Theoretical Computer Science 8 (1979) 31–44

TAUTOLOGY TESTING WITH A GENERALIZED MATRIX REDUCTION METHOD

W. BIBEL

Institut für Informatik, Technische Universität, 8 München 2, Postfach 202420, Federal Republic of Germany

Communicated by Maurice Nivat
Received May 1977
Revised March 1978

Abstract. A formalization of the tautology problem in terms of matrices is given. From that a generalized matrix reduction method is derived. Its application to a couple of selected examples indicates a relatively efficient behaviour in testing the validity of a given formula in propositional logic—not only for machines but also for humans. A further result from that formalization is a reduction of the tautology problem to a part of Presburger arithmetic which involves formulas of the $\forall\exists\exists\cdots\exists$-type where all quantifiers have finite range.

0. Introduction

In 1968 Prawitz [12] proposed a theorem proving method called matrix reduction which has advantages in comparison with the resolution principle and other related methods. Yet, his approach so far has not found much attraction within the theorem proving community.

But in general, there is a growing interest in methods other than those connected with resolution [3], mainly in the following two directions. The first one can be characterized by the idea of using the natural representation of a problem (theorem) in a richer language than that of clausal form as well as natural deduction rules as a source of information for the guidance of the behaviour of a theorem prover ([2, 4, 11] etc.). In the second direction attempts are made to find more efficient derivation rules than resolution ([2, 8, 12] etc.). In the author's view a future powerful theorem proving system will profit from both directions of research, probably in the sense that basically it will behave according to the most efficient known rules but with built-in strategies which take advantage from the natural representation of the given formula.

This paper contributes to the second direction by generalizing Prawitz' matrix reduction method. Roughly speaking, the splitting of the matrix here is not necessarily done on the basis of just two clauses containing two complementary literals

but on that of larger parts determined by a certain strategy resulting in an algorithm which seems to be rather powerful in comparison to other methods (see Example 3.6, e.g.).

The basis for this result is a formalization of the tautology problem given in [1], which also implies that we restrict the algorithm to the case of propositional logic since the methodology is well-known how to "lift" it into first-order logic. In order to make this paper essentially self-contained this formalization together with some basic properties is introduced in the sections 1 and 2. In fact, the present state of this formalization is more elegant than that in [1], and is of interest in itself. For example, the characterization Theorem 2.5 gives a clear insight into the nature of all tautologies in n variables and provides a natural way of enumerating the set of tautologies.

The main reduction theorem then is a rather evident consequence of all these preparations, and, together with the resulting algorithm and a couple of examples, it is given in Section 3.

As a further outcome from our formalization a reduction of the tautology problem to a validity problem in a part of Presburger arithmetic is given in Section 4 which involves formulas of the $\forall\exists\exists\cdots\exists$ type where all quantifiers have finite range determined by the number of occurring variables. It may be used for testing the tautology of a formula as well. But the author feels there might also be an application in connection with the problem whether NP is closed under complementation [6].

1. Complementary matrices

Let V denote the alphabet consisting of the *letters* (or *literals*) $x, \bar{x}, x_1, \bar{x}_1, \ldots, y, \bar{y}, \ldots, z, \bar{z}, \ldots$ Letters without a bar are also called *variables*.

Definition 1.1. *Formula matrices* (shortly *matrices*) are defined inductively as follows.

(a) Any letter from V is a matrix.

(b) If $M_1, \ldots, M_n$ are matrices then the set $\{M_1, \ldots, M_n\}$ is a matrix.

Given a matrix M of the form $\{C_1, \ldots, C_n\}$ the C_i will also be called the *clauses* of M, $i = 1, \ldots, n$.

Matrices will be represented in the plane by assembling the clauses of a matrix horizontally and the matrices of a clause vertically. E.g.

$$\begin{array}{ccc} x_1 & & \bar{x}_1 \\ & x_3 & \\ x_2\,\bar{x}_1 & & \bar{x}_2 \quad x_1 \\ & \bar{x}_2 & \\ x_3 & & \bar{x}_3 \end{array}$$

represents the matrix

$$\{\{x_1, \{x_2, \bar{x}_1\}, x_3\}, \{x_3, \bar{x}_2\}, \{\bar{x}_1, \bar{x}_2, \bar{x}_3\}, x_1\}.$$

Matrices represent formulas in propositional logic which are defined in the usual way using variables $x, x_1, \ldots, y, \ldots$ and the connectives $\neg$, $\vee$, $\wedge$.

Definition 1.2. The *formula positively (negatively) represented by a matrix* is defined inductively as follows.

(a) If the matrix is a literal A then the formula positively (negatively) represented by A is (i) x $(\neg x)$ if $A \equiv x$, (ii) $\neg x$ (x) if $A \equiv \bar{x}$, respectively.

(b) If $F_1, \ldots, F_n$ are the formulas positively (negatively) represented by the matrices of a clause $C = \{M_1, \ldots, M_n\}$ of a matrix, resp., then the formula $F_1 \wedge \cdots \wedge F_n$ $(F_1 \vee \cdots \vee F_n)$ is positively (negatively) represented by C, respectively.

(c) If $F_1, \ldots, F_n$ are the formulas positively (negatively) represented by the clauses of a matrix $M = \{C_1, \ldots, C_n\}$, resp., then the formula $F_1 \vee \cdots \vee F_n$ $(F_1 \wedge \cdots \wedge F_n)$ is positively (negatively) represented by M, respectively.

People who are used to think in terms of validity (inconsistency) of a formula are supposed to use the positive (negative) matrix representation, resp. E.g.

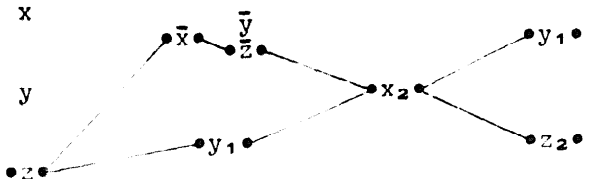
$$\begin{matrix} x \\ \bar{y} \end{matrix} \quad \bar{x} \quad y \begin{cases} \text{positively stands for } (x \wedge \neg y) \vee \neg x \vee y \\ \text{negatively stands for } (\neg x \vee y) \wedge x \wedge \neg y. \end{cases}$$

Definition 1.3. A *path* through a matrix M is a set of literals and is defined inductively as follows.

(a) There is exactly one path through an atom consisting of the atom itself.

(b) If $M = \{C_1, \ldots, C_n\}$, then for any n matrices M_i such that $M_i \in C_i$ and for any n paths P_i through M_i, $i = 1, \ldots, n$, the set $\bigcup_{i=1}^{n} P_i$ is a path through M.

In the following matrix four of its paths are illustrated by lines from left to right, but the reader should keep in mind that our notion of a path actually is much poorer than suggested by Fig. 1.



$P_1 = \{z, \bar{x}, \bar{z}, x_2, y_1\}$, $P_2 = \{z, \bar{x}, \bar{z}, x_2, z_2\}$,
$P_3 = \{z, y_1, x_2\}$, $P_4 = \{z, y_1, x_2, z_2\}$.

Fig. 1.

Definition 1.4. $|S|$ denotes the usual cardinality, i.e. the number of different elements of a set S.

Definition 1.5. A literal x is called *complementary to* $\bar{x}$ and vice versa. A path is called *complementary* if it contains a complementary pair of literals. A matrix is called *complementary* if each path through it is complementary.

E.g. P_1 and P_2 in the previous example are complementary, P_3, P_4, and the whole matrix are *not*. Since any clause is a matrix, it is clear that we also can talk of complementary clauses. For formal reasons the empty set $\emptyset$ also will be called complementary.

If the matrix M positively (negatively) represents the formula F, then the paths through M positively (negatively) represent the d-clauses (c-clauses) in F in the terminology of [1], resp. Therefore [1, Corollaries 8 and 12] now can be restated in the following way.

Theorem 1.6. *If a matrix M positively (negatively) represents a formula F, then F is valid (inconsistent) iff M is complementary, resp. (cf. the example after Definition* 1.2).

Thus the matrix representation unifies both the tautology and the inconsistency approach in an elegant way. An efficient algorithm testing whether a given matrix is complementary is one of the main objectives of this paper.

Definition 1.7. A *partial* matrix M' of a matrix M, in symbols $M' \sqsubseteq M$, is the result of deleting zero or more literals occuring somewhere in M. M' is called *partial w.r.t. paths (in M)*, in symbols $M' \preccurlyeq M$ iff $M' \sqsubseteq M$ and all paths through M' are paths through M.

E.g., if $M \equiv \begin{matrix} x & \bar{x} & x & \bar{x} \\ y & y & \bar{y} & \bar{y} \end{matrix}$ then $\begin{matrix} x \\ y \end{matrix} \bar{x}x \preccurlyeq M$, $x\bar{x} \preccurlyeq M$, etc., but $\begin{matrix} x & \bar{x} \\ y & y \end{matrix} \bar{x} \not\preccurlyeq M$ because $\{y, \bar{x}\}$ is a path through $\begin{matrix} x & \bar{x} \\ y & y \end{matrix} \bar{x}$ but not one through M.

Obviously both relations are reflexive and transitive. Also the following two results immediately follow from the definitions.

Lemma 1.8. *If the matrix M is complementary and $M' \preccurlyeq M$, then M' is complementary.*

Lemma 1.9. *If the matrix M is complementary and C is any clause, then $M \cup \{C\}$ is complementary.*

Definition 1.10. A matrix is in *normal form* if each of its clauses is a set of literals and no clause is complementary.

Obviously, positively (negatively) this represents the usual disjunctive (conjunctive) normal form, but without complementary clauses; here we are not interested in the opposite form; respectively.

2. A characterization of the tautologies

In the previous section the concepts relevant for this paper have been introduced in a more general form than they actually will be used in the sequel because from now on we will restrict ourselves to matrices in normal form. The reason is that this generalization did not cause any additional effort while on the other hand it seems rather promising—but also pretty complicated—to generalize the result of the next section for arbitrary matrices.

We are now going to define a fundamental subset of the set of all complementary matrices.

Definition 2.1. *Complete* matrices in normal form are defined inductively as follows.

(a) $\{x, \bar{x}\}$ is complete for any $x \in V$.

(b) If $M = \{C_1, \ldots, C_n\}$ is complete and $x \in V$ is a variable which does not occur in M, then $\{C_1 \cup \{x\}, \ldots, C_n \cup \{x\}, C_1 \cup \{\bar{x}\}, \ldots, C_n \cup \{\bar{x}\}\}$ is complete.

Examples are: $x\ \bar{x}$, $\begin{matrix} x & \bar{x} \\ y & y \end{matrix}$ $\begin{matrix} x & \bar{x} \\ \bar{y} & \bar{y} \end{matrix}$, etc.

Readers familiar with semantic trees will easily realize a close connection with this concept, but also certain differences which justify an independent study like ours.

If a matrix M contains no other literals than those from $\{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$ and at least one of $x_i, \bar{x}_i$, $i = 1, \ldots, n$, then this will be indicated symbolically by $M(x_1, \ldots, x_n)$. By a trivial inductive argument we have the following lemma.

Lemma 2.2. *For a complete matrix $M = M(x_1, \ldots, x_n)$ it holds that M is uniquely determined by $x_1, \ldots, x_n$ and M is complementary. Further $|M| = 2^n$ and for any non-complementary clause $D \subseteq \{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$ there is a clause $C \in M$ with $D \subseteq C$.*

Definition 2.3. If $N = N(x_1, \ldots, x_n)$ is any matrix, then the complete matrix $M = M(x_1, \ldots, x_n)$ also will be called the *completion* of N. If $D \subseteq \{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$ is any non-complementary clause then the set
$\mathrm{Ext}(D) := \{C \in M : M = M(x_1, \ldots, x_n)$ is complete $\wedge\ D \subseteq C\}$ will be called the *extension of D (in M)*. In the sense of [9] one could also say that any clause $C \in \mathrm{Ext}(D)$ is *covered by D*. $\mathrm{Ext}(D_1, \ldots, D_n) := \bigcup_{i=1}^{n} \mathrm{Ext}(D_i)$.

Lemma 2.4. *For each complementary set N of literals with $N \subseteq \{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$ there exists a path P through the complete matrix $M = M(x_1, \ldots, x_n)$ such that $N = P$.*

Proof by induction on n. We have the following cases.

(a) $\{x_n, \bar{x}_n\} \not\subseteq N$: the lemma follows by induction hypothesis and by definition.

(b) $\{x_n, \bar{x}_n\} = N$: N is a path through M by definition.

(c) $\{x_n, \bar{x}_n\} \cap N \neq \emptyset \wedge \{x_n, \bar{x}_n\} \neq N$: then there exists a literal x_i, $1 \leq i < n$, in N. Let N' be the result of substituting x_n by x_{n-1} and $\bar{x}_n$ by $\bar{x}_{n-1}$ (or x_n by $\bar{x}_{n-1}$ and $\bar{x}_n$ by x_{n-1}) in N such that N' again is complementary. By induction hypothesis N' is a path through the complete matrix $M' = M'(x_1, \ldots, x_{n-1})$ and thus also a path through M which by an appropriate choice of clauses (M' occurs twice in M) can be transformed into a path P with $P = N$.

Theorem 2.5. *A matrix $N = N(\{x_i : i \in I \subseteq \{1, \ldots, n\}\})$ in normal form is complementary iff $N \leq M$, where $M = M(x_1, \ldots, x_n)$ is complete.*

This theorem is an immediate consequence of the definitions and the Lemmata 2.2, 1.8, and 2.4. Together with Theorem 1.6 it characterizes the set of valid formulas in disjunctive normal form. This characterization can be used for a natural way of enumerating all tautologies in n variables since by definition of $\leq$ any matrix N with $N \leq M(x_1, \ldots, x_n)$ is the result of deleting upto $n-1$ literals in each clause of M. It also is one of the fundamentals for the algorithm in this paper. A first step towards its development is the following equivalent version of it.

Theorem 2.6. *A matrix $N = N(\{x_i : i \in I \subseteq \{1, \ldots, n\}\})$ in normal form is complementary iff $\mathrm{Ext}(N) = M$ for the complete matrix $M = M(x_1, \ldots, x_n)$.*

Proof. If the criterium is fulfilled, then with Lemma 2.2 it is obvious that each path through N is also a path through M, hence $N \leq M$, and therefore N is complementary by Theorem 2.5. Conversely, if N is complementary, then $N \leq M$ by Theorem 2.5. This means that for any clause C in M there is a non-empty subset C' of those literals in C which are contained in some path through N. By construction there must be clauses in N containing elements of C'. Therefore, if there would be no clauses $D \in N$ with $D \subseteq C$, each clause in N containing elements from C' would contain at least one literal not in C'. This in turn would mean that there is a path through N not including any literal from C which contradicts $N \leq M$.

As a consequence of this theorem one could think of an algorithm which for each $C \in M$ checks whether there is a $D \in N$ with $D \subseteq C$. Or, alternatively, one could test for each $D \in N$ (possibly in a certain sequence) which clauses $C \in M$ are covered by D and stop as soon as all clauses in M are covered. Let us try the second approach which seems to be more economical.

3. The algorithm

Agreeing on the notation $S(M)$ for the subsumption operation, i.e. for the set $M \backslash M_0$ where M is any set of sets and $M_0 := \{m \in M : \exists m' \in M,\ m' \neq m,\ m' \subset m\}$ we can now state and prove the main reduction theorem.

Theorem 3.1. *Let*
—$M = M(x_1, \ldots, x_n)$ *be a matrix in normal form,*
—$\emptyset \neq I \subseteq \{1, \ldots, n\}$, $\bar{I} = \{1, \ldots, n\} \backslash I$,
—$M_1 = \{C \in M : C \subseteq \{x_i, \bar{x}_i : i \in I\}\}$,
—N_1 *be the complete matrix in* $\{x_i : i \in I\}$,
—$\bar{M}_1 = \{C \in N_1 : C \notin \mathrm{Ext}(M_1)$ *in* $N_1\}$, *i.e. the set of those clauses in* N_1 *which are not covered by some clause from* M_1,
—N_2 *be the complete matrix in* $\{x_i : i \in \bar{I}\}$
—$M_2 = \{C \in M : C \subseteq \{x_i, \bar{x}_i : i \in \bar{I}\}\}$,
—$M_3 = M \backslash (M_1 \cup M_2)$.
Then M *is complementary iff* $\bar{M}_1 = \emptyset$, *or* $\bar{M}_1 \neq \emptyset$ *and for each clause* $C \in \bar{M}_1$ *the set* $M^C = S(M_2 \cup M_3^C)$ *is complementary where* $M_3^C = \{D' \subseteq \{x_i, \bar{x}_i | i \in \bar{I}\} : \exists D \in M_3\ \exists D'' \subseteq C$ *such that* $D = D' \cup D''\}$.

Proof. If $\bar{M}_1 = \emptyset$, then M_1 is complementary by Theorem 2.6, hence M is complementary by Lemma 1.9. Therefore we can assume $\bar{M} \neq \emptyset$.

M^C not complementary $\Rightarrow$ M not complementary: If there is a clause $C \in \bar{M}_1$ such that M^C is not complementary then by Definition 1.5 there is a path P_1 through M^C which is not complementary. Now, for any such given C M can be naturally divided into two disjoint parts, namely the set M' of those clauses from M which contributed to the construction of M^C, and the remaining ones, say M''; i.e. $M' \cup M'' = M$. Thus, P_1 is also a path through M' and contains only variables with indices from $\bar{I}$. The extension of the matrix $\tilde{M}''$ which results from M'' by erasing all variables with indices from $\bar{I}$, is different from N_1, since it does not cover C by construction; therefore, by Theorem 2.6 $\tilde{M}''$ is not complementary; thus there is a non-complementary path P_2 through $\tilde{M}''$ which, by construction, only contains variables with indices from I. Of course, P_2 also is a path through M''. So $P = P_1 \cup P_2$ is a non-complementary path through M since P_1 and P_2 have no variables in common and both are non-complementary. Hence M is non-complementary by Definition 1.5.

M^C complementary $\Rightarrow$ M complementary: On the other hand, assume that M^C is complementary for all $C \in \bar{M}_1$. Then any path through M which "crosses" M^C (i.e. which contains a path through M^C as a subpath) for some C is complementary by Definition 1.5. All other paths through M necessarily contain a literal from each clause in M_1, by Definition 1.3, from each clause in $\bar{M}_1$, by construction of M^C, and therefore, by Definition 2.3, from each clause in N_1 by construction of M^C, and thus they are complementary since this is the case for N_1 by Lemma 2.2.

Theorem 3.1 can be used to set up a formal system for propositional logic: its axioms are the complete matrices and its only rule of inference is that from the set $\{M^C : C \in \bar{M}_1\}$ of premises to the conclusion M. But in this paper the main interest is its transformation into an algorithm which is straightforward as well.

We only have to take some care with an appropriate choice for I. In order to guarantee polynomial behaviour of the algorithm in this introductory step only those I's are taken into consideration for which $2^{|I|} \leq |M|$ or $2^{|I|-1} < |M| \leq 2^{|I|}$ is true. Among those as a strategy it chooses in step b one for which the greatest number of clauses in the completion of M is covered by M_1 with a preference for smaller I's in case that there is a choice among more than one element. Of course, other strategies might serve as well, notably simpler ones which do not require such a sophisticated calculation for the determination of I.

Algorithm 3.2. (tests whether a given matrix $\emptyset \neq M = M(x_1, \ldots, x_n)$ in normal form is complementary)

Step a. Preparatory step.

Step a.1. Remove all clauses containing variables in partial state from M (see [6, 2.2.4 or 2.5.1]) and separate (see [6, 3.3]).

Step a.2. $M \leftarrow S(M)$ (see [6, 2.2.1]).

Step b. Determination of $I \subseteq \{1, \ldots, n\}$.

Step b.1. Set $k \leftarrow 1, j \leftarrow 1, I \leftarrow \emptyset$, and determine l such that $2^{l-1} < |M| \leq 2^l$ is true.

Step b.2. Set

$$j' \leftarrow \max_{\{J:|J|=k\}}(|\operatorname{Ext}(\{C \in M : x_i \in C \vee \bar{x}_i \in C \rightarrow i \in J\})|)$$

where the extension is taken within the complete matrix in $\{x_i : i \in J\}$. If $j' = 2^k$, then M is complementary; else if $j < j' \cdot 2^{n-k}$, then choose a J_0 from $\{J : |J| = k\}$ for which the maximum j' has been reached and set $I \leftarrow J_0$ and $j \leftarrow j' \cdot 2^{n-k}$.

Step b.3. If $k < 1$, then ($k \leftarrow k+1$ and go to Step b.2).

Step c. Reduction step

(Recall the terminology of Theorem 3.1 and observe that $\bar{M}_1 \neq \emptyset$ is now guaranteed by *Step* b.2 if M is complementary.) If $M_1 = \emptyset$, then M is not complementary; else call the whole algorithm for each member of the set $S(\{M^C : C \in \bar{M}_1\})$.

Let us consider a few examples.

Example 3.3. For $\begin{matrix} x_1 \\ x_2 \end{matrix} \bar{x}_1 \bar{x}_2$ Step b yields $j' = 4$ for $k = 2$ and stops with a positive result in the first run.

Example 3.4. Also for $x_1 \ \bar{x}_1 \ M'$ with any $M' = M'(x_1, \ldots, x_n)$ the test $j' = 2^k$ stops the algorithm immediately with a positive result.

Example 3.5. For $\begin{matrix} x_1 \\ x_2 \end{matrix}\ \bar{x}_1\ \bar{x}_2\ \begin{matrix} \bar{x}_3 \\ x_4 \end{matrix}\ \begin{matrix} x_3 \\ x_4 \\ x_5 \end{matrix}$ the problem is reduced to that in Example 3.3 after passing the preparatory Step a.1.

These all are rather trivial examples. Therefore as our next sample we choose one that usually is regarded as a hard one and in fact brings methods like the resolution principle into considerable problems.

Example 3.6. Let M be the following matrix in normal form

$$\begin{array}{cccc|cccc|cccc|cccc|cccc|cccc}
x_1 & \bar{x}_1 & \bar{x}_1 & x_1 & \bar{x}_1 & x_1 & x_1 & \bar{x}_1 & \bar{x}_3 & x_3 & x_3 & \bar{x}_3 & \bar{x}_3 & x_3 & x_3 & \bar{x}_3 & x_1 & \bar{x}_1 & x_1 & x_1 & x_1 & \bar{x}_1 & x_1 & \bar{x}_1 \\
x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_4 & \bar{x}_4 & x_4 & \bar{x}_4 & x_4 & \bar{x}_4 & x_4 & \bar{x}_4 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 \\
x_5 & x_5 & \bar{x}_5 & \bar{x}_5 & x_6 & x_6 & \bar{x}_6 & \bar{x}_6 & x_6 & x_6 & \bar{x}_6 & \bar{x}_6 & x_5 & x_5 & \bar{x}_5 & \bar{x}_5 & x_5 & x_5 & \bar{x}_5 & \bar{x}_5 & x_5 & x_5 & \bar{x}_5 & \bar{x}_5
\end{array}$$

Step b chooses for I any element from $\{\{1, 2, 5\}, \{1, 2, 6\}, \{3, 4, 5\}, \{3, 4, 6\}\}$, say $\{3, 4, 6\}$. Therefore

$$\bar{M}_1 = \{\{x_3, x_4, x_6\}, \{\bar{x}_3, \bar{x}_4, x_6\}, \{\bar{x}_3, x_4, \bar{x}_6\}, \{x_3, \bar{x}_4, \bar{x}_6\}\}.$$

Since every two elements are equal in the set $\{M_3^C : C \in \bar{M}_1\}$ step c calls the algorithm again with the following two subproblems:

$$M' = \begin{array}{cccc|cc|c}
x_1 & \bar{x}_1 & \bar{x}_1 & x_1 & \bar{x}_1 & x_1 & \\
x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & \\
x_5 & x_5 & \bar{x}_5 & \bar{x}_5 & & & \bar{x}_5
\end{array}$$

which belongs to $\{x_3, x_4, x_6\}$ and $\{\bar{x}_3, \bar{x}_4, x_6\}$, and

$$M'' = \begin{array}{cccc|cc|c}
x_1 & \bar{x}_1 & \bar{x}_1 & x_1 & x_1 & \bar{x}_1 & \\
x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & x_2 & \bar{x}_2 & \\
x_5 & x_5 & \bar{x}_5 & \bar{x}_5 & & & x_5
\end{array}$$

which belongs to $\{\bar{x}_3, x_4, \bar{x}_6\}$ and $\{x_3, \bar{x}_4, \bar{x}_6\}$. Let us consider M'. By Step a.2 in this second run the 3rd and 4th clause disappear; Step b sets $I = \{5\}$; and Step c calls for a third run for the matrix

$$\begin{array}{cccc}
x_1 & \bar{x}_1 & \bar{x}_1 & x_1 \\
x_2 & \bar{x}_2 & x_2 & \bar{x}_2
\end{array}$$

which is complete and therefore has the algorithm return a positive message within Step b. The behaviour in the case of M'' is completely analogous to that of M'.

These and other examples indicate a remarkably efficient behaviour of this general tautology testing algorithm which is related to Prawitz' matrix reduction method [12] not only because of the spirit of the approach but also technically since

in a certain sense it is contained in that case of Theorem 3.1 for which I is a singleton. This justifies the title of the present paper and indicates the reason for the success of our algorithm which may split the matrix also on the basis of larger parts not only on those of complementary literals.

Technically, there is a similar and even closer relationship with Galil's "enumeration dags" [8, 13] which has been pointed out to the author by J. Schreiber. Specifically, there is an isomorphism between Galil's method and the method obtained from ours by restricting I to a singleton and extending the application of the subsumption operation in the reduction step c of Algorithm 3.2 to the set of *all* matrices considered so far by the running algorithm (J. Schreiber, personal communication). Since the latter method may be regarded as a special case of the generalized matrix reduction method, we have the following obvious result.

Theorem 3.7. *W.r.t. the length of proofs the generalized matrix reduction method is not worse than Galil's method.*

A comparison w.r.t. proof length of Galil's method with other methods is given in [8, 13] which in turn connects our method with those.

Theorem 3.8. *There are tautologies s.t. the generalized matrix reduction method with unrestricted I needs fewer reduction steps than that with $|I| = 1$.*

Example 3.6 verifies this statement which expresses the fact that the unrestricted method allows a more compact, more readable representation of a proof for the validity of a formula than the restricted (or Galil's) method.

4. Reduction to Presburger arithmetic

In this section we give a further application of the formalization from Sections 1 and 2 and describe a reduction of the tautology problem to Presburger arithmetic, notably to a considerably restricted part of it. This will be possible by agreeing on the following standard form for matrices.

Definition 4.1. A matrix M in normal form is in *standard form* iff

(a) $M = M(x_1, \ldots, x_n)$ for some n and a fixed enumeration $x_1, x_2, \ldots, x_n, \ldots$ of all variables,

(b) each clause in M is ordered with respect to increasing index,

(c) the set of clauses in M is partially ordered by a relation $<_n$, defined inductively as follows: (i) $<_0 := \emptyset$, (ii) $<_{i+1} := \{(C_1, C_2)$: $(x_{i+1} \in C_1 \wedge \bar{x}_{i+1} \in C_2) \vee ((C_1, C_2) \in <_i \wedge (x_{i+1} \in C_1 \vee \bar{x}_{i+1} \in C_2))$.

Obviously, for complete matrices $<_n$ is a total ordering. E.g. the standard form for the complete matrix $M = M(x_1, x_2, x_3)$ is the following:

$$\begin{array}{cccc|cccc} x_1 & \bar{x}_1 & x_1 & \bar{x}_1 & x_1 & \bar{x}_1 & x_1 & \bar{x}_1 \\ x_2 & x_2 & \bar{x}_2 & \bar{x}_2 & x_2 & x_2 & \bar{x}_2 & \bar{x}_2 \\ x_3 & x_3 & x_3 & x_3 & \bar{x}_3 & \bar{x}_3 & \bar{x}_3 & \bar{x}_3 \end{array}$$

In the following M^n (or shortly M if n is understood) denotes the complete matrix $M(x_1, \ldots, x_n)$ in standard form and C_m^n (or C_m) denotes the mth clause in M, $1 \leqslant m \leqslant 2^n$. x_i is abbreviated by $+i$, $\bar{x}_i$ by $-i$. The function $s: \{x_1, \bar{x}_1, \ldots\} \to \{0, 1\}$ is defined by $s(+i) := 0$ and $s(-i) := 1$.

Since the quantifiers needed for what follows in fact are finite disjunctions or conjunctions we use the symbols $\bigvee$ and $\bigwedge$ rather than $\exists$ and $\forall$ resp. $\bigvee_{i=c}^{d} F$ and $\bigvee_{\{i|G\}} F$ is short for $\bigvee i\ (c \leqslant i \leqslant d \wedge F)$ and $\bigvee i(G \wedge F)$.

Definition 4.2. $F(\pm i, m, n) := \bigvee_{l=0}^{p_i} (q(l, \pm i) < m \leqslant q'(l, \pm i))$, where

$$p_i := 2^{n-i} - 1, \quad q(l, \pm i) := l \cdot 2^i + s(\pm i) \cdot 2^{i-1},$$

$$q'(l, \pm i) := q(l, \pm i) + 2^{i-1}, \quad 1 \leqslant i \leqslant n, 1 \leqslant m \leqslant 2^n.$$

The meaning of those expressions will be clear by the following lemma which immediately follows from Definition 4.1.

Lemma 4.3. *$F(\pm i, m, n)$ is true iff $\pm i \in C_m^n$.*

Theorem 4.4. *$N = N(x_1, \ldots, x_n)$ in normal form is complementary iff*

$$J := \{m: \bigvee_{\{D: D \in N\}} \bigwedge_{\{i: \pm i \in D\}} F(\pm i, m, n)\} = \{m: 1 \leqslant m \leqslant 2^n\}$$

$$\text{iff } \{m: 1 \leqslant m \leqslant 2^n\} \subseteq J.$$

This theorem which is a simple consequence of Theorem 2.6 and Lemma 4.3 reduces the tautology problem to a problem in Presburger arithmetic namely whether $1 \leqslant m \leqslant 2^n$ implies the truth of the formula in the expression defining J. Now we want to show that this formula can be brought into an even more specialized form. For this purpose the following two lemmata will be needed.

Lemma 4.5. $q_{k+1} < m \leqslant q'_{k+1} \wedge q_k < m \leqslant q'_k$ (for $0 < l_k < i_{k+1}$) is equivalent with $r_k \leqslant l_k \leqslant r'_k$, where

$$q_k^{(l)} := q^{(l)}(l_k, \pm i_k),$$

$$r'_k := l_{k+1} \cdot 2^{i_{k+1} - i_k} + s(\pm i_{k+1}) \cdot 2^{i_{k+1} - i_k - 1},$$

$$r'_k := r_k + 2^{i_{k+1} - i_k - 1} - 1.$$

Proof. The first two inequalities can be rewritten equivalently in the following way:

$$m = q_{k+1} + j \wedge 0 < j \leq 2^{i_{k+1}-1} \wedge m = q_k + j' \wedge 0 < j' \leq 2^{i_k - 1}.$$

Eliminating m and isolating l_k gives

$$r_k + j \cdot 2^{-i_k} - s(\pm i_k) \cdot 2^{-1} - j' \cdot 2^{-i_k} = l_k \wedge 0 < j \leq 2^{i_{k+1}-1} \wedge 0 < j' \leq 2^{i_k - 1}.$$

This can be combined into the following inequality

$$r_k - s(\pm i_k) \cdot 2^{-1} - 2^{-1} < l_k < r'_k + 1 - s(\pm i_k) \cdot 2^{-1}$$

which obviously is equivalent with the third inequality in the lemma.

Lemma 4.6. $0 \leq l_{k+1} \leq p_{i_{k+1}} \wedge r_k \leq l_k \leq r'_k$ *implies* $0 \leq l_k \leq p_{i_k}$ *for* $0 < i_k < i_{k+1}$.

Proof. Combining the two inequalities in the premise gives

$$0 \leq s(\pm i_{k+1}) \cdot 2^{i_{k+1}-i_k-1} \leq l_k \leq p_{i_k} - 2^{i_{k+1}-i_k-1} \cdot (1 - s(\pm i_{k+1})) \leq p_{i_k}.$$

Definition 4.7. $F(\pm i_k, \ldots, \pm i_1, m, n) := \bigvee_{l_k=0}^{p_{i_k}} R(l_k, \pm i_k, \ldots, \pm i_1, m)$ where R is defined inductively as follows:

(a) $R(l_1, \pm i_1, m)$ is given by Definition 4.2;

(b) For $0 < i_k < i_{k+1} \leq n$ let

$$R(l_{k+1}, \pm i_{k+1}, \pm i_k, \ldots, \pm i_1, m) := \bigvee_{l_k=r_k}^{r'_k} R(l_k, \pm i_k, \ldots, \pm i_1, m).$$

$$F(D, m, n) := F(\pm i_k, \ldots, \pm i_1, m, n) \quad \text{for } D = \{\pm i_1, \ldots, \pm i_k\}.$$

Theorem 4.8. $\bigwedge_{j=1}^{k} F(\pm i_j, m, n)$ *is true iff* $F(\pm i_k, \ldots, \pm i_1, m, n)$ *is true.*

The proof is by induction on k. Since the case $k = 1$ is trivial we assume that the theorem holds for $1 \leq k < n$. $\bigwedge_{j=1}^{k+1} F(\pm i_j, m, n)$ is true, iff (by definition of $F(\pm i_{k+1}, m, n)$ and of $F(\pm i_k, m, n)$)

$$0 \leq l_{k+1} \leq p_{i_{k+1}} \wedge q_{k+1} < m \leq q'_{k+1} \wedge q_k < m \leq q'_k \wedge$$

$$\bigwedge_{j=1}^{k} F(\pm i_j, m, n)$$

is true,

iff (by Lemma 4.5, induction hypothesis, definition of $F(\pm i_k, \ldots)$, Lemma 4.6, and, finally, definition of $F(\pm i_{k+1}, \ldots)$)

$$F(\pm i_{k+1}, \pm i_k, \ldots, \pm i_1, m, n)$$

is true.

Corollary 4.9. *$N = N(x_1, \ldots, x_n)$ in normal form is complementary iff $1 \leqq m \leqq 2^n \rightarrow \bigvee_{\{D: D \in N\}} F(D, m, n)$ is true.*

This is an immediate consequence of Theorem 4.4, Definition 4.7, and Theorem 4.8. Together with Theorem 1.6 it says that the validity of a propositional formula F can be tested by calculating the truth value of a formula G in Presburger arithmetic where G can be derived from F (by Definition 4.7) in linear time.

This result can be considered as the starting point for the development of an algorithm like that in Section 3 which operates on formulas in Presburger arithmetics.

But we believe that it may be of a theoretical interest as well because it says that the test for validity of F in propositional logic which is an $\overline{\text{NP}}$-hard problem is as hard as the test for truth of G which is of the form $\forall\exists\exists \cdots \exists A$. On the other hand it is well-known that the test for satisfiability of $\neg F$ which is an NP-hard problem is as hard as the test for truth of $\exists x_1 \exists x_2 \cdots \exists x_n B$ where B is the formula obtained from $\neg F$ by replacing each occurrence of variable x_i by $x_i = 0, i = 1, 2, \ldots, n$. In other words the difference between NP-hard and $\overline{\text{NP}}$-hard problems has been reduced to a difference of logical structure within Presburger arithmetic, notably to that of one additional quantifier.

5. Summary

In this paper an approach to the validity problem has been given in terms of matrices which allows a unified treatment of both the validity and inconsistency problem (Theorem 1.6) as well as a lucid characterization of the set of tautologies (Theorems 2.5 and 2.6). The clearness of this formal representation of the tautology problem has been further demonstrated by deriving two results in a rather straightforward way.

The first is a matrix reduction theorem (Theorem 3.1) leading to a tautology algorithm (Algorithm 3.2) which for several selected theorems exhibits a surprisingly efficient behaviour. A brief comparison with Galil's enumeration dags has been given.

The second result provides the possibility of calculating whether a given formula is a tautology within a restricted part in Presburger arithmetic, which is characterized by formulas of the type $\forall\exists\exists \cdots \exists$.

Acknowledgements. I thank G. Huet, N. Riedel, H. Saya, J. Schreiber, J. Siekmann and two referees for a number of critical remarks and helpful comments which have been considered in this version. I also thank A. Meyer and M. Rabin for a valuable discussion on the result reported in Section 4.

References

[1] W. Bibel, A syntactic connection between proof procedures and refutation procedures, in *Theoretical Comp. Science 3rd GI Conf.*, Lecture Notes in Computer Science **48** (Springer, Berlin, 1977) 215–224. First presented at the Conference on Automatic Theorem Proving, Oberwolfach, Jan. 1976.

[2] W. Bibel and J. Schreiber, Proof search in a Gentzen-like system of first-order logic, in: *Proc. Int. Computing Symposium.* (North-Holland, Amsterdam, 1975) 205–212.

[3] W.W. Bledsoe. Non-resolution theorem proving, Univ. of Texas, Math. Dept. Memo, ATP 29 (1975).

[4] W.W. Bledsoe. R. Boyer and W. Henneman, Computer proofs of limit theorems, *Artif. Intelligence* **3** (1972) 27–60.

[5] S.A. Cook, The complexity of theorem proving procedures, in: *Proc. 3rd Ann. ACM Symp. on Theory of Computing* (1971).

[6] B. Dunham, H. Wang, Toward feasible solutions of the tautology problem, *Ann. Math. Logic* **10** (1976) 117–154.

[7] M.J. Fischer and M.O. Rabin, Super-exponential complexity of Presburger arithmetic, MIT, MAC Techn. Memo 43 (1974).

[8] Z. Galil, The complexity of resolution procedures for theorem proving in the propositional calculus, TR 75-239, Cornell University (1975).

[9] L.J. Henschen, Theorem proving by covering expressions, Dept. Comput. Sci., Northwestern Univ., Evanston, IL (1976).

[10] R.M. Karp, Reducibility among combinatorial problems, in R.E. Milner and J.W. Thatcher, Eds., *Complexity of Computer Computations* (Plenum Press, New York, 1972) 85–103.

[11] A. Nevins, A human-oriented logic for automatic theorem proving, *J. Assoc. Comput. Mech.* (1974) 606–621.

[12] D. Prawitz, *A Proof Procedure with Matrix Reduction*, Lecture Notes in Mathematics **125** (Springer, Berlin, 1970) 207–213.

[13] R.A. Reckhow, On the lengths of proofs in the propositional calculus, Techn. Report No. 87, Univ. of Toronto (1976).

[14] H. Saya and R. Caferra, A structure sharing technique for matrices and substitutions in Prawitz's theorem proving method, Rapport de Recherche No. 101, Univ. of Grenoble (1977).

[15] G.S. Tseitin, On the complexity of derivations in the propositional calculus, in: A.O. Slisenko, Ed., *Studies in Constructive Mathematics and Mathematical Logic II.* (1968) 115–125.