RUTCOR
RESEARCH
REPORT

On the Complexity of Minimizing the Number of Literals in Horn Formulae

Ondřej Čepek[a] Petr Kučera[b]

RRR 11-2008,

RUTCOR
Rutgers Center for
Operations Research
Rutgers University
640 Bartholomew Road
Piscataway, New Jersey
08854-8003
Telephone: 732-445-3804
Telefax: 732-445-5472
Email: rrr@rutcor.rutgers.edu
http://rutcor.rutgers.edu/∼rrr

[a]Department of Theoretical Computer Science and Mathematical Logic, Charles University, Malostranské náměstí 25, 118 00 Praha 1, Czech Republic (cepek@ksi.ms.mff.cuni.cz)
[b]Department of Theoretical Computer Science and Mathematical Logic, Charles University, Malostranské náměstí 25, 118 00 Praha 1, Czech Republic (kucerap@kti.ms.mff.cuni.cz)

Rutcor Research Report
RRR 11-2008,

On the Complexity of Minimizing the Number of Literals in Horn Formulae

Ondčej Čepek Petr Kučera

Abstract. The problem of pure Horn minimization with respect to the number of literals can be defined as follows. Given a pure Horn CNF, construct a logically equivalent pure Horn CNF which has the minimum possible number of literals. Horn minimization is an important and widely studied problem with many practical applications in artificial intelligence and in database design. In this paper we prove that pure Horn minimization with respect to the number of literals is NP-hard even for CNFs with at most five literals per clause.

Acknowledgements: The authors gratefully acknowledge the support by the the Czech Science Foundation (grant 201/04/1102)

1 Introduction

Horn functions constitute a very important subclass of Boolean functions. Their importance stems from the fact that the satisfiability problem (SAT), which is NP-complete for general Boolean formulae (see e.g. [18]), is solvable in linear time for Horn formulae [15, 25, 30]. This implies that certain real-life problems which require solving SAT become tractable if the underlying Boolean function in the problem is Horn. Such problems arise in several application areas, among others in artificial intelligence [14, 22, 24] and database design [13, 29]. Horn clauses are by far the most popular type of knowledge representation which is due to both the computational efficiency of reasoning and their sufficient richness for capturing essential features of real-life problems.

In some applications an important problem is to find a shortest possible representation of a given Boolean function. For instance, in artificial intelligence this problem is equivalent to finding a most compact representation of a given knowledge base [22, 24]. Such transformation of a knowledge base accomplishes knowledge compression, since the actual knowledge does not change, while the size of the representation can be significantly reduced. The computational complexity of reasoning in the case of Horn knowledge bases is reduced accordingly, since the compressed knowledge base is guaranteed to remain Horn. The procedure of knowledge compression preprocesses the knowledge base, and can be done off-line. This results in speeding up on-line operation while answering queries. Therefore, the computational expense of a single run of knowledge compression will be quickly amortized over a large number of queries to the knowledge base.

Unfortunately, unlike satisfiability, the representation minimization problem (for several different measures of minimality) is NP-hard not only in the general case, but also for Horn CNFs [3, 5, 8, 22, 29]. In this paper, where the number of literals is the selected measure, the Horn Minimization (HM) problem can be stated as follows: given a Horn CNF $\mathcal{F}$ find a CNF $\mathcal{F}'$ representing the same function and such that $\mathcal{F}'$ consists of a minimum possible number of literals.

The first NP-hardeness proof for HM appeared in [29]. Although strictly speaking the measure defined there is slightly different from the one used here, the proof can be easily modified to work also for HM. A simpler proof (this time really using the number of literals as the minimality measure) then appeared in [22]. Both of the these proofs have the drawback, that the reduction constructs clauses of a very high degree (equal to the number of all propositional letters). Therefore it is natural to ask, what is the complexity of HM for CNFs of bounded degree (CNF has a bounded degree if every clause in it does). This question was addressed in [8] where it was proved, that HM stayes NP hard even if we restrict the input to CNFs of degree at most seven. In this paper we shall strengthen the result to CNFs of degree at most five.

2 Basic Definitions and Main Results

A *Boolean function* f on n propositional letters $x_1, \ldots, x_n$ is a mapping $\{0,1\}^n \to \{0,1\}$. The propositional letters $x_1, \ldots, x_n$ and their negations $\overline{x}_1, \ldots, \overline{x}_n$ are called *positive* and *negative literals* respectively. Both sets together are called just *literals.* An elementary disjunction

$$C = \bigvee_{i \in I} \overline{x}_i \bigvee_{j \in J} x_j \tag{1}$$

of literals is called a *clause*, if every propositional letter appears in it at most once, i.e. if $I \cap J = \emptyset$. A *degree* of a clause is the number of literals in it. For two Boolean functions f and g we write $f \leq g$ if

$$\forall (x_1, \ldots, x_n) \in \{0,1\}^n \ : \ f(x_1, \ldots, x_n) = 1 \Longrightarrow g(x_1, \ldots, x_n) = 1 \tag{2}$$

Since each clause is in itself a Boolean function, formula (2) also defines the meaning of inequalities $C_1 \leq C_2$, $C_1 \leq f$, and $f \leq C_1$ where C_1, C_2 are clauses and f is a Boolean function.

We say that a clause C_1 *subsumes* another clause C_2 if $C_1 \leq C_2$ (e.g. the clause $\overline{x} \vee z$ subsumes the clause $\overline{x} \vee \overline{y} \vee z$). A clause C is called an *implicate* of the function f if $f \leq C$. An implicate C of a function is called *prime* if there is no distinct implicate C' subsuming C, or in other words, an implicate of a function is prime if dropping any literal from it produces a clause which is not an implicate of the function.

It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses. Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f. It should be noted that a given Boolean function may have many CNF representations (typically exponentially many in the number of propositional letters). Any two CNFs representing the same function are called *equivalent.* A CNF $\mathcal{F}$ representing a function f is called *prime* if each clause of $\mathcal{F}$ is a prime implicate of the function f. A CNF $\mathcal{F}$ representing a function f is called *irredundant* if dropping any clause from $\mathcal{F}$ produces a CNF that does not represent f.

It is well-known (see e.g. [20]) that each prime implicate of a Horn function is either negative or pure Horn and each prime implicate of a pure Horn function is pure Horn. Thus, in particular, any prime CNF representing a Horn function is Horn and any prime CNF representing a pure Horn function is pure Horn.

For the purposes of measuring the complexity of algorithms we need a way how to measure the "size" of a given CNF. In the Boolean related literature the following two definitions are the most commonly used ones. Let $\mathcal{F}$ be a CNF. Then

- $|\mathcal{F}|_c =$ the number of clauses in $\mathcal{F} = \sum_{C \in \hat{\mathcal{F}}} 1$
- $|\mathcal{F}|_\ell =$ the number of literals in $\mathcal{F} = \sum_{C \in \hat{\mathcal{F}}} |C|$

where $\hat{\mathcal{F}}$ denotes the set of all clauses present in $\mathcal{F}$.

The problem which concerns us in this paper stems from the fact that CNF representations of Boolean functions are not unique. To find a CNF representation having some specific properties (e.g. the minimum number of literals, clauses, etc.) of a given Boolean function is usually a hard problem, not well solved even in the special cases of Horn or pure Horn functions. The decision variant of the problem of finding a Horn CNF with the minimum number of literals for a given Horn function can be stated as follows.

Minimization of the Number of Literals (MNL)

Instance: A pure Horn CNF $\mathcal{F}$ and a positive integer ℓ.
Question: Is there a CNF equivalent to $\mathcal{F}$ which contains at most ℓ literals?

As was remarked in the introduction, this problem is NP-complete (see [29, 3]). Moreover, in [8] it was proved that this problem remains NP-complete even for CNFs of degree at most seven. Here we shall show that this proof can be easily modified to achieve the following result.

Theorem 1 *Let $\mathcal{F}$ be a pure Horn CNF of degree at most five (each clause has degree at most five) and let ℓ be a positive integer. The problem of whether there exists a CNF $\mathcal{F}'$ representing the same function as $\mathcal{F}$ and such that $|\mathcal{F}'|_\ell \leq \ell$ is NP-complete.*

We shall first prove, that problem MNL is in NP. In the rest of the paper (Sections 3–6) we shall present a modification of the original (never published) proof from [8] which strengthens the result from formulae of degree at most seven (claimed in [8]) to formulae of degree at most five, as claimed in Theorem 1.

Lemma 2.1 *The MNL problem is in NP.*

Proof. Since a similar proof was given in [22] (and full details are also in [8]) we shall provide only a sketch of the proof here. If the answer to a given instance of MNL is *yes* and we are given a (not necessarily pure Horn) CNF $\mathcal{F}'$ as the "certificate", we first count the number of literals in $\mathcal{F}'$ to check that this number is not larger than ℓ. If yes, we construct for each clause of $\mathcal{F}'$ (in quadratic time, see [20]) a prime implicate of $\mathcal{F}$ subsuming it. Let $\mathcal{F}''$ be the conjunction of these prime implicates. Clearly, $\mathcal{F}''$ is Horn and consists of at most l literals.

Next we check whether $\mathcal{F}$ and $\mathcal{F}''$ are equivalent. For this we check first the inequality $\mathcal{F}'' \leq \mathcal{F}$ by verifying (in quadratic time, see [15]) that every clause of $\mathcal{F}$ is an implicate of $\mathcal{F}''$. If yes, then we finish by checking the reverse inequality $\mathcal{F} \leq \mathcal{F}''$ in a similar way.

If $\mathcal{F}''$ passed the above test, it itself verifies that the answer to the given instance of MNL is indeed yes. On the other hand if it failed the test, i.e. if $\mathcal{F}''$ is not equivalent to $\mathcal{F}$, then neither is $\mathcal{F}'$ and we can reject it as a certificate. ■

In the rest of this paper we shall show that the known NP-hard problem of finding a maximum matching in a (undirected) hypergraph (also known as the set packing problem, see e.g. [18]) reduces in polynomial time to MNL, thus establishing the NP-completeness of MNL. To simplify notation we shall sometimes use a shorthand $S \vee y$ instead of writing $\bigvee_{x \in S} \overline{x} \vee y$ to denote a pure Horn clause. For the purposes of this paper a CNF is called *minimum* if there is no CNF representing the same function and containing fewer literals. In the next subsection we shall describe a simple procedure which we shall use as a tool for proving our result.

2.1 Forward chaining procedure.

In verifying that a given clause is an implicate of a given pure Horn function, a very useful and simple procedure is the following. Let $\mathcal{F}$ be a pure Horn CNF of a pure Horn function f. We shall define a *forward chaining* procedure which associates to any subset Q of the propositional letters of f a set R in the following way. The procedure takes as input the subset Q of propositional letters, initializes the set $R = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in $\mathcal{F}$ such that $S \subseteq R$, and $y \notin R$. If such a clause is found, the propositional letter y is included into R, and the search is repeated as many times as possible.

Forward Chaining Procedure

Input:	A pure Horn CNF $\mathcal{F} = \bigwedge_{i=1}^{m}(S_i \vee x_{j_i})$ and a subset Q of the propositional letters.
Initialization:	Set $R = Q$.
Main Step:	while $\exists\ i$ such that $S_i \subset R$ and $x_{j_i} \notin R$ do $R = R \cup \{x_{j_i}\}$.
Stop:	Output $R_{\mathcal{F}}(Q) = R$.

In a relational database terminology the propositional letters in $R_{\mathcal{F}}(Q)$ are said to be "chained" to the subset Q (see e.g. [13]). In an expert system terminology the usage of the clause $S \vee y$ is called "firing the rule" $\bigwedge_{x \in S} x \longrightarrow y$ (see e.g. [21]).

The following lemma (proved in [22]) will be frequently used in the rest of this text.

Lemma 2.2 *A pure Horn clause $Q \vee y$ is an implicate of a pure Horn function f given by a pure Horn CNF $\mathcal{F}$ if and only if $y \in R_{\mathcal{F}}(Q)$.*

Lemma 2.2 shows how to verify whether a given clause is an implicate of a given CNF. The following statement proved in [23] shows that the forward chaining procedure can be also efficiently implemented, i.e. that the complexity of performing the above mentioned verification is low.

Lemma 2.3 *Given a pure Horn CNF $\mathcal{F}$ and a subset of Q of the propositional letters used in $\mathcal{F}$, the set $R_{\mathcal{F}}(Q)$ can be determined in $O(|\mathcal{F}|_\ell)$ time.*

Obviously the complexity guaranteed by Lemma 2.3 is the best possible since any implementation of the forward chaining procedure must at least read the entire input CNF. Let us conclude this section with a notational remark.

If $\mathcal{F}_1$ and $\mathcal{F}_2$ are two distinct CNF representations of a given pure Horn function f and if Q is an arbitrary subset of the propositional letters then by Lemma 2.2 $R_{\mathcal{F}_1}(Q) = R_{\mathcal{F}_2}(Q)$ because $\mathcal{F}_1$ and $\mathcal{F}_2$ have the same set of implicates. Therefore the set of propositional letters reachable from Q by forward chaining depends only on the underlying function rather than on a particular CNF representation. For this reason we shall throughout the text use the expression $R_f(Q)$ instead of $R_{\mathcal{F}}(Q)$ whenever we do not want to refer to a specific CNF.

3 Pure Horn functions corresponding to hypergraphs.

The forthcoming reduction (proof of NP-hardness) will use the following correspondence between hypergraphs and pure Horn functions. Let $\mathcal{H}$ be an arbitrary hypergraph on a vertex set X containing n vertices and let us identify each vertex of $\mathcal{H}$ with a Boolean propositional letter x_i, i.e. let $X = \{x_1, x_2, ..., x_n\}$. Let us consider a set of additional Boolean propositional letters, $T = \{t_1, t_2, ..., t_n\}$, and let us associate with the hypergraph $\mathcal{H}$ a pure Horn function $f_{\mathcal{H}}$ defined by the following CNF expression

$$\mathcal{F}_{\mathcal{H}} = [\bigwedge_{i=1}^{n-1} (\bar{t}_i \vee t_{i+1}) \wedge (\bar{t}_n \vee t_1)] \wedge [\bigwedge_{i=1}^{n} (\bar{t}_1 \vee x_i) \wedge \bigwedge_{H \in \mathcal{H}} (H \vee t_1)] \tag{3}$$

Note that the first part of the above formula is in fact a cycle of implications over propositional letters $t_1, \ldots, t_n$, thus establishing an equivalence among them. Therefore, the propositional letters $t_1, \ldots, t_n$ are freely interchangeble in the second part of the formula and it does not matter which t_i's appear there (all such formulas are equivalent, i.e. represent the same function). For instance in the CNF (3) only the propositional letter t_1 is used.

It is easy to see that the CNF (3) is not a minimum one. Let H_1 be an arbitrary hyperedge and let us replace the clause $H_1 \vee t_1$ with $H_1 \vee t_2$. The new clause together with the clauses $\bar{t}_1 \vee x_i,\ x_i \in H_1$ imply the clause $\bar{t}_1 \vee t_2$ and therefore $\bar{t}_1 \vee t_2$ can be deleted from the formula thus reducing the number of literals by two. Similarly, if H_2 is an arbitrary hyperedge disjoint from H_1 and we replace $H_2 \vee t_1$ with $H_2 \vee t_3$ and $\bar{t}_1 \vee x_i$ with $\bar{t}_2 \vee x_i$ for each $x_i \in H_2$, the clause $\bar{t}_2 \vee t_3$ can be eliminated from the formula. This suggests a heuristic approach to minimize the function $f_{\mathcal{H}}$: Find as many pairwise disjoint hyperedges as possible (i.e. a maximum matching on $\mathcal{H}$ would perhaps be the best) and by the above procedure eliminate for each hyperedge in this matching one arc (quadratic clause) from the cycle on the propositional letters in T. The somewhat surprising fact is that this is the best one could do, at least in case of certain hypergarphs. In other words, given a maximum matching on $\mathcal{H}$ one can construct a minimum CNF of $f_{\mathcal{H}}$ in this way, and vice versa, from a minimum CNF of $f_{\mathcal{H}}$ one can deduce a maximum matching of the hypergraph. How this mutual correspondence works precisely will be shown later on.

4 Prime Implicates of $f_{\mathcal{H}}$

Let us consider the pure Horn CNF $\mathcal{F}_{\mathcal{H}}$ given by (3) associated to the hypergraph $\mathcal{H}$ and let us moreover assume that $\mathcal{H}$ has the Sperner's property (i.e. no two hyperedges are comparable by inclusion). In such a case it is easy to see that the CNF (3) is prime, i.e. that dropping a literal from any of the clauses of $\mathcal{F}_{\mathcal{H}}$ would produce a clause which is not an implicate of $f_{\mathcal{H}}$. The following lemma characterizes all prime implicates of $f_{\mathcal{H}}$.

Lemma 4.1 *Let $\mathcal{H}$ be a hypergraph with the Sperner's property and let us consider the associated function $f_{\mathcal{H}}$ defined by the CNF (3). An elementary disjunction C is a prime implicate of $f_{\mathcal{H}}$ iff one of the following is true :*

I) *$C = \bar{t}_i \vee t_j$ for some $t_i, t_j \in T$ such that $i \neq j$; or*

II) *$C = \bar{t}_k \vee x_i$ for some $t_k \in T$ and $x_i \in X$; or*

III) *$C = H \vee t_k$ for some $H \in \mathcal{H}$ and $t_k \in T$; or*

IV) *$C = H \vee x_i$ for some $H \in \mathcal{H}$ and $x_i \in X$ such that $x_i \notin H$*

Proof. One can observe that

$$\begin{aligned} R_{\mathcal{F}_{\mathcal{H}}}(\{t_i\}) &= T \cup X && \text{for every } t_i \in T, \text{ and} \\ R_{\mathcal{F}_{\mathcal{H}}}(H) &= T \cup X && \text{for every } H \in \mathcal{H}, \text{ and} \\ R_{\mathcal{F}_{\mathcal{H}}}(S) &= S && \text{for every } S \subseteq X \text{ such that } \nexists H \in \mathcal{H} \text{ for which } H \subseteq S. \end{aligned} \tag{4}$$

This implies, according to Lemma 2.2 that every clause of type I) - IV) is an implicate of $f_{\mathcal{H}}$, and moreover that no truncation of such an implicate yields an implicate. Thus all such clauses are indeed prime implicates of $f_{\mathcal{H}}$.

On the other hand, if C is a prime implicate of $f_{\mathcal{H}}$, then it must be pure Horn, since all prime implicates of (the pure Horn function) $f_{\mathcal{H}}$ must be pure Horn (see e.g. [20]). Thus we may assume that $C = Q \vee y$ for some $Q \subseteq T \cup X$ and $y \in T \cup X$ such that $y \notin Q$. Then $R_{\mathcal{F}_{\mathcal{H}}}(Q) \neq Q$ (since it must contain y), and thus by (4) either some $t_i \in T$ or an hyperedge $H \in \mathcal{H}$ must be contained in Q. The primeness of C thus implies in the first case that $C = \bar{t}_i \vee y$, i.e. that C is of category I) or II) depending on whether $y \in T$ or $y \in X$, and in the second case that $C = H \vee y$, and so C is of category III) or IV). ■

5 Properties of $f_{\mathcal{H}}$.

Let us now state the hypergraph matching problem (see e.g. [18]) that we shall use for the reduction. Following standard terminology by a matching on a (undirected) hypergraph we understand a collection of pairwise disjoint hyperedges. The problem can be then stated as follows :

Hypergraph Matching (HM)

Instance: A hypergraph $\mathcal{H}$ and an integer k.
Question: Does $\mathcal{H}$ contain a matching of size at least k?

What unfortunatelly complicates the matter a little bit is that the forthcoming reduction requires the given hypergraph to have certain special properties. First of all, in order to be able to use Lemma 4.1, $\mathcal{H}$ must have Sperner's property. Moreover, we will require that every hyperedge contains exactly four vertices and that the difference of any two (different) hyperedges contains at least two vertices. More formally speaking, the hypergraph $\mathcal{H}$ must satisfy the following three properties :

$$\begin{aligned} &1.\quad \forall H_1 \neq H_2 \in \mathcal{H} \,:\, H_1 \not\subseteq H_2;\\ &2.\quad \forall H \in \mathcal{H} \,:\, |H| = 4; \text{and}\\ &3.\quad \forall H_1 \neq H_2 \in \mathcal{H} \,:\, |H_1 \setminus H_2| \geq 2. \end{aligned} \tag{5}$$

The requirement that every hyperedge of $\mathcal{H}$ has cardinality exactly four will guarantee that the corresponding CNF $\mathcal{F}_{\mathcal{H}}$ defined by (3) has degree at most five. The problem that is actually used in the forthcoming reduction can be stated as follows :

Special Hypergraph Matching (SHM)

Instance: A hypergraph $\mathcal{H}$ satisfying properties (5) and an integer k.
Question: Does $\mathcal{H}$ contain a matching of size at least k?

It is quite easy to see that SHM is NP-complete. First of all SHM is a special case of HM and therefore it is indeed in NP. Secondly, HM stays NP-complete even if no hyperedge has cardinality more than three and less than two, i.e. if $\mathcal{H}$ is at most cubic with no singletons (see [18]). Now given an instance $(\mathcal{H}, k)$ of HM one can construct (in polynomial time) an instance $(\mathcal{H}', k)$ of SHM such that $\mathcal{H}$ contains a matching of size at least k if and only if $\mathcal{H}'$ does. If $\mathcal{H}$ is not Sperner's (does not fulfil Property 1 in (5)) we can simply delete every hyperedge H_1 which is a superset of another hyperedge H_2 since if H_1 appears in a matching it can be replaced by its subset H_2 without violating the pairwise disjointness of the matching or decreasing its size. In order to guarantee the validity of Properties 2 or 3 in (5) construct $\mathcal{H}'$ by adding two extra vertices (not present in $\mathcal{H}$) to each hyperedge H of size two and one extra vertex to each H of size three. Now we can observe, that all hyperedges in $\mathcal{H}'$ have the same size four as required by Property 2. Since we assume that $\mathcal{H}$ has Sperner's property, $\mathcal{H}'$ must have it as well, so $\mathcal{H}'$ fulfils also Property 1. To verify Property 3 for $\mathcal{H}'$ consider two different hyperedges H_1 and H_2 in $\mathcal{H}$ and their modifications H_1' and H_2' in $\mathcal{H}'$ respectively. Due to Sperner's property of $\mathcal{H}$ we have $|H_1 \setminus H_2| \geq 1$. Due to the construction of $\mathcal{H}'$, the modified hyperedge H_1' contains at least one extra vertex which is not present in H_1, H_2, and H_2', and so we get $|H_1' \setminus H_2'| \geq 2$ as is required by Property 3.

There is an obvious one-to-one correspondence between hyperedges in $\mathcal{H}$ and $\mathcal{H}'$. This correspondence clearly carries over to an one-to-one correspondence between matchings on $\mathcal{H}$ and $\mathcal{H}'$, because two hyperedges in $\mathcal{H}$ are disjoint if and only if their corresponding "images" in $\mathcal{H}'$ are disjoint. The next proposition shows the correspondence between MNL and SHM.

Proposition 5.1 *Let $\mathcal{H}$ be a hypergraph on n vertices satisfying (5) and let k be a positive integer. The associated pure Horn function $f_\mathcal{H}$ has a CNF representation consisting of at most $\ell = \sum_{H\in\mathcal{H}}(|H|+1) + 4n - 2k$ if and only if $\mathcal{H}$ has a matching of size at least k.*

The proof of Proposition 5.1 consists of several easy lemmas. Since the proof is constructive (i.e. given a matching of size at least k one can construct in a polynomial time the corresponding CNF of size at most ℓ and vice versa) it gives the desired reduction proving Theorem 1. Indeed, if there was a polynomial time algorithm for finding a minimum CNF of an arbitrary pure Horn function of degree at most five, one could solve the special hypergraph matching (SHM) problem (in polynomial time) as follows. Given $\mathcal{H}$ construct the (pure Horn) CNF $\mathcal{F}_\mathcal{H}$, find its minimum CNF $\mathcal{F}$, count the number of literals in $\mathcal{F}$, and compare it to the above defined number ℓ.

First we shall show that given an instance of SHM all minimum CNFs of $f_\mathcal{H}$ fulfil certain properties. Next we shall show that among the minimum CNFs there is always (at least) one which obeys some further properties and moreover that one can get to such a CNF in polynomial time from any minimum CNF.

We shall say that a pure Horn clause $Q \vee y$ is *derivable* from a CNF $\mathcal{F}$ iff $y \in R_\mathcal{F}(Q)$, or in other words if it is an implicate of the function defined by $\mathcal{F}$.

Let $\mathcal{F}'$ be an arbitrary pure Horn CNF of a pure Horn function f. A pure Horn CNF $\mathcal{F}''$ is called a *feasible modification* of $\mathcal{F}'$ if on one hand (a) all clauses of $\mathcal{F}''$ which do not appear in $\mathcal{F}'$ are prime implicates of f and, on the other hand, (b) all clauses of $\mathcal{F}'$ which are not in $\mathcal{F}''$ are derivable from $\mathcal{F}''$, and finally if (c) $\mathcal{F}'$ and $\mathcal{F}''$ contain the same number of literals. Properties (a) and (b) ensure that $\mathcal{F}'$ and $\mathcal{F}''$ are both defining the same function f, and (c) guarantees that if $\mathcal{F}'$ is a minimum CNF of the function f then so is $\mathcal{F}''$. Finally, $\mathcal{F}''$ is called a *reducing modification* of $\mathcal{F}'$ if (a) and (b) hold as before but this time $\mathcal{F}''$ contains strictly less literals than $\mathcal{F}'$. In this case existence of $\mathcal{F}''$ contradicts the minimality of $\mathcal{F}'$.

6 Minimum CNFs of $f_\mathcal{H}$.

Now we are ready to prove the properties of minimum CNFs of $f_\mathcal{H}$.

Lemma 6.1 *Let $\mathcal{F}_0$ be an arbitrary minimum CNF of $f_\mathcal{H}$. Then*

a) *$\mathcal{F}_0$ contains no clause of type $H \vee x_i$ (category IV); and*

b) *$\mathcal{F}_0$ contains exactly one clause $H \vee t_k$ (category III) for each $H \in \mathcal{H}$.*

Proof. a) Let us assume that $\mathcal{F}_0$ contains a clause $H_r \vee x_i$. That means that $\mathcal{F}_0$ contains no clause $H_r \vee t_k,\ k \in \{1, \ldots, n\}$ since otherwise adding the clause $\bar{t}_k \vee x_i$ and removing $H_r \vee x_i$ would be a reducing modification of $\mathcal{F}_0$ because H_r contains at least 3 literals. However, $H_r \vee t_k$ is an implicate of $f_{\mathcal{H}}$ (for every $k \in \{1, \ldots, n\}$) and so it must be derivable from $\mathcal{F}_0$. Actually since $\mathcal{F}_0$ and $\mathcal{F}_{\mathcal{H}}$ are equivalent $R_{\mathcal{F}_0}(H_r) = R_{\mathcal{F}_{\mathcal{H}}}(H_r) = T \cup X$ must hold. Therefore $\mathcal{F}_0$ must contain clauses $H_r \vee x_{i_1}, H_r \vee x_{i_2}, \ldots, H_r \vee x_{i_c}$ such that $\{x_{i_1}, x_{i_2}, \ldots, x_{i_c}\} = H_s \backslash H_r$ for some hyperedge $H_s \in \mathcal{H}$ different from H_r since otherwise $R_{\mathcal{F}_0}(H_r) = H_r$. Let us denote the number of literals in H_r by d. By assumption on $\mathcal{H}$, $c \geq 2,\ d \geq 4$ holds. Now let us modify $\mathcal{F}_0$ by removing the clauses $H_r \vee x_{i_1}, H_r \vee x_{i_2}, \ldots, H_r \vee x_{i_c}$ and adding $\bar{t}_k \vee x_{i_1}, \bar{t}_k \vee x_{i_2}, \ldots, \bar{t}_k \vee x_{i_c}$ and $H_r \vee t_k$ (excluding those clauses that are already present in $\mathcal{F}_0$). Since all the deleted clauses can be derived from the added ones and all the added ones are prime implicates of $f_{\mathcal{H}}$ by Lemma 4.1, the above modification satisfies feasibility conditions (a) and (b). The number of removed literals is $c(d+1)$ while the number of added literals is at most $2c+d+1$. However, since $c(d+1) > 2c+d+1$ (which can be rewritten as $(c-1)(d-1) > 2$) always holds for $c \geq 2,\ d \geq 4$, the described modification is reducing which is a contradiction with the minimality of $\mathcal{F}_0$.

b) If $\mathcal{F}_0$ contains no clause $H \vee t_k,\ k \in \{1, \ldots, n\}$ for a given $H \in \mathcal{H}$, then it must contain a clause $H \vee x_i$ for some x_i since otherwise $R_{\mathcal{F}_0}(H) = H$ instead of $R_{\mathcal{F}_0}(H) = T \cup X$. However, this cannot happen due to the part a) of this lemma. Therefore $\mathcal{F}_0$ contains at least one clause $H \vee t_k$ for some $k \in \{1, \ldots, n\}$. On the other hand let us assume that $\mathcal{F}_0$ contains clauses $H \vee t_k$ and $H \vee t_\ell$ where $k \neq \ell$. This cannot happen as well since adding the clause $\bar{t}_k \vee t_\ell$ and removing $H \vee t_\ell$ would again be a reducing modification. Thus $\mathcal{F}_0$ contains exactly one clause $H \vee t_k,\ k \in \{1, \ldots, n\}$. ■

Given an arbitrary minimum CNF $\mathcal{F}_0$ of the function $f_{\mathcal{H}}$ (which necessarily possesses the properties of Lemma 6.1 it can be modified as follows.

Modification : If $\mathcal{F}_0$ contains clauses $\bar{t}_k \vee x_i$ and $\bar{t}_\ell \vee x_i$ where $k \neq \ell$ then replace $\bar{t}_\ell \vee x_i$ by $\bar{t}_\ell \vee t_k$ and repeat this step as long as there exists $x_i \in X$ which is contained in at least two distinct clauses of category II. Call the resulting CNF $\mathcal{F}_1$ (note that $\mathcal{F}_1$ is not uniquely defined). Obviously, $\mathcal{F}_1$ is a feasible modification of $\mathcal{F}_0$.

Lemma 6.2 *Let $\mathcal{F}_0$ be an arbitrary minimum CNF of $f_{\mathcal{H}}$ and let $\mathcal{F}_1$ be an arbitrary CNF obtained from $\mathcal{F}_0$ by the described modification. Then the CNF $\mathcal{F}_1$ contains exactly one clause $\bar{t}_k \vee x_i$ (category II) for each $x_i \in X$.*

Proof. Let $x_i \in X$ be arbitrary. Due to the modification there is at most one clause $\bar{t}_k \vee x_i$ present in $\mathcal{F}_1$. On the other hand $\mathcal{F}_1$ contains at least one clause $\bar{t}_k \vee x_i$ since otherwise the literal x_i would not appear in $\mathcal{F}_1$ at all (due to Lemma 6.1 part a) and hence the implicates $\bar{t}_k \vee x_i,\ k \in \{1, \ldots, n\}$ could not be derived from $\mathcal{F}_1$. ■

Given a CNF $\mathcal{F}$ of the function $f_{\mathcal{H}}$ let us denote the number of clauses of category I (i.e. clauses containing only t_is and no x_is) by $|\mathcal{F}|_t$.

Lemma 6.3 *Let $\mathcal{F}_0^a$ and $\mathcal{F}_0^b$ be two arbitrary minimum CNFs of $f_\mathcal{H}$ and let $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ be arbitrary CNFs obtained from $\mathcal{F}_0^a$ and $\mathcal{F}_0^b$ by the described modification. Then $|\mathcal{F}_1^a|_t = |\mathcal{F}_1^b|_t$ and moreover $|\mathcal{F}_0^a|_\ell = \sum_{H\in\mathcal{H}}(|H|+1) + 2n + 2|\mathcal{F}_1^a|_t$.*

Proof. We can count the numbers of clauses in of categories I - IV as follows:

- By Lemma 6.1 part a) both $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ contain no clauses of category IV.
- By Lemma 6.1 part b) both $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ contain exactly one clause of category III for each $H \in \mathcal{H}$ and hence the number of category III clauses as well as the total number of literals in them is the same for both CNFs (namely $|\mathcal{H}|$ and $\sum_{H\in\mathcal{H}}(|H|+1)$ respectively).
- By Lemma 6.2 both $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ contain exactly one clause of category II for each $x_i \in X$ and hence the number of category II clauses as well as the total number of literals in them is the same for both CNFs (namely $|X| = n$ and $2n$ respectively).
- Since both $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ are minimum CNFs of $f_\mathcal{H}$ (and hence contain the same number of literals) and since both contain the same total number of literals in clauses of categories II, III, and IV, the CNFs $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ also necessarily contain the same number of literals in clauses of category I. However, each category I clause contains exactly two literals and therefore $\mathcal{F}_1^a$ and $\mathcal{F}_1^b$ contain the same number of clauses of category I.
- The fact that $|\mathcal{F}_0^a|_\ell = \sum_{H\in\mathcal{H}}(|H|+1) + 2n + 2|\mathcal{F}_1^a|_t$ follows trivially from the above facts.

■

Let $\mathcal{F}_0$ be an arbitrary minimum CNF of $f_\mathcal{H}$ and let $\mathcal{F}_1$ be an arbitrary CNF obtained from $\mathcal{F}_0$ by the described modification. Then we may define a directed graph $D = (T, E)$ by

$$T = \{t_1, \dots, t_n\}, \text{ and } E = \{(t_k, t_\ell) \mid \bar{t}_k \vee t_\ell \text{ is a clause in } \mathcal{F}_1\}$$

and denote its connected components by $C_1, C_2, \dots, C_q$. Due to Lemma 6.2 we may also define a *predecessor function* $p : X \longrightarrow T$ where

$$p(x_i) = t_k \text{ if and only if } \bar{t}_k \vee x_i \text{ is a clause in } \mathcal{F}_1.$$

Now the following lemma holds.

Lemma 6.4 *If $q \geq 2$ (i.e. if D has at least two components) then for each component C_ℓ of the digraph D, $\ell \in \{1, \dots, q\}$, there exists a hyperedge $H \in \mathcal{H}$ such that*

$$\forall x_i \in H \ : \ p(x_i) \in C_\ell$$

and a clause $H \vee t_p$ in $\mathcal{F}_1$ such that $t_p \notin C_\ell$.

Proof. Let C_ℓ be an arbitrary component of D. We know that $R_{\mathcal{F}_1}(C_\ell) = T \cup X$ (by Lemma 4.1 since C_ℓ contains at least one $t_i \in T$). Let t_p be the first vertex from $T \setminus C_\ell$ to be included into $R_{\mathcal{F}_1}(C_\ell)$. The only way how t_p can be included is by using a clause $H \vee t_p$ in $\mathcal{F}_1$ such that $\forall x_i \in H \;:\; p(x_i) \in C_\ell$ holds. ■

The following lemma will conclude the proof of Proposition 5.1.

Lemma 6.5 *Let $\mathcal{F}_0$ be an arbitrary minimum CNF of $f_\mathcal{H}$ and let $\mathcal{F}_1$ be an arbitrary CNF obtained from $\mathcal{F}_0$ by the described modification. Then the hypergraph H contains a matching of size at least k if and only if $|\mathcal{F}_1|_t \leq n - k$.*

Proof. Let $H_{s_1}, H_{s_2}, \ldots, H_{s_p}$ where $p \geq k$ be a matching on $\mathcal{H}$. Let us denote $\mathcal{H}_R = \mathcal{H} \setminus \{H_{s_1}, H_{s_2}, \ldots, H_{s_p}\}$ and $X_R = X \setminus \bigcup_{\ell=1}^{p} H_{s_\ell}$. Now let us define

$$\tilde{\mathcal{F}} = [\bigwedge_{i=p+1}^{n-1} (\bar{t}_i \vee t_{i+1}) \wedge (\bar{t}_n \vee t_1)] \wedge [\bigwedge_{\ell=1}^{p} (\bigwedge_{x_i \in H_{s_\ell}} (\bar{t}_\ell \vee x_i)) \wedge (H_{s_\ell} \vee t_{\ell+1})] \wedge [\bigwedge_{x_i \in X_R} (\bar{t}_1 \vee x_i)] \wedge [\bigwedge_{H \in \mathcal{H}_R} (H \vee t_1)]$$

Note that for each $\ell \in \{1, \ldots, p\}$ the second part of $\tilde{\mathcal{F}}$ implies the clause $\bar{t}_\ell \vee t_{\ell+1}$ and therefore $\tilde{\mathcal{F}}$ is a CNF of $f_\mathcal{H}$. Note also that $\tilde{\mathcal{F}}$ contains no clause of category IV, exactly one clause of category III for each $H \in \mathcal{H}$, exactly one clause of category II for each $x_i \in X$, and exactly $n - p$ clauses of category I. Now let $\mathcal{F}_0$ be an arbitrary minimum CNF of $f_\mathcal{H}$ and let $\mathcal{F}_1$ be an arbitrary CNF obtained from $\mathcal{F}_0$ by the described modification. By considerations similiar to the ones in the proof of Lemma 6.3 it can be seen that $|\mathcal{F}_1|_t \leq |\tilde{\mathcal{F}}|_t$ and therefore $|\mathcal{F}_1|_t \leq n - p \leq n - k$ which finishes the proof of the first implication.

Let $\mathcal{F}_0$ be an arbitrary minimum CNF of $f_\mathcal{H}$ and let $\mathcal{F}_1$ be its arbitrary modification such that $|\mathcal{F}_1|_t \leq n-k$. This is equivalent to saying that the digraph D that corresponds to $\mathcal{F}_1$ has at most $n-k$ arcs which in turn implies that D has at least k components, say $C_1, C_2, \ldots, C_q$, where $q \geq k$. The case $q = k = 1$ is trivial since we may assume that H is nonempty and hence contains a matching of size 1. So let us assume $q \geq 2$. Now using Lemma 6.4 (subsequently for each component of D) there exist hyperedges $H_{s_1}, H_{s_2}, \ldots, H_{s_q} \in \mathcal{H}$ such that

$$\forall \ell \in \{1, \ldots, q\}\ \forall x_i \in H_{s_\ell} : p(x_i) \in C_\ell.$$

Therefore $H_{s_1}, H_{s_2}, \ldots, H_{s_q}$ are pairwise disjoint and so they constitute a matching on H of size $q \geq k$. ■

Now Proposition 5.1 follows from Lemma 6.5 and Lemma 6.3 since for every minimum CNF $\mathcal{F}_0$ of the function $f_\mathcal{H}$ the inequality $|\mathcal{F}_0^a|_\ell \leq \sum_{H \in \mathcal{H}}(|H| + 1) + 4n - 2k$ holds if and only if the corresponding CNF $\mathcal{F}_1$ has $|\mathcal{F}_1|_t \leq n - k$.

References

[1] V. Arvind and S. Biswas. An $O(n^2)$ algorithm for the satisfiability problem of a subset of propositional sentences in CNF that includes all Horn sentences. *Information Processing Letters*, **24**(1987), 67–69.

[2] A.V. Aho, M.R. Garey and J.D. Ullman. The transitive reduction of a directed graph. *SIAM J.Comput.*, **1**(1972), 131–137.

[3] G. Ausiello, A. D'Atri and D. Sacca. Minimal Representation of Directed Hypergraphs. *SIAM J.Comput.*, **15**(1986), 418–431.

[4] E. Boros, Y. Crama, and P.L. Hammer. Polynomial time inference of all valid implications for Horn and related formulae. *Annals of Mathematics and Artificial Intelligence*, **8**(1990), 121–123.

[5] E. Boros and O. Čepek. On the Complexity of Horn Minimization. *RUTCOR Research Report RRR 1-94*, Rutgers University, New Brunswick, NJ, January 1994.

[6] E. Boros, P.L. Hammer, and X. Sun. Recognition of q-Horn formulae in linear time. *Discrete Applied Mathematics*, **55**(1994), 1–13.

[7] O. Čepek. Restricted consensus method and quadratic implicates of pure Horn functions. *RUTCOR Research Report RRR 31-94*, Rutgers University, New Brunswick, NJ, September 1994.

[8] O. Čepek. *Structural Properties and Minimization of Horn Boolean Functions. Doctoral dissertation*, Rutgers University, New Brunswick, NJ, October 1995.

[9] V. Chandru and J.N. Hooker. Extended Horn Sets in Propositional Logic. *Journal of the ACM*, **38**(1991), 205–221.

[10] V. Chvátal. On certain polytopes associated with graphs. *Journal of Combinatorial Theory B*, **18**(1975), 138–154.

[11] M. Conforti, G. Cornuéjols, and C. De Francesco. Perfect 0,±1 matrices. *Preprint*, Carnegie Mellon University, 1993.

[12] S.A.Cook. The complexity of theorem-proving procedures. *Proc. 3rd Ann. ACM Symp. on Theory of Computing*, Association for Computing Machinery, New York, (1971), 151–158.

[13] C. Delobel and R.G. Casey. Decomposition of a Data Base and the Theory of Boolean Switching Functions. *IBM J.Res.Develop.*, **17**(1973), 374–386.

[14] R. Dechter and J. Pearl. Structure identification in relational data. *Artificial Intelligence* **58**(1992), 237-270.

[15] W.F. Dowling and J.H. Gallier. Linear time algorithms for testing the satisfiability of propositional Horn formulae. *Journal of Logic Programming*, **3**(1984), 267–284.

[16] D.R. Fulkerson. Anti-blocking polyhedra. *Journal of Combinatorial Theory B*, **12**(1972), 50–71.

[17] G. Gallo and M.G. Scutella. Polynomially soluble satisfiability problems. *Information Processing Letters*, **29**(1988), 221–227.

[18] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, San Francisco, 1979.

[19] M.R. Garey, D.S. Johnson and R.E. Tarjan. The planar Hamiltonian circuit problem is NP-complete. *SIAM J.Comput.*, **5**(1976), 704–714.

[20] P.L. Hammer and A. Kogan. Horn functions and their DNFs. *Information Processing Letters*, **44**(1992), 23–29.

[21] P.L. Hammer and A. Kogan. Horn Function Minimization and Knowledge Compression in Production Rule Bases. *RUTCOR Research Report RRR 8-92*, Rutgers University, New Brunswick, NJ, March 1992.

[22] P.L. Hammer and A. Kogan. Optimal Compression of Propositional Horn Knowledge Bases: Complexity and Approximation. *Artificial Intelligence*, **64**(1993), 131–145.

[23] P.L. Hammer and A. Kogan. Quasi-Acyclic Propositional Horn Knowledge Bases : Optimal Compression. *RUTCOR Research Report RRR 10-93*, Rutgers University, New Brunswick, NJ, September 1993.

[24] P.L. Hammer and A. Kogan. Knowledge compression – logic minimization for expert systems. *Computers As Our Better Partners. Proceedings of the IISF/ACM Japan International Symposium*. World Scientific, Singapore, 1994, 306-312.

[25] A. Itai and J.A. Makowsky. Unification as a complexity measure for logic programming. *Journal of Logic Programming*, **4**(1987), 105-117.

[26] D.E. Knuth. Fundamental Algorithms. *Vol.1 of The Art of Computer Programming*, Addison-Wesley, 1968 (second edition 1973).

[27] L. Lovász. Normal hypergraphs and the perfect graph conjecture. *Discrete Mathematics* **2**(1972), 253–267.

[28] L. Lovász. Normal hypergraphs and the weak perfect graph conjecture. *Annals of Discrete Mathematics* **21**(1984), 29–42.

[29] D. Maier. Minimal Covers in the Relational Database Model. *Journal of the ACM*, **27**(1980), 664–674.

[30] M. Minoux. LTUR: A simplified linear time unit resolution algorithm for Horn formulae and computer implementation. *Information Processing Letters*, **29**(1988), 1-12.

[31] W. Quine. The problem of simplifying the truth functions. *Amer.Math.Monthly*, **59**(1952), 521–531.

[32] W. Quine. A way to simplify truth functions. *Amer.Math.Monthly*, **62**(1955), 627–631.

[33] R.E. Tarjan. Depth first search and linear graph algorithms. *SIAM Journal on Computing*, **2**(1972), 146–160.

[34] S. Yamasaki and S. Doshita. The Satisfiability Problem for a Class Consisting of Horn Sentences and Some Non-Horn Sentences in Propositional Logic. *Information and Control*, **59**(1983), 1–12.