ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.7, July 2006, pp.1517−1526
DOI: 10.1360/jos171517
© 2006 by Journal of Software. All rights reserved.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

MAX(1)和 *MARG*(1)中公式改名的复杂性*

许道云[1+], 董改芳[2], 王 健[3]

[1](贵州大学 计算机科学系,贵州 贵阳 550025)
[2](内蒙古农业大学 计算机与信息工程学院,内蒙古 呼和浩特 010018)
[3](软件工程国家重点实验室(武汉大学),湖北 武汉 430072)

Complexities of Renaming for Formulas in *MAX*(1) and *MARG*(1)

XU Dao-Yun[1+], DONG Gai-Fang[2], WANG Jian[3]

[1](Department of Computer Science, Guizhou University, Guiyang 550025, China)
[2](College of Computer and Information Engineering, Inner Mongolia Agricultural University, Huhehaote 010018, China)
[3](State Key Laboratory of Software Engineering (Wuhan University), Wuhan 430072, China)
+ Corresponding author: Phn: +86-851-3627649, E-mail: dyxu@gzu.edu.cn

Xu DY, Dong GF, Wang J. Complexities of renaming for formulas in *MAX*(1) and *MARG*(1). *Journal of Software*, 2006,17(7):1517−1526. http://www.jos.org.cn/1000-9825/17/1517.htm

Abstract: A renaming is a function mapping propositional variable to itself or its complement, a variable renaming is a permutation over the set of propositional variables of a formula, and a literal renaming is a combination of a renaming and a variable renaming. Renaming for CNF formulas may help to improve DPLL algorithm. This paper investigates the complexity of decision problem: for propositional CNF formulas H and F, does there exist a variable (or literal) renaming φ such that $\varphi(H)=F$? Both *MAX*(1) and *MARG*(1) are subclasses of the minimal unsatisfiable formulas, and formulas in these subclasses can be represented by trees. The decision problem of isomorphism for trees is solvable in linear time. Formulas in the *MAX*(1) and *MARG*(1), it is shown that the literal renaming problems are solvable in linear time, and the variable renaming problems are solvable in quadratic time.
Key words: complexity; renaming; minimal unsatisfiable formula

摘 要: 改名是一个将变元映射到变元本身或它的补的函数,变元改名是公式变元集合上的一个置换,文字改名是一个改名和一个变元改名的组合.研究 CNF 公式的改名有助于改进 DPLL 算法.考虑判定问题“对于给定的 CNF 公式 H 和 F 是否存在一个变元(或文字)改名 φ,使得 $\varphi(H)=F$?”的计算复杂性.*MAX*(1)和 *MARG*(1)是极小不可满足公式的两个子类,这两个子类中的公式可以用树表示.树同构的判定问题在线性时间内是可解的.证明了对于 *MAX*(1)和 *MARG*(1)中的公式,文字改名问题在线性时间内可解,变元改名问题在平方次时间内可解.
关键词: 计算复杂性;改名;极小不可满足公式

---

* Supported by the National Natural Science Foundation of China under Grant Nos.60463001, 10410638 (国家自然科学基金); the Special Foundation for Improving Scientific Research Condition of Guizhou Province of China (贵州省高层次人才科研条件特助经费); the Government Foundation of Guizhou Province of China (贵州省省长基金)

Received 2004-02-12; Accepted 2005-07-08

© 中国科学院软件研究所 http://www.jos.org.cn

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, $C=(L_1\vee\ldots\vee L_m)$, and sometimes written as a set of literals, $C=\{L_1,\ldots,L_m\}$. A formula F in conjunction normal form (CNF) is a conjunctive of clauses, $F=(C_1\wedge\ldots\wedge C_n)$, and sometimes written as a set of clauses, $\{C_1,\ldots,C_n\}$ or a list of clauses, $F=[C_1,\ldots,C_n]$. $var(F)$ is the set of variables occurring in the formula F and $lit(F)$ is the set of literals over the variables of F. Let F be a CNF formula. A *renaming* of F is a function mapping propositional variable x to x or $\neg x$ for $x\in var(F)$, a *variable renaming* of F is a permutation over $var(F)$, and a *literal renaming* of F is a combination of a *renaming* and a *variable renaming* of F. For CNF formulas H and F, a homomorphism φ from formula H to F is a mapping from $lit(H)$ to $lit(F)$ and it preserves complements and clauses, i.e., $\varphi(\neg L)=\neg\varphi(L)$ for $L\in lit(H)$, and $\varphi(C)\in F$ for every clause $C\in H$, where $lit(\cdot)$ is the set of literals over variables occurring in the formula, and φ is an isomorphism from formula H to F if φ is a homomorphism from formula H to F and φ is a bijection. Clearly, if formula H is homomorphic to formula F, then the unsatisfiability of H implies the unsatisfiability of F, and if formula H is isomorphic to formula F, then H and F have the same satisfiability.

We are interested in isomorphism of CNF formulas for motivations of constructing some more efficient algorithms for satisfiability and simplifying the proofs of unsatisfiable formulas[1,2]. In Ref.[1], Krishnamurthy illustrated the power of symmetry for propositional proof systems. He added to the resolution calculus the rule of symmetry and gave short proofs for some hard formulas. For example, the pigeon hole formulas have a proof of polynomial size in this extended calculus. The rule of symmetry allows the following inference: If a clause f has been derived from a set of clauses F and φ is a permutation over the set of variables occurring in F, then the clause $\varphi(f)$ can be inferred as the next step in the derivation. Further interesting results can be found in Urquhart's paper[2]. We call a permutation of variables a variable renaming. Instead of a permutation of variables, we can make use of a more general renaming, namely a so called literal renaming or isomorphism. That means we have a permutation of variables and additionally variables can be simultaneously replaced by its complements. More formally, for formulas H and F with $var(H)=var(F)$, a *variable renaming* ϕ is a one–to–one mapping φ: $var(H)\rightarrow var(F)$ and a literal renaming φ is a one–to–one mapping φ: $lit(H)\rightarrow lit(F)$ with $\varphi(\neg x)=\neg\varphi(x)$ for any variable x. The literal renaming of CNF formulas is the isomorphism or symmetry of CNF formulas for satisfiability.

A deeper understanding of the structures of CNF formulas may help to improve the DPLL-algorithm. In the splitting tree of the DPLL-algorithm, if two formulas are labelled at the different nodes, and one of the formulas can be mapped to the other one by an isomorphism, then we can replace one of the formulas by the empty clause and continue with the remaining formula. By variable (or literal) renaming, we can decrease the size of the splitting tree in the DPLL-algorithms for some hard formulas. Formally, in the splitting tree of the DPLL-algorithm, if formula F_u at one node u can be mapped to formula F_v at the other one node v by a variable (or literal) renaming φ, then we can replace F_u by the empty clause, and continue with the remaining formula. We have shown that the DPLL-algorithm with such a symmetry rule has short proofs for the pigeon hole formulas with n+1 pigeons and n holes, which is a class of hard formulas, and it need only $O(n^3)$ nodes in the splitting tree[3].

A CNF formula F is *minimal unsatisfiable* (MU) if F is unsatisfiable and for any clause $f\in F$, $F-\{f\}$ is satisfiable. In Ref.[4], C. H. Papadimitriou and D. Wolfe showed that for every formula F one can construct a formula $f(F)$ in polynomial time such that F is satisfiable if and only if $f(F)$ is satisfiable, and F is unsatisfiable if and only if $f(F)$ is minimal unsatisfiable, i.e., an unsatisfiable formula can be transformed into a minimal

© 中国科学院软件研究所 http://www.jos.org.cn

unsatisfiable formula in polynomial time. The *deficiency* of CNF formula F is the difference between the number of clauses and the number of variables of F. It is well-known that the deficiency of MU formula is more than one[5,6]. For $k\geq 1$, let $MU(k)$ be the set of minimal unsatisfiable formulas with *deficiency* k. The decision problem for minimal unsatisfiable (MU) formulas is D^P-complete[4]. Fortunately, for fixed k, whether or not a formula belongs to $MU(k)$ can be decided in polynomial time[5]. It has been proved in Ref.[6] that for any $k,t\geq 1$ and any formula $F\in MU(t)$, there exists a formula H in $MU(k)$ and a homomorphism ϕ from H to F such that $\phi(H)=F$. Moreover, for fixed $k,t\geq 1$, the formula H and the homomorphism ϕ can be constructed in polynomial time. For a class C of CNF formulas we have considered the problems:

Problem: $Var-C(Lit-C,Hom-C)$

Instance: $H,F\in C$

Query: Does there exist a variable renaming (literal renaming, homomorphism) φ from H to F: $\varphi(H)=F$?

We call the problem $Var-C(Lit-C,Hom-C)$ the variable renaming (resp. literal renaming, homomorphism) for C.

We investigate the above mentioned problems for the class of minimal unsatisfiable formulas and various natural subclasses. The classes considered first are minimal unsatisfiable Horn formulas ($Hom-MU$) and MU formulas with fixed deficiency k. Additionally, for the homomorphism problem we consider the class $MU(k,t)$. The class $MU(k,t)$ is the set of pairs of formulas (H,F) where $H\in MU(k)$ and $F\in MU(t)$. Since a renaming preserves the number of clauses, these problems are not of interest for $MU(k,t)$. For fixed k and t, the problem $Hom-M(k,t)$ has an instance pair of formulas $H\in MU(k)$ and $F\in MU(t)$. The question is whether there is a homomorphism φ such that $\varphi(H)=F$.

The graph isomorphism problem consists in deciding whether two given graphs, $G_1=(V_1,E_1)$ and $G_2=(V_2,E_2)$, are isomorphic, i.e. whether there is a bijective mapping φ from V_1 to V_2 such that for any $u,v\in V_1$, $(u,v)\in E_1$ if , and only if $(\varphi(u),\varphi(v))\in E_2$.

We write $A\leq_p B$ if the class A is a polynomial one reducible to the class B. $A\equiv_p B$ is an abbreviation for $A\leq_p B$ and $B\leq_p A$. We use GI (resp. UGI) to denote the graph isomorphism problem for directed (resp. undirected) graphs. It is easy to prove $GI\equiv_p UGI$. Thus, we use GI to shortly denote the graph isomorphism problem for directed or undirected graphs. The graph isomorphism problem GI is known to be in NP. But it is an open problem whether GI is NP–complete or solvable in polynomial time[9].

In Ref.[10], we have proved the following results.

(1) For $k\geq 1$, the variable (and literal) renaming problems for formulas in $MU(k)$, even if Horn formulas in $MU(1)$, are equivalent to the graph isomorphism problem.

(2) For $k\geq 1$, the homomorphism problem for formulas in $MU(k)$, even if Horn formulas in $MU(1)$, is NP-complete.

(3) For k, $t\geq 1$, the homomorphism problem for $MU(k,t)$ is NP-complete.

In fact, the variable (or literal) renaming of CNF formulas describes some symmetry properties of formulas. By the symmetry properties of formulas, we can short the length of proof of satisfiability for formulas. However, we do not know exactly the complexity of the graph isomorphism problem. So, it is significant for investigating polynomial decidability of the variable (or literal) renamings for some subclasses of CNF.

In order to see whether the problems will be easier for more restrictive classes, we investigate maximal and marginal formulas. A MU formula F is *maximal* if adding a new literal to any clause of F results in a satisfiable formula. That is strongly minimal unsatisfiable formula defined in Ref.[5]. MAX is the set of maximal MU formulas and $MAX(k)=MU(k)\cap MAX$. A MU formula F is *marginal* if removing an occurrence of a literal from F results in a

© 中国科学院软件研究所 http://www.jos.org.cn

non-minimal unsatisfiable formula. *MARG* is the set of *marginal* formulas and $MARG(k)=MU(k)\cap MARG$. Intuitionally, a formula F in $MU(k)$ has a formula F_{low} in $MARG(k)$ as 'lower bound', and a formula F_{up} in $MAX(k)$ as 'upper bound' for literals. The decision problems for *MAX* and *MARG* are known to be D^P-complete[11,12], whereas the problems for fixed k are in P because the problem for $MU(k)$ is solvable in polynomial time.

In this paper, we will investigate variable renaming and literal renaming for formulas in $MAX(1)$ and $MARG(1)$. We consider the following problems:

Problem: *Var-MAX*(1) (*Var-MARG*(1))

Instance: $H,F\in MAX(1)$ ($MARG(1)$)

Query: Does there exist a variable renaming φ such that $\varphi(H)=F$?

Problem: *Lit-MAX*(1) (*Lit-MARG*(1))

Instance: $H,F\in MAX(1)$ ($MARG(1)$)

Query: Does there exist a literal renaming φ such that $\varphi(H)=F$?

We will prove that the problems *Var-MAX*(1) and *Var-MARG*(1) are solvable in quadratic time, and the problems *Lit-MAX*(1) and *Lit-MARG*(1) are solvable in linear time.

2 *MU*(1) Formulas

Let $F=[C_1,\ldots,C_n]$ be a CNF formula. The integer n, the number of clauses in the formula F, is denoted by $\#cl(F)$. $var(F)$ is the set of variables occurring in formula F and $\#var(F)$ is the number of variables of the formula F. $lit(F)$ is the set of literals occurring in formula F. The length (or size) of formula F is the number of occurrences of literals, i.e. $\sum_{C\in F}|lit(C)|$, denoted by $|F|$. A Horn clause is one with at most one positive literal. A Horn formula is a conjunction of Horn clauses. We denote the number of positive (resp. negative) occurrence of x in F by $pos(x,F)$ (resp. $neg(x,F)$), and write $occ(x,F)=(pos(x,F),neg(x,F))$.

Definition 1. (Representation matrix of a CNF formula)

Let $F=[C_1,\ldots,C_m]$ be a formula with n variables $x_1,\ldots,x_n$ in $CNF(n,m)$. The $n\times m$ matrix(a_{ij}) is called the representation matrix of F, where

$$a_{ij}=\begin{cases}+, & x_i\in C_j\\ -, & \neg x_i\in C_j\\ 0, & x_i,\neg x_i\notin C_j\end{cases}.$$

Sometimes we write *blank* for '0'.

Definition 2 (**variable renaming, renaming, literal renaming**).

Let H and F be formulas in CNF and $var(H)=var(F)$

(1) (Variable renaming) A mapping φ: $var(H)\rightarrow var(F)$ is termed a variable renaming from H to F, if φ is a permutation over $var(H)$ such that $\varphi(H)=F$.

(2) (Renaming) A mapping φ: $lit(H)\rightarrow lit(F)$ is termed a renaming if for all $L\in lit(H)$ we have $\varphi(L)=L$ or $\neg L$ and $\varphi(\neg L)=\neg\varphi(L)$. (We assume $\neg\neg L=L$).

(3) (Lit_renaming) A mapping φ: $lit(H)\rightarrow lit(F)$ is termed a literal renaming over $lit(H)$ if φ is a permutation over $lit(H)$ and for all $L\in lit(H)$ we have $\varphi(\neg L)=\neg\varphi(L)$.

Please note that a literal renaming is the combination of a variable renaming and a renaming.

Definition 3 (**minimal unsatisfiable formula**).

Let F be a CNF formula. F is called minimal unsatisfiable if

(1) F is unsatisfiable and

(2) for any clause $f\in F$, $F-\{f\}$ is satisfiable.

© 中国科学院软件研究所 http://www.jos.org.cn

For a formula $F\in CNF(n,n+k)$, the integer k is called the deficiency of F. For minimal unsatisfiable formulas, we always have $k\geq 1$[2,3]. We denote

$$MU(k)=\{F\in CNF(n,n+k)|F \text{ is minimal unsatisfiable}\}$$

and

$$MU=\{F|F \text{ is minimal unsatisfiable}\}=\bigcup_{k\geq 1} MU(k)\ .$$

In Ref.[6], it is well-known that for $F\in MU(1)$ there exists a variable x such that $occ(x,F)=(1,1)$, and the minimal unsatisfiable Horn formulas are in $MU(1)$.

The following theorem represents that $MU(1)$ is an important subclass of the minimal unsatisfiable formulas.

Theorem 1[12] (**splitting theorem**).

Suppose $F\in MU(k)$, $k>1$, and for every variable x, $occ(x,F)\geq(2,2)$. Let $F=[(x\vee f_1),\ldots,(x\vee f_s),B_x,C,B_{\neg x},(\neg x\vee g_1),\ldots,(\neg x\vee g_t)]$ where B_x, C, $B_{\neg x}$ are some formulas without occurrences of x and $\neg x$, such that

$$F_x=[f_1,\ldots,f_s,B_x,C]\in MU(k_x),\ F_{\neg x}=[g_1,\ldots,g_t,B_{\neg x},C]\in MU(k_{\neg x})$$

for some k_x and $k_{\neg x}$. Then we have $1\leq k_x,k_{\neg x}<k$.

The pair $(F_x,F_{\neg x})$ of formulas is called the *splitting pair* of F on variable x.

In Ref.[6], G. Davydov *et al*. introduced the complete representation of formulas in $MU(1)$, basic matrices.

Definition 4[6] (**basic matrix**).

The following matrix with n rows and $(n+1)$ columns defined inductively is termed a basic matrix:

(1) $(+-)$is a basic matrix.

(2) If B_1 is a basic matrix, then the following matrix is basic.

$$\begin{pmatrix} B_1 & 0 \\ b_1 & - \end{pmatrix}.$$

where b_1 is a vector with $(b_1)_j\in\{0,+\}$ and at least one +-sign.

(3) If B_2 is a basic matrix, then the following matrix is basic.

$$\begin{pmatrix} + & b_2 \\ 0 & B_2 \end{pmatrix}.$$

where b_2 is a vector with $(b_2)_j\in\{0,-\}$ and at least one −-sign.

(4) If both B_1 and B_2 are basic matrices, then the following matrix is basic.

$$\begin{pmatrix} B_1 & 0 \\ b_1 & b_2 \\ 0 & B_2 \end{pmatrix}.$$

where b_1 is a vector with $(b_1)_j\in\{0,+\}$ and at least one +-sign, and b_2 is a vector with $(b_2)_j\in\{0,-\}$ and at least one −-sign.

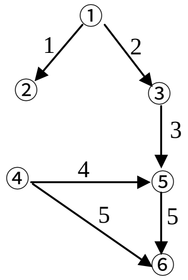
The basic matrix is a complete representation of formulas in $MU(1)$, which means that $F\in MU(1)$ if and only if the representation matrix of F is a basic matrix up to a permutation of rows and columns[6].

Definition 5 (**Representation graph of a formula in *MU*(1)**).

Let F be a formula with n variables in $MU(1)$ and $M=(m_{ij})_{n\times(n+1)}$ is the representation matrix of F. The directed label graph $G=(V,E,\lambda)$ is termed the representation graph of F, where $V=(1,2,\ldots,n,n+1)$, $E=\{(i,j)|m_{ki}=+$ and $m_{kj}=-$ for some $1\leq k\leq n, 1\leq i,j\leq(n+1)\}$ and $\lambda(i,j)=k$ if $m_{ki}=+$ and $m_{kj}=-$.

Example 1. The formula $F=[(x_1\vee x_2),\neg x_1,(\neg x_2\vee x_3),(x_4\vee x_5),(\neg x_3\vee\neg x_4\vee x_5),\neg x_5]$ is in $MU(1)$. The representation matrix M and the representation graph G of F are respectively.

$$\begin{array}{c} \\ 1\\2\\3\\4\\5 \end{array}\begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} + & - & & & & \\ + & 0 & - & & & \\ 0 & 0 & + & 0 & - & 0 \\ & & & + & - & \\ & & & + & + & - \end{pmatrix} \end{array}$$



3 Renaming Problems for Formulas in *MAX*(1)

We know that the isomorphism problem for trees is decidable in linear time[13]. We will show that formulas in *MAX*(1) and *MARG*(1) can be associated to trees in this section and next section.

In this section, we investigate the complexities of variable renaming problem and literal renaming problem for formulas in *MAX*(1). We prove that the variable and the literal renaming problems for formulas in *MAX*(1) are solvable in quadratic time, and the literal renaming problem for formulas in *MAX*(1) is solvable in linear time.

Lemma 1. Let F be a formula with n variables in *MAX*(1), then there is a unique variable x such that $pos(x,F)+neg(x,F)=n+1$.

Proof: Induction on n. It is clear for $n=1$. For $n>1$, let M be the basic matrix of F. Then, M is one of the following basic matrices:

$$\begin{pmatrix} B_1 & 0 \\ b_1 & - \end{pmatrix},\ \begin{pmatrix} + & b_2 \\ 0 & B_2 \end{pmatrix} \text{ and } \begin{pmatrix} B_1 & 0 \\ b_1 & b_2 \\ 0 & B_2 \end{pmatrix}.$$

where b_1 is a vector with $(b_1)_j=+$, and b_2 is a vector with $(b_2)_j=-$. From the structure of the above matrices, we see that only variable x corresponding to the row containing b_1 (or b_2) satisfies the condition: $pos(x,F)+neg(x,F)$ is equal to the number of columns in the matrix. By the induction hypothesis, we get a unique variable x for which $pos(x,F)+neg(x,F)=n+1$. □

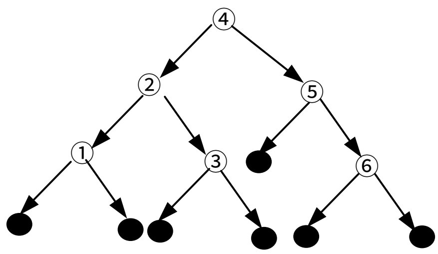
We call the variable x in Lemma 1 the *axis* variable of F. Note that the axis variable x is the unique variable occurring in every clause of F.

Let x be the axis variable of F. Then F is of the forms $[(x\vee f_1),\ldots,(x\vee f_p),(\neg x\vee g_1),\ldots,(\neg x\vee g_q)]$, where $p+q=\#var(F)+1$ and $F|_x=[f_1,\ldots,f_p]$ and $F|_{\neg x}=[g_1,\ldots,g_q]$, where $F|_x=F(x=0)$ and $F|_{\neg x}=F(x=1)$. Clearly, if $p,q>1$, then $(F|_x,F|_{\neg x})$ is the unique splitting pair of$F$on$x$, both $F|_x$ and $F|_{\neg x}$ are maximal, and $var(F|_x)\cap var(F|_{\neg x})=\varnothing$. If $p=1$ and $q>1$, then $F|_{\neg x}$ is maximal. If $p>1$ and $q=1$, then $F|_x$ is maximal.

Based on Lemma 1, a formula F in *MAX*(1) can be associated only to a binary tree T_F.

Example 2. The formula $F=[(x_1\vee x_2\vee x_4),(\neg x_1\vee x_2\vee x_4),(\neg x_2\vee x_3\vee x_4),(\neg x_2\vee\neg x_3\vee x_4),(\neg x_4\vee x_5),(\neg x_4\vee\neg x_5\vee x_6),(\neg x_4\vee\neg x_5\vee\neg x_6)]$ is in *MAX*(1). The representation matrix M_F and the binary tree T_F associated to F are respectively.

$$\begin{array}{c} 1\\2\\3\\4\\5\\6 \end{array}\begin{pmatrix} + & - & & & & & \\ + & + & - & - & & & \\ & & + & - & & & \\ + & + & + & + & - & - & - \\ & & & & + & - & - \\ & & & & & + & - \end{pmatrix}$$



In the binary tree T_F, labels at internal nodes correspond to variables in F, and the order of axis variables during the recursive splitting of the formula corresponds to the order searching internal nodes of T_F in middle root search. A leave of T_F corresponds to a clause of F, and the path from the root to the leave is associated to this clause. The edge from internal node to its left child is associated to a positive occurrence of the variable corresponding to the internal node, and the edge from internal node to its right child is associated to a negative occurrence of the variable corresponding to the internal node. For example, let 3_r be the right child of node 3, then the path from the root to 3_r corresponds to the clause: $(x_4\vee\neg x_2\vee\neg x_3)$.

Theorem 2. The problem *Var-MAX*(1) is solvable in quadratic time.

Proof: Let F and H be two formulas in *MAX*(1). If $\#var(F)\neq\#var(H)$, then F cannot be renamed into H. If F can be renamed into H, then the axial variable x_f of F must be mapped to the axial variable x_h of H, and $pos(x_f,F)=pos(x_h,H)$. Finally, we split F and H on axial variables respectively and apply the induction to the splitted formulas.

We now consider the following algorithm.

Algorithm 1. (var_renaming for *MAX*(1))

Input: Formula F and H in *MAX*(1).

Output: Yes or No.

procedure *Var_ren*(*F*,*H*);

begin

  $n_f:=\#var(F)$; $n_h:=\#var(H)$;

  if $n_f\neq n_h$ **then** return No;

  if (n_f==1) **then** return Yes;

  x_f:= the axial variable of F;

  x_h:= the axial variable of H;

  $pos_f:=pos(x_f,F)$; $neg_f:=neg(x_f,F)$;

  $pos_h:=pos(x_h,H)$; $neg_h:=neg(x_h,H)$;

  if $pos_f\neq pos_h$ **then** return No;

  if pos_f=1 **then** call $Var\_ren(F|_{\neg x_f}, H|_{\neg x_h})$;

  if neg_f=1 **then** call $Var\_ren(F|_{x_f}, H|_{x_h})$;

  if $(Var\_ren(F|_{x_f}, H|_{x_h})=\text{Yes})\ \&\ (Var\_ren(F|_{\neg x_f}, H|_{\neg x_h})=\text{Yes})$

    then return Yes;

  return No;

end;

Let F and H be formulas with n variables in *MAX*(1), and let x and y be the axial variables of F and H, respectively. It is easy to prove that: There exists a var_renaming φ with $\varphi(F)=H$ if and only if $\varphi(x)=y$, $pos(x,F)=pos(y,H)$ and there are two variable renamings, φ_+ and φ_-, with $\varphi_+(F|_x)=H|_y$ and $\varphi_-(F|_{\neg x})=H|_{\neg y}$. Therefore, there exists a variable renaming φ with $\varphi(F_1)=F_2$ if and only if Algorithm 1 returns *Yes*. Please note that we can compute the axial variable x of F and the pair $(F|_x,F|_{\neg x})$ in $O(n)$ time. The number of recursive calls is $O(n)$. Thus, the complexity of Algorithm 1 is $O(n^2)$, where $n=\#var(F)$. Therefore, the problem *MAX*(1)-*VR* is solvable in quadratic time. □

Note in variable renaming that we must consider the difference of positive and negative literals, which corresponds to the difference of the left and right children of internal node. This is why we do not apply directly the method of tree isomorphism.

Theorem 3. The problem *Lit-MAX*(1) is solvable in linear time.

Proof: Based on Lemma 1, we can associate a $MAX(1)$ formula F with n variables to a binary tree T_F with n internal nodes and n+1 leaves, and each internal node has exactly two children. Note in literal renaming that the sign of literals can be ignored. Let F and H be formulas with $var(F)=var(H)$ in $MAX(1)$. We associate F and H to binary trees T_F and T_H, respectively. Thus, there exists a literal renaming ϕ from H to F if and only if T_H is isomorphic to T_F. We know that the isomorphism problem for binary trees is solvable in linear time. Therefore, the problem *Lit*-$MAX(1)$ is solvable in linear time. □

Corollary 1. The homomorphism problem for formulas in $MAX(1)$ is solvable linear time.

Proof: Let F be a formula in $MAX(1)$. By the induction on $n=\#var(F)$ and the basic matrix, it can easily be proved that: For any different clauses f and g, there exists exactly one pair of complementary literals, L and $\neg L$, such that $L\in f$ and $\neg L\in g$. Thus, any homomorphism for a formula in $MAX(1)$ must be a literal renaming. □

4 Renaming Problems for Formulas in *MARG*(1)

In this section, we investigate the complexities of variable renaming problem and literal renaming problem for formulas in $MARG(1)$. We prove that the variable renaming problem for formulas in $MARG(1)$ is solvable in quadratic time, and the literal renaming problem for formulas in $MARG(1)$ is solvable in linear time.

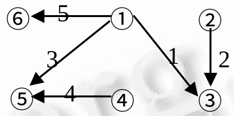
Based on the characterization of basic matrix of formulas in $MARG(1)$, it is easy to prove the following lemma.

Lemma 2. Let F be a formula in $MARG(1)$. Then, for every variable $x\in\#var(F)$ we have $occ(x,F)=(1,1)$.

By Lemma 2, we can associate a formula F to a directed graph G_F with labels, which is the representation graph of F. Based on the basic matrix of F and the induction, we can show that G_F has no cycle and the resulting undirected graph by deleting the directions of edges in G_F is a tree.

Example 3. The formula $F=[(x_1\vee x_3\vee x_5),x_2,(\neg x_1\vee\neg x_2),x_4,(\neg x_3\vee\neg x_4),\neg x_6]$ is in $MARG(1)$. The representation matrix M and the representation graph G of F are respectively

$$\begin{array}{c}\begin{array}{cccccc}1&2&3&4&5&6\end{array}\\ \begin{array}{c}1\\2\\3\\4\\5\end{array}\begin{pmatrix}+& &-& & & \\ &+&-& & & \\ +& & & &-& \\ & & &+&-& \\ +& & & & &-\end{pmatrix}\end{array}$$



Theorem 4. The problem *Lit*-$MARG(1)$ is decidable in linear time.

Proof: Let $F=[C_1,\ldots,C_{n+1}]$ be a formula with variables $x_1,\ldots,x_n$ in $MARG(1)$. By Lemma 2, every variable in F occurs exactly once positively and once negatively. Then, the representation graph G_F of F contains exactly n edges, and the different edge has different labels. Based on the basic matrix of F and the induction, we can show that G_F has no cycle and the resulting undirected graph by deleting the directions of edges in G_F is a tree. So, we can introduce a new node at each edge to replace the label on the edge. Formally, we define an undirected graph $T_F=(V_F, E_F)$, where $V_F=\{x_1,\ldots,x_n,c_1,\ldots,c_{n+1}\}$ and $E_F=\{(c_i,x_k),(x_k,c_j)|x_k\in C_i,\neg x_k\in C_j,1\le k\le n,1\le i,j\le n+1\}$.

Thus, T_F is a tree, and we have the fact: $deg(x_k)=2$ for every $1\le k\le n$. It shows that every vertex x_k is an internal node of T_F.

Let H and F be formulas with $var(H)=var(F)$ in $MARG(1)$, and let T_H and T_F be the associated trees, respectively. By the structures of T_H and T_F, we have that there exists a lit_renaming φ with $\varphi(H)=F$ if and only if T_H is isomorphic to T_F. Note that both T_H and T_F contain $2n$+1 nodes. Therefore, the problem *Lit*-$MARG(1)$ is decidable in $O(n)$ time, since the tree isomorphism problem is solvable in linear time. □

Theorem 5. The problem *Var*-$MARG(1)$ is decidable in quadratic time.

© 中国科学院软件研究所 http://www.jos.org.cn

Proof: Let $F=[C_1,\ldots,C_{n+1}]$ be a formula with variables $x_1,\ldots x_n$ in *MARG*(1). Similar to the proof for the problem *Lit-MARG*(1), we now associate F to an undirected graph $T=(V,E)$ in $O(n^2)$ time as follows:

(1) We define $V=V_{var}\cup V_{var}^{+}\cup V_{var}^{-}\cup V_{cl}\cup V_{var+}^{*}\cup V_{var-}^{*}\cup V_{cl}^{*}$, where $V_{var}=\{x_1,\ldots,x_n\}$, $V_{cl}=\{c_1,\ldots,c_{n+1}\}$, $V_{var}^{+}=\{x_1^+,\ldots,x_n^+\}$, $V_{var}^{-}=\{x_1^-,\ldots,x_n^-\}$, $V_{var+}^{*}=\{y_1^1,\ldots,y_1^{n+2},\ldots,y_n^1,\ldots,y_n^{n+2}\}$, $V_{var-}^{*}=\{z_1^1,\ldots,z_1^{n+1},\ldots,z_n^1,\ldots,z_n^{n+1}\}$, and $V_{cl}^{*}=\{c_1^1,c_1^2,\ldots,c_{n+1}^1,c_{n+1}^2\}$, we have $|V|=2n^2+9n+3$.

(2) We define $E=E_0\cup E_1^+\cup E_1^-\cup E_2\cup E_3$, where $E_0=\{(c_i,x_k^+),(x_k^-,c_j)\mid x_k\in C_i,\neg x_k\in C_j,\ 1\le k\le n,\ 1\le i,j\le n+1\}$, $E_1^+=\{(x_k^+,y_k^i)\mid 1\le k\le n, 1\le i\le n+2\}$, $E_1^-=\{(x_k^-,z_k^i)\mid 1\le k\le n, 1\le i\le n+1\}$, $E_2=\{(c_k,c_k^1),(c_k,c_k^2)\mid 1\le k\le n+1\}$ and $E_3=\{(x_k^+,x_k),(x_k,x_k^-)\mid 1\le k\le n\}$.

Based on the proof of Theorem 4 and the construction of T, T is a tree and we have

(a) $deg(x_k^+)=n+4$ for every $1\le k\le n$;

(b) $deg(x_k^-)=n+3$ for every $1\le k\le n$;

(c) $deg(x_k)=2$ for every $1\le k\le n$;

(d) $3\le deg(c_k)\le n+2$ for every $1\le k\le n+1$;

(e) $deg(v)=1$ for every $v\in V_{var+}^{*}\cup V_{var-}^{*}\cup V_{cl}^{*}$.

Our idea is to identify positive literals and negative literals, and to distinguish nodes corresponding to the variables and nodes corresponding to clauses by different degrees of vertices.

Let $H=[C_1,\ldots,C_{n+1}]$ and $F=[C_1',\ldots,C_{n+1}']$ be formulas over variables $x_1,\ldots x_n$ in *MARG*(1), and let T_H and T_F the associated trees. By the structures of T_H and T_F, we will show that there exists a variable renaming φ with $\varphi(H)=F$ if and only if T_H is isomorphic to T_F.

($\Rightarrow$) Suppose that there exists a variable renaming φ with $\varphi(H)=F$. We have a permutation π_v over $\{1,\ldots,n\}$ and a permutation π_c over $\{1,\ldots,n+1\}$ such that $\varphi(x_k)=x_{\pi_v(k)}$ for $1\le k\le n$ and $\varphi(C_i)=C'_{\pi_c(i)}$ for $1\le i\le n+1$.

By the construction of T_H, we have that for any variable x_k, $x_k\in C_i$ and $\neg x_k\in C_j$ if, and only if $(c_i,x_k^+),(x_k^-,c_j),(x_k^+,x_k),(x_k,x_k^-)$ are edges in T_H.

Now we define an isomorphism ϕ with $\phi(T_H)=T_F$ as follows:

(1) $\phi(x_k)=x_{\pi_v(k)}$, $\phi(x_k^+)=x_{\pi_v(k)}^+$, $\phi(x_k^-)=x_{\pi_v(k)}^-$ $(1\le k\le n)$;

(2) $\phi(c_i)=c'_{\pi_c(i)}$ $1\le i\le n+1$;

(3) $\phi(y_k^p)=y_{\pi_v(k)}^p$ for $(1\le k\le n)$ and $(1\le p\le n+2)$;

(4) $\phi(z_k^p)=z_{\pi_v(k)}^p$ for $(1\le k\le n)$ and $(1\le p\le n+1)$;

(5) $\phi(c_i^p)=c_{\pi_c(i)}^p$ $1\le i\le n+1$ and $p=1,2$.

($\Leftarrow$) Let ϕ be an isomorphism with $\phi(T_H)=T_F$. By the difference of degrees of nodes, we have that $\phi(V_{var})=V_{var}$, $\phi(V_{var}^+)=V_{var}^+$, $\phi(V_{var}^-)=V_{var}^-$, and $\phi(V_{cl})=V_{cl}$. The restriction $\phi|_{V_{var}}$ is the desired variable renaming, since $x_k\in C_i$ and $\neg x_k\in C_j$ if, and only if there is a unique path, $c_i x^+ x_k x_k^- c_j$, from c_i to c_j through x_k. Please note that T_H contains $3n^2+9n+3$ nodes. Thus, the problem *Var-MARG*(1) is decidable in $O(n^2)$ time, since the isomorphism problem for trees is solvable in linear time. □

5 Conclusions

The variable (or literal) renaming of formulas is helpful for improving proof system and DPLL algorithm. From Refs.[3,10], we know that the variable renaming and literal renaming problems for formulas in *MU*(1) are related closely to the graph isomorphism problem. So, it is significant for investigating solvable variable and literal

© 中国科学院软件研究所 http://www.jos.org.cn

renaming problems in polynomial time. In this paper, we investigate variable and literal renaming problems for two subclasses, *MAX*(1) and *MARG*(1), of minimal unsatisfiable formulas. We have proved that the literal renaming problems for formulas in *MAX*(1) and *MARG*(1) are solvable in linear time, and the variable renaming problem for formulas in *MAX*(1) and *MARG*(1) are solvable in quadratic time.

For $k\geq 2$, it is still open whether the variable and literal renaming problems for formulas in *MAX*(k) and *MARG*(k) are solvable in polynomial time.

References:

[1] Krishnamurthy B. Short proofs for tricky formulas. Acta Informatica, 1985,22(3):253–275.

[2] Urquhart A. The symmetry rule in propositional logic. Discrete Applied Mathematics, 1999,96-97(1):177–193.

[3] Xu DY. On the complexity of renamings and homomorphisms for minimal unsatisfiable formulas [Ph.D. Thesis]. Nanjing: Nanjing University, 2002.

[4] Papadimitriou CH, Wolfe D. The complexity of facets resolved. Journal of Computer and System Sciences, 1988,37(1):2–13.

[5] Aharoni R. Minimal non-two-colorable hypergraphs and minimal unsatisfiable formulas. Journal of Combinatorial Theory, 1996,43(A):196–204.

[6] 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(3-4):229–245.

[7] 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.

[8] Kleine Büning H, Zhao XS. Polynomial time algorithms for computing a representation for minimal unsatisfiable formulas with fixed deficiency. Information Processing Letters, 2002,84(3):147–151.

[9] Köbler J, Schöning U, Toran J. The Graph Isomorphism Problem: Its Structural Complexity. Birkhäuser Verlag, 1993.

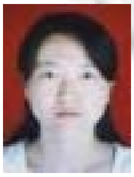
[10] Kleine Büning H, Xu DY. The complexity of homomorphisms and renamings for minimal unsatisfiable formulas. Annals of Mathematics and Artificial Intelligence, 2005,43(1-4):113–127.

[11] Kleine Büning H, Zhao XS. The complexity of some subclasses of minimal unsatisfiable formulas. Discrete Applied Mathematics, 2005,130(2):185–207.

[12] Kleine Büning H. On subclasses of minimal unsatisfiable formulas. Discrete Applied Mathematics, 2000,107(1-3):83–98.

[13] Aho AV, Hopcropt JE, Ullman JD. The Design and Analysis of Computer Algorithms. Addison-Wesley Publishing Company, 1976. 84–86.

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



DONG Gai-Fang was born in 1979. She is a lecturer at the Inner Mongolia Agricultural University. Her current research area is complexity.

WANG Jian was born in 1980. He is a Ph.D. candidate at the Wuhan University. His current research areas are complexity and software engineering.

© 中国科学院软件研究所 http://www.jos.org.cn