\( \text{MAX}(1) \quad \text{MARG}(1) \)\*  

\begin{align*}
&1^\text{(1+)} & 2 & 3 \\
&1 & 2 & 3 & 550025 \\
&0 & 1 & 0 & 010018 \\
&0 & 1 & 0 & 430072
\end{align*}

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

XU Dao-Yun\textsuperscript{1+}, DONG Gai-Fang\textsuperscript{2}, WANG Jian\textsuperscript{3}

\textsuperscript{1}(Department of Computer Science, Guizhou University, Guiyang 550025, China)  
\textsuperscript{2}(College of Computer and Information Engineering, Inner Mongolia Agricultural University, Huhehaote 010018, China)  
\textsuperscript{3}(State Key Laboratory of Software Engineering (Wuhan University), Wuhan 430072, China)

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

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 \( \varphi \) such that \( \varphi(H) = F \)? Both \( \text{MAX}(1) \) and \( \text{MARG}(1) \) 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 \( \text{MAX}(1) \) and \( \text{MARG}(1) \), 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

[Abstract translated from Chinese]

\* Supported by the National Natural Science Foundation of China under Grant Nos.60463001, 10410638 (\( \text{MAX}(1) \)) and the Special Foundation for Improving Scientific Research Condition of Guizhou Province of China (\( \text{MARG}(1) \)).

Received 2004-02-12; Accepted 2005-07-08
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 conjunction of clauses, $F=\{C_1 \land \ldots \land C_n\}$, and sometimes written as a set of clauses, $\{C_1, \ldots, C_n\}$ or a list of clauses, $F=[C_1, \ldots, C_n]$. $\text{var}(F)$ is the set of variables occurring in the formula $F$ and $\text{lit}(F)$ 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 $x \in \text{var}(F)$, a variable renaming of $F$ is a permutation over $\text{var}(F)$, 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 $\varphi$ from formula $H$ to $F$ is a mapping from $\text{lit}(H)$ to $\text{lit}(F)$ and it preserves complements and clauses, i.e., $\varphi(\neg L)=\neg \varphi(L)$ for $L \in \text{lit}(H)$, and $\varphi(C) \in F$ for every clause $C \in H$, where $\text{lit}(C)$ is the set of literals over variables occurring in the formula, and $\varphi$ is an isomorphism from formula $H$ to $F$ if $\varphi$ is a homomorphism from formula $H$ to $F$ and $\varphi$ 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 $\varphi$ 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 their complements. More formally, for formulas $H$ and $F$ with $\text{var}(H)=\text{var}(F)$, a variable renaming $\varphi$ is a one–to–one mapping $\varphi : \text{var}(H) \rightarrow \text{var}(F)$ and a literal renaming $\varphi$ is a one–to–one mapping $\varphi : \text{lit}(H) \rightarrow \text{lit}(F)$ 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 $\varphi$, 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 $f \in F$, $F\setminus \{f\}$ 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
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 \( \text{MU}(k) \) 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 \( \text{MU}(k) \) can be decided in polynomial time\(^{[5]}\). It has been proved in Ref.\([6]\) that for any \( k,t \geq 1 \) and any formula \( F \in \text{MU}(t) \), there exists a formula \( H \) in \( \text{MU}(k) \) and a homomorphism \( \phi \) from \( H \) to \( F \) such that \( \phi(H)=F \). Moreover, for fixed \( k,t \geq 1 \), the formula \( H \) and the homomorphism \( \phi \) can be constructed in polynomial time. For a class \( C \) of CNF formulas we have considered the problems:

**Problem:** \( \text{Var}−C(\text{Lit}−C,\text{Hom}−C) \)

**Instance:** \( H,F \in C \)

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

We call the problem \( \text{Var}−C(\text{Lit}−C,\text{Hom}−C) \) 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 (\( \text{Hom}−\text{MU} \)) and MU formulas with fixed deficiency \( k \). Additionally, for the homomorphism problem we consider the class \( \text{MU}(k,t) \). The class \( \text{MU}(k,t) \) is the set of pairs of formulas \((H,F)\) where \( H \in \text{MU}(k) \) and \( F \in \text{MU}(t) \). Since a renaming preserves the number of clauses, these problems are not of interest for \( \text{MU}(k,t) \). For fixed \( k \) and \( t \), the problem \( \text{Hom}−\text{M}(k,t) \) has an instance pair of formulas \( H \in \text{MU}(k) \) and \( F \in \text{MU}(t) \). The question is whether there is a homomorphism \( \phi \) such that \( \phi(H)=F \).

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 \( \phi \) from \( V_1 \) to \( V_2 \) such that for any \( u,v \in V_1 \), \( (u,v) \in E_1 \) if, and only if, \( (\phi(u),\phi(v)) \in E_2 \).

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 \text{GI} (resp. \text{UGI}) to denote the graph isomorphism problem for directed (resp. undirected) graphs. It is easy to prove \( \text{GI} \equiv_p \text{UGI} \). Thus, we use \text{GI} to shortly denote the graph isomorphism problem for directed or undirected graphs. The graph isomorphism problem \( \text{GI} \) is known to be in \( \text{NP} \). But it is an open problem whether \( \text{GI} \) is \( \text{NP} \)-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 \( \text{MU}(k) \), even if Horn formulas in \( \text{MU}(1) \), are equivalent to the graph isomorphism problem.
2. For \( k \geq 1 \), the homomorphism problem for formulas in \( \text{MU}(k) \), even if Horn formulas in \( \text{MU}(1) \), is \( \text{NP} \)-complete.
3. For \( k,t \geq 1 \), the homomorphism problem for \( \text{MU}(k,t) \) is \( \text{NP} \)-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.\([5]\). \( \text{MAX} \) is the set of maximal MU formulas and \( \text{MAX}(k)=\text{MU}(k)\cap \text{MAX} \). A MU formula \( F \) is *marginal* if removing an occurrence of a literal from \( F \) results in a
non-minimal unsatisfiable formula. \textit{MARG} is the set of \textit{marginal} formulas and \textit{MARG}(k)=\textit{MU}(k)\cap\textit{MARG}. Intuitively, a formula \(F\) in \textit{MU}(k) has a formula \(F_{\text{low}}\) in \textit{MARG}(k) as ‘lower bound’, and a formula \(F_{\text{up}}\) in \textit{MAX}(k) as ‘upper bound’ for literals. The decision problems for \textit{MAX} and \textit{MARG} are known to be \(\mathcal{NP}\)-complete\cite{11,12}, whereas the problems for fixed \(k\) are in \(\mathcal{P}\) because the problem for \textit{MU}(k) is solvable in polynomial time.

In this paper, we will investigate variable renaming and literal renaming for formulas in \textit{MAX}(1) and \textit{MARG}(1).

We consider the following problems:

\textbf{Problem:} \textit{Var-MAX}(1) (\textit{Var-MARG}(1))

\textbf{Instance:} \(H.F\in\text{MAX}(1) (\text{MARG}(1))\)

\textbf{Query:} Does there exist a variable renaming \(\varphi\) such that \(\varphi(H)=F?\)

\textbf{Problem:} \textit{Lit-MAX}(1) (\textit{Lit-MARG}(1))

\textbf{Instance:} \(H.F\in\text{MAX}(1) (\text{MARG}(1))\)

\textbf{Query:} Does there exist a literal renaming \(\varphi\) such that \(\varphi(H)=F?\)

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

\section{\textit{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 \#\text{cl}(F). \text{var}(F) is the set of variables occurring in formula \(F\) and \#\text{var}(F) is the number of variables of the formula \(F\). \text{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. \(\sum_{C\in\text{lit}(F)}|\text{lit}(C)|\), 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 \text{pos}(x,F) (resp. \text{neg}(x,F)), and write \text{occ}(x,F)=(\text{pos}(x,F),\text{neg}(x,F))\).

\textbf{Definition 1.} (Representation matrix of a CNF formula)

Let \(F=[C_1,\ldots,C_n]\) 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 \textit{blank} for ‘0’.

\textbf{Definition 2} (variable renaming, renaming, literal renaming).

Let \(H\) and \(F\) be formulas in CNF and \text{var}(H)=\text{var}(F)

(1) (Variable renaming) A mapping \(\varphi\): \text{var}(H)\rightarrow \text{var}(F)\) is termed a variable renaming from \(H\) to \(F\), if \(\varphi\) is a permutation over \text{var}(H) such that \(\varphi(H)=F\).\n
(2) (Renaming) A mapping \(\varphi\): \text{lit}(H)\rightarrow \text{lit}(F)\) is termed a renaming if for all \(L\in\text{lit}(H)\) we have \(\varphi(L)=L\) or \(\neg L\) and \(\varphi(-L)=-\varphi(L).\) (We assume \(\neg\neg L=L\).)

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

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

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

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

and

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

In Ref. [6], it is well-known that for $F \in MU(1)$ there exists a variable $x$ such that $\text{occ}(x,F) = (1,1)$, 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** (splitting theorem).

Suppose $F \in MU(k)$, $k > 1$, and for every variable $x$, $\text{occ}(x,F) \geq (2,2)$. Let $F = ((x \lor f_1), \ldots, (x \lor f_j), B_0, C, B_m, (\neg x \lor g_1), \ldots, (\neg x \lor g_t))$ where $B_0, C, B_m$ are some formulas without occurrences of $x$ and $\neg x$, such that $F_x = [f_1, \ldots, f_j, B_0, C] \in MU(k_1)$, $F_{\neg x} = [g_1, \ldots, g_t, B_m, C] \in MU(k_{m+1})$ for some $k_1$ and $k_m$. Then we have $1 \leq k_1, k_m \leq k$.

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

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 $V = \{1, 2, \ldots, n, n+1\}$, $E = [(i,j) \mid m_{ij} = +$ and $m_{ji} = -$ for some $1 \leq k \leq n, 1 \leq i, j \leq (n+1)\}$ and $\lambda(i,j) = k$ if $m_{ij} = +$ and $m_{ji} = -$.

**Example 1.** The formula $F = [(x_1 \lor x_2), (\neg x_3), (\neg x_2 \lor x_3), (x_2 \lor x_3), (\neg x_3 \lor x_4), (\neg x_4 \lor x_3), (\neg x_3), (\neg x_4)]$ is in $MU(1)$. The representation matrix $M$ and the representation graph $G$ of $F$ are respectively.
3 Renaming Problems for Formulas in \( \text{MAX}(1) \)

We know that the isomorphism problem for trees is decidable in linear time\(^{[13]} \). We will show that formulas in \( \text{MAX}(1) \) and \( \text{MARG}(1) \) 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 \( \text{MAX}(1) \). We prove that the variable and the literal renaming problems for formulas in \( \text{MAX}(1) \) are solvable in quadratic time, and the literal renaming problem for formulas in \( \text{MAX}(1) \) is solvable in linear time.

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

**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:

\[
\begin{pmatrix}
B_1 & 0 \\
0 & 0 & B_2
\end{pmatrix}
\quad \text{and} \quad
\begin{pmatrix}
B_1 & 0 \\
0 & 0 & B_2
\end{pmatrix}
\]

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: \( \text{pos}(x,F)+\text{neg}(x,F) \) is equal to the number of columns in the matrix. By the induction hypothesis, we get a unique variable \( x \) for which \( \text{pos}(x,F)+\text{neg}(x,F)=n+1 \).

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 \( p+q=\#\text{var}(F)+1 \) and \( F_{[a]}=[f_1,\ldots,f_p] \) and \( F_{[a]}=[g_1,\ldots,g_q] \), where \( F_{[a]}=F(x=0) \) and \( F_{[a]}=F(x=1) \). Clearly, if \( p,q>1 \), then \( (F_{[a]},F_{[a]}) \) is the unique splitting pair of \( \text{Fonx} \), both \( F_{[a]} \) and \( F_{[a]} \) are maximal, and \( \text{var}(F_{[a]})\cap\text{var}(F_{[a]})=\emptyset \). If \( p=1 \) and \( q>1 \), then \( F_{[a]} \) is maximal. If \( p>1 \) and \( q=1 \), then \( F_{[a]} \) is maximal.

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

**Example 2.** The formula \( F=\neg(x_1\vee x_2\vee x_3),(-x_1\vee x_2\vee x_4),(-x_2\vee -x_3\vee x_4),(-x_3\vee -x_1\vee x_4),(-x_4\vee -x_3\vee x_5),(-x_4\vee -x_3\vee x_6),(-x_4\vee -x_5\vee -x_6) \) is in \( \text{MAX}(1) \). The representation matrix \( M_F \) and the binary tree \( T_F \) associated to \( F \) are respectively.
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 $3$ be the right child of node $3$, then the path from the root to $3$, corresponds to the clause: $(x_4 \lor \neg x_2 \lor \neg x_3)$.

**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;

\[ x_f := \text{the axial variable of } F; \]

\[ x_h := \text{the axial variable of } H; \]

\[ pos_f := pos(x_f,F); neg_f := neg(x_f,F); \]

\[ pos_h := pos(x_h,H); neg_h := neg(x_h,H); \]

if $pos \neq pos_f$ then return No;

if $pos = 1$ then call Var_ren($F |_{\neg x_f}, H |_{\neg x_h}$);

if $neg = 1$ then call Var_ren($F |_{x_f}, H |_{x_h}$);

if (Var_ren($F |_{\neg x_f}, H |_{\neg x_h}$) = Yes) & (Var_ren($F |_{x_f}, H |_{x_h}$) = 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 $\varphi$ with $\varphi(F) = H$ if and only if $\varphi(x) = y$, $pos(x,F) = pos(y,H)$ and there are two variable renamings, $\varphi_1$ and $\varphi_2$, with $\varphi_1(F|_x) = H|_y$ and $\varphi_2(F|_{\neg x}) = H|_{\neg y}$. Therefore, there exists a variable renaming $\varphi$ with $\varphi(F|_x) = F_2$ 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 $\text{var}(F)=\text{var}(H)$ in MAX(1). We associate $F$ and $H$ to binary trees $T_F$ and $T_H$, respectively. Thus, there exists a literal renaming $\phi$ 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 $n=\#\text{var}(F)$ 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 $x \in \#\text{var}(F)$ we have $\text{occ}(x,F)=(1,1)$.

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

\[
\begin{pmatrix}
1 & 2 & 3 & 4 & 5 & 6 \\
1 & + & - \\
2 & + & - \\
3 & + & - \\
4 & + & - \\
5 & + & - \\
\end{pmatrix}
\]

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

Proof. Let $F=[C_1, \ldots, C_n]$ 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 $E_F=\{(c_i, x_k), (x_k, c_j) | x_k \in C_i, \neg x_k \in C_j, 1 \leq k \leq n, 1 \leq i, j \leq n+1\}$.

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

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

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

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

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 $1 \leq k \leq n$;
(b) $\deg(x_k^c) = n+3$ for every $1 \leq k \leq n$;
(c) $\deg(x_k) = 2$ for every $1 \leq k \leq n$;
(d) $3 \leq \deg(C_i) \leq n+2$ for every $1 \leq k \leq n+1$;
(e) $\deg(v) = 1$ for every $v \in V_{\text{var}} \cup V_{\text{var}}^c \cup V_{cl}$.

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 $F=\{C_1^c,\ldots,C_{n+1}^c\}$ be formulas over variables $x_1,\ldots,x_n$ in $\text{MARG}(1)$, 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 $\phi$ with $\phi(H)=F$ if and only if $T_H$ is isomorphic to $T_F$.

($\Rightarrow$) Suppose that there exists a variable renaming $\phi$ with $\phi(H)=F$. We have a permutation $\pi_c$ over $\{1,\ldots,n\}$ and a permutation $\pi_v$ over $\{1,\ldots,n+1\}$ such that $\phi(x_k) = x_{\pi_v(k)}$ for $1 \leq k \leq n$ and $\phi(C_i) = C_{\pi_c(i)}$ for $1 \leq i \leq n+1$.

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, $(c_i,x_k^c),(x_k,c_j)$, $(x_k^c,x_k),(x_k,c_j)$ are edges in $T_H$.

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

(1) $\phi(x_k) = x_{\pi_v(k)}^c, \phi(x_k^c) = x_{\pi_v(k)}$ for $1 \leq k \leq n$;
(2) $\phi(C_i) = c_{\pi_c(i)}$ for $1 \leq i \leq n+1$;
(3) $\phi(y_k) = y_{\pi_v(k)}$ for $1 \leq k \leq n$ and $1 \leq k \leq n+2$;
(4) $\phi(z_k^c) = z_{\pi_v(k)}^c$ for $1 \leq k \leq n$ and $1 \leq k \leq n+1$;
(5) $\phi(c_i^p) = c_{\pi_c(i)}^p$ for $1 \leq i \leq n+1$ and $p=1,2$.

($\Leftarrow$) Let $\phi$ be an isomorphism with $\phi(T_H)=T_F$. By the difference of degrees of nodes, we have that $\phi(V_{\text{var}})=V_{\text{var}}$, $\phi(V_{\text{var}}^c)=V_{\text{var}}^c$, $\phi(V_{cl})=V_{cl}$, and $\phi(V_{cl})=V_{cl}^c$. The restriction $\phi |_{\text{var}}$ 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, $c_i x_k^c x_k c_j$, from $c_i$ to $c_j$ through $x_k$. Please note that $T_H$ contains $3n^2+9n+3$ nodes. Thus, the problem $\text{Var-MARG}(1)$ 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 $\text{MU}(1)$ are related closely to the graph isomorphism problem. So, it is significant for investigating solvable variable and literal
renaming problems in polynomial time. In this paper, we investigate variable and literal renaming problems for two subclasses, \( \text{MAX}(1) \) and \( \text{MARG}(1) \), of minimal unsatisfiable formulas. We have proved that the literal renaming problems for formulas in \( \text{MAX}(1) \) and \( \text{MARG}(1) \) are solvable in linear time, and the variable renaming problem for formulas in \( \text{MAX}(1) \) and \( \text{MARG}(1) \) are solvable in quadratic time.

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

References:


**XU Da-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.