dense minimization problems.

MIN kCSP(EDNF) and MIN kCSP(ESAT) are the variants of the above problems where the clauses of each constraint representation of the collection have the same size.

Density. An instance of MIN kSAT (MIN kDNF) is δ-dense for some constant δ, if for each variable the total number of occurrences (reads) of the variable (and its negation) is at least δn^{k-1}. An instance of MIN kCSP(SAT) (MIN kCSP(DNF)) is δ-dense if for each variable the total number of occurrences of the variable (and its negation) is at least δn^{k-1} and an instance of MIN kCSP(SAT) (MIN kCSP(DNF)) is dense if there is a constant δ such that the instance is δ-dense. An instance of MIN kCSP(SAT) (MIN kCSP(DNF)) is average δ-dense if the number of boolean functions in it is at least δn^k and it is average-dense if there is a constant δ such that the instance is average δ-dense.

We show in the following that Dense MIN kCSP(SAT) and Dense MIN kCSP(DNF) are NP-hard. In fact we show that Dense MIN 2SAT is NP-hard and it is easy to see that MIN 2SAT is a particular case of MIN kCSP(SAT) and MIN kCSP(DNF).

We construct a direct reduction from MIN 2SAT (the problem proven NP-hard in [KKM94]) to Dense MIN 2SAT.

Given an instance F of MIN 2SAT on n variables, $x_1, \ldots, x_n$ and with m clauses $C_1, \ldots, C_m$ we define an instance F' of Dense MIN 2SAT as follows: We add n new variables

$y_1, \ldots, y_n$. F' will contain m clauses of F and the clauses $x_i \vee y_j, \bar{x}_i \vee y_j, 1 \leq j \leq n, 1 \leq i \leq n$. The total number of occurrences of x_i is at least $2n$ and the total number of occurrences of y_j is also at least $2n$. So, F' is a dense instance. Also, it is easy to see that $opt(F') = opt(F) + n^2$.

Remark that the reduction that prove the inapproximability of MIN 3DNF ([KT94]) use copying of variables but this process is not allowed without the introducing of new variables and by the introduction of new variables the density is lost.

3 Some useful L-reductions

We need firstly some technical results. More exactly we are going to show that if dense instances of MIN EkSAT problem (where the clauses have exactly the size k) have a polynomial time approximation scheme then dense instances of MIN kSAT have also a polynomial time approximation scheme. The same result holds also for MIN kDNF, MIN kCSP(SAT) and MIN kCSP(DNF).

Lemma 1 *For any $k \geq 2$, Dense* MIN kSAT $\leq_L$ *Dense* MIN EkSAT.

Proof: Let F be an instance δ-dense of MIN kSAT on n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$. We consider n new variables $y_1, \ldots, y_n$. The instance of MIN EkSAT that we construct, F', will contain the variables $x_1, \ldots, x_n, y_1, \ldots, y_n$. If a clause of F contains t literals, $C_i = \ell_1 \vee \ldots \vee \ell_t$ where $t < k$, then we define the clause $C'_i = \ell_1 \vee \ldots \vee \ell_t \vee y_1 \vee \ldots \vee y_{k-t}$. If a clause C_i contains k literals then we define $C'_i = C_i$. The instance F' contains the clauses C'_i, $1 \leq i \leq m$ at which we add $\begin{pmatrix} k \\ n \end{pmatrix}$ clauses of size k that are in fact all the possibilities to combine the variables $y_1, \ldots, y_n$ in sets of size k.

Since the density of F is δ, for each variable x_i, the total number of occurrences of the variable (and its negation) in F' is at least $\delta n^{k-1} = \frac{\delta (2n)^{k-1}}{2^{k-1}}$. The variables y_i, $1 \leq i \leq n$ have at least $\begin{pmatrix} k-1 \\ n-1 \end{pmatrix}$ occurrences, so at least $\frac{(2n)^{k-1}}{2^{2k-2}(k-1)!}$. So, the density of F' is at least $\min\{\frac{\delta}{2^{k-1}}, \frac{1}{2^{2k-2}(k-1)!}\}$.

Let justify in the following that this is a L-reduction. If $opt(F)$ is the value of an optimum solution of F then the optimum value of F', $opt(F') \leq opt(F)$ (we consider that the variables y take the value 0). Each solution of F' could be transformed in a solution of F' with a smaller value and where the variables y take the value 0. This solution give us a solution of F that satisfy the same number of clauses as it satisfy in F'. $\square$

In a similar way we can prove that

Lemma 2 *For any $k \geq 2$, Dense* MIN kDNF $\leq_L$ *Dense* MIN EkDNF.

Lemma 3 *For any $k \geq 2$, Dense* MIN kCSP(DNF) $\leq_L$ *Dense* MIN $(2k-1)$CSP(EDNF).

Proof: Let F be an instance δ-dense of MIN kCSP(DNF) on n variables, $x_1, \ldots, x_n$ and with m functions. We construct an instance F' of Dense MIN (2k-1)CSP(EDNF) as following: We add n new variables $y_1, \ldots, y_n$. F' will contain as variables $x_1, \ldots, x_n, y_1, \ldots, y_n$. If a function f_j of F has a clause with t literals, $C_i = \ell_1 \wedge \ldots \wedge \ell_t$ where $t < k$ then we define the clause $C'_i = \ell_1 \wedge \ldots \wedge \ell_t \wedge y_1 \wedge \ldots \wedge y_{k-t}$. If a clause C_i contains k literals then we define $C'_i = C_i$. Suppose $f_j = C_{j,1} \vee \ldots \vee C_{j,t_j}$. We define $f'_j = C'_{j,1} \vee \ldots \vee C'_{j,t_j}$. We define in the following the clause $A_{i,j}$, $1 \leq i \leq n, 1 \leq j \leq \begin{pmatrix} k-1 \\ n \end{pmatrix}$ as the clause that contains the variable x_i and the jth set of $k-1$ variables y among the n variables. For example $A_{j,1} = x_i \wedge y_1 \wedge \ldots \wedge y_{k-1}$.

The instance F' contains the functions $f'_j \vee A_{p,q}$, $1 \leq j \leq m, 1 \leq p \leq n, 1 \leq q \leq \begin{pmatrix} k-1 \\ n \end{pmatrix}$. F' is an instance of Min $(2k-1)$CSP(EDNF) since the clauses of the functions of F' have the same size k and each depends of at most $2k-1$ variables.

Since the density of F is δ, for each variable x_i, the total number of occurrences of the variable (and its negation) in F' is at least $\delta n^{k-1} \begin{pmatrix} k-1 \\ n \end{pmatrix} \geq c(\delta, k) n^{2k-1}$, where c is a constant function depending only on δ, k. A variable y_i has at least $\begin{pmatrix} k-2 \\ n \end{pmatrix} nm$ occurrences, so at least $c(k)n^{2k-1}$, where c is a constant function depending only on k.

Let justify in the following that this is an L-reduction. If $opt(F)$ is the value of an optimum solution of F, then the optimum of F', $opt(F') \leq opt(F)$ (the variables y take the value 0). Each solution of F' can be transformed in a solution of F' with a smaller value and where the variables y take the value 0. This solution give us a solution of F that satisfy the same number of functions as it satisfy in F'.

□

In a similar way we prove that

Lemma 4 *For any $k \geq 2$, Dense* Min kCSP(Sat) $\leq_L$ *Dense* Min $(2k-1)$CSP(ESat).

4 PTAS for Dense Min kCSP(EDNF) and Dense Min kCSP(ESat)

In this section we give firstly an approximation scheme for Dense Min E3Sat, after this we generalized it to Dense Min EkSat and Dense Min EkDNF. At the end of the section we give also a PTAS for Dense Min kCSP(EDNF) and Dense Min kCSP(ESat).

The idea of these approximation schemes is to run for each δ-dense instance two distinct algorithms and to select the solution with the smallest value. Let ε be the error within which we want to obtain the solution. One of the algorithm gives an $(1+\varepsilon)$-approximation for the instances whose minimum value is at least $f(\varepsilon, \delta, n, k)$ where f is a certain function. This algorithm consists in writing the problem as a smooth Integer Programming (IP) and to use the algorithm of [AKK95] in order to obtain an $(1+\varepsilon)$-approximation. The second algorithm gives a good solution for the instances with an optimum value greater than $f(\varepsilon, \delta, n, k)$. The idea of this algorithm is to do an exhaustive sampling and to combine it with a certain greedy algorithm. The both algorithms can be derandomized as in [AKK95].

4.1 Min E3Sat

Theorem 5 *Dense* Min E3Sat *has a PTAS.*

Let F be an input δ-dense of Min E3Sat with m clauses on a set $X = \{x_1, \ldots, x_n\}$ of

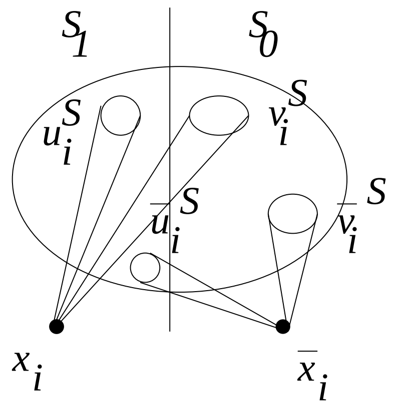

Figure 1:

variables. Let ε be an allowed error and $\alpha = \sqrt{\frac{6\delta^3\varepsilon}{10^3}}$. Similarly to [AKK95] we run two distinct algorithms and select the solution with the smallest value. The first algorithm gives a good solution for the instances whose minimum value is at least αn^3 and the second for the instances whose minimum value is less than αn^3.

1. **First algorithm** (Algorithm for the case of 'large" instances)

An ' aritmetization " method can be used to write a Min E3Sat instance as a degree 3 smooth integer program. We introduce n binary variables $y_1, \ldots, y_n$ and for each i, $1 \leq i \leq n$, we replace each occurrence of variable x_i by $1 - y_i$, each negated occurrence of variable x_i by y_i, each operator ' $\vee$ " by multiplication and for each clause we subtract the resulting term from 1. Let t_j be the polynomial obtained in this way from the jth clause.

So, Min E3Sat problem can be written as a degree 3 smooth IP as following :

$$\begin{cases} min \sum_{j=1}^{m} t_j(y_1, \ldots, y_n) \\ y_i \in \{0,1\} \; \forall i, \; 1 \leq i \leq n. \end{cases}$$

Using the algorithm of [AKK95] we find in polynomial time $(1+\varepsilon)$-approximation to it.

2. **Second algorithm** (Algorithm for the case of 'small" instances)

Given a set S of variables and an assignment to these variables. Denote by S_1 (S_0) the set of literals corresponding to the variables of S that are true (false). Let $X_1(X_0)$ be the set of literals that are true (false) in an optimum solution of F. We call a literal a *neighbor* of another literal if the two literals occur in a same clause in F. A number of such neighbors will be counted with multiplicities among the clauses of F.

For a variable x_i we define (Figure 1):

$$u_i^S = |\{neighbors \; of \; x_i \; among \; the \; literals \; of \; S_1\}$$
$$v_i^S = |\{neighbors \; of \; x_i \; among \; the \; literals \; of \; S_0\}$$
$$\bar{u}_i^S = |\{neighbors \; of \; \bar{x}_i \; among \; the \; literals \; of \; S_1\}$$
$$\bar{v}_i^S = |\{neighbors \; of \; \bar{x}_i \; among \; the \; literals \; of \; S_0\}$$

Let S be a set of $\ell = O((\log n)/\delta)$ variables picked randomly. For each possible assignment S_1, S_0 of the variables of S:

1. Let $V_1 = S_1$ and $V_0 = S_0$ be the current assignment to the literals. Let

$$T_1 = \{x_i \notin S : v_i^S \leq (u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)/8\}$$

$$T_2 = \{x_i \notin S : \bar{v}_i^S \leq (u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)/8\}.$$

   For each $x_i \in T_1$ we assign the value true at x_i and we introduce x_i in V_1 and $\bar{x}_i$ in V_0. For each $x_i \in T_2$ we assign the value false at x_i and we introduce x_i in V_0 and $\bar{x}_i$ in V_1.

2. Let U be the set of variables that have no value after the first step of the algorithm and let $U_1 = U_0 = \emptyset$. Suppose that $U = \{x_k, \ldots, x_n\}$.

   For $i = k$ to n we define :

$$val_i = 1/2|\{neighbors\, of\ x_i\ in\ V_0\, that\ not\ appear\ in\, the\ same\ clause\ as\ a\ literal\ of\ V_1\}|$$
$$+1/2|\{clauses\ of\ F\ with\ a\ literal\ x_i,\ a\ literal\ in\ V_0\ and\ a\ literal\ in\ U_0\}|$$
$$+1/2|\{clauses\ of\ F\ with\ a\ literal\ \bar{x}_i,\ a\ literal\ in\ V_0\ and\ a\ literal\ in\ U_1\}|$$
$$\overline{val_i} = 1/2|\{neighbors\, of\ \bar{x}_i\ in\ V_0\, that\ not\ appear\ in\, the\ same\ clause\ as\ a\ literal\ of\ V_1\}|$$
$$+1/2|\{clauses\ of\ F\ with\ a\ literal\ \bar{x}_i,\ a\ literal\ in\ V_0\ and\ a\ literal\ in\ U_0\}|$$
$$+1/2|\{clauses\ of\ F\ with\ a\ literal\ x_i,\ a\ literal\ in\ V_0\ and\ a\ literal\ in\ U_1\}|$$

   If $val_i \leq \overline{val_i}$ then assign to x_i the value true and put x_i in U_1 and $\bar{x}_i$ in U_0 and let $bias_i(U) = val_i$

   otherwise assign to x_i the value false and put x_i in U_0 and $\bar{x}_i$ in U_1 and let $bias_i(U) = \overline{val_i}$.

The clauses with at least a literal in V that are satisfied (and that were not satisfied by the current assignment V_1, V_0) when $x_i =$ true are the clauses with a literal x_i, a literal in V_0 and a literal in U_0 and the clauses with a literal $\bar{x}_i$, a literal in V_0 and a literal in U_1. For a clause $x_i \vee \ell_1 \vee \ell_2$ where $\ell_1 \in V_0$ and $\ell_2 \in U_0$ we added $1/2 + 1/2$ to the sum $\sum_{i=t}^{n} bias_i(U)$ when we place x_i in U_1. For a clause $\bar{x}_i \vee \ell_1 \vee \ell_2$ where $\ell_1 \in V_0$ and $\ell_2 \in U_1$ we added $1/2$ to this sum when we placed ℓ_2 in U_1 and we added $1/2$ when we placed $\bar{x}_i$ in U_0.

Let us sketch now a proof of correctness of the second algorithm. We denote by $opt(F)$ the value of an optimum solution of F and by $m(F, sol)$ the value of the solution given by the second algorithm.

Lemma 6 *With high probability,*

1. *T_1 contains each variable x_i with the property that in an optimum solution of F*

$$v_i^X \leq (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

2. *T_2 contains each variable x_i with the property that in an optimum solution of F*

$$\bar{v}_i^X \leq (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

Also with high probability, each variable in the set $T_1 \cup T_2$ is placed as in the optimum solution.

Proof: 1. Let x_i be a variable with the property that in the optimum solution

$$v_i^X \leq (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

Applying the Sampling Lemma for a random sample S of $O(\log n)$ variables, the probability that

$$v_i^S > (u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)/8$$

is $n^{-\Omega(1)}$. So, the probability that $x_i \in T_1$ is $1 - n^{-\Omega(1)}$.

2. Let x_i be a variable in T_1. Then

$$v_i^S \leq (u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)/8.$$

Applying the Sampling Lemma for a random sample of $O(\log n)$ variables, the probability that

$$v_i^X > (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10$$

is $n^{-\Omega(1)}$. Then the probability that

$$v_i^X \leq (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10$$

is $1 - n^{-\Omega(1)}$. □

Lemma 7 *With high probability, $n - (\ell + |T_1| + |T_2|) \leq \frac{10 opt(F)}{\delta n^2}$.*

Proof: If a variable x_i has no a truth value after the first step of the algorithm then from Lemma 6 with high probability

$$v_i^X > (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10$$

and

$$\bar{v}_i{}^X > (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

Since the sum $u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X \geq 2\delta n^2$, $opt(F) \geq 1/2 min\{v_i^X, \bar{v}_i{}^X\}|U| \geq (n - (\ell + |T_1| + |T_2|))2\delta n^2/2 \cdot 10$ (we divided by 2 since we can count in the above sum an edge twice). □

Lemma 8 *If $opt(F) < \alpha n^3$ then with high probability the number of clauses satisfied by the assignment given by the Algorithm 2 is at most $(1+\varepsilon)opt(F)$ where $\varepsilon = \frac{10^3\alpha^2}{6\delta^3}$.*

Proof: Let $v(V)$ be the number of clauses of F with at least a literal in V_1 and let $t(U)$ be the number of clauses with the three literals in U and at least one in U_1.

The number of clauses satisfied by the solution given by the second algorithm is

$$m(F, sol) = v(V) + \sum_{i=k}^{n} bias_i(U) + t(U).$$

The function $t(U) \leq \binom{3}{|U|} \leq |U|^3/6$ and the variables are placed in U such that

$$\sum_{i=k}^{n} bias_i(U) \leq \sum_{i=k}^{n} bias_i(U_{opt}).$$

So,

$$m(F, sol) \leq v(V) + \sum_{i=k}^{n} bias_i(U_{opt}) + t(U_{opt}) + t(U) - t(U_{opt}) \leq$$

$$\leq opt(F) + t(U) \leq opt(F) + |U|^3/6 \leq opt(F) + 10^3 opt(F)^3/(6\delta^3 n^6) \leq$$

$$\leq opt(F)(1 + 10^3\alpha^2 n^6/(6\delta^3 n^6)) = opt(F)(1 + 10^3\alpha^2/(6\delta^3)).$$

□

4.2 Min EkSat

Theorem 9 *Dense* Min EkSat *has a PTAS.*

Let F be an input δ-dense of Min EkSat with m clauses on n variables. Let ε be the allowed error and α a certain function that depends of ε, δ, k. We run two distinct algorithms and select the solution with the smallest value. The first algorithm gives a good solution for the instances whose minimum value is at least αn^k and the second for the instances whose minimum value is less than αn^k.

1.**First algorithm** (Algorithm for the case of 'large" instances)

We use the ' aritmetization " method as for Min E3Sat in order to write this problem as a degree k smooth IP.

2.**Second algorithm** (Algorithm for the case of 'small" instances)

Given a set S of variables and an assignment of these variables denote by S_1 (S_0) the set of literals of S that are true (false). Let $X_1(X_0)$ be the set of literals that are true(false) in an optimum solution of F. Again a literal will be a *neighbor* of another literal if the two literals occur in a same clause in F.

For a variable x_i we define by:

$$u_i^S = |\{neighbors\ of\ x_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_1\}$$

$$v_i^S = |\{neighbors\ of\ x_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_0\}$$

$$\bar{u}_i^S = |\{neighbors\ of\ \bar{x}_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_1\}$$

$$\bar{v}_i^S = |\{neighbors\ of\ \bar{x}_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_0\}$$

Let S be a set of $O((\log n)/\delta)$ variables picked randomly. For each possible assignment S_1, S_0 of the variables of S:

1. Let $V_1 = S_1$ and $V_0 = S_0$ be the current assignment of the literals.

   Let
$$T_1 = \{x_i \notin S : v_i^S \le (u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)/8\}$$
$$T_2 = \{x_i \notin S : \bar{v}_i^S \le (u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)/8\}.$$

   For each $x_i \in T_1$ we assign the value true at x_i and we introduce x_i in V_1 and $\bar{x}_i$ in V_0. For each $x_i \in T_2$ we assign the value false at x_i and we introduce x_i in V_0 and $\bar{x}_i$ in V_1.

2. Let U be the set of variables that have no value assigned after the first step of the algorithm and let $U_1 = U_0 = \emptyset$. Suppose that $U = \{x_t, \ldots, x_n\}$.

   For $i = t$ to n we define :

   - val_i is the size of the set of clauses of F with a literal x_i and the other literals with a truth value: at least a literal in V_0 and no literal in V_1
   - $\overline{val_i}$ is the size of the set of clauses of F with a literal $\bar{x}_i$ and the other literals with a truth value: at least a literal in V_0 and no literal in V_1

   If $val_i \le \overline{val_i}$ then assign to x_i the value true and put x_i in U_1 and $\bar{x}_i$ in U_0 and let $bias_i(U) = val_i$

   otherwise assign to x_i the value false and put x_i in U_0 and $\bar{x}_i$ in U_1 and let $bias_i(U) = \overline{val_i}$.

Let us sketch now a proof of correctness of the second algorithm. We denote by $opt(F)$ the value of an optimum solution of F by $m(F, sol)$ the value of the solution given by the second algorithm.

Lemma 10 *With high probability,*

1. *T_1 contains each variable x_i with the property that in an optimum solution of F*
$$v_i^X \le (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

2. *T_2 contains each variable x_i with the property that in an optimum solution of F*
$$\bar{v_i}^X \le (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

Also with high probability, each variable in the set $T_1 \cup T_2$ is placed as in the optimum solution.

Proof: Similarly with the proof of Lemma 6. □

Lemma 11 *With high probability,* $|U| \le \frac{10k\,opt(F)}{\delta n^{k-1}(k-1)}$.

Proof: If a variable x_i has no value after the first step of the algorithm then with high probability

$$v_i^X > (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10$$

and

$$\bar{v_i}^X > (u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/10.$$

So, $opt(F) \geq |U|\delta n^{k-1}(k-1)/(10k)$. $\square$

Lemma 12 *If $opt(F) < \alpha n^k$ then with high probability the number of clauses satisfied by the assignment given by the second algorithm is at most $(1+\varepsilon)opt(F)$ where $\varepsilon = \frac{10^k k^k \alpha^{k-1}}{\delta^k (k-1)^k}$.*

Proof: Let $v(V)$ be the number of clauses of F with at least a literal in V_1 and let $t(U)$ be the number of clauses with the k literals in U and at least one in U_1.

The number of clauses satisfied by the solution given by the second algorithm is

$$m(F, sol) = v(V) + \sum_{i=t}^{n} bias_i(U) + t(U).$$

The function $t(U) \leq \begin{pmatrix} k \\ |U| \end{pmatrix} \leq |U|^k$ and the variables are placed in U such that

$$\sum_{i=t}^{n} bias_i(U) \leq \sum_{i=t}^{n} bias_i(U_{opt}).$$

So,

$$m(F, sol) \leq v(V) + \sum_{i=t}^{n} bias_i(U_{opt}) + t(U_{opt}) + t(U) - t(U_{opt}) \leq$$

$$\leq opt(F) + t(U) \leq opt(F) + |U|^k \leq opt(F)(1+\varepsilon)$$

$\square$

4.3 Min EkDNF

We will turn now to the Min DNF problems.

Theorem 13 *Dense* Min EkDNF *has a PTAS.*

Let F be an instance δ-dense of Min kDNF with m clauses and n variables. Let ε be an allowed error and α a certain function depending of δ, ε, k. We again run two distinct algorithms and select the solution with the smallest value. The first algorithm gives a good solution for the instances whose minimum value is at least αn^k and the second for the instances whose minimum value is less than αn^k.

1. **First algorithm** (Algorithm for the case of 'large" instances)

Again an ' aritmetization " method can be used to write Min Ek DNF as a degree k smooth IP. We introduce n binary variables $y_1, \ldots, y_n$ and for each i, $1 \leq i \leq n$, we replace each occurrence of variable x_i by y_i, each negated occurrence of variable x_i by $1 - y_i$, each operator ' $\wedge$ " by multiplication. Let t_j be the polynomial obtained in this way from the jth clause.

So, MIN EkDNF can be written as a degree k smooth IP as following :

$$\begin{cases} min \sum_{j=1}^{m} t_j(y_1, \ldots, y_n) \\ y_i \in \{0,1\} \; \forall i, \; 1 \le i \le n. \end{cases}$$

Using the algorithm of [AKK95] we can find a polynomial time $(1+\varepsilon)$-approximation for it.

2. **Second algorithm** (Algorithm for the case of 'small" instances)

Given a set S of variables and an assignment of these variables denote by S_1 (S_0) the set of literals of S that are true (false). Let $X_1(X_0)$ be the set of literals that are true(false) in an optimum solution of F. A literal is a *neighbor* of another literal if the two literals occur in a same clause in F.

For a variable x_i we define by:

$$u_i^S = |\{neighbors\ of\ x_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_1\}$$
$$v_i^S = |\{neighbors\ of\ x_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_0\}$$
$$\bar{u}_i^S = |\{neighbors\ of\ \bar{x}_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_1\}$$
$$\bar{v}_i^S = |\{neighbors\ of\ \bar{x}_i\ (with\ multiplicity)\ among\ the\ literals\ of\ S_0\}$$

Let S be a set of $O((\log n)/\delta)$ variables picked randomly. For each possible assignment S_1, S_0 of the variables of S:

1. Let $V_1 = S_1$ and $V_0 = S_0$ be the current assignment of the literals.

   Let
   $$T_1 = \{x_i \notin S : u_i^S \le \frac{k}{4k+1}(u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)\}$$
   $$T_2 = \{x_i \notin S : \bar{u}_i^S \le \frac{k}{4k+1}(u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)\}.$$

   For each $x_i \in T_1$ we assign the value true at x_i and we introduce x_i in V_1 and $\bar{x}_i$ in V_0. For each $x_i \in T_2$ we assign the value false at x_i and we introduce x_i in V_0 and $\bar{x}_i$ in V_1.

2. Let U be the set of variables that have no value after the first step of the algorithm and let $U_1 = U_0 = \emptyset$. Suppose that $U = \{x_t, \ldots, x_n\}$.

   For $i = t$ to n we define :

   - val_i is the size of the set of clauses of F with a literal x_i and the other literals with a truth value and that are satisfied if $x_i = true$
   - $\overline{val_i}$ is the size of the set of clauses of F with a literal $\bar{x}_i$ and the other literals with a truth value and that are satisfied if $_i = false$

If $val_i \leq \overline{val_i}$ then assign to x_i the value true and put x_i in U_1 and $\bar{x}_i$ in U_0 and let $bias_i(U) = val_i$

otherwise assign to x_i the value false and put x_i in U_0 and $\bar{x}_i$ in U_1 and let $bias_i(U) = \overline{val_i}$.

Let us sketch now a proof of correctness of the second algorithm. We denote by $opt(F)$ the value of an optimum solution of F by $m(F, sol)$ the value of the solution given by the second algorithm.

Lemma 14 *With high probability,*

1. *T_1 contains each variable x_i with the property that in an optimum solution of F*

$$u_i^X \leq \frac{2k}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

2. *T_2 contains each variable x_i with the property that in an optimum solution of F*

$$\bar{u_i}^X \leq \frac{2k}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

Also with high probability, each variable in the set $T_1 \cup T_2$ is placed as in the optimum solution.

Proof: Similarly with the proof of Lemma 6. □

Lemma 15 *With high probability, $|U| \leq \frac{c(k)opt(F)}{\delta n^{k-1}}$.*

Proof: If a variable x_i has no value after the first step of the algorithm then with high probability

$$u_i^X > \frac{2k}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$$

and

$$\bar{u_i}^X > \frac{2k}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

So, the value $v_i^X + \bar{v}_i^X < \frac{1}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$ and thus $u_i^X > 2kv_i^X$ and $\bar{u}_i^X > 2k\bar{v}_i^X$.

The size of the multi-set of neighbors of x_i that make part (in an optimum solution) of a clause that is satisfied is at least $u_i^X - kv_i^X > \frac{k}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$. So, the number of clauses that contain x_i and are satisfied is at least $\frac{1}{4k+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$. The value $u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X \geq (k-1)\delta n^{k-1}$. So, the number of clauses that contain x_i and are satisfied is at least $\frac{k-1}{4k+1}\delta n^{k-1}$. Also, there are at least $\frac{k-1}{4k+1}\delta n^{k-1}$ clauses that contain $\bar{x}_i$ and are satisfied.

So, $opt(F) \geq |U|\delta n^{k-1}\frac{k-1}{(4k+1)k}$ since we could count a clause at most k times. □

Lemma 16 *If $opt(F) < \alpha n^k$ then with high probability the number of clauses satisfied by the assignment given by the second algorithm is at most $(1+\varepsilon)opt(F)$.*

Proof: Let $v(V)$ be the number of clauses of F with all the literals in V_1.

The number of clauses satisfied by the solution given by the second algorithm is

$$m(F, sol) = v(V) + \sum_{i=t}^{n} bias_i(U) + v(U).$$

The function $v(U) \leq \binom{k}{|U|} \leq |U|^k$ and the variables are placed in U such that

$$\sum_{i=t}^{n} bias_i(U) \leq \sum_{i=k}^{n} bias_i(U_{opt}).$$

So,

$$m(F, sol) \leq v(V) + \sum_{i=t}^{n} bias_i(U_{opt}) + v(U_{opt}) + v(U) - v(U_{opt}) \leq$$

$$\leq opt(F) + v(U) \leq opt(F) + |U|^k \leq opt(F)(1+\varepsilon)$$

□

4.4 Min kCSP(EDNF)

Theorem 17 *Dense* Min kCSP(EDNF) *has a PTAS.*

Let F be an input δ-dense of Min kCSP(EDNF) on n variables with m constraints $f_1, \ldots, f_m$ in DNF each function depending of at most k variables and the clauses of the functions of F have the size exactly ℓ, $1 \leq \ell \leq k$. Let ε be the allowed error and $\alpha = g(k, \varepsilon, \delta, \ell)$ a certain function depending of $k, \varepsilon, \delta, \ell$ that will be determinated in the following. We run two distinct algorithms and select the solution with the smallest value. The first algorithm gives a good solution for the instances whose minimum value is at least αn^k and the second for the instances whose minimum value is less than αn^k.

1. **First algorithm** (Algorithm for the case of 'large" instances)

This problem can be written as a degree k smooth IP. We introduce n binary variables $y_1, \ldots, y_n$. For an assignment of $x_1, \ldots, x_k$ for which f_j has the value 1 we define a degree k polynomial as following: if the variable x_i has the value 0 then we replace it by $1 - y_i$ and it has the value 1 we replace it by y_i. After we multiply these degree 1 polynomials obtaining a degree k polynomial. The degree k polynomial associated to f_j, t_j, is the sum of these degree k polynomials corresponding to the assignments of $x_1, \ldots, x_k$ for which f_j takes the value 1.

So, Min kCSP(EDNF) can be written as a degree k smooth IP as following :

$$\begin{cases} min \sum_{j=1}^{m} t_j(y_1, \ldots, y_n) \\ y_i \in \{0, 1\} \; \forall i, \; 1 \leq i \leq n. \end{cases}$$

Using the algorithm of [AKK95] we can find a polynomial time $(1+\varepsilon)$-approximation for it.

2. **Second algorithm** (Algorithm for the case of 'small" instances)

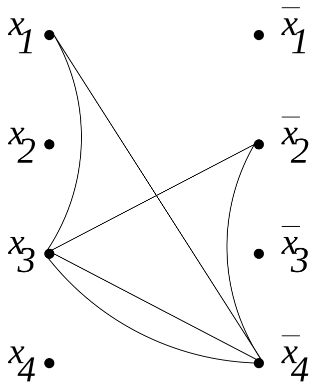

Figure 2: The graph $G(F)$ when $k = 4, \ell = 3$ and $F = (x_1 \wedge x_3 \wedge \bar{x}_4) \vee (\bar{x}_2 \wedge x_3 \wedge \bar{x}_4)$

Given a set S of variables and an assignment of these variables denote by S_1 (S_0) the set of literals of S that are true (false). Let $X_1(X_0)$ be the set of literals that are true(false) in an optimum solution of F.

We define $Lit(f_j)$ the set of the literals of the function f_j.

We define a multi-graph associated to the collection of functions F as following: The vertex set is the set of the $2n$ literals. For each clause $C_{j,i}$ of the function $f_j = C_{j,1} \vee \ldots \vee C_{j,t_j}$ of F we add in $G(F)$ the edges of a complete graph where the vertices are the literals of $C_{j,i}$.

In the following by the neighbors of a literal we understand the neighbors of the corresponding vertex in $G(F)$.

Let S be a set of $O((\log n)/\delta)$ variables picked randomly. For each possible assignment S_1, S_0 of the variables of S and for a variable x_i we define by:

- u_i^S is the size of the multi-set of neighbors of x_i in S_1,
$$u_i^S = |\{neighbours\ of\ x_i\ in\ S_1\}|$$
- v_i^S is the size of the multi-set of neighbors of x_i in S_0,
$$v_i^S = |\{neighbours\ of\ x_i\ in\ S_0\}|$$
- $\bar{u}_i^S$ is the size of the multi-set of neighbors of $\bar{x}_i$ in S_1,
$$\bar{u}_i^S = |\{neighbours\ of\ \bar{x}_i\ in\ S_1\}|$$
- $\bar{v}_i^S$ is the size of the multi-set of neighbors of $\bar{x}_i$ in S_0,
$$\bar{v}_i^S = |\{neighbours\ of\ \bar{x}_i\ in\ S_0\}|.$$

1. Let $V_1 = S_1$ and $V_0 = S_0$ be the current assignment of the literals.

   Let
$$T_1 = \{x_i \notin S : u_i^S \leq \frac{\ell}{4\ell + 1}(u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)\}$$
$$T_2 = \{x_i \notin S : \bar{u}_i^S \leq \frac{\ell}{4\ell + 1}(u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)\}.$$

For each $x_i \in T_1$ we assign the value true at x_i and we introduce x_i in V_1 and $\bar{x}_i$ in V_0. For each $x_i \in T_2$ we assign the value false at x_i and we introduce x_i in V_0 and $\bar{x}_i$ in V_1.

2. Let U be the set of variables that have no value after the first step of the algorithm and let $U_1 = U_0 = \emptyset$. Suppose that $U = \{x_t, \dots, x_n\}$.

   For $i = t$ to n we define :

   - val_i is the size of the collection of functions of F that contains only literals with a truth value and x_i, $\bar{x}_i$ and that are satisfied if x_i is true,

   $$val_i = |\{f_j : Lit(f_j) \subseteq V \cup U \cup \{x_i, \bar{x}_i\},\ f_j\ satisfied\ if\ x_i = true\}|$$

   - $\overline{val_i}$ is the size of the collection of F that contains only literals with a truth value and x_i, $\bar{x}_i$ and that are satisfied if x_i is false,

   $$val_i = |\{f_j : Lit(f_j) \subseteq V \cup U \cup \{x_i, \bar{x}_i\},\ f_j\ satisfied\ if\ x_i = false\}|$$

   If $val_i \leq \overline{val_i}$ then assign to x_i the value true and put x_i in U_1 and $\bar{x}_i$ in U_0 and let $bias_i(U) = val_i$

   otherwise assign to x_i the value false and put x_i in U_0 and $\bar{x}_i$ in U_1 and let $bias_i(U) = \overline{val_i}$.

Let us sketch now a proof of correctness of the second algorithm. We denote by $opt(F)$ the value of an optimum solution of F by $m(F, sol)$ the value of the solution given by the second algorithm.

Lemma 18 *With high probability,*

1. *T_1 contains each variable x_i with the property that in an optimum solution*

$$u_i^X \leq \frac{2\ell}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

2. *T_2 contains each variable x_i with the property that in an optimum solution*

$$\bar{u}_i^X \leq \frac{2\ell}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

Also with high probability, each variable in the set $T_1 \cup T_2$ is placed as in the optimum solution.

Proof: Similar with the proof of Lemma 6. $\square$

Lemma 19 *With high probability, $|U| \leq c(k, \delta, \ell)\frac{opt(F)}{n^{k-1}}$, where c is a constant.*

Proof: If a variable x_i has no truth value after the first step of the algorithm then with high probability

$$u_i^X > \frac{2\ell}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$$

and

$$\bar{u}_i^X > \frac{2\ell}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

So, the value $v_i^X + \bar{v}_i^X < (1 - \frac{4\ell}{4\ell+1})(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$ and thus $u_i^X > 2\ell v_i^X$ and $\bar{u}_i^X > 2\ell \bar{v}_i^X$.

The size of the multi-set of neighbors of x_i that make part (in an optimum solution) of a clause that is satisfied is at least $u_i^X - \ell v_i^X > \frac{\ell}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$. So, the number of clauses that contain x_i and are satisfied is at least $\frac{\ell}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)/\ell = \frac{1}{4\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$.

The value $u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X \geq (\ell-1)\delta n^{k-1}$. Since in the worst case the clauses that contain x_i and are satisfied are in the same function there are at least $\frac{1}{4\ell+1}(\ell-1)\delta n^{k-1}/2^k$ functions that contain x_i and are satisfied. Also, there are at least $\frac{1}{4\ell+1}(\ell-1)\delta n^{k-1}/2^k$ functions that contain $\bar{x}_i$ and are satisfied.

So, $opt(F) \geq |U|\delta n^{k-1}\frac{\ell-1}{(4\ell+1)2^k \cdot k}$ since we could count a function at most k times (one time for each variable that it depends). □

Lemma 20 *If $opt(F) < \alpha n^k$ then with high probability the number of functions satisfied by the assignment given by the second algorithm is at most $(1+\varepsilon)opt(F)$.*

Proof: Let $f(V)$ be the number of constraints f_j with $Lit(f_j) \subseteq V$ and that are satisfied.

The number of constraints satisfied by the solution given by the second algorithm is

$$m(F, sol) = f(V) + \sum_{i=t}^{n} bias_i(U) + f(U).$$

The function $f(U) \leq \binom{k}{|U|} h(k,\ell) \leq |U|^k h(k,\ell)$ where $h(k,\ell)$ is the number of ℓDNF functions on a fixed set of k variables. Also, the variables are placed in U such that

$$\sum_{i=t}^{n} bias_i(U) \leq \sum_{i=t}^{n} bias_i(U_{opt}).$$

So,

$$m(F, sol) \leq f(V) + \sum_{i=t}^{n} bias_i(U_{opt}) + f(U_{opt}) + f(U) - f(U_{opt}) \leq$$

$$\leq opt(F) + f(U) \leq opt(F) + |U|^k h(k,\ell) \leq opt(F)(1+\varepsilon)$$

□

4.5 Min kCSP(ESat)

Theorem 21 *Dense* Min kCSP(ESat) *has a PTAS.*

Let F be an input δ-dense of Min kCSP(ESat) on n variables with m constraints $f_1, \ldots, f_m$ in CNF each function depending of at most k variables and the clauses of the functions of F have the size exactly ℓ, $1 \leq \ell \leq k$. Let ε be the allowed error and $\alpha = g(k, \varepsilon, \delta, \ell)$ a certain function depending of $k, \varepsilon, \delta, \ell$ that will be determinated in the following. We run two distinct algorithms and select the solution with the smallest value. The first algorithm gives a good solution for the instances whose minimum value is at least αn^k and the second for the instances whose minimum value is less than αn^k.

1. **First algorithm** (Algorithm for the case of 'large" instances)

The same as in the case of MIN kCSP(EDNF).

2. **Second algorithm** (Algorithm for the case of 'small" instances)

Given a set S of variables and an assignment of these variables denote by S_1 (S_0) the set of literals of S that are true (false). Let $X_1(X_0)$ be the set of literals that are true(false) in an optimum solution of F.

We define $Lit(f_j)$ the set of the literals of the function f_j.

We define a multi-graph associated to the collection of functions F as in the case of MIN kCSP(EDNF).

Let S be a set of $O((\log n)/\delta)$ variables picked randomly. For each possible assignment S_1, S_0 of the variables of S and for a variable x_i we define by:

- u_i^S is the size of the multi-set of neighbors of x_i in S_1,
$$u_i^S = |\{neighbours\ of\ x_i\ in\ S_1\}|$$
- v_i^S is the size of the multi-set of neighbors of x_i in S_0,
$$v_i^S = |\{neighbours\ of\ x_i\ in\ S_0\}|$$
- $\bar{u}_i^S$ is the size of the multi-set of neighbors of $\bar{x}_i$ in S_1,
$$\bar{u}_i^S = |\{neighbours\ of\ \bar{x}_i\ in\ S_1\}|$$
- $\bar{v}_i^S$ is the size of the multi-set of neighbors of $\bar{x}_i$ in S_0,
$$\bar{v}_i^S = |\{neighbours\ of\ \bar{x}_i\ in\ S_0\}|$$

1. Let $V_1 = S_1$ and $V_0 = S_0$ be the current assignment of the literals.

   Let
$$T_1 = \{x_i \notin S : u_i^S \leq \frac{2^{k-2}\ell}{2^k\ell+1}(u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)\}$$
$$T_2 = \{x_i \notin S : \bar{u}_i^S \leq \frac{2^{k-2}\ell}{2^k\ell+1}(u_i^S + v_i^S + \bar{u}_i^S + \bar{v}_i^S)\}.$$

   For each $x_i \in T_1$ we assign the value true at x_i and we introduce x_i in V_1 and $\bar{x}_i$ in V_0. For each $x_i \in T_2$ we assign the value false at x_i and we introduce x_i in V_0 and $\bar{x}_i$ in V_1.

2. Let U be the set of variables that have no value after the first step of the algorithm and let $U_1 = U_0 = \emptyset$. Suppose that $U = \{x_t, \ldots, x_n\}$.

   For $i = t$ to n we define :

- val_i is the size of the collection of functions of F that contains only literals with a truth value and x_i, $\bar{x}_i$ and that are satisfied if x_i is true,

$$val_i = |\{f_j : Lit(f_j) \subseteq V \cup U \cup \{x_i, \bar{x}_i\},\ f_j \ satisfied \ if \ x_i = true\}|$$

- $\overline{val_i}$ is the size of the collection of functions of F that contains only literals with a truth value and x_i, $\bar{x}_i$ and that are satisfied if x_i is false,

$$val_i = |\{f_j : Lit(f_j) \subseteq V \cup U \cup \{x_i, \bar{x}_i\},\ f_j \ satisfied \ if \ x_i = false\}|$$

If $val_i \leq \overline{val_i}$ then assign to x_i the value true and put x_i in U_1 and $\bar{x}_i$ in U_0 and let $bias_i(U) = val_i$

otherwise assign to x_i the value false and put x_i in U_0 and $\bar{x}_i$ in U_1 and let $bias_i(U) = \overline{val_i}$.

Let us sketch now a proof of correctness of the second algorithm. We denote by $opt(F)$ the value of an optimum solution of F by $m(F, sol)$ the value of the solution given by the second algorithm.

Lemma 22 *With high probability,*

1. *T_1 contains each variable x_i with the property that in an optimum solution*

$$u_i^X \leq \frac{2^{k-1}\ell}{2^k\ell + 1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

2. *T_2 contains each variable x_i with the property that in an optimum solution*

$$\bar{u_i}^X \leq \frac{2^{k-1}\ell}{2^k\ell + 1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

Also with high probability, each variable in the set $T_1 \cup T_2$ is placed as in the optimum solution.

Lemma 23 *With high probability, $|U| \leq c(k, \delta, \ell)\frac{opt(F)}{\delta n^{k-1}}$, where c is a constant.*

Proof: If a variable x_i has no value after the first step of the algorithm then with high probability

$$u_i^X > \frac{2^{k-1}\ell}{2^k\ell + 1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$$

and

$$\bar{u_i}^X > \frac{2^{k-1}\ell}{2^k\ell + 1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X).$$

So, the value $v_i^X + \bar{v}_i^X < (1 - \frac{2^k\ell}{2^k\ell+1})(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X) = \frac{1}{2^k\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$. So, in an optimum solution at most $\frac{1}{2^k\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$ clauses of size ℓ containing x_i are not satisfied and also at most $\frac{1}{2^k\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$ clauses of size ℓ containing $\bar{x}_i$ are not satisfied.

In the worst case these clauses are placed in different functions, and thus at most $\frac{1}{2^k\ell+1}(u_i^X + v_i^X + \bar{u}_i^X + \bar{v}_i^X)$ functions of F containing x_i are not satisfied and at most

$\frac{1}{2^k\ell+1}(u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X)$ functions of F containing $\bar{x}_i$ are not satisfied. So, in an optimum solution at most $\frac{1}{2^k\ell+1}(u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X)$ functions containing x_i or $\bar{x}_i$ are not satisfied. Since the literals x_i and $\bar{x}_i$ could appear in the 2^k clauses of a function, there are at least $\frac{1}{(\ell-1)2^k}(u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X)$ functions that contain the literals x_i or $\bar{x}_i$.

So there are at least

$$\frac{1}{\ell 2^k}(u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X)-\frac{1}{2^k\ell+1}(u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X)=\frac{1}{2^k\ell+1}(u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X)$$

functions containing x_i and $\bar{x}_i$ and are satisfied.

The value $u_i^X+v_i^X+\bar{u}_i^X+\bar{v}_i^X\geq(\ell-1)\delta n^{k-1}$. So, $opt(F)\geq|U|\delta n^{k-1}\frac{\ell-1}{(2^k\ell+1)k}$ since we could count a function at most k times (ones for each variable that it depends). □

Lemma 24 *If $opt(F)<\alpha n^k$ then with high probability the number of functions satisfied by the assignment given by the second algorithm is at most $(1+\varepsilon)opt(F)$.*

Proof: Let $f(V)$ be the number of constraints f_j with $Lit(f_j)\subseteq V$ and that are satisfied.

The number of constraints satisfied by the solution given by the second algorithm is

$$m(F,sol)=f(V)+\sum_{i=t}^{n}bias_i(U)+f(U).$$

The function $f(U)\leq\begin{pmatrix}k\\|U|\end{pmatrix}h(k,\ell)\leq|U|^kh(k,\ell)$ where $h(k,\ell)$ is the number of ℓCNF functions on a fixed set of k variables and the variables are placed in U such that

$$\sum_{i=t}^{n}bias_i(U)\leq\sum_{i=t}^{n}bias_i(U_{opt}).$$

So,

$$m(F,sol)\leq f(V)+\sum_{i=t}^{n}bias_i(U_{opt})+f(U_{opt})+f(U)-f(U_{opt})\leq$$

$$\leq opt(F)+f(U)\leq opt(F)+|U|^kh(k,\ell)\leq opt(F)(1+\varepsilon)$$

□

5 Conclusions

We observe that all the problems considered in this paper are hard to approximate for the case of average density. More exactly, Min X$\leq_L$ Average Dense Min X for all X$\in${EkSat, EkDNF, kCSP(ESat), kCSP(EDNF)}. We remark also that Average Dense Max kCSP does have PTAS (cf. [AKK95]) for all k.

Our results imply existence of PTASs for dense instances of several minimization problems that are very hard to approximate on general instances.

Min Hitting Set and Min Implicative Hitting Set restricted to sets of bounded size are constant approximable and MAX SNP-hard [H96]. Min 2CNF Deletion is MAX SNP-hard [KPRT97] and it is $O(\log n\log\log n)$-approximable. Min UnCut is proved

$O(\log n)$-approximable and MAX SNP-hard [GVY93]. Tight lower bounds for MAX Ek LIN2 were given in [H97]. The minimization version MIN Ek LIN2 is also hard to approximate since a special case of MIN E3 LIN2, NEAREST CODEWORD, was proven [ABSS93] to be hard to approximate within a factor of $2^{\log^{1-\varepsilon} n}$ for any $\varepsilon > 0$ unless NP $\subseteq$ QP. MIN HORN DELETION is a problem that is as hard to approximate as NEAREST CODEWORD.

Remark that all the above problems are included in the class MIN CSP defined by [KST97]. What happens with dense instances of problems from MIN ONES? MIN VERTEX COVER belong to MIN ONES and since Dense MIN VERTEX COVER has no PTAS ([CT96],[KZ97]) the class Dense MIN ONES has no PTAS also.

References

[ABSS93] S. Arora, L. Babai, J. Stern and Z. Sweedyk, *The hardness of approximate optima in lattice, codes, and systems of linear equations*, Proc. of 34th IEEE FOCS, 1993, 724–733.

[AFK96] S. Arora, A. Frieze and H. Kaplan, *A new rounding procedure for the assignment problem with applications to dense graph arrangements*, Proc. of 37th IEEE FOCS, 1996, 21–30.

[AKK95] S. Arora, D. Karger and M. Karpinski, *Polynomial time approximation schemes for dense instances of NP-hard problems*, Proc. of 27th ACM STOC, 1995, 284–293. The full paper will appear in Journal of Computer and System Sciences, 1999.

[BFdV99] C. Bazgan and W. Fernandez de la Vega, *A Polynomial Time Approximation Scheme for Dense* MIN 2SAT, Fundamentals of Computation Theory, LNCS 1684, Springer, 1999, 91–99.

[BTV96] D. Bertsimas, C-P. Teo and R. Vohra, *On dependent randomized rounding algorithms*, Conference on Integer Programming and Combinatorial Optimization, LNCS 1084, Springer, 1996, 330–344.

[CT96] A.E.F. Clementi and L. Trevisan, *Improved non-approximability results for vertex cover with density constraints*, Proc. of 2nd Conference on Computing and Combinatorics, COCOON'96, Springer, 1996, 333–342.

[C95] N. Creignou, *A dichotomy theorem for maximimum generalized satisfiability problems*, Journal of Computer and System Sciences 51 (1995), 511–522.

[FdV96] W. Fernandez de la Vega, *Max-Cut has a Randomized Approximations Scheme in Dense Graphs*, Random Structures and Algorithms, 8(3) (1996), 187–198.

[FdVK99] W. Fernandez de la Vega and M. Karpinski, *On approximation hardness of dense TSP and other path problem*, Information Proccesing Letters 70 (1999), 53–55.

[FK96] A. Frieze and R. Kannan, *The Regularity Lemma and approximation schemes for dense problems* , Proc. of 37th IEEE FOCS, 1996, 12–20.

[FK99] A. Frieze and R. Kannan, *Quick Approximation to Matrices and Applications*, Combinatorica 19 (1999), 175–220.

[GVY93] N. Garg, V. Vazirani and M. Yannakakis *Approximate max-flow min-(multi)cut theorems and theirs applications*, SIAM Journal of Computing 25(1996), 235–251.

[GGR96] O. Goldreich, S. Goldwasser and D. Ron, *Property Testing and its Connection to Learning and Approximation*, Proc. of 37th IEEE FOCS, 1996, 339–348. The full paper has appeared in Journal of the ACM, 45 (4) (1998), 653–750.

[H96] M. M. Halldórsson, *Approximating kset cover and complementary graph coloring*, Proc. 5th International Conference on Integer Programming and Combinatorial Optimization, LNCS 1084, Springer, 1996, 118–131.

[H97] J. Hastad, *Some optimal inapproximability results*, Proc. of 29th ACM STOC, 1997, 1–10.

[K97] M. Karpinski, *Polynomial Time Approximation Schemes for Some Dense Instances of NP-Hard Optimization Problems*, Randomization and Approximation Techniques in Computer Science, LNCS 1269, Springer, 1997, 1–14.

[KZ97] M. Karpinski and A. Zelikovsky, *Approximating Dense Cases of Covering Problems*, ECCC Technical Report TR 97-004, 1997, appeared also in DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 40, 1998, 169–178.

[KWZ97] M. Karpinski, J. Wirtgen and A. Zelikovsky, *An approximation algorithm for the* BANDWIDTH *problem on dense graphs* , ECCC Technical Report TR 97-017, 1997.

[KSW97] S. Khanna, M. Sudan and D. Williamson, *A complete classification of the approximability of maximization problems derived from boolean constraint satisfaction*, Proc. of 29th ACM STOC, 1997, 11–20.

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

[KPRT97] P. Klein, S. Plotkin, S. Rao and E. Tardos, *Approximation algorithms for steiner and directed multicuts*, Journal of Algorithms 22(1997), 241–269.

[KKM94] R. Kohli, R. Krishnamurti and P. Mirchandani, *The Minimum Satisfiability Problem*, SIAM Journal on Discrete Mathematics 7(1994), 275–283.

[KT94] P.G. Kolaitis and M.N. Thakur, *Logical definability of NP optimization problems*, Information and Computation 115 (1994), 321–353.

[PY91] C. Papadimitriou and M. Yannakakis, *Optimization, Approximation and Complexity Classes*, Journal of Computer and System Science 43 (1991), 425–440.

[S78] T. Schaefer, *The complexity of satisfiability problems*, Proc. of 10th ACM STOC, 1978, 216–226.