ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.3, March 2008, pp.511–521
DOI: 10.3724/SP.J.1001.2008.00511

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

k-LSAT ($k\geq3$)是 NP-完全的*

许道云+, 邓天炎, 张庆顺

(贵州大学 计算机科学系,贵州 贵阳 550025)

k-LSAT is NP-Complete for $k\geq3$

XU Dao-Yun+, DENG Tian-Yan, ZHANG Qing-Shun

(Department of Computer Science, Guizhou University, Guiyang 550025, China)

+ Corresponding author: Phn: +86-851-3627649, Fax: +86-851-3627649, E-mail: dyxu@gzu.edu.cn

Xu DY, Deng TY, Zhang QS. *k*-LSAT is NP-complete for $k\geq3$. *Journal of Software*, 2008,19(3):511–521. http://www.jos.org.cn/1000-9825/19/511.htm

Abstract: A CNF formula *F* is linear if any distinct clauses in *F* contain at most one common variable. A CNF formula *F* is exact linear if any distinct clauses in *F* contain exactly one common variable. All exact linear formulas are satisfiable[1], and for the class LCNF of linear formulas, the decision problem LSAT remains NP-complete. For the subclasses $LCNF_{\geq k}$ of LCNF, in which formulas have only clauses of length at least *k*, the NP-completeness of the decision problem $LSAT_{\geq k}$ is closely relevant to whether or not there exists an unsatisfiable formula in $LCNF_{\geq k}$, i.e., the NP-completness of SAT for $LCNF_{\geq k}$ ($k\geq3$) is the question whether there exists an unsatisfiable formula in $LCNF_{\geq k}$. S. Porschen et al. have shown that both $LCNF_{\geq 3}$ and $LCNF_{\geq 4}$ contain unsatisfiable formulas by the constructions of hypergraphs and latin squares. It leaves the open question whether for each $k\geq5$ there is an unsatisfiable formula in $LCNF_{\geq k}$. This paper presents a simple and general method to construct unsatisfiable formulas in *k*-LCNF for each $k\geq3$ by the application of minimal unsatisfiable formulas to reductions for formulas. It is shown that for each $k\geq3$ there exists a minimal unsatisfiable formula in *k*-LCNF. Therefore, the stronger result is shown that *k*-LSAT is NP-complete for $k\geq3$.
Key words: linear CNF formula; unsatisfiability; NP-completeness; minimal unsatisfiable formula; reduction

摘 要: 合取范式(conjunctive normal form,简称 CNF)公式 *F* 是线性公式,如果 *F* 中任意两个不同子句至多有一个公共变元.如果 *F* 中的任意两个不同子句恰好含有一个公共变元,则称 *F* 是严格线性的.所有的严格线性公式均是可满足的,而对于线性公式类 LCNF,对应的判定问题 LSAT 仍然是 NP-完全的.$LCNF_{\geq k}$是子句长度大于或等于 *k* 的 CNF 公式子类,判定问题 $LSAT_{\geq k}$ 的 NP-完全性与 $LCNF_{\geq k}$ 中是否含有不可满足公式密切相关.即 $LSAT_{\geq k}$ 的 NP-完全性取决于 $LCNF_{\geq k}$ 是否含有不可满足公式.S.Porschen 等人用超图和拉丁方的方法构造了 $LCNF_{\geq 3}$ 和 $LCNF_{\geq 4}$ 中的不可满足公式,并提出公开问题:对于 $k\geq5$,$LCNF_{\geq k}$ 是否含有不可满足公式?将极小不可满足公式应用于公式的归约,引入了一个简单的一般构造方法.证明了对于 $k\geq3$,*k*-LCNF 含有不可满足公式,从而证明了一个更强的结果:对于 $k\geq3$,*k*-LSAT 是 NP-完全的.
关键词: 线性 CNF 公式;不可满足性;NP-完全性;极小不可满足公式;归约

* Supported by the National Natural Science Foundation of China under Grant Nos.10410638, 60310213 (国家自然科学基金)
Received 2006-12-13; Accepted 2007-01-24

中图法分类号: TP301 文献标识码: A

1 Introduction

A literal is a propositional variable or a negated propositional variable. A clause C is a disjunction of literals, $C=(L_1\vee\ldots\vee L_m)$ or a set $\{L_1,\ldots,L_m\}$ of literals. A formula F in conjunctive normal form (CNF) is a conjunction of clauses, $F=(C_1\wedge\ldots\wedge C_n)$ or a set $\{C_1,\ldots,C_n\}$ of clauses, or a list $[C_1,\ldots,C_n]$ of clauses. $var(F)$ is the set of variables occurring in the formula F and $var(C)$ is the set of the variables in the clause C. We denote $\#cl(F)$ as the number of clauses of F and $\#var(F)$ (or $|var(F)|$) as the number of variables occurring in F. CNF(n,m) is the class of CNF formulas with n variables and m clauses. The *deficiency* of a formula F is defined as $\#cl(F)-\#var(F)$, denoted by $d(F)$. A formula F is minimal unsatisfiable (MU) if F is unsatisfiable and $F-\{C\}$ is satisfiable for any clause $C\in F$. It is well known that F is not minimal unsatisfiable if $d(F)\le 0$[1,2]. So, we denote $MU(k)$ as the set of minimal unsatisfiable formulas with deficiency $k\ge 1$. Whether or not a formula belongs to $MU(k)$ can be decided in polynomial time[3].

A CNF formula F is linear if any two distinct clauses in F contain at most one common variable. A CNF formula F is exact linear if any two distinct clauses in F contain exactly one common variable. We define $k\text{-}CNF:=\{F\in CNF|(\forall C\in F)(|C|=k)\}$, $LCNF:=\{F\in CNF|F \text{ is linear}\}$, $XLCNF:=\{F\in CNF|F \text{ is exact linear}\}$, $LCNF_{\ge k}:=\{F\in LCNF|(\forall C\in F)(|C|\ge k)\}$ and $k\text{-}LCNF:=\{F\in LCNF|(\forall C\in F)(|C|=k)\}$. The decision problems of satisfiability are denoted as k-SAT, LSAT, XLSAT and k-LSAT for restricted instances to the corresponding to the above subclasses, respectively.

It is shown that every exact linear formulas is satisfiable[4], but LSAT remains NP-completeness[4−6]. For the subclasses $LCNF_{\ge k}$, $LSAT_{\ge k}$ remains NP-completeness if there exists an unsatisfiable formula in $LCNF_{\ge k}$[4−6]. Therefore, the NP-completeness of $LSAT_{\ge k}$ for $k\ge 3$ is the question whether there exists an unsatisfiable formula in $LCNF_{\ge k}$. We are interested in some NP-complete problems for linear formulas, and get some simplified NP-complete problem by constructing unsatisfiable linear formulas. It is helpful to analyze complexity of resolutions, and to find some effective algorithm for satisfiability.

In Refs.[4,6], by the constructions of hypergraphs and latin squares, the unsatisfiable formulas in $LCNF_{\ge 3}$ and $LCNF_{\ge 4}$ are constructed, respectively. But, the method is too complex and has no generalization. In Ref.[4], it leaves the open question whether for each $k\ge 5$ there is an unsatisfiable formula in $LCNF_{\ge k}$.

It is well known that 3-*SAT* is NP-complete. In the transformation from a CNF formula to a 3-CNF formula, we found a basic application of minimal unsatisfiable: for a clause $C=(L_1\vee L_2\vee\ldots\vee L_p)$ $(p>3)$ one can introduce $(p-3)$ new $y_1,y_2,\ldots,y_{p-3}$ variables, and split C into a partition $\{L_1,L_2\},\{L_3\},\ldots,\{L_{p-2}\},\{L_{p-1},L_p\}$ of C, and then construct $(p-2)$ clauses $(L_1\vee L_2\vee y_1),(L_3\vee\neg y_1\vee y_2),\ldots,(L_{p-2}\vee\neg y_{p-4}\vee y_{p-3}),(L_{p-1}\vee L_p\vee y_{p-3})$. In fact $[y_1,(\neg y_1\vee y_2),\ldots,(\neg y_{p-4}\vee y_{p-3}),\neg y_{p-3}]$ is a minimal unsatisfiable in $MU(1)$, and the partition $\{L_1,L_2\},\{L_3\},\ldots,\{L_{p-3}\},\{L_{p-1},L_p\}$ of C corresponds to a CNF formula $[(L_1\vee L_2),L_3,\ldots,L_{p-2},(L_{p-1}\vee L_p)]$. Thus, the formula $[(L_1\vee L_2\vee y_1),(L_3\vee\neg y_1\vee y_2),\ldots,(L_{p-2}\vee\neg y_{p-4}\vee y_{p-3}),(L_{p-1}\vee L_p\vee\neg y_{p-3})]$ is viewed as *clauses-disjunction* of $[(L_1\vee L_2),L_3,\ldots,L_{p-2},(L_{p-1}\vee L_p)]$ and $[y_1,(\neg y_1\vee y_2),\ldots,(\neg y_{p-4}\vee y_{p-3}),\neg y_{p-3}]$ at the corresponding positions of clauses, respectively. Additionally, an unit clause L corresponds to the formula $[(L\vee y\vee z),(L\vee y\vee\neg z),(L\vee\neg y\vee z),(L\vee\neg y\vee\neg z)]$, where $[(y\vee z),(y\vee\neg z),(\neg y\vee z),(\neg y\vee\neg z)]$ is a minimal unsatisfiable formula $MU(2)$, and a clause $(L_1\vee L_2)$ corresponds to the formula $[(L_1\vee L_2\vee y),(L_1\vee L_2\vee\neg y)]$, where $[y,\neg y]=y\wedge\neg y$ is a minimal unsatisfiable formula MU(1). It implies that a subclause of the original clause can be copied.

Based on this observation and the characterization of minimal unsatisfiable formulas, we introduce a generalize method in Lemma 1 and Lemma 2, which we can transform a CNF formula into a required CNF formula by constructing proper minimal unsatisfiable formulas. We have applied this method to reduction for formulas. In Ref.[7], we present an algorithm to solve an open problem in Ref.[8], which for fixed k and t ($3\leq t<k$), one can transform a k-CNF formula F to a t-CNF formula F' in linear time on the size of F with the same satisfiability. For some simplified NP-complete problems restricted instances to the subclass (k,s)-CNF the method is also used[9,10], where (k,s)-CNF is a subclass of CNF, $F\in(k,s)$-CNF if and only if (iff) F has only clauses of length k, and the number of occurrences of each variable in F is less than s.

In this paper, we present a simple and general method to construct unsatisfiable formulas in k-LCNF for each $k\geq 3$ by the application of minimal unsatisfiable formulas and the induction. It is shown for each $k\geq 3$ that there exists a minimal unsatisfiable formula in k-LCNF. Based on existences of minimal unsatisfiable formulas in k-LCNF, the stronger result is shown that k-LSAT is NP -complete for $k\geq 3$. In our proof, we introduce two algorithms: Algorithm 1 is for transforming a k-CNF to a linear formula and Algorithm 2 is for lengthening clauses of linear formulas.

2 Minimal Unsatisfiable Formulas and Its Applications

A clause $C=(L_1\vee L_2\vee\ldots\vee L_n)$ can be represented as a set $\{L_1,L_2,\ldots,L_n\}$ of literals. Similarly, A CNF formulas $F=(C_1\wedge C_2\wedge\ldots\wedge C_m)$ can be represented as a set $\{C_1,C_2,\ldots,C_m\}$ of clauses, or a list $[C_1,C_2,\ldots,C_m]$ of clauses. $var(F)$ is the set of variables occurring in the formula F and $var(C)$ is the set of the variables in the clause C. We define $|F|=\sum_{1\leq i\leq m}|C_i|$ as the size of F. In this paper, the formulas mean CNF formulas.

A formula $F=[C_1,\ldots,C_m]$ with n variables $x_1,\ldots,x_n$ in $CNF(n,m)$ can be represented as a $n\times m$ matrix $(a_{i,j})$, called the representation matrix of F, where $a_{ij}=+$ if $x_i\in C_j$, $a_{ij}=-$ if $\neg x_i\in C_j$, otherwise $a_{ij}=0$ (or, blank).

A formula F is called *minimal unsatisfiable* if F is unsatisfiable, and for any clause $f\in F$, $F-\{f\}$ is satisfiable. We denote MU as the class of minimal unsatisfiable formulas, and $MU(k)$ as the class of minimal unsatisfiable formulas with deficiency k. Let $C=(L_1\vee\ldots\vee L_n)$ be a clause. We view a clause as a set of literals. The collection $C_1,\ldots,C_m$ of subsets of C (as a set) is a partition of C, where $C=\bigcup_{1\leq i\leq m}C_i$ and $C_i\cap C_j=\phi$ for any $1\leq i\neq j\leq m$, which corresponds to a formula $F_C=C_1\wedge\ldots\wedge C_m$. We call F_C as a partition formula of C. Specially, the collection $\{L_1\},\ldots,\{L_n\}$ of singleton subsets of C is called the simple partition of C, and the formula $[L_1,\ldots,L_n]=L_1\wedge\ldots\wedge L_n$ is called the *simple partition formula* of C.

Let $F_1=[f_1,\ldots,f_m]$ and $F_2=[g_1,\ldots,g_m]$ be formulas. We denote $F_1\vee_{cl}F_2=[f_1\vee g_1,\ldots,f_m\vee g_m]$. Similarly, let C be a clause and $F=[f_1,\ldots,f_m]$ a formula, denote $C\vee_{cl}F=[(C\vee_{cl}f_1),\ldots,(C\vee_{cl}f_m)]$.

Lemma 1. Let $C=(L_1\vee\ldots\vee L_n)$ ($n\geq 2$) be a clause and $F_C=[C_1,\ldots,C_m]$ ($m\geq 2$) a partition formula of C. For any MU formula $H=[f_1,\ldots,f_m]$ with $var(C)\cap var(H)=\phi$, if a truth assignment ν satisfies the formula $F_C\vee_{cl}H$, then $\nu(C)=1$. Conversely, for any truth assignment ν_0 satisfying C, ν_0 can be extended into a truth assignment ν satisfying $F_C\vee_{cl}H$.

Proof: Let $C=(L_1\vee\ldots\vee L_n)$ be a clause and $F_C=[C_1,\ldots,C_m]$ ($m\geq 2$) a partition formula of C. Without losses of generality (w.l.o.g.), we assume $C_1=(L_1\vee\ldots\vee L_{i_1})$, $C_2=(L_{i_1+1}\vee\ldots\vee L_{i_2})$,…, $C_m=(L_{i_{m-1}+1}\vee\ldots\vee L_n)$.

Let ν be a truth assignment satisfying $F_C\vee_{cl}H$. Since H is minimal unsatisfiable, we have $\nu(f_k)=0$ for some ($1\leq k\leq m$). It must be $\nu(C_k)=1$. It implies $\nu(C)=1$ since C_k is a subclause of C.

Conversely, suppose that C is satisfied by a truth assignment ν_0. Since C is disjunction of literals $L_1,\ldots,L_n$, there exists some k ($1\leq k\leq n$) such that $\nu_0(L_k)=1$. W.l.o.g., we assume $\nu_0(L_1)=1$, then $\nu_0(C_1)=1$. Since H is minimal unsatisfiable, we have $H-\{f_1\}$ is satisfiable, thus there exists a truth assignment ν_1 such that $\nu_1(H-\{f_1\})=1$. Note that $var(C)\cap var(H)=\phi$, we can join into a truth assignment ν from ν_0 and ν_1, which for $x\in var(C)\cup var(H)$, $\nu(x)=\nu_0(x)$ for $x\in var(C)$, and $\nu(x)=\nu_1(x)$ for $x\in var(H)$. It is clear that ν is a truth assignment satisfying $F_C\vee_{cl}H$. □

Based on the method in Lemma 1 for a clause, we have the following Lemma 2. It presents a method constructing the required formulas.

Lemma 2. Let $F=C_1\wedge\ldots\wedge C_n$ be a formula with $|C_i|\ge 2$ for $1\le i\le n$. Suppose that for each $1\le i\le n$, F_i is a partition formula of C_i and $\#cl(F_i)=m_i\ge 2$. Let $H_1,\ldots,H_n$ be MU formulas satisfying the following conditions:

(1) For each $1\le i\le n$, $\#cl(H_i)=m_i$.

(2) $(\bigcup_{1\le i\le n} var(H_i))\cap var(F)=\varnothing$.

(3) For any $1\le i\ne j\le n$, $var(H_i)\cap var(H_j)=\varnothing$.

We define $F^*:=(F_1\vee_{cl}H_1)\wedge(F_2\vee_{cl}H_2)\wedge\ldots\wedge(F_n\vee_{cl}H_n)$. Then, F is satisfiable iff F^* is satisfiable.

Proof: (⇒) Assume that F is satisfiable. We have a truth assignment ν_0 over $var(F)$ such that $\nu_0(F)=1$. It implies $\nu_0(C_i)=1$ for each $1\le i\le n$. By the proof of Lemma 1, we can extend ν_0 into a truth assignment ν_i over $var(F)\cup var(H_i)$ such that $\nu_i(F_i\vee_{cl}H_i)=1$. By condition (3), we can combine $\nu_1,\ldots,\nu_n$ into a truth assignment ν^* over $var(F)\cup var(H_1)\cup\ldots\cup var(H_n)$ such that $\nu^*(F_i\vee_{cl}H_i)=1$ for each $1\le i\le n$, where $\nu^*(x):=\nu_0(x)$ for $x\in var(F)$ and $\nu^*(x):=\nu_i(x)$ for $x\in var(H_i)$ $(1\le i\le n)$. It means that F^* is satisfiable.

(⇐) Assume that F^* is satisfiable. We have a truth assignment ν over $var(F)\cup var(H_1)\cup\ldots\cup var(H_n)$ such that $\nu(F^*)=1$. It implies $\nu(F_i\vee_{cl}H_i)=1$ for each $1\le i\le n$. Note that for each $1\le i\le n$, H_i is minimal unsatisfiable and $\#cl(H_i)=\#cl(F_i)=m_i$. We have $\nu_i(H_i)=0$ for each $1\le i\le n$, where ν_i is the restriction of ν over $var(H_i)$. By the definition of $F_i\vee_{cl}H_i$ and $\nu(F_i\vee_{cl}H_i)=1$, there exists a clause $C_{i,j}$ of F_i such that $\nu_0(C_{i,j})=1$, where ν_0 is the restriction of ν over $var(F)$. Since $C_{i,j}$ is a subclause of C_i, we have $\nu_0(C_i)=1$. So, we have $\nu_0(C_i)=1$ for each $1\le i\le n$. It means that F is satisfiable. □

We now introduce the following four MU formulas.

(1) $A_n=[(x_1\vee\ldots\vee x_n),(\neg x_1\vee x_2),(\neg x_2\vee x_3),\ldots,(\neg x_{n-1}\vee x_n),(\neg x_n\vee x_1),(\neg x_1\vee\ldots\vee\neg x_n)]\in MU(2)$. Its representation matrix is

$$\begin{matrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-1} \\ x_n \end{matrix}\begin{pmatrix} + & - & & & & + & - \\ + & + & - & & & & - \\ \vdots & & + & & & & \vdots \\ \vdots & & & \cdots & & & \vdots \\ & & & & - & & \\ + & & & & + & - & - \end{pmatrix}.$$

We take a formula $A_n^c=[(\neg x_1\vee x_2),(\neg x_2\vee x_3),\ldots,(\neg x_{n-1}\vee x_n),(\neg x_n\vee x_1)]$. Clearly, both $A_n^c+\{(x_1\vee\ldots\vee x_n)\}$ and $A_n^c+\{(\neg x_1\vee\ldots\vee\neg x_n)\}$ are satisfiable, and $A_n^c+\{(x_1\vee\ldots\vee x_n)|=(x_1\wedge\ldots\wedge x_n)\}$ and $A_n^c+\{(\neg x_1\vee\ldots\vee\neg x_n)|=(\neg x_1\wedge\ldots\wedge\neg x_n)\}$.

Clearly, the subformula A_n^c of A_n is satisfiable, and for any truth assignment τ satisfying A_n^c it holds that $\tau(x_1)=\ldots=\tau(x_n)$. The formula A_n^c represents a cycle of implication: $x_1\to x_2\to\ldots\to x_n\to x_1$.

(2) $B_n=[(x_1\vee x_3),(\neg x_1\vee x_2),\ldots,(\neg x_s\vee x_{s+1}),\ldots,(\neg x_{n-2}\vee x_{n-1}),(\neg x_{n-1}\vee\neg x_3)]\in MU(1)$, where $n\ge 6$. The representation matrix of B_6 is

$$\begin{matrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{matrix}\begin{pmatrix} + & - & & & & \\ & + & - & & & \\ + & & + & - & & - \\ & & & + & - & \\ & & & & + & - \end{pmatrix}.$$

Note that $\#cl(B_n)=n$ and $\#var(B_n)=n-1$, and B_n is a linear formula for $n\ge 6$.

(3) The standard MU formulas S_n with n variables, $x_1,\ldots,x_n$, is defined by

$$S_n=\wedge_{(\varepsilon_1,\ldots,\varepsilon_n)\in\{0,1\}^n}(x_1^{\varepsilon_1}\vee\ldots\vee x_n^{\varepsilon_n}),$$

where $x_i^0 = x_i$ and $x_i^1 = \neg x_i$ for $1 \le i \le n$. Denote the clause $X_{\varepsilon_1,\dots,\varepsilon_n} = x_1^{\varepsilon_1} \vee \dots \vee x_n^{\varepsilon_n}$.

The representation matrix of S_3 is

$$\begin{matrix} x_1 \\ x_2 \\ x_3 \end{matrix}\begin{pmatrix} + & + & + & + & - & - & - & - \\ + & + & - & - & + & + & - & - \\ + & - & + & - & + & - & + & - \end{pmatrix}.$$

The above MU formulas are useful in constructions of the required formulas in this paper.

3 Construction of Linear Minimal Unsatisfiable Formulas

In this section, we introduce a subclass of CNF, called linear CNF formulas, and present a general constructing method of linear MU formulas.

Definition 1.

(1) A formula $F \in CNF$ is called linear if

(a) F contains no pair of complementary unit clauses, and

(b) For all $C_1, C_2 \in F$ with $C_1 \ne C_2$, $|var(C_1) \cap var(C_2)| \le 1$.

Let LCNF denote the class of all linear formulas.

(2) A formula $F \in CNF$ is called exact linear if F is linear, and for all $C_1, C_2 \in F$ with $C_1 \ne C_2$, $|var(C_1) \cap var(C_2)| = 1$.

For example, the formula B_n is linear for $n \ge 6$. Let (XLCNF) LCNF denote the class of all (exact) linear formulas. Similarly, denote by ($XLCNF_{\ge k}$) $LCNF_{\ge k}$ the class of all (exact) linear formulas, in which formulas have only clauses of length at least $k \in N$.

Lemma 3. Let $F=[C_1,\dots,C_m]$ be a MU formula with $|C_i|=l_i \ge 2$ for each $1 \le i \le m$, and let $G_i = [f_1^i,\dots,f_{l_i}^i]$ be a linear MU formula for $1 \le i \le m$, where $var(G_i) \cap var(G_j) = \phi$ for any $1 \le i \ne j \le m$. Then, the formula $F^* := \wedge_{1 \le i \le m}(F_{C_i} \vee_{cl} G_i)$ is a linear MU formula, where F_{C_i} is the simple partition formula of clause C_i for $1 \le i \le m$, and $var(\ \mathrm{var}(F) \cap (\bigcup_{1 \le i \le m} var(G_i)) = \phi$.

Proof: Let $F=[C_1,\dots,C_m]$ be a MU formula with $|C_i|=l_i \ge 2$ for each $1 \le i \le m$. For $1 \le i \le m$, we assume that $C_i = (L_{i,1} \vee \dots \vee L_{i,l_i})$ and define a *block* formula: $F_{C_i} \vee_{cl} G_i := [(L_{i,1} \vee f_1^i),\dots,(L_{i,l_i} \vee f_{l_i}^i)]$, where $F_{C_i} = [L_{i,1},\dots,L_{i,l_i}]$, and the the formula: $F^* := \wedge_{1 \le i \le m}(F_{C_i} \vee_{cl} G_i)$.

(1) F^* is minimal unsatisfiable.

Firstly, by Lemma 2, F^* is unsatisfiable since F is unsatisfiable and $G_1,\dots,G_m$ are minimal unsatisfiable.

Secondly, F^* is minimal unsatisfiable. For any clause $g \in F^*$, w.l.o.g., we assume $g = (L_{1,1} \vee f_1^1)$, and consider the satisfiability of $F^* - \{g\}$.

Since F is minimal unsatisfiable, there exists a truth assignment τ_0 over $var(F)$ satisfying $[C_2,\dots,C_m]$, and τ_0 forces each literal in C_1 to be false, i.e., $\tau_0(L_{1,1}) = \dots = \tau_0(L_{1,l_1}) = 0$, and $\tau_0(C_2)=\dots=\tau_0(C_m)=1$. Since G_1 is minimal unsatisfiable, there exists a truth assignment τ_1 over $var(G_1)$ satisfying $G_1 - \{f_1^1\}$. Thus, we have a truth assignment τ_1^* satisfying $(F_{C_1} \vee_{cl} G_1) - \{(L_{1,1} \vee f_1^1)\}$ by joining τ_0 and τ_1, where $\tau_1^*(x) = \tau_0(x)$ for $x \in var(F)$ and $\tau_1^*(x) = \tau_1(x)$ for $x \in var(G_1)$.

For each $2 \le k \le m$, since $\tau_0(C_k)=1$, there is a literal $L_{k,j_k} (1 \le j_k \le l_k)$ such that $\tau_0(L_{k,j_k}) = 1$. By the minimal satisfyability of G_k, we have that $G_k - \{f_{j_k}^k\}$ is satisfiable. Therefore, we have a truth assignment τ_k over $var(G_k)$ satisfying $G_k - \{f_{j_k}^k\}$. Thus, we have a truth assignment τ_k^* satisfying $(F_{C_k} \vee_{cl} G_k)$ by joining τ_0 and τ_k, where $\tau_k^*(x) = \tau_0(x)$ for $x \in var(F)$ and $\tau_k^*(x) = \tau_k(x)$ for $x \in var(G_k)$.

Finally, we have a truth assignment τ^* satisfying $F^*-\{g\}$ by combining $\tau_0,\tau_1,\ldots,\tau_m$, where $\tau^*(x)=\tau_0(x)$ for $x\in var(F)$ and $\tau^*(x)=\tau_k(x)$ for $x\in var(G_k)$ $(1\le k\le m)$.

(2) F^* is linear.

For any distinct clauses $f,g\in F^*$, we consider the following cases.

Case 1: Both f and g are in the same block formula.

There exists some k $(1\le k\le m)$ such that $f=(L_{k,s}\vee f_s^k)$ and $g=(L_{k,s'}\vee f_{s'}^k)$ for some $1\le s\ne s'\le l_k$. By $s\ne s'$, $var(f)\cap var(g)\subseteq var(f_s^k)\cap var(f_{s'}^k)$. Since G_k is linear, we have $|var(f_s^k)\cap var(f_{s'}^k)|\le 1$. Thus, $|var(f)\cap var(g)|\le 1$.

Case 2: f and g are in the different block formulas.

There exist some k and k' $(1\le k\ne k'\le m)$ such that $f\in(F_{C_k}\vee_{cl} G_k)$ and $g\in(F_{C_{k'}}\vee_{cl} G_{k'})$. By constructions of block formulas, we have $f=(L_{k,s}\vee f_s^k)$ for some $1\le s\le l_k$ and $g=(L_{k',s'}\vee f_{s'}^{k'})$ for some $1\le s'\le l_{k'}$. By $k\ne k'$, we have $var(G_k)\cap var(G_{k'})=\varnothing$. Thus, $var(f)\cap var(g)\subseteq var(L_{k,s})\cap var(L_{k',s'})$. It implies that $|var(f)\cap var(g)|\le 1$. □

In Lemma 3, we present a method constructing MU formulas k-LCNF for $k\ge 3$ by S_n and B_n $(n\ge 6)$.

We consider firstly the construction of formulas for the case of $k=3$.

We take MU formulas S_6 and B_6 with $var(S_6)\cap var(B_6)=\phi$ in Section 2. Note that B_6 is a linear MU formula, and $|C|=6$ for each $C\in S_6$, and $|C|=2$ for each $C\in B_6$.

For each clause $X_{\varepsilon_1,\ldots,\varepsilon_6}=(x_1^{\varepsilon_1}\vee\ldots\vee x_6^{\varepsilon_6})\in S_6$, we take the simple partition formula $F_{\varepsilon_1,\ldots,\varepsilon_6}=[x_1^{\varepsilon_1},\ldots,x_6^{\varepsilon_6}]=x_1^{\varepsilon_1}\wedge\ldots\wedge x_6^{\varepsilon_6}$ of $X_{\varepsilon_1,\ldots,\varepsilon_6}$, and take a copy of B_6, denoted by $B_6^{\varepsilon_1,\ldots,\varepsilon_6}$, and define a formula $(F_{\varepsilon_1,\ldots,\varepsilon_6}\vee_{cl} B_6^{\varepsilon_1,\ldots,\varepsilon_6})$.

It restricts $var(B_6^{\varepsilon_1,\ldots,\varepsilon_6})\cap var(B_6^{\varepsilon_1',\ldots,\varepsilon_6'})=\varnothing$ for any distinct $(\varepsilon_1,\ldots,\varepsilon_6),(\varepsilon_1',\ldots,\varepsilon_6')\in\{0,1\}^6$, and $var(B_6^{\varepsilon_1,\ldots,\varepsilon_6})\cap var(S_6)=\varnothing$ for any $(\varepsilon_1,\ldots,\varepsilon_6)\in\{0,1\}^6$.

We now define the following formula

$$SL_3:=\wedge_{(\varepsilon_1,\ldots,\varepsilon_6)\in\{0,1\}^6}(F_{\varepsilon_1,\ldots,\varepsilon_6}\vee_{cl} B_6^{\varepsilon_1,\ldots,\varepsilon_6}).$$

SL_3 is a linear MU formula by Lemma 3.

Note that $\#cl(SL_3)=6\cdot 2^6$, and $|C|=3$ for each $C\in SL_3$.

We define inductively a counting functions of clauses $cl(k)$ for $k\ge 3$: $cl(3)=6\cdot 2^6$ and $cl(k+1)=cl(k)\cdot 2^{cl(k)}$ for $k\ge 3$. For the case of $k\ge 3$, suppose that the linear formula SL_k has been constructed such that SL_k is a linear MU formula, and the length of each clause in SL_k equals to k.

By Lemma 3, we define inductively the following linear MU formula

$$SL_{k+1}:=\wedge_{(\varepsilon_1,\ldots,\varepsilon_{cl(k)})\in\{0,1\}^{cl(k)}}(F_{\varepsilon_1,\ldots,\varepsilon_{cl(k)}}\vee_{cl} SL_k^{\varepsilon_1,\ldots,\varepsilon_{cl(k)}})$$

where, for $(\varepsilon_1,\ldots,\varepsilon_{cl(k)})\in\{0,1\}^{cl(k)}$.

(a) $F_{\varepsilon_1,\ldots,\varepsilon_{cl(k)}}$ is the simple partition formula of clause $X_{\varepsilon_1,\ldots,\varepsilon_{cl(k)}}\in S_{cl(k)}$.

(b) $SL_k^{\varepsilon_1,\ldots,\varepsilon_{cl(k)}}$ is a copy SL_k with new variables.

$S_{cl(k)}$ is minimal unsatisfiable, SL_k is both minimal unsatisfiable and linear. By Lemma 3, SL_{k+1} is a linear MU formula. Thus, we have the following result:

Theorem 1. For each positive integer $k\ge 3$, k-LCNF contains MU formulas.

4 NP-Completeness of SAT for Linear Formulas

In this section, we consider complexities of decision problems of satisfiability for restricted instances in LCNF and $LCNF_{\ge k}$ $(k\ge 3)$, respectively.

Let F be a formula, we denote $pos(x,F)$ (resp. $neg(x,F)$) as the number of positive (resp. negative) occurrence of variable x in F, and write $occs(x,F)=pos(x,F)+neg(x,F)$. Sometimes, we denote F_{rest} as a subformula of F, which consists of rest clauses of F.

For a formula $F=[C_1,\ldots,C_m]$, the following facts are clear:

(1) If $pos(x,F)>0$ and $neg(x,F)=0$ (or, $pos(x,F)=0$ and $neg(x,F)>0$) for some $x\in var(F)$, then the resulting formula F' by deleting clauses, in which x occurs, has the same satisfiability with F.

(2) If $F=[(x\vee y\vee C_1'),(\neg x\vee\neg y\vee C_2'),F_{rest}]$ (or $F=[(x\vee\neg y\vee C_1'),(\neg x\vee y\vee C_2'),F_{rest}]$), where $F_{rest}=[C_3,\ldots,C_m]$, such that $pos(x,F)=neg(x,F)=1$ and $pos(y,F)=neg(y,F)=1$, then the formula $F'=[(x\vee y\vee C_1'),(\neg x\vee z\vee C_2'),(\neg y\vee\neg z\vee C_2'),F_{rest}]$ (or $F'=[(x\vee\neg y\vee C_1'),(\neg x\vee z\vee C_2'),(y\vee\neg z\vee C_2'),F_{rest}]$) has the same satisfiability with F, where z is a new variable.

From now on, for the sake of description, we assume that the formulas satisfy the following conditions: (for a formula F)

(1) For each $x\in var(F)$, $pos(x,F)>0$ and $neg(x,F)>0$, and

(2) For any $x,y\in var(F)$ ($x\neq y$), if $pos(x,F)=neg(x,F)=1$ and $pos(y,F)=neg(y,F)=1$ then the number of clauses containing x or y is at least three.

Lemma 4. Let $F=[(x\vee f_1),\ldots,(x\vee f_s),(\neg x\vee g_1),\ldots,(\neg x\vee g_t),F_{rest}]$ be a CNF formula with $pos(x,F)=s$ and $neg(x,F)=t$ and $occs(x,F)=s+t\geq 3$, where F_{rest} is the subformula of F. By introducing $(s+t)$ new variables $x_1,\ldots,x_{s+t}$, we define a formula

$$F^{[x]}:=[(x_1\vee f_1),\ldots,(x_s\vee f_s),(\neg x_{s+1}\vee g_1),\ldots,(\neg x_{s+t}\vee g_t),F_{rest}]+[(\neg x_1\vee x_2),(\neg x_2\vee x_3),\ldots,(\neg x_{s+t-1}\vee x_{s+t}),(\neg x_{s+t}\vee x_1)].$$

Then, we have that:

(1) F is satisfiable if and only if $F^{[x]}$ is satisfiable, and

(2) For any distinct clauses $C,C'\in F^{[x]}$, $|var(C)\cap var(C')\cap\{x_1,\ldots,x_{s+t}\}|\leq 1$.

Proof: Note that $var(F)\cap\{x_1,\ldots,x_{s+t}\}=\phi$ and $var(F^{[x]})=(var(F)-\{x\})\cup\{x_1,\ldots,x_{s+t}\}$.

(1) Assume that F is satisfied by a truth assignment τ over $var(F)$, then $F^{[x]}$ is satisfied by the truth assignment $\tau^{[x]}$ over $var(F^{[x]})=(var(F)-\{x\})\cup\{x_1,\ldots,x_{s+t}\}$, where $\tau^{[x]}(y)=\tau(y)$ if $y\in(var(F)-\{x\})$, and $\tau^{[x]}(y)=\tau(y)$ if $y\in\{x_1,\ldots,x_{s+t}\}$.

Conversely, we assume that $F^{[x]}$ is satisfied by a truth assignment τ' over $var(F^{[x]})$. It implies that τ' satisfies the subformula $[(\neg x_1\vee x_2),(\neg x_2\vee x_3),\ldots,(\neg x_{s+t-1}\vee x_{s+t}),(\neg x_{s+t}\vee x_1)]$ of $F^{[x]}$. The subformula $[(\neg x_1\vee x_2),(\neg x_2\vee x_3),\ldots,(\neg x_{s+t-1}\vee x_{s+t}),(\neg x_{s+t}\vee x_1)]$ represents a cycle of implication: $x_1\to x_2\to x_3\to\ldots\to x_{s+t}\to x_1$. Thus, $\tau'(x_1)=\ldots=\tau'(x_{s+t})$. Therefore, F is satisfied by a truth assignment τ'' over $var(F)$, where $\tau''(y)=\tau'(y)$ for $y\in(var(F)-\{x\})$, and $\tau''(x)=\tau'(x_1)$.

(2) It is clear that for any distinct clauses $C,C'\in F^{[x]}$, $|var(C)\cap var(C')\cap\{x_1,\ldots,x_{s+t}\}|\leq 1$, since the formula $[x_1,\ldots,x_{s+t},(\neg x_1\vee x_2),(\neg x_2\vee x_3),\ldots,(\neg x_{s+t-1}\vee x_{s+t}),(\neg x_{s+t}\vee x_1)]$ is linear when $s+t\geq 3$. □

The following example help readers to observe the resulting formula by replacing a variable with new variables in proof of Lemma 4.

Example 1. Let F be a formula. Its representation matrix is

$$\begin{matrix} x \\ y \\ z \end{matrix}\begin{pmatrix} + & + & & - & - \\ + & & - & - & - \\ & - & + & - & + \end{pmatrix}.$$

Then, the representation matrix of $F^{[x]}$ is

$$\begin{array}{c}x_1\\x_2\\x_3\\x_4\\y\\z\end{array}\begin{pmatrix}+ & & & & & - & & & +\\ & + & & & & + & - & & \\ & & & - & & & + & - & \\ & & & & - & & & + & -\\ + & & - & - & - & & & & \\ & - & + & - & + & & & & \end{pmatrix}.$$

By Lemma 4, we have the following algorithm for reducing a formula F to a linear formula F^{lin} in polynomial time of $|F|$.

Algorithm 1. Linear transformation for CNF formulas.

Input: A formula F with variables $x_1,\ldots,x_n$;

Output: A linear formulas F^{lin}.

begin

F^{lin}:=F; i:=1;

while $(i\le n)\wedge(occs(x_i,F^{lin})\ge 3)$ **do**

(let $F^{lin}=[(x_i\vee f_1),\ldots,(x_i\vee f_s),(\neg x_i\vee g_1),\ldots,(\neg x_i\vee g_t),F^{lin}_{rest}]$, $(s+t=(occs(x_i,F^{lin})))$.

Introducing new variables $y_{i,1},\ldots,y_{i,s+t}$;

$F^{lin}:=[(y_{i,1}\vee f_1),\ldots,(y_{i,s}\vee f_s),(\neg y_{i,s+1}\vee g_1),\ldots,(\neg y_{i,s+t}\vee g_t),F^{lin}_{rest}]+$

$[(\neg y_{i,1}\vee y_{i,2}),(\neg y_{i,2}\vee y_{i,3}),\ldots,(\neg y_{i,s+t-1}\vee y_{i,s+t}),(\neg y_{i,s+t}\vee y_{i,1})]$;

i:=i+1;

end_do;

output F^{lin};

end;

Algorithm 1 can be completed in times of $O(mn)$, and we have $|F^{lin}|=2n_2+3\sum_{n_2+1\le i\le n}occs(x_i,F))\le 3|F|$, where $n=|var(F)|$ and $m=\#cl(F)$, $n_2=|\{x\in var(F)|occs(x,F)=2|$.

Theorem 2. LSAT is NP-complete, where LSAT is the decision problem of satisfiability for restricted instances in LCNF.

Proof: Let F be a 3-CNF formula with variables $x_1,\ldots,x_n$ We assume that F satisfies the following conditions:

(1) For each $x\in var(F)$, $pos(x,F)>0$ and $neg(x,F)>0$, and

(2) For any $x,y\in var(F)$ $(x\neq y)$, if $pos(x,F)=neg(x,F)=1$ and $pos(y,F)=neg(y,F)=1$, then the number of clauses containing x or y is at least three.

W.l.o.g., let $var(F)=\{x_1,\ldots,x_n\}=\{x_1,\ldots,x_m\}\cup\{x_{m+1},\ldots,x_n\}$, where $0\le m\le n$, and $occs(x_i,F)=2$ for $1\le i\le m$, and $occs(x_j,F)\ge 3$ for $m+1\le j\le n$.

By the assumption, for any distinct clauses $C,C'\in F$, we have

$$|var(C)\cap var(C')\cap\{x_1,\ldots,x_m\}|\le 1 \qquad (*)$$

By Algorithm1, F can be transformed into F^{lin} in polynomial times of $|F|$, and only variables $x_{m+1},\ldots,x_n$ are replaced by new variables.

For any distinct clauses $f,g\in F^{lin}$, the followings are true:

(1) If both f and g come from the original clauses in F by replacing variables, then $|var(f)\cap var(g)\cap\{x_1,\ldots,x_m\}|\le 1$ by Eq.(*), and $var(f)\cap var(g)\cap(var(F^{lin})-\{x_1,\ldots,x_m\})=\phi$ by the proof of Lemma 4. It implies $|var(f)\cap var(g)|\le 1$.

(2) If either f or g comes from the original clause in F by replacing variables, and the other is a new additional clause in Algorithm 1, then $|var(f)\cap var(g)|=|var(f)\cap var(g)\cap(var(F^{lin})-\{x_1,\ldots,x_m\})|\le 1$ by the proof of Lemma 4.

(3) If neither f nor g comes from the original clauses in F by replacing variables, then $var(f)\cap var(g)\cap\{x_1,\ldots,x_m\}=\phi$ and $|var(f)\cap var(g)\cap(var(F^{lin})-\{x_1,\ldots,x_m\})|\le 1$ by the proof of Lemma 4.

Finally, $|var(f)\cap var(g)|\le 1$. Thus, F^{lin} is linear.

By Lemma 4, F is satisfiable if and only if F^{lin} is satisfiable.

F^{lin} can be computed from F in polynomial time of F. By NP-completeness of 3-SAT we have LSAT is NP-complete. □

Lemma 5. Let $F=[C_1,\ldots,C_m]$ be a linear formula and $G=[f_1,\ldots,f_n]$ a linear MU formula. We define a formula $F':=[(C_1\vee f_1'),C_2,\ldots,C_m,f_2,\ldots,f_n]$, where $var(F)\cap var(G)=\phi$ and f_1' is a nonempty subclause of f_1. Then, F' is a linear formula, and F is satisfiable if and only if F' is satisfiable.

Proof: It is clear that F' is linear, because of $var(F)\cap var(G)=\phi$ and linearity of F and G.

By renaming of literals in G, i.e., $\neg x$ is renamed to x, we can assume that f_1 contains only positive literals. Let $f_1=(y_1\vee\ldots\vee y_t)$, and $f_s'=(y_1\vee\ldots\vee y_s)$, where $1\le s\le t$.

Since G is minimal unsatisfiable, any truth assignment τ_G satisfying subformula $[f_2,\ldots,f_n]$ forces variables $y_1,\ldots,y_t$ to be false.

Assume that F is satisfiable, then there exists a truth assignment τ_1 satisfying F. Since G is minimal unsatisfiable, $[f_2,\ldots,f_n]$ is satisfiable, and then there exists a truth assignment τ_2 satisfying $[f_2,\ldots,f_n]$, and $\tau_2(y_1)=\ldots=\tau_2(y_t)=0$. We have a truth assignment τ over $var(F)\cup var(G)$ satisfying F', where $\tau(x)=\tau_1(x)$ for $x\in var(F)$, and $\tau(x)=\tau_2(x)$ for $x\in var(G)$.

Conversely, we assume that F' is satisfiable, then there exists a truth assignment τ satisfying F'. Thus, the restriction $\tau|_{var(G)}$ of τ over $var(G)$ satisfies $[f_2,\ldots,f_n]$, and $\tau|_{var(G)}(y_1)=\ldots=\tau|_{var(G)}(y_t)=0$. Similarly, the restriction $\tau|_{var(F)}$ of τ over $var(F)$ satisfies $[C_2,\ldots,C_m]$. Since $\tau(C_1\vee f_s')=1$ and $\tau|_{var(G)}(y_1)=\ldots=\tau|_{var(G)}(y_s)=0$, we have $\tau(C_1)=1$ It means that $\tau|_{var(F)}$ satisfies F. □

Lemma 5 represents a method lengthening clauses.

Lemma 6. For any fixed positive integer $k\ge 3$, k-SAT is NP-complete.

Proof: It is sufficient to show that 3-SAT can be reduced polynomially to k-SAT for $k>3$. Let $F=[C_1,\ldots,C_m]$ be a 3-CNF formula, and $l=k-3$. We define a k-CNF formula $F':=\wedge_{1\le i\le m}(C_i\vee_{cl}S_l^{(i)})$, where $S_l^{(i)}$ is a copy of the standard MU formula S_l (in Section 2) with new variables for $1\le i\le m$. Clearly, $|F'|=2^l|F|$, where 2^l is a constant for fixed k. Similar to the proof of Lemma 2, we can show that F is satisfiable if and only if F' is satisfiable. □

Theorem 3. For any fixed positive integer $k\ge 3$, k-LSAT is NP-complete, where k-LSAT is the decision problem of satisfiability for restricted instances in k-LCNF.

Proof: It is sufficient to show that k- SAT can be reduced polynomially to k-LSAT by Lemma 6.

Let $F=[C_1,\ldots,C_m]$ be a k-CNF. W.l.o.g., we assume $occs(x,F)\ge 3$ for each $x\in var(F)$. We now transform F into a formula F^* in k-LCNF by the following two stages.

Stage 1: Call Algorithm 1 (Linear Transformation for CNF formulas) to transform F into a linear formula F^{lin}. Note that for any clause $C\in F^{lin}$ $|C|=k$ or $|C|=2$.

Stage 2: Lengthen clauses of the length 2 in F^{lin}.

By Theorem 1, we can take a linear MU formula G in k-LCNF. Further, we can assume $G=[(y_1\vee\ldots\vee y_k),f_1,\ldots,f_l]$ where $|f_i|=k$ for $1\le i\le l$. Define $H:=[(y_3\vee\ldots\vee y_k),f_1,\ldots,f_l]$. The following algorithm generates a linear formula F^* in k-LCNF.

Algorithm 2. Lengthening clauses in linear formulas.

Input: The formula F^{lin};

Output: A linear formula F^* in k-LCNF.

begin

$F^*:=F^{lin}$;

while $((\exists C\in F^{fin})(|C|=2))$ **do**

taking a copy $[(y_3^c\vee\ldots\vee y_k^c), f_1^c,\ldots, f_l^c]$ of H with new variables;

$F^*:=(F^*-\{C\})+(C\vee y_3^c\vee\ldots\vee y_k^c)+[f_1^c,\ldots,f_l^c]$;

end_do;

output F^*;

end;

(For formulas F_1 and F_2, F_1+F_2 means $F_1\wedge F_2$).

The above stages can be completed in polynomial time of $|F|$, and we have $|F^*|=|F|\cdot|H|$.

By Lemma 4, F is satisfiable iff F^{lin} is satisfiable. By Lemma 5, F^{lin} is satisfiable iff F^* is satisfiable. Thus, k-SAT can be reduced polynomially to k-LSAT. □

5 Conclusions and Future Work

Based on the application of minimal unsatisfiable formulas and the induction, we present a simple and general method to construct some linear formulas minimal unsatisfiable in k-CNF for each $k\geq3$, which is stronger than the open problem whether or not there are unsatisfiable formulas in $LCNF_{\geq k}$[5,6]. Based on existences of minimal unsatisfiable formula in k-LCNF for $k\geq3$, we show that the decision problem k-LSAT is NP-complete for $k\geq3$. Additionally, we present two algorithms in the proof for transforming a k-CNF to a linear formula and lengthening clauses of linear formulas, respectively. The idea of algorithms is helpful for constructing other linear formulas. The future work is to investigate deeply structures and characterizations of linear formulas, and to apply linear formulas to analyzing complexity of resolutions and modifying effective algorithms for satisfiability.

References:

[1] Aharoni R. Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas. Journal of Combinatorial Theory, Series, 1986,A43:196−204.

[2] Davydov G, Davydova I, Kleine Büning H. An efficient algorithm for the minimal unsatisfiability problem for a subclass of CNF. Annals of Mathematics and Artificial Intelligence, 1998,23:229−245.

[3] Fleischner H, Kullmann O, Szeider S. Polynomial-Time recognition of minimal unsatisfiable formulas with fixed clause-variable difference. Theoretical Computer Science, 2002,289(1):503−516.

[4] Porschen S, Speckenmeyer E, Randerath B. On linear CNF formulas. In: Biere A, Gomes CP, eds. Proc. of the 19th Int'l Conf. on Theory and Applications of Satisfiability Testing-SAT 2006. LNCS 4121, New York: Springer-Verlag, 2006. 212−225.

[5] Porschen S, Speckenmeyer E. On linear CNF formulas and satisfiability. Tech. Report zaik, 2006.520, University Köln, 2006.

[6] Porschen S, Speckenmeyer E. NP-Completeness of SAT for restricted linear formulas classes. In: Proc. of the Guangzhou Symp. on Satisfiability in Logic-Based Modeling. 111−123.

[7] Wang J, Xu D. An effective algorithm for reducing k-CNF to t-CNF. Journal of Nanjing University Mathematical Biquartely, 2005, 22(1):53−65.

[8] Impagilazzo R, Paturi R. Complexity of k-SAT. In: Proc. of the 14th Annual IEEE Conf. on Computational Complexity. 1999. 237−240.

[9] Gong P, Xu D. A new lower bound of critical function for (k,s)-SAT. In: Cai J, Cooper SB, Li A, eds. Proc. of the 3rd Int'l Conf. on Theory and Applications of Models of Computation (TAMC 2006). LNCS 3959, Berlin, Heidelberg: Springer-Verlag, 2006. 274−282.

[10] Xu DY. Applications of minimal unsatisfiable formulas to polynomially reduction for formulas. Journal of Software, 2006,17(5): 1204−1212 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/17/1204.htm

附中文参考文献:

[10] 许道云.极小不可满足公式在多项式归约中的应用.软件学报,2006,17(5):1204–1212. http://www.jos.org.cn/1000-9825/17/1204.htm

XU Dao-Yun was born in 1959. He is a professor of the Department of Computer Science, Guizhou University and a CCF senior member. His research areas are complexity and computable analysis.

DENG Yian-Yan was born in 1964. He is a Ph.D. candidate at the Department of Computer Science, Guizhou University. His research areas are complexity and computable analysis.

ZHANG Qing-Shun was born in 1973. He is a Ph.D. candidate at the Department of Computer Science, Guizhou University. His research areas are complexity and computable analysis.

****