RUTCOR
RESEARCH
REPORT

A subclass of Horn CNFs optimally compressible in polynomial time.

Endre Boros[a] Ondřej Čepek[b] Alexander Kogan[c]
Petr Kučera[d]

RRR 11-2009, June 2009

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]RUTCOR, Rutgers University, P.O. Box 5062, New Brunswick, NJ 08903, USA (boros@rutcor.rutgers.edu)

[b]Department of Theoretical Computer Science and Mathematical Logic, Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic and Institute of Finance and Administration - VŠFS, Estonská 500, 100 00, Praha 10, Czech Republic (cepek@ksi.ms.mff.cuni.cz)

[c]Department of Accounting, Business Ethics, and Information Systems, Rutgers Business School, Rutgers University, Newark, NJ 07102, and RUTCOR, Rutgers University, P.O. Box 5062, New Brunswick, NJ 08903, USA (kogan@rutgers.edu)

[d]Department of Theoretical Computer Science and Mathematical Logic, Charles University, Malostranské nám. 25, 118 00 Praha 1, Czech Republic (kucerap@ktiml.mff.cuni.cz)

Rutcor Research Report
RRR 11-2009, June 2009

A subclass of Horn CNFs optimally compressible in polynomial time.

Endre Boros Ondřej Čepek Alexander Kogan Petr Kučera

Abstract. The problem of Horn Minimization (HM) can be stated as follows: given a Horn CNF representing a Boolean function f, find a CNF representation of f which consists of the minimum possible number of clauses. This problem is the formalization of the problem of knowledge compression for speeding up queries to propositional Horn expert systems, and it is known to be NP-hard. There are two subclasses of Horn functions for which HM is known to be solvable in polynomial time: acyclic and quasi-acyclic Horn functions. In this paper we define a new class of Horn functions properly containing both of the known classes and design a polynomial time HM algorithm for this new class.

Acknowledgements: The second author gratefully acknowledges the support by the Czech science Foundation (grant 201/07/0205). The fourth author gratefully acknowledges the support by the Czech Science Foundation (grant 201/07/P168).

1 Introduction

Horn functions are 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. [10]) is solvable in linear time for Horn formulae [8, 17, 20]. 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 [6, 14, 15] and database design [7, 19]. Horn clauses constitute a very 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 [14, 15]. 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 is NP-hard not only in the general case, but also for Horn CNFs [1, 2, 5, 14, 19]. The Horn Minimization (HM) problem can be stated as follows: given a Horn CNF ϕ find a CNF ϕ' representing the same function and such that ϕ' consists of the minimum possible number of clauses. Paper [16] introduced two subclasses of Horn functions, acyclic and quasi-acyclic functions, for which HM is solvable in polynomial time.

In the present paper we shall introduce another subclass of Horn functions which properly contains both of the above subclasses and develop a polynomial time algorithm solving HM for the new class. The correctness of this algorithm heavily depends on nontrivial results about certain sets of implicates of Boolean functions proved in [4].

The paper is structured as follows. In Section 2 we introduce the necessary notation and present several elementary results important for the subsequent presentation. In Section 3 we describe the classes of Horn functions for which HM is known to be solvable in polynomial time, and define a new class of Horn functions (component-wise quadratic or simply CQ functions), as well as prove some basic properties of this new class. Section 4 studies certain sets of implicates of (general) Boolean functions. We recall several key properties of these sets (proved in [4]), which are needed in the remainder of the paper. In Section 5 we return to the study of Horn functions. We associate with each Horn function two different directed graphs and derive several useful properties which these graphs possess. Finally, Section 6

contains the main result of this paper, namely the polynomial time HM algorithm for CQ functions, a proof of its correctness, and an upper bound on its time complexity.

2 Basic Notation, Definitions, and Results

In this section we shall introduce the necessary notation and summarize the basic known results that will be needed later in the text. The first subsection will present some basic facts about (general) Boolean functions and about the subclass of Horn functions. The second subsection will introduce a so called "forward chaining procedure" which constitutes a very useful tool for the study of Horn functions. Finally, the third subsection will present the standard graph terminology that will be used throughout this paper.

2.1 Boolean functions

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

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

is called a *clause*, if every propositional variable 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

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

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 a function f if $f \leq C$. An implicate C 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 that function.

It is a well-known fact that every Boolean function f can be represented by a conjunction of clauses (see e.g. [11]). 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 variables). If two distinct CNFs, say ϕ_1 and ϕ_2, represent the same function, we say that they are *equivalent*, and denote this fact by $\phi_1 \equiv \phi_2$. A CNF ϕ representing a function f is called *prime* if each clause of ϕ is a prime implicate of the function f. A CNF ϕ representing

a function f is called *irredundant* if dropping any clause from ϕ produces a CNF that does not represent f.

For example, in the CNF

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_2 \vee \overline{x}_3 \vee x_4) \wedge (\overline{x}_3 \vee \overline{x}_4)$$

the 2nd clause can be dropped (although it is prime), and the 4th clause can be shortened (i.e. it is not prime). In fact, the same (Horn) function can be represented by the CNF

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_1 \vee x_4) \wedge (\overline{x}_2 \vee \overline{x}_3) \wedge (\overline{x}_3 \vee \overline{x}_4)$$

which is both prime and irredundant.

Two clauses C_1 and C_2 are said to be *resolvable* if they contain exactly one complementary pair of literals, i.e. if there exists exactly one propositional variable that appears uncomplemented in one of the clauses and complemented in the other. That means that we can write $C_1 = \tilde{C}_1 \vee x$ and $C_2 = \tilde{C}_2 \vee \overline{x}$ for some propositional variable x and clauses $\tilde{C}_1$ and $\tilde{C}_2$ which contain no complementary pair of literals. The clauses C_1 and C_2 are called *parent clauses* and the disjunction $R(C_1, C_2) = \tilde{C}_1 \vee \tilde{C}_2$ is called the *resolvent* of the parent clauses C_1 and C_2. Note that the resolvent is a clause (does not contain a propositional variable and its negation). The following is an easy lemma.

Lemma 2.1 *Let C_1 and C_2 be two resolvable implicates of a Boolean function f. Then $R(C_1, C_2)$ is also an implicate of f.*

Resolutions have a very important property usually called the *completeness of resolution*. Sometimes this property is also referred to as the Quine theorem after the author of one of the first papers in which this property was proved [21, 22].

Theorem 2.2 *Let ϕ be a CNF representation of a Boolean function f and let C be a prime implicate of f. Then C can be derived from ϕ by a series of resolutions.*

Throughout this paper we shall also use the following notation. For an arbitrary set of clauses $\mathcal{C}$ the *resolution closure* of $\mathcal{C}$ denoted by $\mathcal{R}(\mathcal{C})$ is the set of all clauses obtainable through series of resolutions from the set $\mathcal{C}$ (allowing the resolvents to become parent clauses in subsequent resolutions).

The following two notational conventions will allow us to switch back and forth between sets of clauses and CNFs. For an arbitrary set of clauses $\mathcal{C}$ the symbol $\phi(\mathcal{C})$ denotes the CNF obtained by taking a conjunction of all clauses in $\mathcal{C}$. On the other hand, for an arbitrary CNF ϕ the symbol $\mathcal{C}(\phi)$ denotes the set of all clauses present in ϕ. We shall use the notion of "representing a given function" interchangeably for both CNFs and sets of clauses, i.e. if a CNF ϕ represents a function f we shall also say that the set of clauses $\mathcal{C}(\phi)$ represents f.

For a Boolean function f let us denote by $\mathcal{I}^p(f)$ the set of its prime implicates, and let $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$. Note that not all implicates of f may belong to $\mathcal{I}(f)$. For instance, if f is

defined by the CNF $\phi = (x_1 \vee x_2) \wedge (x_2 \vee x_3)$, then we have $\mathcal{I}(f) = \mathcal{I}^p(f) = \{(x_1 \vee x_2), (x_2 \vee x_3)\}$, however the clause $(x_1 \vee x_2 \vee x_3)$ is also a implicate of f.

For the purpose of measuring the complexity of algorithms we need ways of measuring the "size" of a given CNF ϕ. The following two definitions are the ones used most commonly in the literature:

- $|\phi|_c = \sum_{C \in \mathcal{C}(\phi)} 1$ (the number of clauses in ϕ),
- $|\phi|_\ell = \sum_{C \in \mathcal{C}(\phi)} |C|$ (the number of literals in ϕ).

For instance for the small CNF $\phi = (x_1 \vee x_2) \wedge (x_2 \vee x_3)$ of the above example, we have $|\phi|_c = 2$ and $|\phi|_\ell = 4$.

In this paper we shall mainly focus on the first measure, though many of our statements hold for the second measure, as well. Let us now turn our attention to the subclass of Boolean functions which is the focus of this paper, i.e. the class of Horn functions.

A clause C defined by (1) is called *negative* if it contains no positive literals (i.e. if $J = \emptyset$). It is called *pure Horn* (or in some literature *definite Horn*) if it contains exactly one positive literal (i.e. if $|J| = 1$). To simplify notation, we shall sometimes write a pure Horn clause $C = \bigvee_{x \in S} \overline{x} \vee y$ simply as $C = S \vee y$. Each propositional variable $x \in S$ is called a *subgoal of* C and the propositional variable y is called the *head of* C.[1] We shall denote $Head(C) = y$, $Subg(C) = S$, and $Vars(C) = S \cup \{y\}$.

A CNF is called *Horn* if it contains only negative and pure Horn clauses. A CNF is called *pure Horn* if it contains only pure Horn clauses. Finally, a Boolean function is called *Horn* if it has at least one representation by a Horn CNF, and similarly a Boolean function is called *pure Horn* if it has at least one representation by a pure Horn CNF.

It is known (see [12]) 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. The next statement was proved in [12].

Theorem 2.3 *Given a Horn CNF ϕ one can find in $O(|\phi|_\ell^2)$ time an irredundant and prime CNF ϕ' equivalent with ϕ.*

We shall use Theorem 2.3 as a (polynomial time) "preprocessing step" which will allow us to make an assumption that the CNF we work with (input CNF) is irredundant and prime (usually only primality will be needed).

2.2 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 η be a pure Horn CNF of a pure Horn function

[1] This terminology comes from the area of artificial intelligence, where the clause C is thought of as a "rule" $S \longrightarrow y$.

h. We shall define a *forward chaining* procedure which associates to any subset Q of the propositional variables of h a set M in the following way. The procedure takes as input the subset Q of propositional variables, initializes the set $M = Q$, and at each step it looks for a pure Horn clause $S \vee y$ in η such that $S \subseteq M$, and $y \notin M$. If such a clause is found, the propositional variable y is included into M, and the search is repeated as many times as possible.

In the relational database terminology the propositional variables in M are said to be "chained" to the subset Q (see e.g. [7]). In the expert system terminology the usage of the clause $S \vee y$ is called "firing the rule" $\bigwedge_{x \in S} x \to y$ (see e.g. [13]).

FORWARD CHAINING PROCEDURE$(\mathcal{C}, Q)$

Input:	A set $\mathcal{C}$ of pure Horn clauses, and a subset Q of propositional variables.
Initialization:	**Set** $M = Q$.
Main Step:	**While** $\exists\, C \in \mathcal{C} : Subg(C) \subseteq M$ and $Head(C) \notin M$ **do** $M = M \cup \{Head(C)\}$.
Stop:	**Output** $FC_{\mathcal{C}}(Q) = M$.

The following lemma, proved in [14], shows how the above procedure can help in determining whether a given clause is an implicate of a given CNF, or not.

Lemma 2.4 *Given a set $\mathcal{C}$ of pure Horn clauses, a subset Q of its propositional variables, and another its variable y, we have $y \in FC_{\mathcal{C}}(Q)$ if and only if $Q \vee y$ is an implicate of a function represented by $\mathcal{C}$.*

In what follows we will frequently refer to CNFs as well as their sets of clauses, and thus for $\mathcal{C} = \mathcal{C}(\eta)$ we shall write both $FC_\eta(Q) = FC_{\mathcal{C}}(Q)$. The following statement, proved in [16], shows that the forward chaining procedure can be efficiently implemented, i.e. that the complexity of performing the above mentioned verification is low.

Lemma 2.5 *Given a pure Horn CNF η and a subset Q of its propositional variables, the set $FC_\eta(Q)$ can be determined in $O(|\eta|_\ell)$ time.*

Obviously, the complexity guaranteed by Lemma 2.5 is asymptotically the best possible one, 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 η' and η'' are two distinct CNF representations of a given pure Horn function h and if Q is an arbitrary subset of the propositional variables, then by Lemma 2.4 $FC_{\eta'}(Q) = FC_{\eta''}(Q)$ because η' and η'' have the same set of implicates. Therefore, the set of propositional variables reachable from Q by forward chaining depends only on the underlying function rather than on a particular CNF representation. For this reason, we shall also use the expression $FC_h(Q)$ instead of $FC_\eta(Q)$ whenever we do not want to refer to a specific CNF.

2.3 Implication graphs of Horn functions

Let us recall first some standard notions from graph theory. A *directed graph* (sometimes abbreviated to *digraph*) is an ordered pair $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ where $\mathbf{N}$ is the set of *nodes* and $\mathbf{A}$ is the set of *arcs*, and where an arc is an ordered pair of nodes.

A *directed path* is a sequence of arcs a_1, a_2,...,a_p such that $a_i = (x_i, x_{i+1})$ for some vertices x_1, x_2, ..., x_{p+1}. A *cycle* is a path such that $x_1 = x_{p+1}$. A directed graph is called *strongly connected* if for any two nodes x and y there exist both a directed path from x to y and a directed path from y to x. If a graph $\mathbf{D}$ is not strongly connected then its vertex set can be decomposed in a unique way into maximal strongly connected subsets, called the *strong components* of $\mathbf{D}$.

A subset $X \subseteq \mathbf{N}$ of nodes is called an *initial set* of $\mathbf{D}$ if $(u, v) \in \mathbf{A}$ and $v \in X$ imply $u \in X$, and it is called a *terminal set* of $\mathbf{D}$ if $(u, v) \in \mathbf{A}$ and $u \in X$ imply $v \in X$. We shall denote by $Cone_{\mathbf{D}}(X)$ the smallest (with respect to inclusion) initial set of $\mathbf{D}$ that contains X, and by $Anticone_{\mathbf{D}}(X)$ the smallest (with respect to inclusion) terminal set of $\mathbf{D}$ that contains X.

A directed graph is called *acyclic* if it contains no directed cycle. Note that in such a case every strong component consists of a single node. If $\mathbf{D}$ is an acyclic directed graph with a node set $\mathbf{N} = \{x_1, \ldots, x_n\}$, then an ordering of the nodes $(x_{i_1}, \ldots, x_{i_n})$ is called a *topological order* on $\mathbf{N}$ if for every arc $(x_{i_j}, x_{i_k}) \in \mathbf{A}$ we have $i_j < i_k$.

If $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ is a directed graph, then the *transitive closure* of $\mathbf{D}$ is a directed graph $\overline{\mathbf{D}} = (\mathbf{N}, \overline{\mathbf{A}})$ where $(x, y) \in \overline{\mathbf{A}}$, whenever there is a directed path from x to y in the digraph $\mathbf{D}$. Obviously, for a given $\mathbf{D}$ the transitive closure $\overline{\mathbf{D}}$ is uniquely defined. A *transitive reduction* of $\mathbf{D}$ is a graph $\mathbf{D}_R = (\mathbf{N}, \mathbf{A}_R)$ such that $\overline{\mathbf{D}}_R = \overline{\mathbf{D}}$ (the digraphs $\mathbf{D}_R$ and $\mathbf{D}$ have the same transitive closure) and $|\mathbf{A}_R|$ is minimum. As opposed to the transitive closure, there may be several distinct transitive reductions of the same digraph $\mathbf{D}$.

Finally, if $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ is a directed graph with strong components $C_1, \ldots, C_s$, then the directed graph $\mathbf{D}' = (\mathbf{N}', \mathbf{A}')$ on the set of nodes $\mathbf{N}' = \{C_1, \ldots, C_s\}$ with arcs

$$(C_i, C_j) \in \mathbf{A}' \text{ iff } \exists x \in C_i \ \exists y \in C_j \text{ such that } (x, y) \in \mathbf{A}$$

is called the *acyclic condensation* of the digraph $\mathbf{D}$.

Let us note that the strong components of a directed graph $\mathbf{D} = (\mathbf{N}, \mathbf{A})$ along with the topological order of its acyclic condensation can be found in $O(|\mathbf{A}|)$ time, [24].

Let us recall next some very useful definitions from [16], associating directed graphs to Horn CNFs and Horn functions.

Definition 2.6 *For a Horn CNF ϕ let $\mathbf{G}_\phi = (\mathbf{N}, \mathbf{A}_\phi)$ be the digraph defined by*

$$\mathbf{N} = \{x | \ x \text{ is a propositional variable in } \phi\}$$

$$\mathbf{A}_\phi = \{(x, y) \mid \exists \text{ a clause } C \in \mathcal{C}(\phi) \text{ such that } C \geq \overline{x} \vee y\}.$$

In other words, for each pure Horn clause C in ϕ, the graph $\mathbf{G}_\phi$ contains as many arcs as is the number of subgoals in C, with each arc going from the corresponding subgoal to the head of C. Since a Horn function can be represented by several different Horn CNFs, seemingly we can associate in this way several different graphs to a Horn function. However, as it was shown in [3], all these graphs share several important features.

Theorem 2.7 *Let ϕ_1 and ϕ_2 be two distinct prime CNFs representing the same Horn function f and let x, y be arbitrary propositional variables from f. Then there is a directed path from x to y in $\mathbf{G}_{\phi_1}$ if and only if there is a directed path from x to y in $\mathbf{G}_{\phi_2}$. Moreover, it then follows that $\mathbf{G}_{\phi_1}$ and $\mathbf{G}_{\phi_2}$ have identical transitive closures, identical strong components, and identical acyclic condensations.*

Theorem 2.7 allows us to associate a graph directly to a Horn function rather than to its particular CNF representations.

Definition 2.8 *Let f be a Horn function and ϕ its arbitrary prime CNF representation. Then we define $\mathbf{G}_f$ as the transitive closure of $\mathbf{G}_\phi$.*

Theorem 2.7 suggests that for a given Horn function f the strong components of $\mathbf{G}_f$ play an important role in how the set of all prime CNF representations of f are structured.

In what follows we shall call $\mathbf{G}_\phi$ and $\mathbf{G}_f$ the *implication graphs* of ϕ and f, respectively.

Let us state finally an important property of implication graphs, slightly generalizing a statement of [16].

Lemma 2.9 *Let h be a pure Horn function, and let $C \in \mathcal{I}(h)$. Then $(x, Head(C))$ is an arc in $\mathbf{G}_h = (\mathbf{N}, \mathbf{A})$ for every $x \in Subg(C)$.*

Proof. This statement was shown in [16] for all prime implicates $C \in \mathcal{I}^p(h)$. Since $\mathcal{I}(h) = \mathcal{R}(\mathcal{I}^p(h))$, every implicate $C \in \mathcal{I}(h)$ can be derived by a series of resolutions from prime implicates of h. Thus, the statement will follow by the inductive use of the following argument:

Let $C_1 = B_1 \vee x$ and $C_2 = B_2 \vee \overline{x} \vee y$ be two resolvable clauses such that $(z, x) \in \mathbf{A}$ for every $z \in B_1$ and $(z, y) \in \mathbf{A}$ for every $z \in B_2 \cup \{x\}$. Then, the clause $C = R(C_1, C_2) = B_1 \vee B_2 \vee y$ has the property that $(z, y) \in \mathbf{A}$ for every $z \in B_1 \cup B_2$, since $(z, y) \in \mathbf{A}$ is assumed for $z \in B_2$ and since $\mathbf{G}_h$ is transitively closed and $(x, y) \in \mathbf{A}$ and $(z, x) \in \mathbf{A}$ for all $z \in B_1$ by our assumptions. ■

3 Polynomially Solvable Cases of HM

The notion of an implication graph of a Horn function f carries a lot of information about the CNF representations of f, allowing the characterization of important special classes for which HM is polynomially solvable.

A Horn CNF ϕ is said to be *acyclic* if its associated implication graph $\mathbf{G}_\phi$ is acyclic. A Horn function f is called *acyclic* if it admits at least one acyclic CNF representation. It was shown in [16] that every acyclic function has a unique irredundant and prime representation. Obviously, that implies that this unique CNF also constitutes a minimal representation of the given function with respect to both of the introduced complexity measures. Hence the "preprocessing phase" corresponding to Theorem 2.3 (i.e. transforming the input CNF into an irredundant and prime one) itself represents a polynomial time HM algorithm for acyclic functions.

Let us call two propositional variables x and y *logically equivalent* in a Horn function f if the clauses $\overline{x} \vee y$ and $\overline{y} \vee x$ are implicates of f. A Horn CNF ϕ is then said to be *quasi-acyclic* (see [16]) if every strong component of its associated implication graph $\mathbf{G}_\phi$ consists of a set of logically equivalent propositional variables. A Horn function f is called *quasi-acyclic* if it admits at least one quasi-acyclic CNF representation.

Note that every acyclic CNF ϕ is also quasi-acyclic since each strong component of $\mathbf{G}_\phi$ is a singleton. The name quasi-acyclic comes from the fact that picking a representative in each set of logically equivalent propositional variables and substituting this representative for all the other logically equivalent variables in the set results in an acyclic CNF (i.e. the CNF is essentially acyclic except for the fact that each variable can have several "names"). In order to understand the structure of quasi-acyclic functions it is important to realize that if f is a quasi-acyclic function and x, y are propositional variables from the same strong component of $\mathbf{G}_f$ then, since both $\overline{x} \vee y$ and $\overline{y} \vee x$ are implicates of f, no prime pure Horn implicate of f with degree three or more may contain a subgoal from the same strong component of $\mathbf{G}_f$ as the head. This means that the pure Horn clauses in any prime CNF representation of f can be partitioned into two groups. The first group (let us call it group A) contains clauses where all the subgoals are in different strong component(s) of $\mathbf{G}_f$ than the head, while the second group (group B) contains quadratic clauses with both the subgoal and the head belonging to the same strong component of $\mathbf{G}_f$. Loosely speaking, the clauses in group B "generate" the strong components of $\mathbf{G}_f$ while the clauses in group A "generate" its acyclic condensation. It was proved in [16] that HM can be solved in polynomial time for quasi-acyclic functions.

Generalizing further the above classes (still using the implication graph) leads us to the main concept of this paper.

Definition 3.1 *Let us call a pure Horn clause C* component-wise quadratic *(or CQ for short) with respect to a Horn function f if at most one subgoal of C belongs to the strong component of $\mathbf{G}_f$ containing the head of C. A Horn CNF ϕ representing a function f is said to be CQ if every pure Horn clause of ϕ is CQ with respect to f. Finally, a Horn function f is called CQ if it admits at least one CQ CNF representation.*

The intuition behind the name, component-wise quadratic, is that if we restrict any such CNF to variables from a single strong component of its implication graph, the pure Horn part of the resulting CNF is always quadratic. The negative part may contain clauses of

higher degrees than two, but as we shall see later in this paper, negative clauses play no essential role in the Horn minimization problem.

Once again, it is possible to ask what happens if a CQ function is represented by a CNF which is not CQ. Fortunately, as in the acyclic and quasi-acyclic cases, it is enough to do the preprocessing phase to arrive to a CQ representation, as the following statement shows.

Theorem 3.2 *Let f be a CQ function. Then any prime CNF representation of f is CQ.*

Proof. By Definition 3.1 there exists a CQ CNF ϕ which represents f. Let us replace every clause in ϕ by a prime implicate of f which subsumes the given clause and let us denote the resulting prime CNF of f by ϕ'. Note that each pure Horn clause C of ϕ' has two properties:

1. C is a CQ clause[2];
2. there is an arc in $\mathbf{G}_f$ from every subgoal of C to the head of C.

The first property follows from the fact that deleting literals from a CQ clause must yield another CQ clause. The second property follows from the fact that $\mathbf{G}_f$ is the transitive closure of $\mathbf{G}_{\phi'}$ by Definition 2.8 (here we need primality and that is the reason why we transformed ϕ into ϕ'). We shall show that a resolvent of any two clauses which satisfy the above two properties has again both of these properties.

So let $C_1 = A \vee x$ and $C_2 = B \vee \overline{x} \vee y$ be two arbitrary resolvable clauses belonging to $\mathcal{I}(f)$ and satisfying the above two properties. Let us denote their resolvent by $C = A \vee B \vee y$, and observe that $C \in \mathcal{I}(f)$ by Lemma 2.1 and by the definition of $\mathcal{I}(f)$. Thus, the second property follows by Lemma 2.9.

To verify the first property, let us consider the strong components S_x and S_y of $\mathbf{G}_f$ which contain x and y respectively.

- If $S_x \neq S_y$, then S_x precedes S_y in the partial order imposed by the acyclic condensation of $\mathbf{G}_f$ (because there is an arc from x to y in $\mathbf{G}_f$), and hence so do all strong components which have a nonempty intersection with the set A (because for every $z \in A$ there is an arc from z to x in G_f). Hence $A \cap S_y = \emptyset$. Now the fact that C_2 is CQ implies $|B \cap S_y| \leq 1$ and thus $|(A \cup B) \cap S_y| \leq 1$ follows, implying that C is CQ.
- If $S_x = S_y$, then $\{x, y\} \subseteq S_y$, and thus the fact that C_1 and C_2 are CQ implies $|A \cap S_y| \leq 1$ and $B \cap S_y = \emptyset$. Therefore again $|(A \cup B) \cap S_y| \leq 1$ follows, proving that C is CQ.

By completeness of resolution (Theorem 2.2) every prime implicate of f can be derived from ϕ' by a series of resolutions. This implies that every prime implicate of f satisfies the above two properties, i.e. in particular every prime implicate of f is a CQ clause, which completes the proof. ■

[2]Here we mean CQ with respect to f. Whenever it is obvious which function is meant, we shall omit referring to it in the subsequent text.

Note that the above proof actually implies a bit more: if f is a CQ function then not only all prime implicates of f are CQ but even all clauses in $\mathcal{I}(f)$ are CQ. Let us formulate this easy observation as a corollary.

Corollary 3.3 *Let f be a CQ function and C an arbitrary clause in $\mathcal{I}(f)$. Then C is CQ with respect to f.*

The main aim of this paper is to show that HM is polynomially solvable for CQ-functions.

Theorem 3.4 *Let h be a CQ-function on n variables, represented by an irredundant and prime CNF $\mathcal{C}$ consisting of m clauses and l literals. Then, a minimum CNF representation $\mathcal{C}^*$ of h can be found in $O(n^2 + m \cdot l)$ time.*

In the rest of the paper we shall present a decomposition based proof for the above statement. To arrive at such a proof, we first need to analyze the structure of potentially useful decompositions. We accomplish this by studying the structure of certain subfamilies of implicates of a Boolean function (in general, we do not restrict ourselves to Horn functions only). In particular, in Section 4 we consider subfamilies of implicates which define subfunctions, the representation of which can be chosen independently from the other clauses of the considered CNF representation. We also consider subfamilies from which every CNF representation must contain some clauses, leading to a min-max relation. In Section 5 we return to Horn functions, and provide tools to identify the above mentioned useful subfamilies of implicates in a constructive way. The main tool in this will be a new graph associated to a Horn function h, the vertices of which are the implicates of h, and which we shall call the *clause graph* of h. Finally, in Section 6 we present an algorithm for HM, and prove its correctness and complexity, as claimed in Theorem 3.4.

A natural question to ask is whether it is possible to further extend the class of CQ functions by allowing the pure Horn clauses to have not at most one but at most two (three, four, etc.) subgoals in the same strong component as the head, and still maintain the polynomial time solvability of HM. The answer is no unless P=NP. The reason is that even allowing just two subgoals to fall in the same strong component as the head would include all cubic Horn functions into the extended class (a Horn function is cubic if it admits a CNF representation with the highest clause degree at most three). However, it was proved in [2, 5] that HM is NP-hard for cubic Horn CNFs.

4 Exclusive and Essential Sets of Implicates

In this section we recall properties of CNF representations of Boolean functions proved in [4]. These properties are applicable to all Boolean functions not just Horn ones. We also state and prove a decomposition lemma which is a consequence of the recalled properties. Later in the paper we shall return to Horn functions, and more specifically to CQ-functions, and apply the general results developed in this section.

In the remainder of this section let us consider an arbitrary but fixed Boolean function f, the set $\mathcal{I}^p(f)$ of all prime implicates of f, and the set $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}^p(f))$ of all implicates of f that can be generated from the prime implicates of f by series of resolutions. Note that $\mathcal{I}(f) = \mathcal{R}(\mathcal{I}(f))$, i.e. the set $\mathcal{I}(f)$ is closed under resolution.

Let us start by recalling two simple technical lemmas from [4] (appearing there as Lemmas 4.3 and 4.4) which deal with properties of resolution closures of sets of clauses.

Lemma 4.1 *Let $\mathcal{C}_1$ and $\mathcal{C}_2$ be two sets of clauses. Then $\mathcal{R}(\mathcal{C}_1) = \mathcal{R}(\mathcal{C}_2)$ implies that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2)$, i.e. if the sets have the same resolution closure then they represent the same function.*

Lemma 4.2 *Let $\mathcal{X}, \mathcal{Z} \subseteq \mathcal{I}(f)$ be two arbitrary sets of clauses. Then $\mathcal{R}(\mathcal{X} \cup \mathcal{Z}) = \mathcal{R}(\mathcal{X} \cup \mathcal{R}(\mathcal{Z}))$.*

4.1 Exclusive components of functions

Let us now define the first key concept of this section, which helps us to decompose the problem of Horn minimization.

Definition 4.3 *Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{X} \subseteq \mathcal{C}$ is called an* exclusive subset of $\mathcal{C}$ *if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:*

$$R(C_1, C_2) \in \mathcal{X} \Longrightarrow C_1 \in \mathcal{X} \text{ and } C_2 \in \mathcal{X},$$

i.e. the resolvent belongs to $\mathcal{X}$ only if both parent clauses are in $\mathcal{X}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call such a subset $\mathcal{X}$ an exclusive set of clauses of f (or simply an exclusive set, if f or $\mathcal{C}$ is clear from the context).

Let us first claim in the next lemma [4] some simple properties possessed by exclusive sets. Since all these properties follow directly from Definition 4.3 we shall omit the proofs.

Lemma 4.4 *Let $\mathcal{C}$ be an arbitrary set of clauses. Then,*

(a) if $\mathcal{A}$ is an exclusive subset of $\mathcal{B}$ and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A}$ is an exclusive subset of $\mathcal{C}$;

(b) if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{C}$, and $\mathcal{A}$ is an exclusive subset of $\mathcal{C}$, then it is also an exclusive subset of $\mathcal{B}$;

(c) if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$ are both exclusive subsets of $\mathcal{C}$, then $\mathcal{A} \cup \mathcal{B}$ and $\mathcal{A} \cap \mathcal{B}$ are also exclusive (and hence all exclusive subsets of $\mathcal{C}$ form a lattice). ■

The following simple technical lemma dealing with exclusive sets of implicates was proved in [4] (as Lemma 5.4).

Lemma 4.5 *Let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses (of f) and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses such that $\mathcal{X} \subseteq \mathcal{R}(\mathcal{C})$. Then $\mathcal{R}(\mathcal{X}) = \mathcal{R}(\mathcal{C} \cap \mathcal{X})$.*

To see an interesting example of exclusive sets of clauses, let us for a moment return to Horn functions. Let h be a Horn function and let us partition the set $\mathcal{I}(h)$ into two subsets $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ where $\mathcal{H}$ is the set of all pure Horn clauses in $\mathcal{I}(h)$ and $\mathcal{N}$ is the set of all negative clauses in $\mathcal{I}(h)$[3]. Then it is not hard to see that $\mathcal{H}$ is an exclusive set of h (the resolvent is in $\mathcal{H}$ only if both parent clauses are in $\mathcal{H}$).

The partition $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ has some important properties, shown in [12]). The first such property states, that if ϕ_1 and ϕ_2 are two distinct prime CNFs representing h, then the pure Horn parts of ϕ_1 and ϕ_2 (i.e. the conjunctions of all pure Horn clauses in the given CNFs) also represent the same pure Horn function, called in [12] the *pure Horn component* of h.

Proposition 4.6 ([12]) *Let ϕ_1 and ϕ_2 be two distinct prime CNFs of a Horn function h. Then $\phi(\mathcal{C}(\phi_1) \cap \mathcal{H}) \equiv \phi(\mathcal{C}(\phi_2) \cap \mathcal{H})$.*

Proposition 4.6 was generalized in [4] to all exclusive sets (it appears there as Theorem 5.5).

Proposition 4.7 ([4]) *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2) \equiv f$, i.e. such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\phi(\mathcal{C}_1 \cap \mathcal{X}) \equiv \phi(\mathcal{C}_2 \cap \mathcal{X})$.*

It is immediate to see that Proposition 4.6 is just a special case of Proposition 4.7. We can also generalize the notion of a "pure Horn component".

Definition 4.8 *Let f be an arbitrary Boolean function, $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses of f, and $\mathcal{C} \subseteq \mathcal{I}(f)$ be a set of clauses which represents f (i.e. $\phi(\mathcal{C}) \equiv f$). The Boolean function $f_{\mathcal{X}}$ represented by the set $\mathcal{C} \cap \mathcal{X}$ is called the* $\mathcal{X}$-component *of the function f. We shall simply call a function g an* exclusive component of f, *if $g = f_{\mathcal{X}}$ for some exclusive subset $\mathcal{X} \subseteq \mathcal{I}(f)$.*

Proposition 4.7 guarantees that the $\mathcal{X}$-component $f_{\mathcal{X}}$ is well defined for every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$. Moreover, Proposition 4.7 has an important consequence (appearing in [4] as Corollary 5.7) that will prove to be instrumental later in the minimization of CQ functions.

Corollary 4.9 *Let $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{I}(f)$ be two distinct sets of clauses such that $\phi(\mathcal{C}_1) \equiv \phi(\mathcal{C}_2) \equiv f$, i.e. such that both sets represent f, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\phi((\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})) \equiv f$.*

[3]It is left to the reader to verify the easy fact that $\mathcal{H}$ and $\mathcal{N}$ indeed constitute a partition of $\mathcal{I}(h)$, i.e. that no clause which is neither pure Horn nor negative can appear in $\mathcal{I}(h)$ (recall that each prime implicate of a Horn function is either pure Horn or negative.)

Loosely speaking, Corollary 4.9 says that if $\mathcal{C}_1, \mathcal{C}_2$ both represent f and $\mathcal{X}$ is exclusive, then removing $\mathcal{C}_1 \cap \mathcal{X}$ from $\mathcal{C}_1$ and replacing it with $\mathcal{C}_2 \cap \mathcal{X}$ produces another representation of f. This statement can be strengthened in the following way: if $\mathcal{C}_1$ is a prime and irredundant representation of f and $\mathcal{C}_2 \cap \mathcal{X}$ is a prime and irredundant representation of $f_{\mathcal{X}}$ then also $(\mathcal{C}_1 \setminus \mathcal{X}) \cup (\mathcal{C}_2 \cap \mathcal{X})$ is a prime and irredundant representation of f. Before stating this stronger version of Corollary 4.9 we first prove a simple technical lemma.

Lemma 4.10 *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an irredundant and prime set of clauses such that $\phi(\mathcal{C}) \equiv f$, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Then $\mathcal{C} \cap \mathcal{X}$ is an irredundant and prime set of clauses such that $\phi(\mathcal{C} \cap \mathcal{X}) \equiv f_{\mathcal{X}}$.*

Proof. The fact that $\mathcal{C} \cap \mathcal{X}$ represents $f_{\mathcal{X}}$ follows directly from the definition of $f_{\mathcal{X}}$. The irredundancy of $\mathcal{C} \cap \mathcal{X}$ trivially follows from the irredundancy of $\mathcal{C}$ (if we can drop a clause from $\mathcal{C} \cap \mathcal{X}$ without changing the function represented by $\mathcal{C} \cap \mathcal{X}$ then the same clause can be dropped from $\mathcal{C}$ without changing the function represented by $\mathcal{C}$). To show the primality of $\mathcal{C} \cap \mathcal{X}$ let us assume that there exists a nonprime implicate $C' \in \mathcal{C} \cap \mathcal{X}$ of function $f_{\mathcal{X}}$. However, then C' is also a nonprime implicate of f contradicting the primality of $\mathcal{C}$. ■

Corollary 4.11 *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an irredundant and prime set of clauses such that $\phi(\mathcal{C}) \equiv f$, and let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive set of clauses. Moreover, let $\mathcal{C}' \subseteq \mathcal{I}(f)$ be an irredundant and prime set of clauses such that $\phi(\mathcal{C}') \equiv f_{\mathcal{X}}$. Then $(\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}'$ is an irredundant and prime set of clauses such that $\phi((\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}') \equiv f$.*

Proof. The fact that $(\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}'$ represents f follows directly from Corollary 4.9 (where $\mathcal{C}'$ plays the role of $\mathcal{C}_2 \cap \mathcal{X}$). The irredundancy and primality of $(\mathcal{C} \setminus \mathcal{X}) \cup \mathcal{C}'$ follows by similar arguments as in the proof of Lemma 4.10 ■

Let us recall next that a subset of the implicates $\mathcal{S} \subseteq \mathcal{I}(f)$ of f is called redundant with respect to f, if $\mathcal{S} \cap \mathcal{C} = \emptyset$ for all irredundant representations $\mathcal{C} \subseteq \mathcal{I}(f)$ of f (i.e., for all minimal sets of implicates for which $\mathcal{R}(\mathcal{C}) = \mathcal{I}(f)$). The following statement was proved in [4] (as Corollary 5.8).

Lemma 4.12 *For every exclusive set $\mathcal{X} \subseteq \mathcal{I}(f)$ we have $\mathcal{R}(\mathcal{X}) = \mathcal{I}(f_{\mathcal{X}})$, furthermore the set $\mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$ is redundant with respect to $f_{\mathcal{X}}$, as well as with respect to f.*

Let us finally close this section by an important consequence of Corollary 4.9, namely that in an arbitrary representation $\mathcal{C}$ of f we can replace $\mathcal{C} \cap \mathcal{X}$ by an arbitrary representation of $f_{\mathcal{X}}$ and obtain again a representation of f, whenever $\mathcal{X}$ is an exclusive set. This suggests a decomposition of problem HM, which we summarize in the following statement:

Lemma 4.13 (Decomposition lemma) *Given a function f, let $\emptyset = \mathcal{X}_0 \subseteq \mathcal{X}_1 \subseteq \cdots \subseteq \mathcal{X}_t$ be a chain of exclusive subsets in which $\mathcal{R}(\mathcal{X}_t) = \mathcal{I}(f)$ and let $\mathcal{C}_i^* \subseteq \mathcal{X}_i \setminus \mathcal{X}_{i-1}$ be minimal subsets such that $\mathcal{R}(\mathcal{X}_{i-1} \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{X}_i)$ for $i = 1, ..., t$. Then, $\mathcal{C}^* = \bigcup_{i=1}^{t} \mathcal{C}_i^*$ is a minimal representation of f (where minimality is with respect to any one of the complexity measures introduced earlier).*

Proof. Using Lemma 4.2 we can show by induction that $\mathcal{R}(\mathcal{C}_1^* \cup \cdots \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{R}(\mathcal{C}_1^* \cup \cdots \cup \mathcal{C}_{i-1}^*) \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{X}_{i-1} \cup \mathcal{C}_i^*) = \mathcal{R}(\mathcal{X}_i)$, for $i = 1, .., t$, which implies by Lemma 4.1 that $\bigcup_{j=1}^{i} \mathcal{C}_j^*$ is a representation of the $\mathcal{X}_i$-component $f_{\mathcal{X}_i}$ of f, for $i = 1, .., t$.

Let us now consider an arbitrary CNF representation $\mathcal{C} \subseteq \mathcal{I}(f)$ of f, and define $\mathcal{C}_i = \mathcal{C} \cap (\mathcal{X}_i \setminus \mathcal{X}_{i-1})$ for $i = 1, ..., t$. By (b) of Lemma 4.4 we have that $\mathcal{X}_{i-1}$ is an exclusive subset of $\mathcal{X}_i$, for $i = 1, ..., t$. Thus, we can apply Lemma 4.5 to the function $f_{\mathcal{X}_i}$ and its exclusive subset $\mathcal{X}_{i-1}$, for $i = 1, 2, ..., t$, and obtain inductively by Lemma 4.2 that $\mathcal{R}(\mathcal{X}_{i-1} \cup \mathcal{C}_i) = \mathcal{R}(\mathcal{X}_i)$, for $i = 1, ..., t$. Therefore, by our choice of $\mathcal{C}_i^*$ we have that the size of $\mathcal{C}_i$ is not smaller than that of $\mathcal{C}_i^*$, for $i = 1, ..., t$ (by the complexity measure we use). Since both considered complexity measures are additive, the statement follows. ■

4.2 Essential sets and an orthogonality relation

Let us now introduce the second key concept of this section, which will establish a certain orthogonality relation between sets of implicates and CNF representations.

Definition 4.14 *Given a set $\mathcal{C}$ of clauses, a subset $\mathcal{E} \subseteq \mathcal{C}$ is called an* essential subset of $\mathcal{C}$ *if for every pair of resolvable clauses $C_1, C_2 \in \mathcal{C}$ the following implication holds:*

$$R(C_1, C_2) \in \mathcal{E} \Longrightarrow C_1 \in \mathcal{E} \text{ or } C_2 \in \mathcal{E},$$

i.e. the resolvent belongs to $\mathcal{E}$ only if at least one of the parent clauses are from $\mathcal{E}$. In particular, if $\mathcal{C} = \mathcal{I}(f)$ for a Boolean function f, we call $\mathcal{E}$ an essential set of clauses of f (or simply an essential set, if f or $\mathcal{C}$ is clear from the context).

It is easy to see that every exclusive set of clauses (and the set $\mathcal{I}(f)$ in particular) is also essential. We summarize in the following lemma a few simple properties of essential sets. Since all these properties follow directly from Definitions 4.3 and 4.14 we shall omit the proofs.

Lemma 4.15 ([4]) *Let $\mathcal{C}$ be an arbitrary set of clauses. Then,*

(a) if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$ are both essential subsets of $\mathcal{C}$, then $\mathcal{A} \cup \mathcal{B}$ is also essential;

(b) if $\mathcal{R}(\mathcal{C}) = \mathcal{C}$ and $\mathcal{A}$ is an essential subset of $\mathcal{C}$, then $\mathcal{C} \setminus \mathcal{A}$ is closed under resolution, i.e. $\mathcal{C} \setminus \mathcal{A} = \mathcal{R}(\mathcal{C} \setminus \mathcal{A})$;

(c) if $\mathcal{R}(\mathcal{A}) = \mathcal{A}$ and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{B} \setminus \mathcal{A}$ is an essential subset of $\mathcal{C}$;

(d) if $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{C}$, $\mathcal{A}$ is an essential subset of $\mathcal{B}$, and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A}$ is an essential subset of $\mathcal{C}$, as well;

(e) if $\mathcal{A}, \mathcal{B} \subseteq \mathcal{C}$, $\mathcal{A} \cap \mathcal{B} \neq \emptyset$, $\mathcal{A}$ is an essential subset of $\mathcal{C}$, and $\mathcal{B}$ is an exclusive subset of $\mathcal{C}$, then $\mathcal{A} \cap \mathcal{B}$ is also an essential subset of $\mathcal{C}$. ■

To see an interesting example of essential sets, let us consider again a Horn function h and return to the partition of the set $\mathcal{I}(h)$ into two subsets $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ where $\mathcal{H}$ is the set of all pure Horn clauses in $\mathcal{I}(h)$ and $\mathcal{N}$ is the set of all negative clauses in $\mathcal{I}(h)$. Then, it is not hard to see that $\mathcal{N}$ is essential for h (since no two clauses in $\mathcal{N}$ are resolvable, the resolvent is in $\mathcal{N}$ only if *exactly* one of the parent clauses is in $\mathcal{N}$ and the other one is in $\mathcal{H}$).

The orthogonality property of essential sets was proved in [4]. This key proposition (which appears there as Theorem 6.4) shows that every essential set has one (or more) of its clauses present in every representation of f and moreover that this condition is not only necessary but also sufficient.

Proposition 4.16 ([4]) *Let $\mathcal{C} \subseteq \mathcal{I}(f)$ be an arbitrary set of clauses. Then $\mathcal{C}$ represents f if and only if $\mathcal{C} \cap \mathcal{E} \neq \emptyset$ for every nonempty essential set of clauses $\mathcal{E} \subseteq \mathcal{I}(f)$.*

Proposition 4.16 has an obvious corollary: if there exist nonempty essential sets $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_k \subseteq \mathcal{I}(f)$ which are pairwise disjoint, then every representation of f must consist of at least k clauses. Hence, any collection of pairwise disjoint essential sets of clauses provides an easy lower bound on the size (i.e. number of clauses) of a minimal representation of f.

A second important property (proved in [12]) of the partition $\mathcal{I}(h) = \mathcal{H} \cup \mathcal{N}$ states that, if ϕ_1 and ϕ_2 are two distinct irredundant CNFs representing h, then ϕ_1 and ϕ_2 both contain the same number of negative clauses.

Proposition 4.17 ([12]) *Let ϕ_1 and ϕ_2 be two distinct irredundant CNFs of a Horn function h. Then $|\mathcal{C}(\phi_1) \cap \mathcal{N}| = |\mathcal{C}(\phi_2) \cap \mathcal{N}|$.*

Let us finish this section by recalling a generalization of Proposition 4.17 (it appears as Theorem 6.12 in [4]), which will come handy in the subsequent sections of this paper.

Proposition 4.18 ([4]) *Given a Boolean function f, let $\mathcal{X} \subseteq \mathcal{I}(f)$ be an exclusive subset of f such that no two clauses from $\mathcal{E} = \mathcal{I}(f) \setminus \mathcal{R}(\mathcal{X})$ are resolvable. Then, there exists an integer $k = k(\mathcal{E}) > 0$, and pairwise disjoint essential subsets $\mathcal{Q}_j \subseteq \mathcal{E}$, $j = 1, .., k$ such that $|\mathcal{Q}_j \cap \mathcal{C}| = 1$ for $j = 1, ..., k$ and $|(\mathcal{E} \setminus \bigcup_{j=1}^{k} \mathcal{Q}_j) \cap \mathcal{C}| = 0$ for any irredundant set $\mathcal{C} \subseteq \mathcal{I}(f)$ of clauses representing f.*

Of course, it is clear that Proposition 4.18 implies the following corollary which more closely resembles the statement of Proposition 4.17.

Corollary 4.19 ([4]) *Let f, $\mathcal{X}$, and $\mathcal{E}$ be as in the statement of Proposition 4.18, and let ϕ_1 and ϕ_2 be two distinct irredundant CNFs of f. Then $|\mathcal{C}(\phi_1) \cap \mathcal{E}| = |\mathcal{C}(\phi_2) \cap \mathcal{E}|$.*

Clearly, Proposition 4.17 is just a special case of Corollary 4.19 if we set $\mathcal{X}$ to be the set of all pure Horn clauses in $\mathcal{I}(f)$ (in this case $\mathcal{X} = \mathcal{R}(\mathcal{X})$) and $\mathcal{E}$ to be the set of all negative clauses in $\mathcal{I}(f)$.

5 Horn Minimization

Let us return now to the problem of Horn minimization, and let us recall first a few preprocessing steps.

By standard Boolean terminology a *unit* clause is a clause consisting of exactly one literal. If x or $\overline{x}$ is a unit prime implicate of a Boolean function f, then clearly no other prime implicates of f may contain the variable x (negated or not). Therefore any Horn function f represented by a CNF ϕ can be decomposed (in $O(|\phi|_\ell^2)$ time due to Theorem 2.3) into a conjunction of unit clauses f_1 and a Horn function f_2 which has no unit prime implicates, in such a way that f_1 and f_2 are defined on disjoint sets of variables, and $f = f_1 \wedge f_2$. Since the aim of this paper is Horn minimization, and the above described decomposition of course outputs the shortest possible representation of the exclusive component f_1 of f, we can restrict our attention (without loss of generality) solely to functions with no unit prime implicates. Note that both $\mathcal{I}(f_1)$ and $\mathcal{I}(f_2)$ are exclusive sets, so we can minimize them independently due to Corollary 4.9.

Moreover, due to Proposition 4.17 (and its more general form Proposition 4.18) we may restrict our attention even further to pure Horn functions because the minimization of a Horn function really amounts to the minimization of its pure Horn component. Again note that the set of pure Horn implicates is exclusive and so it can be minimized idependently due to Corollary 4.9.

Therefore, throughout the remainder of this section we can assume that the Horn function h to be minimized is pure Horn and has no unit prime implicates. Let us denote by η the given CNF representation of it.

5.1 Implication graphs

Recall that in Section 3 we have defined the implication graphs $\mathbf{G}_\eta = (\mathbf{N}, \mathbf{A}_\eta)$ and $\mathbf{G}_h = (\mathbf{N}, \mathbf{A}_h)$ associated to η and h, respectively, where $\mathbf{N}$ is the set of variables of the function h represented by the CNF η. We shall first use these implication graphs to define exclusive subsets of clauses for h. In what follows, we shall simply write $\mathbf{G}$ instead of $\mathbf{G}_h$ or $\mathbf{G}_\eta$, whenever h and/or η will be clear from the context.

Definition 5.1 *Given a subset Y of variables of the function h, let us denote by*

$$Clauses(Y) = \{C \in \mathcal{I}(h) \mid Vars(C) \subseteq Y\}$$

the set of all clauses from $\mathcal{I}(h)$ which have all their variables belonging to the set Y.

Lemma 5.2 *Let $X \subseteq \mathbf{N}$ be a set of nodes of the implication graph $\mathbf{G} = \mathbf{G}_h$ (variables of h). Then the sets $Clauses(Cone_{\mathbf{G}}(X))$ and $Clauses(Anticone_{\mathbf{G}}(X))$ are both exclusive sets of clauses of h.*

Proof. Let $C_1 = B_1 \vee z$ and $C_2 = B_2 \vee \overline{z} \vee y$ be two resolvable clauses in $\mathcal{I}(h)$ and let $C = R(C_1, C_2) = B_1 \vee B_2 \vee y$ be their resolution.

If $Vars(C) \subseteq Cone_{\mathbf{G}}(X)$ then $y \in Cone_{\mathbf{G}}(X)$ and $(z, y) \in \mathbf{A}_h$ imply $z \in Cone_{\mathbf{G}}(X)$ and thus $Vars(C_1) \subseteq Cone_{\mathbf{G}}(X)$ and $Vars(C_2) \subseteq Cone_{\mathbf{G}}(X)$ proving that $Clauses(Cone_{\mathbf{G}}(X))$ is exclusive.

Similarly, if $Vars(C) \subseteq Anticone_{\mathbf{G}}(X)$ then the fact that $w \in Anticone_{\mathbf{G}}(X)$ and $(w, z) \in \mathbf{A}$ for every $w \in B_1$ implies $z \in Anticone_{\mathbf{G}}(X)$ and thus $Vars(C_1) \subseteq Anticone_{\mathbf{G}}(X)$ and $Vars(C_2) \subseteq Anticone_{\mathbf{G}}(X)$ proving that $Clauses(Anticone_{\mathbf{G}}(X))$ is exclusive. ■

Corollary 5.3 *Let X and Y be two arbitrary sets of nodes (variables) in $\mathbf{G}$, and let us define $Int_{\mathbf{G}}(X, Y) = Clauses(Anticone_{\mathbf{G}}(X)) \cap Clauses(Cone_{\mathbf{G}}(Y))$ to be the "interval" between X and Y. Then $Int_{\mathbf{G}}(X, Y)$ is an exclusive set of clauses.*

Proof. Follows immediately from Lemma 5.2 and Lemma 4.4 (exclusiveness is closed under intersection). ■

Let us show next a few technical claims about forward chaining and implication graphs.

Lemma 5.4 (Forward chaining) *Let S be a set of variables, $v \in FC_h(S)$, and let $\mathcal{C} \subseteq \mathcal{I}(h)$ be a minimal set of implicates of h for which $v \in FC_{\mathcal{C}}(S)$. Then, we have*

$$\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq Cone_{\mathbf{G}}(\{v\}).$$

Proof. By Lemma 2.9 we have $Vars(C) \subseteq Cone_{\mathbf{G}}(Head(C))$ for all $C \in \mathcal{I}(h)$. Since $\mathcal{C}$ is a minimal set with the property $v \in FC_{\mathcal{C}}(S)$, we can index its clauses $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ so that $v = Head(C_k)$ and $Head(C_j) \in Subg(C_i)$ for some $i > j$ for all $j = 1, ..., k-1$. This implies hence that $Vars(C_j) \subseteq Cone_{\mathbf{G}}(Head(C_i))$ for some $i > j$ for all $j < k$, from which the statement readily follows, since $v = Head(C_k)$. ■

We shall call such a minimal set of clauses $\mathcal{C} \subseteq \mathcal{I}(h)$ for which $v \in FC_{\mathcal{C}}(S)$ a *minimal derivation* of v from S.

Definition 5.5 *Let Y be an arbitrary set of variables and $\mathcal{C} \subseteq \mathcal{I}(h)$ be a set of clauses. Then we shall denote by $\mathcal{C}|_Y$ the set of clauses from $\mathcal{C}$ which have all variables in the set Y, i.e. $\mathcal{C}|_Y = \mathcal{C} \cap Clauses(Y)$. Furthermore, if Y is such that the set $\mathcal{E} = Clauses(Y)$ forms an exclusive set of clauses (of h), we denote by $h|_Y = h_{\mathcal{E}}$ the $\mathcal{E}$-component of h (see Definition 4.8).*

Proposition 4.7 guarantees that the function $h|_Y$ is well defined whenever the set $Clauses(Y)$ is exclusive. Due to Lemma 5.2 and Corollary 5.3 this happens for instance whenever Y is an initial, terminal, or interval set in $\mathbf{G}$.

Lemma 5.6 (Initial set) *Let I be an initial set in $\mathbf{G}$, and let S be an arbitrary set of variables. Then $FC_h(S) \cap I = FC_{h|_I}(S \cap I)$.*

Proof. Let us start by proving the inclusion $FC_h(S) \cap I \subseteq FC_{h|_I}(S \cap I)$. Let us consider an arbitrary variable $v \in FC_h(S) \cap I$, and let $\mathcal{C} \subseteq \mathcal{I}(h)$ be a minimal derivation of v from S. It follows from Lemma 5.4 that $\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq Cone_{\mathbf{G}}(\{v\})$, and because $v \in I$ and I is an initial set, we also have $Cone_{\mathbf{G}}(\{v\}) \subseteq I$. Therefore, $\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq I$ follows, and hence $v \in FC_{h|_I}(S \cap I)$, which is the desired result.

Now let us prove the opposite inclusion. Since $h|_I$ is defined only on variables from the set I, it follows that $FC_{h|_I}(S \cap I) \subseteq I$. The inclusion $FC_{h|_I}(S \cap I) \subseteq FC_h(S)$ is trivial, and so we get that $FC_{h|_I}(S \cap I) \subseteq FC_h(S) \cap I$. ■

5.2 Clause graphs

We shall define a yet another directed graph, associated to a set of pure Horn clauses and/or to a pure Horn function. As opposed to $\mathbf{G}_h$ and $\mathbf{G}_{\mathcal{C}}$ which are defined on the set of variables, this so called *clause graph* is defined on the set of clauses.

Given a set $\mathcal{C}$ of pure Horn clauses let us define its *clause graph* $\mathbf{D}_{\mathcal{C}} = (\mathbf{V}_{\mathcal{C}}, \mathbf{E}_{\mathcal{C}})$ on the given set of clauses as its vertex set $\mathbf{V}_{\mathcal{C}} = \mathbf{V}(\mathbf{D}_{\mathcal{C}}) = \mathcal{C}$, and where the edge set $\mathbf{E}_{\mathcal{C}} = \mathbf{E}(\mathbf{D}_{\mathcal{C}})$ is defined as follows: For $C_1, C_2 \in \mathcal{C}$ we have $(C_1, C_2) \in \mathbf{E}_{\mathcal{C}}$ if and only if both

(1) $Head(C_1) \in Cone_{\mathbf{G}_{\mathcal{C}}}(Head(C_2))$, and

(2) $Subg(C_1) \subseteq FC_{\mathcal{C}}(Subg(C_2))$.

Recall that $FC_{\mathcal{C}}(S)$ is the forward chaining closure of the set of variables S as defined in Section 2.

It is easy to see by the definitions of the implication graph and forward chaining that the clause graph $\mathbf{D}_{\mathcal{C}}$ is transitively closed. In the special case when $\mathcal{C} = \mathcal{I}(h)$ for a pure Horn function h we shall denote the clause graph of $\mathcal{C}$ by $\mathbf{D}_h$, furthermore, whenever the function h will be clear from the context, we shall simply write $\mathbf{D}$ instead of $\mathbf{D}_h$. In this latter case condition (1) simplifies to $(Head(C_1), Head(C_2)) \in \mathbf{A}_h$, due to the fact that $\mathbf{G}_h$ itself is transitively closed.

Let us remark that in the sequel, we shall primarily consider clause graphs of Horn functions, though some of those will be specified implicitly via the set of their implicates, such as exclusive components of a given function. For this reason we keep both notations $\mathbf{D}_{\mathcal{I}(h)} = \mathbf{D}_h$.

We shall also need to consider induced subgraphs of clause graphs. For given sets $\mathcal{B} \subseteq \mathcal{C}$ of pure Horn clauses, let us denote by $\mathbf{D}_{\mathcal{C}}(\mathcal{B})$ the subgraph of $\mathbf{D}_{\mathcal{C}}$ induced by the (vertex) set $\mathcal{B}$. Note that by the definitions $\mathbf{D}_{\mathcal{B}}$ is a subgraph of $\mathbf{D}_{\mathcal{C}}$, which may not be induced, i.e., in general we have $\mathbf{D}_{\mathcal{B}} \neq \mathbf{D}_{\mathcal{C}}(\mathcal{B})$. However, it is an induced subgraph when $\mathcal{C} = \mathcal{R}(\mathcal{B})$.

Lemma 5.7 *Given a set $\mathcal{B}$ of pure Horn clauses, let $\mathcal{C} = \mathcal{R}(\mathcal{B})$. Then, we have $\mathbf{D}_{\mathcal{B}} = \mathbf{D}_{\mathcal{C}}(\mathcal{B})$.*

Proof. By the definitions of implication graphs and forward chaining we have $Cone_{\mathbf{G}_{\mathcal{B}}}(v) = Cone_{\mathbf{G}_{\mathcal{C}}}(v)$ for every variable v, and $FC_{\mathcal{B}}(S) = FC_{\mathcal{C}}(S)$ for every set of variables S. ■

Let us also remark that while the implication graph $\mathbf{G}_h$ can be built in polynomial time from any given CNF representation $\eta = \phi(\mathcal{C})$, $\mathcal{C} \subseteq \mathcal{I}(h)$ of the function h (according to its definition and Theorem 2.7, it is enough to construct the transitive closure of the graph $\mathbf{G}_\eta$, which clearly can be done in $O(n^2 + |\eta|_l)$ time), it is quite clear that the same goal is impossible to achieve for the clause graph $\mathbf{D}_h$ simply because the set $\mathcal{I}(h)$ may be exponentially large with respect to the size of its CNF representation η. However, induced subgraphs of a clause graph can be built efficiently.

Lemma 5.8 *Given the pure Horn function h represented by a pure Horn CNF $\eta = \phi(\mathcal{C})$, $\mathcal{C} \subseteq \mathcal{I}(h) = \mathcal{R}(\mathcal{C})$, and two clauses $C_1, C_2 \in \mathcal{I}(h)$, it is possible to verify in linear $O(|\eta|_l)$ time whether (C_1, C_2) is an arc of $\mathbf{D}_h$, or not.*

Proof. Forward chaining works in linear time in the size of η. Since $Cone_{\mathbf{G}_\mathcal{C}}(Subg(C_2)) = Cone_{\mathbf{G}_h}(Subg(C_2))$, condition (1) can also be checked in linear time in the size of η by a direct labelling procedure using only the clauses of $\mathcal{C}$. ■

We can save on the above complexity somewhat when building a clause graph, by constructing all arcs entering an implicate essentially at the same price as constructing one of those arcs.

Corollary 5.9 *For any CNF representation η of the function h and subset $\mathcal{B} \subseteq \mathcal{I}(h)$ the induced subgraph $\mathbf{D}_h(\mathcal{B})$ can be built in $O(|\mathcal{B}|(|\eta|_l + |\mathcal{B}|_l))$ time. In particular, the induced subgraph $\mathbf{D}_h(\mathcal{C}(\eta))$ can be built in $O(|\eta|_c|\eta|_l) = O(m\ell)$ time, where $m = |\eta|_c$ is the number of clauses in η, and $\ell = |\eta|_l$ is the number of literals in η.* ■

Proof. Let us build first the implication graph $\mathbf{G}_h$ in $O(|\eta|_l)$ time. Next, compute the sets $FC_h(Subg(C))$ for all clauses of $\mathcal{B}$ in $O(|\mathcal{B}||\eta|_l)$ time, and store them in a $|\mathcal{B}| \times n$ binary matrix. Finally, for each $C \in \mathcal{B}$ we can check in $O(|C||\mathcal{B}|)$ time which other clauses of $\mathcal{B}$ are reachable from C, thus in another $O(|\mathcal{B}||\mathcal{B}|_l)$ time we can construct $\mathbf{D}_h(\mathcal{B})$. ■

5.3 Strong components of clause graphs

Let us now derive several important properties of clause graphs.

Lemma 5.10 *Let $\mathcal{C}$ be a set of Horn clauses, and $C_1, C_2 \in \mathcal{C}$ be two resolvable clauses such that $C = R(C_1, C_2) \in \mathcal{C}$. Then (C_1, C) and (C_2, C) are arcs in $\mathbf{D}_\mathcal{C}$.*

Proof. Let $C_1 = B_1 \vee v$, $C_2 = B_2 \vee \overline{v} \vee u$, and $C = R(C_1, C_2) = B_1 \vee B_2 \vee u$. Since $C_2 \in \mathcal{C}$ we get that (v, u) is an arc in the implication graph $\mathbf{G}_\mathcal{C}$ by definition, and so condition (1) in the definition of $\mathbf{D}_\mathcal{C}$ is satisfied for both (C_1, C) and (C_2, C).

To prove condition (2) we have to show that $B_1 \subseteq FC_\mathcal{C}(B_1 \cup B_2)$ and $B_2 \cup \{v\} \subseteq FC_\mathcal{C}(B_1 \cup B_2)$. Since the inclusions $B_1 \subseteq FC_\mathcal{C}(B_1 \cup B_2)$ and $B_2 \subseteq FC_\mathcal{C}(B_1 \cup B_2)$ are trivial, we need only to show that $v \in FC_\mathcal{C}(B_1 \cup B_2)$, which follows by the fact that $C_1 = B_1 \vee v$ is a clause of $\mathcal{C}$ and hence $v \in FC_\mathcal{C}(B_1) \subseteq FC_\mathcal{C}(B_1 \cup B_2)$. ■

Theorem 5.11 *Let $\mathcal{C}$ be a set of Horn clauses, and $\mathcal{I} \subseteq \mathcal{C}$ be an initial set in $\mathbf{D}_\mathcal{C}$. Then $\mathcal{I}$ is an exclusive subset of $\mathcal{C}$.*

Proof. Let $C_1, C_2 \in \mathcal{C}$ be two resolvable clauses such that $C = R(C_1, C_2) \in \mathcal{C}$. Then by Lemma 5.10 $(C_1, C) \in \mathbf{E}_\mathcal{C}$ and $(C_2, C) \in \mathbf{E}_\mathcal{C}$ and hence both $C_1 \in \mathcal{I}$ and $C_2 \in \mathcal{I}$, because $\mathcal{I}$ is an initial set in $\mathbf{D}_\mathcal{C}$. ■

Let us show next some important properties of the strong components of the clause graph.

Theorem 5.12 *Let h be a pure Horn function, and $\mathcal{K}$ be a strong component of $\mathbf{D}_h$. Then, either $\mathcal{K}$ is redundant or $\mathcal{K}$ contains a nonempty essential set of clauses.*

Proof. Let us note first that since $\mathcal{K}$ is a strong component of $\mathbf{D} = \mathbf{D}_h$, both $Cone_\mathbf{D}(\mathcal{K})$ and $\mathcal{X} = Cone_\mathbf{D}(\mathcal{K}) \setminus \mathcal{K}$ are initial sets of $\mathbf{D}$, and hence by Theorem 5.11 both of them are exclusive (for h). Furthermore, by Lemma 4.12 the set $\mathcal{S} = \mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$ is redundant. Thus, if $\mathcal{K} \subseteq \mathcal{S}$, then $\mathcal{K}$ itself is redundant. On the other hand, if $\mathcal{K} \not\subseteq \mathcal{S}$, then the set $\mathcal{E} = \mathcal{K} \setminus \mathcal{R}(\mathcal{X}) = Cone_\mathbf{D}(\mathcal{K}) \setminus \mathcal{R}(\mathcal{X}) \subseteq \mathcal{K}$ is essential by (c) of Lemma 4.15. ■

An important consequence of the above theorem is that the number of non-redundant strong components of $\mathbf{D}$ is limited by the size of an arbitrary representation of h.

Corollary 5.13 *Given an arbitrary irreddundant representation $\mathcal{C} \subseteq \mathcal{I}(h)$ of h, the strong components of the graph $\mathbf{D}_\mathcal{C} = \mathbf{D}(\mathcal{C})$ are in a one-to-one correspondence with the non-redundant strong components of $\mathbf{D}$.*

Proof. For any strong component $\mathcal{K}$ of $\mathbf{D}$, the set $\mathcal{K} \cap \mathcal{C}$ forms a strong component of $\mathbf{D}(\mathcal{C})$, by definition of an induced subgraph. Since $\mathcal{R}(\mathcal{C}) = \mathcal{I}(h)$, we have the equality $\mathbf{D}_\mathcal{C} = \mathbf{D}(\mathcal{C})$ by Lemma 5.7. Thus, by Proposition 4.16 and Theorem 5.12, non-redundant strong components of $\mathbf{D}$ correspond in a one to one way to the (nonempty) strong components of $\mathbf{D}_\mathcal{C}$. ■

Now we are ready to describe an efficient procedure to construct a chain of exclusive sets, in the spirit of Lemma 4.13.

Theorem 5.14 *Let h be a pure Horn function on n variables with no unit implicates, represented by an irredundant and prime CNF $\eta = \phi(\mathcal{C})$, $\mathcal{C} \subseteq \mathcal{I}(h)$. Let $K_1, \ldots, K_t$ be all strong components of graph $D_\mathcal{C}$ sorted according to some topological order. Let us define sets of clauses $\mathcal{X}_0 = \emptyset$ and*

$$\mathcal{X}_i = \bigcup_{j=1}^{i} Cone_{\mathbf{D}_h}(K_j),$$

then the following properties are satisfied:

(i) $\mathcal{X}_{i-1}$ is an exclusive subset of $\mathcal{X}_i$ for $i = 1, \ldots, t$;

(ii) $\mathcal{X}_t$ is an exclusive subset of $\mathcal{I}(h)$ and $\mathcal{R}(\mathcal{X}_t) = \mathcal{I}(f)$;

(iii) $K_i = \mathcal{C} \cap (\mathcal{X}_i \setminus \mathcal{X}_{i-1})$ *and the set* K_i *is the unique terminal strong component of* $\mathbf{D}_{Cone_{\mathbf{D}_{\mathcal{X}_i \cap \mathcal{C}}}(K_i)}$*, for* $i = 1, ..., t$*;*

(iv) the strong components $K_1, \ldots, K_t$ *can be found in time* $O(n^2 + m \cdot \ell)$*, where* m *is the number of clauses of* η *and* ℓ *is the number of literals of* η*.*

Proof. Since $\mathcal{X}_{i-1}$ is by definition an initial set of graph $\mathbf{D}_h$ for every $i = 1, \ldots, t$, it is an exclusive subset of $\mathcal{I}(h)$ according to Theorem 5.11. Therefore according to Lemma 4.4, proposition (b), it is also an exclusive subset of $\mathcal{X}_i$. Therefore property (i) is satisfied.

Since $\mathcal{C} \subseteq \mathcal{X}_t$ and $\eta = \phi(\mathcal{C})$ is a prime representation of h, clearly $\mathcal{R}(\mathcal{X}_t) = \mathcal{I}(h)$. Because $\mathcal{X}_t$ is an initial set of graph D_h, it is also an exclusive subset of $\mathcal{I}(h)$ due to Theorem 5.11. Hence (ii) is satisfied.

The fact that $K_i = \mathcal{C} \cap (\mathcal{X}_i \setminus \mathcal{X}_{i-1})$ follows directly from the definition of sets $\mathcal{X}_i$ and $\mathcal{X}_{i-1}$. Set K_i is clearly the unique strong component of $D_{Cone_{\mathbf{D}_{\mathcal{X}_i \cap \mathcal{C}}}(K_i)}$ by the definition of $Cone_{\mathbf{D}_{\mathcal{X}_i \cap \mathcal{C}}}(K_i)$. Hence also property (iii) is satisfied.

Now, let us examine, how quickly we can find strong components $K_1, \ldots, K_t$. We first construct an implication graph $\mathbf{G}_{\mathcal{C}}$, which can be done in $O(l)$ time and then its transitive closure $\mathbf{G}_h$ can be found in $O(n^2)$ time. Therefore $O(n^2 + l)$ time suffices to perform this step.

Then the subgraph $\mathbf{D}_{\mathcal{C}}$ of the clause graph $\mathbf{D}_h$ is constructed. For this purpose it is necessary to test for each pair of clauses $C_1, C_2 \in \mathcal{C}$, whether (C_1, C_2) is an arc in $\mathbf{D}_{\mathcal{C}}$, which amounts to verifying the conditions:

1. $(Head(C_1), Head(C_2))$ is an arc in $\mathbf{G}_h$ or $Head(C_1) = Head(C_2))$. This can be tested in constant time because $\mathbf{G}_h$ is already constructed (assume for simplicity that the first step constructs its adjacency matrix).

2. $Subg(C_1) \subseteq FC_h(Subg(C_2)))$. In order to test this condition we shall first construct for each clause C the set $FC_h(Subg(C))$. By Lemma 2.5, this takes $O(l)$ time per clause and thus $O(m \cdot l)$ time in total (the results can be stored e.g. in an $m \times n$ matrix). Now we can easily decide whether $Subg(C_1) \subseteq FC_h(Subg(C_2))$ in $O(|C_1|)$ time, and thus we can find all arcs leading to the clause C_2 in $O(l)$ time.

Therefore all arcs of graph $D_{\mathcal{C}}$ can be discovered in $O(m \cdot l)$ time. Strong components and their topological order can be found in time linear in the size of $D_{\mathcal{C}}$, i.e. in $O(m^2) = O(m \cdot l)$ time. By this also the property (iv) is satisfied. ■

An important consequence of the above theorem is that the problem of Horn Minimization can be reduced to the following special incremental problem, of finding the optimal representation of a terminal strong component of the clause graph (or in short, the problem of ORTSC).

PROBLEM ORTSC($\mathcal{F}, \mathcal{Q}$)

Input: A pure Horn function g, represented by a prime and irredundant set of clauses $\mathcal{F} \cup \mathcal{Q}$, for which $\mathcal{F} \subseteq \mathcal{X}$ for some exclusive subset $\mathcal{X}$ of $\mathcal{I}(g)$ and $\mathcal{Q} \subseteq \mathcal{K} = \mathcal{I}(g) \setminus \mathcal{X}$, and where $\mathcal{K}$ is the unique terminal strong component of the clause graph $\mathbf{D}_g$ (in other words, $\mathcal{I}(g) = \mathcal{X} \cup \mathcal{K} = Cone_{\mathbf{D}_g}(\mathcal{K})$).

Output: A minimum cardinality subset $\mathcal{Q}^* \subseteq \mathcal{K}$, such that $\mathcal{R}(\mathcal{F} \cup \mathcal{Q}^*) = \mathcal{R}(\mathcal{F} \cup \mathcal{Q}) = \mathcal{I}(g)$.

Theorem 5.15 *If problem ORTSC can be solved in polynomial time, then problem HM can be solved in polynomial time.*

Proof. Given a Horn function represented by a Horn CNF, let us first bring it to a prime and irredundant form. We can then delete the negative clauses, according to Propositions 4.6 and 4.17. Let us next perform the preprocessing steps as described in the beginning of Section 5. All these can be done in $O(\ell^2)$ time, and reduces the minimization of the original input to the minimization of a pure Horn function h which has no unit prime implicates, and which is represented by an irredundant and prime set $\mathcal{C}$ of clauses.

Let us next find strong components $K_1, \ldots, K_t$ of graph $\mathbf{D}_\mathcal{C}$ and their topological order in time $O(n^2 + m \cdot \ell)$ as in Theorem 5.14, and let $\mathcal{X}_1, \ldots, \mathcal{X}_t$ be defined as in Theorem 5.14.

For every $i = 1, \ldots, t$ the set $\mathcal{X}_i \setminus \mathcal{X}_{i-1}$ obviously contains the (non-redundant) strong component L_i of the graph $\mathcal{D}_h$ which corresponds by Corollary 5.13 to the strong component K_i of the graph $\mathbf{D}_\mathcal{C}$. In other words $K_i = \mathcal{C} \cap L_i \subseteq L_i \subseteq \mathcal{X}_i \setminus \mathcal{X}_{i-1}$ holds (where the last inclusion may be proper, if $\mathcal{X}_i \setminus \mathcal{X}_{i-1}$ contains some redundant strong components of $\mathbf{D}_h$). Therefore, in order to satisfy the assumptions of Lemma 4.13 (which yields a minimum cardinality set of clauses which represents h and thus solves HM) it suffices to show, that by solving an instance of ORTSC with suitable input we can find for every $i = 1, \ldots, t$ a subset $K_i^* \subseteq \mathcal{X}_i \setminus \mathcal{X}_{i-1}$ such that

$$K_i^* \text{ is a minimum cardinality set such that } \mathcal{R}(\mathcal{X}_{i-1} \cup K_i^*) = \mathcal{R}(\mathcal{X}_i). \tag{3}$$

The set $\mathcal{X}_i$ can be split into two disjoint parts: $\mathcal{V} = Cone_{\mathbf{D}_h}(K_i)$ and $\mathcal{W} = \mathcal{X}_i \setminus \mathcal{V}$. Let us denote $\mathcal{Y} = \mathcal{V} \cap \mathcal{X}_{i-1}$. Note that both $\mathcal{V}$ and $\mathcal{Y}$ are again initial sets of $\mathbf{D}_h$, so both are exclusive subsets of $\mathcal{I}(h)$ by Theorem 5.11. Moreover $\mathcal{W} = \mathcal{X}_{i-1} \setminus \mathcal{Y}$ holds and so $\mathcal{X}_{i-1}$ can be rewritten as disjoint union of $\mathcal{W}$ and $\mathcal{Y}$. Thus (3) can be rewritten as

$$K_i^* \text{ is a minimum cardinality set such that } \mathcal{R}(\mathcal{W} \cup \mathcal{Y} \cup K_i^*) = \mathcal{R}(\mathcal{W} \cup \mathcal{V}). \tag{4}$$

Let us show that condition (3) is equivalent to the condition

$$K_i^* \text{ is a minimum cardinality set such that } \mathcal{R}(\mathcal{Y} \cup K_i^*) = \mathcal{R}(\mathcal{V}) \tag{5}$$

To show that (4) implies (5) it suffices to use Lemma 4.5 where we set $\mathcal{X} = \mathcal{V}$ and $\mathcal{C} = \mathcal{W} \cup \mathcal{Y} \cup K_i^*$ and $\mathcal{C} = \mathcal{W} \cup \mathcal{Y} \cup K_i^*$. For the reverse implication we need to use Lemma 4.2 twice. First we set $\mathcal{X} = \mathcal{W}$ and $\mathcal{Z} = \mathcal{Y} \cup K_i^*$ obtaining $\mathcal{R}(\mathcal{W} \cup (\mathcal{Y} \cup K_i^*)) = \mathcal{R}(\mathcal{W} \cup \mathcal{R}(\mathcal{Y} \cup K_i)) = \mathcal{R}(\mathcal{W} \cup \mathcal{R}(\mathcal{V}))$. In the next step we set $\mathcal{X} = \mathcal{W}$ and $\mathcal{Z} = \mathcal{V}$ getting $\mathcal{R}(\mathcal{W} \cup \mathcal{R}(\mathcal{V})) = \mathcal{R}(\mathcal{W} \cup \mathcal{V})$.

A set K_i^* satisfying condition (5) can be found by solving ORTSC for the instance in which we set g to be the $\mathcal{Y}$-component of h, $\mathcal{F} = \mathcal{C} \cap \mathcal{Y} = Cone_{D_\mathcal{C}}(K_i) \setminus K_i$, and $\mathcal{Q} = \mathcal{C} \cap (\mathcal{V} \setminus \mathcal{Y}) = K_i$. According to (i) – (iii) of Theorem 5.14, the function g represented by $\mathcal{F} \cup \mathcal{Q} = \mathcal{C} \cap \mathcal{X}_i$ satisfies the conditions of an input for ORTSC. Thus the set $K_i^* = Q^*$ which is a solution of ORTSC to this input satisfies condition (5) and therefore by the above argumets also condition (3). Hence the sets $K_1^*, \ldots, K_t^*$ satisfy requirements of Lemma 4.13, and $\mathcal{C}^* = \bigcup_{i=1}^t K_i^*$ is a minimum cardinality set representing h. ■

Remark 5.16 *If ORTSC is reformulated to output a set $\mathcal{Q}^*$ consisting of a minimum number of literals such that $\mathcal{R}(\mathcal{F} \cup \mathcal{Q}^*) = \mathcal{R}(\mathcal{F} \cup \mathcal{Q})$, then the proof of Theorem 5.15 can be easily modified to show that using ORTSC one can find a representation of a given input pure Horn function with the minimum number of literals. In this case the minimization procedure does not necessarily work for Horn CNFs which contain negative clauses. The same number of negative clauses may have a different total number of literals, so dropping them in the preprocessing step cannot be justified in this case (see Section 6.2 for more details on the minimization of the number of literals).*

5.4 Switching lemmas

To arrive at a polynomial algorithm for Horn minimization (at least for CQ-functions) we need to tackle problem ORTSC, according to Theorem 5.15. For this, we shall need to analyze further the structure of clause graphs and in particular, the structure of clause graphs of CQ-functions.

Let us remark that until this point, all of our statements remained valid for arbitrary additive complexity measure of CNF-s, in particular, for both measures $|\eta|_l$ and $|\eta|_c$. From now on however, we need to restrict our claims to the minimality of the number of clauses, i.e., to measure $|\eta|_c$.

In this subsection, we prove a series of "switching" claims for pure Horn, and more specifically for CQ functions. In each of these claims some literals of implicates connected in the clause graph are switched, resulting in another implicate of h. We assume h to be a fixed pure Horn function, and simply write $\mathbf{D}$ instead of $\mathbf{D}_h$, and $\mathbf{G}$ instead of $\mathbf{G}_h$.

Lemma 5.17 *Let $A \vee u$, $B \vee v \in \mathcal{I}(h)$ be implicates of h such that $(B \vee v, A \vee u)$ is an arc of the clause graph $\mathbf{D}$. Then, there exists a subset $A' \subseteq A$ such that $A' \vee v$ is a prime implicate of h.*

Proof. Since $(B \vee v, A \vee u)$ is an arc of $\mathbf{D}$, we have $B \subseteq FC_h(A)$ by definition, and thus $v \in FC_h(A)$ follows, since $B \vee v \in \mathcal{I}(h)$ is assumed. Therefore, $A \vee v$ is an implicate of h by Lemma 2.4. Consequently, there must exist a pure Horn prime implicate $A' \vee v$ subsuming $A \vee v$. ■

It follows from the definition of the clause graph $\mathbf{D}$ that whenever both (C_1, C_2) and (C_2, C_1) are arcs in $\mathbf{D}$ for some pair of clauses $C_1, C_2 \in \mathcal{I}(h)$, then both $(Head(C_1), Head(C_2))$ and $(Head(C_2), Head(C_1))$ are arcs in the implication graph $\mathbf{G}$. Therefore, all clauses from the same strong component of $\mathbf{D}$ must have their heads in a single strong component of $\mathbf{G}$ (however, heads of clauses from several strong components of $\mathbf{D}$ may also belong to the same strong component of $\mathbf{G}$). This fact allows us to state the following definition.

Definition 5.18 *For a strong component $\mathcal{K}$ of the clause graph $\mathbf{D}$ let us denote by $Q(\mathcal{K})$ the strong component of the implication graph $\mathbf{G}$ which contains the heads of the clauses belonging to $\mathcal{K}$.*

In the rest of this section we shall separate, for each implicate $C \in \mathcal{I}(h)$, the subgoals of C which are in the same strong component of the implication graph $\mathbf{G}$ as its head $Head(C)$ from those subgoals of C which are in preceding strong components. This separation will allow us to state and prove several stronger "switching" results. Let us start with a simple auxiliary lemma.

Lemma 5.19 *Let $\mathcal{K}$ be a strong component of graph $\mathbf{D}$ and let $(A \vee X \vee u)$, $(B \vee Y \vee v) \in \mathcal{K}$ be two implicates of h, such that $A \cap Q(\mathcal{K}) = \emptyset$, $B \cap Q(\mathcal{K}) = \emptyset$, and $X, Y \subseteq Q(\mathcal{K})$. Let further $I = Cone_{\mathbf{G}}(Q(\mathcal{K})) \setminus Q(\mathcal{K})$ be an initial set of $\mathbf{G}$, and denote, as before, by h_I the $Clauses(I)$-component of h, and let $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$. Then, we have*

$$B \subseteq FC_{h_I}(A) \cap FC_{\mathcal{X}}(A) \subseteq FC_h(A) \qquad \text{and} \qquad A \subseteq FC_{h_I}(B) \cap FC_{\mathcal{X}}(B) \subseteq FC_h(B).$$

Proof. Let us first recall that h_I is well defined by Definition 5.5 (applied to the initial set I of the implication graph $\mathbf{G}$). Let us note next that, since $\mathcal{K}$ is a strong component, $(B \vee Y \vee v, A \vee X \vee u)$ is an arc in $\mathbf{D}$, and thus the inclusion $B \cup Y \subseteq FC_h(A \cup X)$ holds (by the definition of the clause graph $\mathbf{D}$). By Lemma 5.6 applied to the initial set I and the set of variables $A \cup X$ we get

$$B = (B \cup Y) \cap I \subseteq FC_h(A \cup X) \cap I = FC_{h|_I}((A \cup X) \cap I) = FC_{h|_I}(A) \subseteq FC_h(A).$$

Consider next an arbitrary variable $b \in B \setminus A$, and choose a minimal set $\mathcal{C}$ of prime implicates of h_I such that $b \in FC_{\mathcal{C}}(A)$ holds. We claim that $\mathcal{C} \subseteq \mathcal{X}$. To see this claim, let us note first that $\bigcup_{C \in \mathcal{C}} Vars(C) \subseteq Cone_{\mathbf{G}}(b)$ follows by Lemma 5.4, implying that $(Head(C), b)$ is an arc in $\mathbf{G}$ for every clause $C \in \mathcal{C}$. We also have (b, v) as an arc in $\mathbf{G}$ for all $b \in B$ by Lemma 2.9, since $B \vee Y \vee v$ is an implicate of h. Finally, we have (v, u) as an arc of $\mathbf{G}$, since $A \vee X \vee u$ and $B \vee Y \vee v$ are implicates from the same strong component $\mathcal{K}$ of $\mathbf{D}$. Since $\mathbf{G}$ is

transitively closed, these imply that $(Head(C), u)$ is an arc in $\mathbf{G}$ for all $C \in \mathcal{C}$. On the other hand, for all $C \in \mathcal{C}$ we have $Subg(C) \subseteq FC_{\mathcal{C}}(A) \subseteq FC_h(A) \subseteq FC_h(A \cup X)$ by our choice of $\mathcal{C}$. Thus, $(C, A \vee X \vee u) \in \mathbf{A}_h$ follows, implying $\mathcal{C} \subseteq Cone_{\mathbf{D}}(\mathcal{K})$. Since $\mathcal{C} \subseteq Clauses(I)$ and $Clauses(I) \cap \mathcal{K} = \emptyset$, $\mathcal{C} \subseteq \mathcal{X}$ follows, as claimed.

Applying the above claim for all $b \in B \setminus A$, we get $B \subseteq FC_{\mathcal{X}}(A) \subseteq FC_h(A)$, completing the proof of the first part of the theorem.

The second part, i.e., the inclusion $A \subseteq FC_{h_I}(B) \cap FC_{\mathcal{X}}(B) \subseteq FC_h(B)$ can be proved analogously by interchanging the roles of clauses $A \vee X \vee u$ and $B \vee Y \vee v$. ■

Now we are ready to state the main result of this subsection, a generalization of Lemma 5.17.

Theorem 5.20 (Switching Theorem) *Let $\mathcal{K}$ be a strong component of the clause graph $\mathbf{D} = \mathbf{D}_h$ and let $A \vee X \vee u,\ B \vee Y \vee v \in \mathcal{K}$ be two implicates of h, such that $A \cap Q(\mathcal{K}) = \emptyset$, $B \cap Q(\mathcal{K}) = \emptyset$, and $X, Y \subseteq Q(\mathcal{K})$. Then, there exist subsets $A' \subseteq A$ and $Y' \subseteq Y$, such that*

(a) $A' \vee Y' \vee v$ is a prime implicate of h belonging to $Cone_{\mathbf{D}}(\mathcal{K})$.

(b) Additionally, if $B \vee Y \vee v$ is a prime implicate of h and $Y \neq \emptyset$, then also $Y' \neq \emptyset$.

(c) Furthermore, if $B \vee Y \vee v$ is not redundant, then $A' \vee Y' \vee v \in \mathcal{K}$.

(d) Finally, if $A \vee X \vee u$ is prime, and $A' \vee Y' \vee v \in \mathcal{K}$, then $A = A'$.

Proof. By Lemma 5.19 we get $B \subseteq FC_h(A)$. Moreover, $B \vee Y \vee v$ is an implicate of h, and so it follows that $v \in FC_h(A \cup Y)$, which by Lemma 2.4 implies that $A \vee Y \vee v$ is an implicate of h. Thus there must exist sets $A' \subseteq A$ and $Y' \subseteq Y$, such that $A' \vee Y' \vee v \in \mathcal{I}^p(h)$. Furthermore, since $(B \vee Y \vee v, A \vee X \vee u)$ is an arc in the clause graph $\mathbf{D}$, (v, u) must be an arc of $\mathbf{G}$ and $A' \cup Y' \subseteq A \cup Y \subseteq A \cup B \cup Y \subseteq FC_h(A \cup X)$ must hold, implying that $(A' \vee Y' \vee v, A \vee X \vee u)$ is an arc in the implication graph $\mathbf{D}$, completing the proof of (a).

To see (b), let us assume that $B \vee Y \vee v \in \mathcal{I}^p(h)$, $Y \neq \emptyset$, and by contradiction that $Y' = \emptyset$. These imply that $A' \vee v \in \mathcal{I}^p(h)$ and so $v \in FC_h(A')$. By Lemma 5.19 we get $A' \subseteq A \subseteq FC_h(B)$, implying $v \in FC_h(B)$, from which $B \vee v \in \mathcal{I}(h)$ would follow, contradicting the primality of $B \vee Y \vee v$.

Next, to see (c), let us define $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$, and observe that $A \subseteq FC_{\mathcal{X}}(B)$ follows by Lemma 5.19, implying $A' \cup Y' \subseteq A \cup Y \subseteq FC_{\mathcal{X}}(B \cup Y)$. Now, if $A' \vee Y' \vee v$ were not belonging to $\mathcal{K}$, then by (a) it would belong to $\mathcal{X}$, and thus $v \in FC_{\mathcal{X}}(B \cup Y)$ would also follow, implying $B \vee Y \vee v \in \mathcal{R}(\mathcal{X}) \setminus \mathcal{X}$, that is that $B \vee Y \vee v$ is redundant by Lemma 4.12, since $\mathcal{X}$ is an exclusive set.

Finally, to see (d), let us apply (a) for the pair $A' \vee Y' \vee v, A \vee X \vee u \in \mathcal{K}$ of clauses, and get that $A'' \vee X'' \vee u$ is also an implicate of h for some $X'' \subseteq X$ and $A'' \subseteq A' \subseteq A$. Since $A \vee X \vee u$ is assumed to be prime, $A'' = A' = A$ and $X'' = X$ are implied. ■

Corollary 5.21 *Let $\mathcal{C}$ be a prime and irredundant representation of the pure Horn function h, $\mathcal{K}$ be a strong component of $\mathbf{D}$, and $A \vee X \vee u,\ B \vee Y \vee v \in \mathcal{C} \cap \mathcal{K}$. Then, there exists a subset $Y' \subseteq Y$ such that the set $\mathcal{C}' = (\mathcal{C} \setminus \{B \vee Y \vee v\}) \cup \{A \vee Y' \vee v\}$ is again a prime and irredundant representation of h. Furthermore, if $|Y| \leq 1$, then we must have $Y' = Y$.*

Proof. By (a), (c) and (d) of Theorem 5.20 we know that there exists a subset $Y' \subseteq Y$ such that $A \vee Y' \vee v \in \mathcal{K}$ is a prime implicate of h. Thus both the primeness and irredundancy of $\mathcal{C}'$ is implied trivially by that of $\mathcal{C}$. The only claim remained to show is that $\mathcal{C}'$ represents h, that is that $B \vee Y \vee v \in \mathcal{R}(\mathcal{C}')$, or equivalently by Lemma 2.4 that $v \in FC_{\mathcal{C}'}(B \cup Y)$.

Let us denote by $\mathcal{X} = Cone_{\mathbf{D}}(\mathcal{K}) \setminus \mathcal{K}$ and note that by Lemma 5.19 we have $A \subseteq FC_{\mathcal{X}}(B)$. Thus, by $\mathcal{C} \cap \mathcal{X} = \mathcal{C}' \cap \mathcal{X}$, we get $A \subseteq FC_{\mathcal{C}'}(B)$, from which $A \cup Y' \subseteq A \cup Y \subseteq FC_{\mathcal{C}'}(B \cup Y)$ follows. Since we also have $A \vee Y' \vee v \in \mathcal{C}'$, $v \in FC_{\mathcal{C}'}(B \cup Y)$ follows too, as claimed.

Finally, by (b) of Theorem 5.20, we have $Y' \neq \emptyset$ whenever $Y \neq \emptyset$, which implies $Y' = Y$, since $|Y| \leq 1$ is assumed. ■

5.5 A classification of strong components of clause graphs

In this section we introduce a useful classification of the strong components of the clause graph $\mathbf{D}$ by the number of their subgoal variables which belong to the same strong component of the implication graph as their head. Let us start with an analogous classification of the clauses.

Definition 5.22 *A pure Horn clause $C \in \mathcal{I}(h)$ is said to be of type i (with respect to h), if exactly i subgoals of C belong to the same strong component of the graph $\mathbf{G}$ as the head of C.*

Let us note next an easy consequence of Lemma 5.19.

Corollary 5.23 *Let $\mathcal{K}$ be a strong component of the clause graph $\mathbf{D}$ and let $(A \vee X \vee u)$, $(B \vee Y \vee v) \in \mathcal{K}$ be two prime implicates of h, such that $A \cap Q(\mathcal{K}) = \emptyset$, $B \cap Q(\mathcal{K}) = \emptyset$, and $X, Y \subseteq Q(\mathcal{K})$. Then, if $X = \emptyset$ we must also have $Y = \emptyset$.*

Proof. Since $(B \vee Y \vee v, A \vee X \vee u)$ is an arc of $\mathbf{D}$, we have $B \cup Y \subseteq FC_h(A \cup X)$ by the definition of the arcs in the clause graph. By Lemma 5.19 we also have $A \subseteq FC_h(B)$. Thus, if $X = \emptyset$, then $Y \subseteq FC_h(B)$ follows, implying by Lemma 2.4 that $B \vee v$ is also an implicate of h. Thus, $Y = \emptyset$ is implied by the primality of $B \vee Y \vee v$. ■

Corollary 5.24 *Let h be a pure Horn function, and let $\mathcal{K}$ be a strong component of $\mathbf{D}$. Then, either all or none of the prime implicates belonging to $\mathcal{K}$ are of type 0.*

Proof. Immediate by Corollary 5.23. ■

If h is a CQ function then Corollary 5.24 justifies a complete classification of strong components of $\mathbf{D}$. Indeed, every prime implicate of h is either of type 0 or of type 1 and hence every strong component of $\mathbf{D}$ either contains only prime implicates of type 0 or it contains only prime implicates of type 1.

Definition 5.25 *Let $\mathcal{K}$ be a strong component of the graph $\mathbf{D}$. Then we say that $\mathcal{K}$ is of type 0 if all prime implicates belonging to $\mathcal{K}$ are of type 0, and we say that $\mathcal{K}$ is of type 1 if all prime implicates belonging to $\mathcal{K}$ are of type 1.*

6 Algorithm for Minimizing CQ Functions

Now we are ready to design an algorithm for the minimization of CQ functions.

Algorithm 6.1 (Minimization of CQ Functions)
Input: Irredundant and prime CNF $\mathcal{C}$ representing a function h from the class CQ.
Output: A CNF $\mathcal{C}_{min}$ representing h such that $\mathcal{C}_{min}$ has the minimum possible number of clauses.

1. [**Implication graph**] Construct the graph $\mathbf{G}_{\mathcal{C}}$, its transitive closure $\mathbf{G}_h$ and find its strong components.

2. [**Clause graph**] Construct the graph $\mathbf{D}_{\mathcal{C}}$ (i.e. the subgraph of $\mathbf{D}_h$ induced by the clauses in $\mathcal{C}$), find its strong components, and find some topological order $T = (K_1, \ldots K_t)$ of these strong components.

3. [**Main loop**] Process the strong components of the graph $\mathbf{D}_{\mathcal{C}}$ in the topological order T found in the preceding step, and for each component K perform one of the following actions:

   - **Action $\mathcal{K}_0$:** If the component K is of type $\mathcal{K}_0$, then do nothing.
   - **Action $\mathcal{K}_1$:** If the component K is of type $\mathcal{K}_1$, then do the following:
     (a) Switch all subgoal sets outside of $Q(K)$ of all clauses in K to some representative set $A(K)$ (i.e. to a set of subgoals outside of $Q(K)$ of an arbitrary clause from K) obtaining set of clauses K' and new representation $\mathcal{C}'$.
     (b) Construct the subgraph $Q'(K')$ of $Q(K')$ whose arcs are generated only by the clauses from $Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$ and mark all arcs generated by clauses from $(Cone_{\mathbf{D}_{\mathcal{C}'}}(K') \setminus K')$ as *fixed* and all arcs generated by clauses from K' as *free*.

(c) Find the transitive reduction of $Q'(K')$ with the added restriction that the reduction must contain all fixed arcs. Free arcs can be removed and replaced by other arcs (however the transitive closure of $Q'(K')$ must of course remain the same). This task can be performed using a generalization of the augmentation algorithm described in [9] and [23].

(d) For each removed arc, remove from $\mathcal{C}'$ the corresponding clause. For each added arc, add to $\mathcal{C}'$ the corresponding quadratic clause and extend the clause by adding the representative set $A(K)$ of subgoals. Update the graph $\mathbf{D}_{\mathcal{C}'}$ accordingly.

4. [**Output**] Output the final CNF $\mathcal{C}$.

The rest of this section is organized as follows. Subsection 6.1 proves the correctness of Algorithm 6.1, i.e. it shows that Algorithm 6.1 outputs a minimum CNF representation of the CQ-function given by the input CNF. Subsection 6.2 shows that Algorithm 6.1 not only minimizes the number of clauses but it also outputs a minimum CNF with respect to the number of literals. Finally, Subsection 6.3 deals with the time complexity of Algorithm 6.1 and proves that it runs in $O(n^2 + m \cdot \ell)$ time where n is the number of variables, m is the number of clauses, and ℓ is the number of literals in the input CNF.

6.1 Correctness of Algorithm 6.1

Theorem 6.2 *Algorithm 6.1 works correctly and outputs a minimum CNF representation of the given input function h.*

Proof. Algorithm 6.1 follows the idea from the proof of Theorem 5.15. The only thing we have to prove is, that we are able to solve problem ORTSC if we restrict our attention to CQ functions. Action $\mathcal{K}_0$ solves problem ORTSC in case of strong components of type $\mathcal{K}_0$ and its correctness is justified by Lemma 6.3. Action $\mathcal{K}_1$ solves problem ORTSC in case of strong components of type $\mathcal{K}_1$ and its correctness is justified by Lemma 6.4.

In both lemmas we will use the following notation (it is same notation as in the proof of Theorem 5.15):

$$\begin{aligned} L_i &= \text{ strong component of graph } \mathbf{D}_h \text{ corresponding to } K_i \\ \mathcal{V} &= Cone_{\mathbf{D}_h}(K_i) \\ \mathcal{Y} &= Cone_{\mathbf{D}_h}(K_i) \cap \bigcup_{j=1}^{i-1} Cone_{\mathbf{D}_h}(K_j) \end{aligned}$$

Note that $Q = K_i$ and $\mathcal{F} = \mathcal{C} \cap \mathcal{Y} = Cone_{D_\mathcal{C}}(K_i) \setminus K_i$ in the notation from the definition of ORTSC. Moreover, note that $L_i \subseteq \mathcal{V} \setminus \mathcal{Y}$, where the inclusion is proper if $\mathcal{X}_i \setminus \mathcal{X}_{i-1}$ (where $\mathcal{X}_i = \bigcup_{j=1}^{i} Cone_{\mathbf{D}_h}(K_j)$) contains some redundant strong component of $\mathbf{D}_h$.

Lemma 6.3 (Action $\mathcal{K}_0$) *Let K_i be of type $\mathcal{K}_0$, then K_i is a minimum cardinality set such that $\mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$.*

Proof. We shall show that the irredundancy of $\mathcal{C}$ is in this case sufficient for the intersection $\mathcal{C} \cap L_i$ to have the minimum possible cardinality and no further action is required. Let us denote $K_i = \mathcal{C} \cap L_i = \{A_1 \vee u_1, \ldots, A_l \vee u_l\}$. By Corollary 5.21 we may switch the subgoal sets of all clauses in K_i to some representative set A, such that after this switch we get $K' = \mathcal{C}' \cap L_i = \{A \vee u_1, \ldots, A \vee u_l\}$ and $\mathcal{C}'$ is still a prime and irredundant representation of h. The irredundancy of $\mathcal{C}'$ now implies that for every $i \neq j$ we have $u_i \neq u_j$. Moreover, by Proposition 4.7 we have $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$.

Now let us assume by contradiction that there exists a set of clauses $P = \{B_1 \vee w_1, \ldots, B_l \vee w_{l'}\}$ where $l' < l$ such that $\mathcal{R}(\mathcal{Y} \cup P) = \mathcal{R}(\mathcal{V})$. By Proposition 4.7 and exclusiveness of $\mathcal{Y}$ and $\mathcal{V}$ it follows that replacing K' by P produces again a CNF representation of h and we may w.l.o.g. assume that P is such that this CNF is irredundant and prime. Now we may once more use Corollary 5.21 to switch the subgoal sets of all clauses in P to some representative set B, such that after this switch we get $P' = \mathcal{C}'' \cap L_i = \{B \vee w_1, \ldots, B \vee w_{l'}\}$ and $\mathcal{C}''$ is still a prime and irredundant representation of h. Clearly, after such a switch we again have $\mathcal{R}(\mathcal{Y} \cup P') = \mathcal{R}(\mathcal{Y} \cup P) = \mathcal{R}(\mathcal{V})$.

Due to irredundancy of $\mathcal{C}'$ and $\mathcal{C}''$ no clause in $K' \cup P'$ can be in $\mathcal{R}(\mathcal{Y})$. On the other hand $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup P')$ implies $K' \subseteq \mathcal{R}(\mathcal{Y} \cup P')$ and $P' \subseteq \mathcal{R}(\mathcal{Y} \cup K')$. Therefore any forward chaining derivation of a variable u_k, $k = 1, \ldots, l$, from the set A using the set of clauses $\mathcal{Y} \cup P'$ must involve at least one clause $B \vee w_j \in P'$ (and symmetrically any forward chaining derivation of a variable w_k from the set B using the set of clauses $\mathcal{Y} \cup K'$ must involve at least one clause $A \vee u_j \in K'$). Then, by the forward chaining Lemma 5.4, $w_j \in Cone_{G_{\mathcal{Y} \cup P'}}(u_k)$, and so there must be a path in $G_{\mathcal{Y} \cup P'}$ leading from w_j to u_k. Since L_i is of type $\mathcal{K}_0$, no arc in $Q(L_i)$ can be generated by clauses in $P' \subseteq L_i$. Therefore, for every u_k there exists a w_j such that there is a path in $G_{\mathcal{Y}}$ leading from w_j to u_k. Since $l' < l$, there is some w_j (w.l.o.g. w_1), from which paths lead to (at least) two different u_k's (w.l.o.g., to u_1 and u_2). However, by a symmetric argument, for every w_k (and in particular for w_1) there exists a u_j such that there is a path in $G_{\mathcal{Y}}$ leading from u_j to w_k. Thus there is a path in $G_{\mathcal{Y}}$ from some u_j to both u_1 and u_2, and so no matter what the value of j is, there is a path in $G_{\mathcal{Y}}$ from u_j to u_k where $j \neq k$. We shall finish the proof by showing that the existence of such a path contradicts the irredundancy of $\mathcal{C}'$.

Due to Theorem 2.7 we can assume that all arcs on the path from u_j to u_k are generated by clauses present in $\mathcal{C}' \cap \mathcal{Y}$, namely by some $B_1 \vee \overline{u}_i \vee w_1, B_2 \vee \overline{w}_1 \vee w_2, \ldots, B_{\ell-1} \vee \overline{w}_{\ell-2} \vee w_{\ell-1}, B_\ell \vee \overline{w}_{\ell-1} \vee u_j \in \mathcal{Y}$, where $B_i \cap Q(K') = \emptyset$ for every $i = 1, \ldots, \ell$. Since $A \vee u_j \in L_i$, $\mathcal{Y} \subseteq Cone_{\mathbf{D}_h}(L_i)$, and $\mathbf{D}_h$ is transitively closed, the graph $\mathbf{D}_h$ contains arcs $(B_i \vee \overline{w}_{i-1} \vee w_i, A \vee u_j)$ for every $i = 1, \ldots, \ell$ (here we denote $u_j = w_0$ and $u_k = w_\ell$), and therefore $B_i \subseteq FC_h(A)$ holds for each such index i. Let $I = Cone_{G_h}(Q(K')) \setminus Q(K')$. Since $A, B_i \subseteq I$ for all i, it follows from Lemma 5.6 that the inclusion $B_i \subseteq FC_{h|_I}(A)$ holds for each index i. Let us now define $\mathcal{C}^* = \mathcal{C}' \setminus \{A \vee u_k\}$, and let us investigate the set $FC_{\mathcal{C}^*}(A)$. First of all, since $u_k \notin I$, the inclusion $B_i \subseteq FC_{h|_I}(A) \subseteq FC_{\mathcal{C}^*}(A)$ holds for each index i. Secondly, since $A \vee u_j \in \mathcal{C}^*$, we get $B_1 \cup \{u_j\} \subseteq FC_{\mathcal{C}^*}(A)$ and so $w_1 \in FC_{\mathcal{C}^*}(A)$. Similarly, $B_2 \cup \{w_1\} \subseteq FC_{\mathcal{C}^*}(A)$ implies

$w_2 \in FC_{\mathcal{C}^*}(A)$, and so on, eventually obtaining $u_k \in FC_{\mathcal{C}^*}(A)$. Thus $\phi(\mathcal{C}^*) \equiv \phi(\mathcal{C}')$ which is a contradiction to the irredundancy of $\mathcal{C}'$. ■

Lemma 6.4 (Action $\mathcal{K}_1$) *Let K_i be of type $\mathcal{K}_1$, then the set K_i after being modified by Action $\mathcal{K}_1$ is a minimum cardinality set such that $\mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$.*

Proof. Before we start the proof, let us describe its main steps. We shall start by proving that the set K' created in step (a) of Action $\mathcal{K}_1$ satisfies that $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$ and that $Q(K_i) = Q(K')$. Then we shall show that the set K'', which denotes the set K_i after being modified by Action $\mathcal{K}_1$, satisfies that $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{V})$. After that we shall show, that $\mathcal{C}''$ which denotes the set $\mathcal{C} \cap \mathcal{V}$ after being modified by Action $\mathcal{K}_1$ is again the prime and irredundant representation of $\mathcal{V}$-component $h_{\mathcal{V}}$ of h. We will finish the proof by showing that K'' is a minimum cardinality set for which $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{V})$.

Let us denote $K_i = \mathcal{C} \cap L_i = \{A_1 \vee \overline{u}_1 \vee v_1, \ldots, A_k \vee \overline{u}_k \vee v_k\}$ where $A_j \cap Q(K_i) = \emptyset$ and $\{u_j, v_j\} \subseteq Q(K_i)$ for each index j. By Corollary 5.21 we can switch all sets of subgoals outside of $Q(K_i)$ of all clauses from K_i to some representative set A, such that after this switch we obtain $K' = \mathcal{C}' \cap L_i = \{A \vee \overline{u}_1 \vee v_1, \ldots, A \vee \overline{u}_k \vee v_k\}$. Corollary 5.21 guarantees that after this switch is performed in step (a) of Action $\mathcal{K}_1$ the "current" CNF $\mathcal{C}'$ is still a prime and irredundant representation of h. Note also that K' is now a strong component of $\mathbf{D}_{\mathcal{C}'}$ (such that L_i is its corresponding strong component of $\mathbf{D}_h$) and $\mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{Y} \cup K_i) = \mathcal{R}(\mathcal{V})$. Moreover, since the switch did not change anything inside of $Q(K_i) = Q(K')$ the subgraphs of $Q(K')$ generated by clauses in K_i and K' are identical.

Let us investigate the exclusive set $\mathcal{V} = Cone_{\mathbf{D}_h}(K_i) = Cone_{\mathbf{D}_h}(K')$ of h and the corresponding exclusive component $h_{\mathcal{V}}$ of h. Since $\mathcal{C}'$ is a prime and irredundant representation of h, $\mathcal{C}' \cap \mathcal{V}$ is, due to Lemma 4.10, a prime and irredundant representation of $h_{\mathcal{V}}$. Moreover, the fact $\mathcal{I}(h) = \mathcal{R}(\mathcal{C}')$ implies by Lemma 5.7 that $\mathbf{D}_{\mathcal{C}'} = \mathbf{D}_{\mathcal{I}(h)}(\mathcal{C}') = \mathbf{D}_h(\mathcal{C}')$ which in turn implies $Cone_{\mathbf{D}_{\mathcal{C}'}}(K') = \mathcal{C}' \cap Cone_{\mathbf{D}_h}(K') = \mathcal{C}' \cap \mathcal{V}$.

In step (b) of Action $\mathcal{K}_1$ the graph $Q'(K')$ is defined as the subgraph of $Q(K')$ generated by clauses from $Cone_{\mathbf{D}_{\mathcal{C}'}}(K') = \mathcal{C}' \cap \mathcal{V}$. We shall show that if there is a path in $Q'(K')$ from u to v, then $A \vee \overline{u} \vee v \in \mathcal{R}(\mathcal{V})$ (i.e. $A \vee \overline{u} \vee v$ is an implicate of $h_{\mathcal{V}}$). Since $\mathcal{C}' \cap \mathcal{V}$ represents $h_{\mathcal{V}}$ this is the same as proving $A \vee \overline{u} \vee v \in \mathcal{R}(\mathcal{C}' \cap \mathcal{V})$). The last fact is (by Lemma 2.4) equivalent to showing that $v \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$. Let $B_1 \vee \overline{u} \vee w_1, B_2 \vee \overline{w}_1 \vee w_2, \ldots, B_k \vee \overline{w}_{k-1} \vee v$ be the clauses from $\mathcal{C}' \cap \mathcal{V} = Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$ that generate a path from u to v in $Q'(K')$, where $w_1, \ldots w_{k-1} \in Q(K')$. Since $A \vee \overline{u}_1 \vee v_1 \in K'$ (we could pick any clause from K' here) the arc $(B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell, A \vee \overline{u}_1 \vee v_1)$ is in $\mathbf{D}_{\mathcal{C}'}$ for every $\ell = 1, \ldots, k$ (here we denote $w_0 = u$ and $w_k = v$) by the definition of $Cone_{\mathbf{D}_{\mathcal{C}'}}(K')$ and the fact that $\mathbf{D}_{\mathcal{C}'}$ is transitively closed. However, the definition of the graph $\mathbf{D}_{\mathcal{C}'}$ now implies $B_\ell \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u_1\})$ for every $\ell = 1, \ldots, k$. However, $B_\ell \subseteq Cone_{G_h}(Q(K')) \setminus Q(K')$ and so by Lemma 5.4 no clause with literals in $Q(K')$ (and in particular no clause containing u_1) is necessary to derive B_ℓ by forward chaining from $A \cup \{u_1\}$. Hence $B_\ell \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A)$ for every $\ell = 1, \ldots, k$. This means that $B_1 \cup \{u\} \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$ which together with $B_1 \vee \overline{u} \vee w_1 \in \mathcal{C}' \cap \mathcal{V}$ gives $w_1 \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$. Similarly, $B_2 \cup \{w_1\} \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$ together with $B_2 \vee \overline{w}_1 \vee w_2 \in$

$\mathcal{C}' \cap \mathcal{V}$ gives $w_2 \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$ and so on, eventually obtaining $v \in FC_{\mathcal{C}' \cap \mathcal{V}}(A \cup \{u\})$, which finishes the proof that $A \vee \overline{u} \vee v \in \mathcal{R}(\mathcal{V})$.

Obviously, in step (c) an arc (u, v) is added into the transitive reduction of $Q'(K')$ only if there is a path in $Q'(K')$ from u to v. Therefore the argument in the previous paragraph proves that every clause added in step (d) is in $\mathcal{R}(\mathcal{V})$. Thus $\mathcal{R}(\mathcal{Y} \cup K'') \subseteq \mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{V})$ holds after K' is modified to K'' in step (d). Let us denote $\mathcal{C}_{old} = (\mathcal{C}' \cap \mathcal{V}) \setminus (K' \setminus K'')$, $\mathcal{C}_{new} = (K'' \setminus K')$ and $\mathcal{C}'' = \mathcal{C}_{old} \cup \mathcal{C}_{new}$. To prove the reverse inclusion $\mathcal{R}(\mathcal{Y} \cup K') \subseteq \mathcal{R}(\mathcal{Y} \cup K'')$ we need to show that for every clause $A \vee \overline{u} \vee v$ removed from K' in step (d), v can be derived by forward chaining from $A \cup \{u\}$ using the new set of clauses $\mathcal{C}''$. Note that an arc (u, v) is removed from $Q'(K')$ in step (c) only if there is a path from u to v in the constructed transitive reduction which means that clauses in $\mathcal{C}''$ generate a path from u to v. However, now we can repeat the argument from the previous paragraph to prove that $v \in FC_{\mathcal{C}''}(A \cup \{u\})$ provided we can show $B_\ell \subseteq FC_{\mathcal{C}''}(A)$ for every clause $B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell \in \mathcal{C}''$ generating an arc on the path from u to v. If $B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell \in \mathcal{C}_{old}$ then we already know that $B_\ell \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A)$. However, $B_\ell \subseteq Cone_{G_h}(Q(K')) \setminus Q(K')$ and so by Lemma 5.4 no clause with its head in $Q(K')$ is necessary to derive B_ℓ by forward chaining from A. Hence $B_\ell \subseteq FC_{(\mathcal{C}' \cap \mathcal{V}) \setminus K'}(A)$ which implies $B_\ell \subseteq FC_{\mathcal{C}''}(A)$ as $\mathcal{C}' \setminus K'$ and $\mathcal{C}'' \setminus K''$ are identical. If $B_\ell \vee \overline{w}_{\ell-1} \vee w_\ell \in \mathcal{C}_{new}$ then $B_\ell \subseteq FC_{\mathcal{C}''}(A)$ holds trivially as $B_\ell = A$. This finishes the proof of $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{Y} \cup K')$.

The use of Corollary 5.21 in step (a) at every iteration of the algorithm requires not only that all added clauses be implicates of $h_{\mathcal{V}}$ (and thus implicates of h) but also that all added clauses be prime implicates, maintaining the primality of the CNF representation of h. Let us assume that an arc (u, v) is added into the transitive reduction of $Q'(K')$ in step (c). Since the minimum possible number of arcs has been added to get back the same transitive closure, it implies that all paths from u to v in $Q'(K')$ contain at least one free arc which is then removed in step (d) (note that no fixed arcs are removed and no arc spanning a path which entirely stays in $Q'(K')$ can be added into its transitive reduction). The addition of the arc (u, v) in step (c) implies the addition of the clause $A \vee \overline{u} \vee v$ in step (d). Let us show that $A \vee \overline{u} \vee v$ is a prime implicate of $h_{\mathcal{V}}$. Let us distinguish two cases:

- First let us assume by contradiction that $A \vee v$ is an implicate of $h_{\mathcal{V}}$, and hence $\mathcal{C}^* = \mathcal{C}'' \setminus \{A \vee \overline{u} \vee v\} \cup \{A \vee v\}$ still represents $h_{\mathcal{V}}$. However, the arc (u, v) which was added in step (c) is now missing from $Q'(K')$, and since the minimum number of arcs was added to keep the transitive closure of $Q'(K)$ the same, the transitive closure will change as a result of removing (u, v). That means that the graphs $\mathbf{G}_{\mathcal{C}''}$ and $\mathbf{G}_{\mathcal{C}^*}$ have different transitive closures, which by Theorem 2.7 contradicts the assumption that $\mathcal{C}''$ represents the same function as $\mathcal{C}^*$. This argument also proves that $\mathcal{C}''$ is an irredundant representation of $h_{\mathcal{V}}$ and thus $\mathcal{C}' \setminus (\mathcal{C}' \cap \mathcal{V}) \cup \mathcal{C}''$ is an irredundant representation of h.

- Now let us assume by contradiction that $A' \vee \overline{u} \vee v$, where $A' \subset A$, is an implicate of $h_{\mathcal{V}}$, and hence $\mathcal{C}^* = \mathcal{C}'' \setminus \{A \vee \overline{u} \vee v\} \cup \{A' \vee \overline{u} \vee v\}$ still represents $h_{\mathcal{V}}$. This implies that $v \in FC_{\mathcal{C}' \cap \mathcal{V}}(A' \cup \{u\})$ while $v \notin FC_{\mathcal{C}' \cap \mathcal{V}}(A')$ due to the above first case. Therefore, any forward chaining derivation of v from $A' \cup \{u\}$ must use some clause C_1 in which u is a subgoal. By Lemma 5.4 also the head (say w_1) of this clause must be in $Q(K')$, and

so we can write $C_1 = B_1 \vee \overline{u} \vee w_1$ for some $B_1 \cap Q(K') = \emptyset$ (since C_1 is a CQ clause). Assuming that we are looking at a forward chaining derivation of v from $A' \cup \{u\}$ in which all derived variables are subsequently used in some future step, also w_1 must be a subgoal of some clause C_2 used in the derivation chain. By similar considerations as above we can write $C_2 = B_2 \vee \overline{w}_1 \vee w_2$ where $w_2 \in Q(K')$ and $B_2 \cap Q(K') = \emptyset$, and so on, until some clause $C_\ell = B_\ell \vee \overline{w}_{\ell-1} \vee v$ is used. Clauses C_1 to C_ℓ of course generate a path from u to v in $Q'(K')$. However, every such path contains at least one free arc generated by a clause from K' and so $A \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A' \cup \{u\})$ must hold. But now using the fact that $A \subseteq Cone_{G_h}(Q(K')) \setminus Q(K')$ and Lemma 5.4 we get that no clause with literals in $Q(K')$ (and in particular no clause containing u) is necessary to derive A by forward chaining from A' and so $A \subseteq FC_{\mathcal{C}' \cap \mathcal{V}}(A')$ contradicting the primality of C'.

The above considerations show that the CNF $\mathcal{C}'' = (\mathcal{C}' \cap \mathcal{V}) \setminus K' \cup K''$ resulting from Action $\mathcal{K}_1$ is an irredundant and prime representation of $h_\mathcal{V}$ such that $\mathcal{R}(\mathcal{Y} \cup K'') = \mathcal{R}(\mathcal{Y} \cup K') = \mathcal{R}(\mathcal{V})$. Using Corollary 4.11 we moreover get that $\mathcal{C}' \setminus (\mathcal{C}' \cap \mathcal{V}) \cup \mathcal{C}''$ is an irredundant and prime representation of h. To finish the proof we shall show that K'' is a minimum cardinality set with this property.

Let us assume by contradiction that there exists a set of clauses K^*, where $|K^*| < |K''|$, such that $\mathcal{R}(\mathcal{Y} \cup K^*) = \mathcal{R}(\mathcal{Y} \cup K'')$. Let $Q''(K')$ and $Q^*(K')$ be the subgraphs of $Q(K')$ generated by clauses from $\mathcal{Y} \cup K''$ and $\mathcal{Y} \cup K^*$. Since the set of arcs generated by clauses from $\mathcal{Y}$ is exactly the same in both cases, and the set of arcs K'' has the minimum cardinality subject to the condition that the transitive closure of $Q''(K')$ is the same as that of $Q'(K')$, then the transitive closure of $Q^*(K')$ must be different. That means that the graphs $\mathbf{G}_{\mathcal{Y} \cup K''}$ and $\mathbf{G}_{\mathcal{Y} \cup K^*}$ have different transitive closures, which by Theorem 2.7 contradicts the assumption that $\mathcal{Y} \cup K''$ represents the same function as $\mathcal{Y} \cup K^*$. ■

6.2 Minimization of the number of literals

In this subsection we shall show that given a CQ function f Algorithm 6.1 not only outputs a minimum CNF with respect to the number of clauses, but if a simple preprocessing and postprocessing step is employed (adding and removing an extra variable which turns an arbitrary CQ CNF into a CQ CNF with no negative clauses) then the output CNF is also minimum with respect to the number of literals. Throughout this subsection let us simply denote the clause graph D_f by D and let K be an arbitrary strong component of D. We shall show that all prime implicates of f which belong to K have the same degree, i.e. consist of the same number of literals. We shall start with two preparatory lemmas.

Lemma 6.5 *Let f be a CQ function and let $C = A \vee X \vee u$ be a prime implicate of f, let Q be the strong component of $G_{Cone_D(C)}$ to which u belongs and let us assume that $A \cap Q = \emptyset$, $X \subseteq Q \setminus \{u\}$, $|X| \leq 1$. Let K denote the strong component of $D = D_f$, to which C belongs. Then each strong component of $G_{Cone_D(C) \setminus K}$ contains at most one element of A.*

Proof. Let us denote $\mathcal{X} = Cone_D(C) \setminus K$. According to Theorem 5.11, we have that $\mathcal{X}$ is an exclusive set of clauses of f. To prove the lemma, let us proceed by contradiction. Let us assume, that there are two distinct variables $a_1, a_2 \in A$ which belong to the same strong component S of $G_{\mathcal{X}}$. Since there is a path from a_1 to a_2 in $G_{\mathcal{X}}$ there must exist clauses $C_1 = B_1 \vee \overline{x}_1 \vee x_2, C_2 = B_2 \vee \overline{x}_2 \vee x_3, \ldots, C_k = B_k \vee \overline{x}_k \vee x_{k+1}$ in $\mathcal{X}$, where $x_1 = a_1$, $x_{k+1} = a_2$, and $\forall\ 1 \leq i \leq k+1 : x_i \in S$. The following properties can now be derived for every $1 \leq i \leq k$:

- From the fact that $\mathcal{X} \subseteq \mathcal{I}(f)$ and from Corollary 3.3 it follows that C_i is a CQ-clause w.r.t. f, which in turn implies that $B_i \subseteq Cone_{G_{\mathcal{X}}}(S) \setminus S$.
- Moreover, since $C_i \in \mathcal{X} = Cone_D(C) \setminus K$, we get $B_i \subseteq FC_{\mathcal{X}}(A \cup X)$ by the definition of graph D, and using Lemma 5.6 (where we set $I = Cone_{G_{\mathcal{X}}}(S) \setminus S$) we can strengthen this to $B_i \subseteq FC_{\mathcal{X}}(A)$.

Let us denote $B = \bigcup_{i=1}^{k} B_i$. Clearly, the properties of individual B_i's carry over to set B and so $B \subseteq Cone_{G_{\mathcal{X}}}(S) \setminus S$ and $B \subseteq FC_{\mathcal{X}}(A)$ hold. Using the resolution chain of C_i clauses along the path from a_1 to a_2 also $C' = B \vee \overline{a}_1 \vee a_2 \in \mathcal{R}(\mathcal{X}) \subseteq \mathcal{I}(f)$ holds. Now there are two possibilities:

- Either $B \subseteq FC_{\mathcal{X}}(A \setminus \{a_2\})$ but then using the clause $C' \in \mathcal{I}(f)$ we get $a_2 \in FC_f(A \setminus \{a_2\})$, contradicting the primality of C,
- or there must exist a variable $b \in B$ such that each forward chaining derivation proving $b \in FC_{\mathcal{X}}(A)$ uses a_2. Let $\{E_1, \ldots, E_\ell\} \subseteq \mathcal{X}$ be a minimal (under inclusion) forward chaining derivation of $b \in FC_{\mathcal{X}}(A)$, then due to the above considerations $a_2 \in \bigcup_{j=1}^{k} Vars(E_j)$ must hold. However, Lemma 5.4 now implies that there is a path from a_2 to b in $G_{\mathcal{X}}$ contradicting the fact that $B \subseteq Cone_{G_{\mathcal{X}}}(S) \setminus S$.

■

Lemma 6.6 *Let f be a CQ function and let $C = A \vee X \vee u$ and $C' = B \vee Y \vee v$ be two prime implicates of f that belong to the same strong component K of clause graph $D = D_f$. Let Q be the strong component of G_f to which both u and v belong, and let us assume that $A \cap Q = \emptyset$, $X \subseteq Q \setminus \{u\}$, $|X| \leq 1$, $B \cap Q = \emptyset$, $Y \subseteq Q \setminus \{v\}$, $|Y| \leq 1$. Let S be an arbitrary strong component of $G_{Cone_D(C) \setminus K}$. Then $|S \cap A| = |S \cap B| \leq 1$.*

Proof. Similarly as in the proof of Lemma 6.5 let us denote the exclusive set of clauses $Cone_D(C) \setminus K = Cone_D(C') \setminus K$ by $\mathcal{X}$. Using Lemma 5.19 we get $A \subseteq FC_{\mathcal{X}}(B)$ and $B \subseteq FC_{\mathcal{X}}(A)$. By Lemma 6.5, $|S \cap A| \leq 1$ and $|S \cap B| \leq 1$, so we only need to show that if $|S \cap A| = 1$ (say $S \cap A = \{a\}$) then there exists some $b \in S \cap B$ (the opposite direction is of course similar).

Now $A \subseteq FC_{\mathcal{X}}(B)$ implies $a \in FC_{\mathcal{X}}(B)$, i.e. $B \vee a$ is an implicate of function $f_{\mathcal{X}}$, the $\mathcal{X}$-component of f. Therefore there must exist $B' \subseteq B$ such that $B' \vee a$ is a prime

implicate of $f_{\mathcal{X}}$, which implies $a \in FC_{\mathcal{X}}(B')$. Using Theorem 2.7 there must be a path from every variable $b \in B'$ to a in the graph $G_{\mathcal{X}}$. Furthermore, the inclusion $B \subseteq FC_{\mathcal{X}}(A)$ of course implies $B' \subseteq FC_{\mathcal{X}}(A)$, and so, similarly as in the proof of Lemma 6.5, there are two possibilities:

- Either $B' \subseteq FC_{\mathcal{X}}(A \setminus \{a\})$ but then also $a \in FC_f(A \setminus \{a\})$ (recall that $a \in FC_{\mathcal{X}}(B')$), contradicting the primality of C,
- or there must exist a variable $b \in B'$ such that each forward chaining derivation proving $b \in FC_{\mathcal{X}}(A)$ uses a. Let $\{E_1, \ldots, E_\ell\} \subseteq \mathcal{X}$ be a minimal (under inclusion) forward chaining derivation of $b \in FC_{\mathcal{X}}(A)$, then due to the above considerations $a \in \bigcup_{j=1}^{k} Vars(E_j)$ must hold. However, Lemma 5.4 now implies that there is a path from a to b in $G_{\mathcal{X}}$. Thus $G_{\mathcal{X}}$ contains both a path from b to a and a path from a to b proving that $b \in S$. ■

Now we are ready to derive the desired result.

Corollary 6.7 *Let f be a CQ function and let $C = A \vee u$ and $C' = B \vee v$ be two prime implicates of f, which belong to the same strong component K of $D = D_f$, then $|A| = |B|$.*

Proof. Let Q be the strong component of G_f which contains both u and v. According to Corollary 5.24 both C and C' are of the same type (either $\mathcal{K}_0$ or $\mathcal{K}_1$) and hence $|A \cap Q| = |B \cap Q|$. According to Lemma 6.6, $|S \cap A| = |S \cap B|$ for every strong component S of the graph $G_{Cone_D(C)\setminus K}$, and hence $|A \setminus Q| = |B \setminus Q|$. Therefore $|A| = |B|$. ■

As an easy corollary, we now get that Algorithm 6.1 for minimizing the number of clauses in a CQ CNF minimizes also the number of literals, if the input CNF is pure Horn.

Corollary 6.8 *Let f be a pure Horn CQ function and let $\mathcal{C}$ be a CNF with the minimum number of clauses which represents f and which was produced by Algorithm 6.1. Then $\mathcal{C}$ is also a CNF with the minimum number of literals which represents f.*

Proof. Algorithm 6.1 in each step solves one ORTSC problem by minimizing the number of clauses in some strong component K of D_f. Using Corollary 6.7 we get that all prime clauses in K have the same number of literals. It then follows that a minimum cardinality set $Q^* \subseteq K$ which solves the current ORTSC is also a set with the minimum number of literals which solves the ORTSC reformulated to output a set $\mathcal{Q}^*$ consisting of a minimum number of literals such that $\mathcal{R}(\mathcal{F} \cup \mathcal{Q}^*) = \mathcal{R}(\mathcal{F} \cup \mathcal{Q})$ (as in Remark 5.16). ■

Using a simple transformation we can extend Corollary 6.8 to all CQ functions including those that have negative prime implicates.

Corollary 6.9 *Let f be a CQ function given by a prime and irredundant CQ CNF $\mathcal{F}$. Then a CNF with the minimum number of literals which represents f can be constructed using a simple modification of Algorithm 6.1.*

Proof. If $\mathcal{F}$ is pure Horn then Algorithm 6.1 constructs the desired CNF due to Corollary 6.8. If $\mathcal{F}$ contains negative clauses then introduce a new variable z and replace each negative clause $\bigvee_{i\in I}\overline{x}_i$ in $\mathcal{F}$ with a pure Horn clause $\bigvee_{i\in I}\overline{x}_i \vee z$. Notice that the new pure Horn CNF $\mathcal{F}_z$ resulting from such a replacement is CQ, because all newly introduced arcs in $G_{\mathcal{F}_z}$ point to z, which means that z constitutes a new singleton strong component of $G_{\mathcal{F}_z}$, while the structure of all the other strong components of $G_{\mathcal{F}_z}$ remains the same as in $G_{\mathcal{F}}$. Let f_z be the CQ function represented by $\mathcal{F}_z$. Note that $f_z(z=0)=f$, and $f_z(z=1)=f_{\mathcal{H}}$, where $f_{\mathcal{H}}$ is the pure Horn component of f.

Let $\mathcal{F}'_z$ be a CNF produced by Algorithm 6.1 on input $\mathcal{F}_z$. Using Corollary 6.8 we get that $\mathcal{F}'_z$ is a representation of f_z with the minimum number of literals. Let $\mathcal{F}'$ be the CNF produced from $\mathcal{F}'_z$ by dropping all occurences of literal z (or equivalently, by substituting $z=0$). We claim that $\mathcal{F}'$ is a representation of f with the minimum number of literals.

Note first that $\mathcal{F}'$ represents f, since $\mathcal{F}'=\mathcal{F}'_z(z=0)$, and $\mathcal{F}'_z$ represents f_z and $f_z(z=0)=f$. Let k be the number of negative clauses in $\mathcal{F}$. Let us assume by contradiction that $\mathcal{F}''$ is a prime and irredundant representation of f with fewer literals than $\mathcal{F}'$. By Proposition 4.17 both $\mathcal{F}'$ and $\mathcal{F}''$ have exactly k negative clauses. Let $\mathcal{F}''_z$ be the CNF constructed from $\mathcal{F}''$ by adding z to all negative clauses, as above. Clearly, $\mathcal{F}''_z$ represents f_z since $\mathcal{F}''_z(z=0)$ represents f and $\mathcal{F}''_z(z=1)$ represents $f_{\mathcal{H}}$. However, now we have: $|\mathcal{F}''_z|_\ell = |\mathcal{F}''|_\ell + k < |\mathcal{F}'|_\ell + k = |\mathcal{F}'_z|_\ell$ contradicting the minimality of $\mathcal{F}'_z$. ∎

6.3 Time complexity of Algorithm 6.1

Now let us turn our attention to the complexity of CQ minimization. First let us recall that due to Theorem 2.3, we can transform any given Horn CNF representation (having l literals) of a CQ function h into an irredundant and prime CNF $\mathcal{C}$ in $O(l^2)$ time. By Lemma 3.2 we know that such a prime representation must be a CQ CNF, which can be easily tested in $O(l)$ time (by building the strong components of $G_{\mathcal{C}}$ in $O(l)$ time, and then testing whether each clause is CQ in $O(l)$ time). Once we have an irredundant and prime CNF representation of a CQ function, we can run Algorithm 6.1.

Theorem 6.10 *Let h be a Horn function on n variables which is from the class CQ. Let $\mathcal{C}$ be an irredundant and prime CNF representing h which is given as an input to Algorithm 6.1. Then, if properly implemented, Algorithm 6.1 runs in $O(n^2 + m\cdot l)$ time, where m is the number of clauses and l is the number of literals in $\mathcal{C}$.*

Proof. As we have shown in property (iv) of Theorem 5.14, we can perform the first and the second step in time $O(n^2+m\cdot\ell)$. To classify the components according to Definition 5.25 (types $\mathcal{K}_0$ and $\mathcal{K}_1$) it suffices to look at one clause in $\mathcal{C}$ from each strong component, which can also be easily done in $O(m\cdot l)$ time. Note that $\mathbf{D}_{\mathcal{C}}$ is transitively closed, so it is sufficient to keep the graph of the acyclic condensation of $\mathbf{D}_{\mathcal{C}}$ and for each strong component a list of participating clauses.

The third step is the main loop and it is executed for each strong component of $D_{\mathcal{C}}$ exactly once. Thus the body of the loop is repeated at most m times. Performing Action

$\mathcal{K}_0$ of course takes constant time. We shall show that Action $\mathcal{K}_1$ can be performed in $O(l)$ time for each strong component of type $\mathcal{K}_1$. Let us analyze Action $\mathcal{K}_1$ in detail.

- Step (a): switching all subgoal sets outside of $Q(K)$ of all clauses in $\mathcal{C} \cap K$ to some representative set $A(K)$ can be easily performed in $O(l)$ time.
- Step (b): identifying the subgraph $Q'(K')$ of $Q(K')$ generated only by clauses from $Cone_{D_{\mathcal{C}'}}(K')$ also takes only $O(l)$ time.
- Step (c): finding a transitive reduction of $Q'(K')$ with the added restriction that the reduction must contain all fixed arcs can be done in three steps:
  1. Detecting the strong components of $Q'(K')$ takes $O(|Q'(K')|)$ time (which is of course $O(l)$).
  2. Arcs between different strong components all stay in place. This is due to the fact that the irredundancy of the CNF representation $\mathcal{C}'$ guarantees that these arcs already span the skeleton graph of the acyclic condensation of $Q'(K')$ which is of course the unique transitive reduction of the acyclic condensation of $Q'(K')$. Therefore this step takes no time. To see that the irredundancy of $\mathcal{C}'$ is sufficient, let us assume by contradiction that there is a free arc (u, v) (generated by a clause C' in K') between two distinct strong components of $Q'(K')$ which is not in the skeleton graph of the acyclic condensation. That means that there is a path from u to v in $Q'(K')$ (i.e. a path generated by clauses in $Cone_{D_{\mathcal{C}'}}(K')$) not using the arc (u, v). By the definition of the graph $D_{\mathcal{C}'}$, for each such clause C'' we have $Subg(C'') \subseteq FC_{\mathcal{C}'}(Subg(C'))$. In fact, by Lemma 5.4 all such forward chaining derivations can be done without ever using a clause containing v (in particular, without using the clause C'), and so $Subg(C'') \subseteq FC_{\mathcal{C}' \setminus C'}(Subg(C'))$. However, this means that $v \in FC_{\mathcal{C}' \setminus C'}(Subg(C'))$ (i.e. v can be derived from $Subg(C')$ along the path from u to v without using the clause C') which is a contradiction to the irredundancy of $\mathcal{C}'$.
  3. In each strong component of $Q'(K')$ its transitive reduction that keeps all the fixed arcs can be found in $O(|Q'(K')|)$ (and hence also $O(l)$) time using the algorithm from [9] and its correction from [23].
- Step (d): removing clauses that correspond to the removed arcs and adding clauses that correspond to the added arcs in step (c) amounts to updating the list that represents the strong component K', which can be performed in $O(l)$ time as the total length of the added clauses is at most equal to the total length of the removed clauses (this also takes care of updating the graph $\mathbf{D}_{\mathcal{C}'}$).

The fourth and last step obviously takes only $O(l)$ time, so adding all time requirements together we get the desired $O(n^2 + m \cdot l)$ time bound. Moreover, if we assume that each variable appears as a head of some clause, then the complexity of Algorithm 6.1 can be simplified to $O(m \cdot l)$ time. ■

7 Conclusions

This paper introduces a new subclass of Horn functions - the class of CQ (component-wise quadratic) functions, which strictly contains the class of quasi-acyclic functions defined in [16]. For this new class a polynomial time algorithm is designed, which for a given prime and irredundant input CNF of a CQ function f outputs a minimum (shortest) CNF representation of f. The algorithm minimizes the number of clauses and, with an additional preprocessing step, can be also used to minimize the number of literals.

Moreover, the algorithm in fact works for a broader subclass of Horn functions than just the class of CQ functions. The property which suffices for the correctness of the algorithm is the following: every prime implicate $C = A \vee u$ of f has at most one subgoal from A in the strong component $Q_{Cone_{D_f}(C)}(u)$ (i.e. the strong component which contains u) of the implication graph induced by the exclusive component of f represented by clauses in $Cone_{D_f}(C)$. Since this implication graph is a subgraph of G_f (has fewer edges), the strong component $Q_{Cone_{D_f}(C)}(u)$ can be strictly smaller than the strong component $Q_f(u)$ in G_f. Thus, it may happen that some prime implicate $C = A \vee u$ has two or more subgoals in $Q_f(u)$ (which means that f is not a CQ function) while it still has at most one subgoal in $Q_{Cone_{D_f}(C)}(u)$ (thus, still enabling the minimization algorithm to work).

References

[1] G. Ausiello, A. D'Atri and D. Sacca. Minimal representation of directed hypergraphs. *SIAM Journal on Computing*, **15** (1986), 418–431.

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

[3] E. Boros, O. Čepek, and A. Kogan. Horn minimization by iterative decomposition. *Annals of Mathematics and Artificial Intelligence*, **23** (1998), 321-343.

[4] E. Boros, O. Čepek, A. Kogan, and P. Kučera. Exclusive and essential sets of implicates of Boolean functions. *RUTCOR Research Report RRR 10-08*, Rutgers University, New Brunswick, NJ, June 2008.

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

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

[7] C. Delobel and R.G. Casey. Decomposition of a Data Base and the Theory of Boolean Switching Functions. *IBM Journal of Research and Development*, **17**(1973), 374-386.

[8] 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.

[9] K.P. Eswaran and R.E. Tarjan Augmentation problems. *SIAM Journal on Computing*, **5**(1976), 653-665

[10] 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.

[11] M.R. Genesereth and N.J. Nilsson. *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann, Los Altos, CA, 1987.

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

[13] 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.

[14] P.L. Hammer and A. Kogan. Optimal compression of propositional Horn knowledge bases: Complexity and approximation. *Artificial Intelligence*, **64**(1993), 131-145.

[15] 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.

[16] P.L. Hammer and A. Kogan. Quasi-acyclic propositional Horn knowledge bases: Optimal compression. *IEEE Transactions on Knowledge and Data Engineering*, **7**(5)(1995), 751-762.

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

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

[19] D. Maier. Minimal covers in the relational database model. *Journal of the ACM*, **27**(1980), 664-674.

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

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

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

[23] S. Raghavan. A Note on Eswaran and Tarjan's Algorithm for the Strong Connectivity Augmentation Problem. pp 19-26, in The Next Wave in Computing, Optimization, and Decision Technologies, edited by Golden, Raghavan, and Wasil (Springer) 2005.

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