DIMACS Technical Report 94-50
November 1994

Restricted consensus method and quadratic implicates of pure Horn functions.

by

Ondřej Čepek [1] [,2]
RUTCOR
Rutgers University
P.O.Box 5062
New Brunswick, New Jersey 08903
email : cepek@rutcor.rutgers.edu

[1] Affiliated Graduate Student Member

[2] This work was partially supported by the Office of Naval Research (grants N0001492F1375 and N0001492F4083) and by the Air Force Office of Scientific Research (grant F49620-93-1-0041).

DIMACS is a cooperative project of Rutgers University, Princeton University, AT&T Bell Laboratories and Bellcore.

DIMACS is an NSF Science and Technology Center, funded under contract STC–91–19999; and also receives support from the New Jersey Commission on Science and Technology.

ABSTRACT

In this paper we deal with a special subclass of Boolean functions - the pure Horn functions. First we prove for this subclass a variant of the well-known Quine theorem which has a certain "non-expanding" property (every clause resulting from a consensus has a degree bounded by the maximum degree of a clause in the input CNF). Then we show how to use this result for generating all quadratic implicates of a given pure Horn function and we prove that the generating algorithm has the best possible worst case complexity.

1 Introduction.

The well-known Quine theorem (se e.g. [12, 13]) states that given an arbitrary CNF $\mathcal{F}$ representing a Boolean function f, every prime implicate of f can be derived from $\mathcal{F}$ by a chain (sequence) of consensuses. The theorem can be used in an algorithmic manner as follows. Given $\mathcal{F}$ one can generate all prime implicates of f by performing consensuses (starting with clauses in $\mathcal{F}$) and adding them to the CNF until every possible consensus is already either in the CNF or subsumed by some clause in the CNF. There are several ways how to organize such a procedure but each of them has an inevitable drawback, namely that the generated (intermediate) consensuses may have very high degrees compared to the clauses in the input CNF.

In this paper we will state a variant of the Quine theorem for the class of pure Horn functions. This variant does not have the above mentioned drawback. It gives a bound on the degree of all intermediate consensuses as a function of the maximum degree of a clause in the input CNF. The strongest bound is achieved in the case when the objective is to generate only quadratic prime implicates. This case is then studied in the rest of the paper.

We explain the importance of quadratic implicates for the minimization of pure Horn functions (for more on Horn minimization see e.g. [8, 9, 2]) and present a simple algorithm for generating all quadratic implicates of a given pure Horn function. We also prove that this algorithm achieves the best possible worst case complexity.

The paper is organized as follows. In the next section we introduce the basic Boolean terminology. Section 3 describes the forward chaining procedure which will be later used as a powerful tool for dealing with pure Horn functions. In Section 4 the pure Horn variant of the Quine theorem is stated and proved. Finally, Section 5 contains the discussion about how to generate quadratic implicates.

2 Basic Definitions.

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

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

of literals is called a *clause*, if every variable appears in it at most once, i.e. if $I \cap J = \emptyset$. A clause C is called *Horn* if it contains at most one positive literal (i.e. if $|J| \leq 1$), and it is called *pure Horn* (or in some literature *definite Horn*) if it contains exactly one positive literal. A *degree* of a clause C is the number of literals in it and is denoted $\deg(C)$.

It is well known that every Boolean function f can be represented as a conjunction of elementary disjunctions. Such an expression is called a *conjunctive normal form* (or CNF) of the Boolean function f. A CNF is called *Horn* (*pure Horn*) if it contains only Horn (resp. pure Horn) clauses. Finally, a Boolean function is called *Horn* (or *pure Horn*) if it can be represented by at least one Horn (resp. pure Horn) CNF. To simplify notation, we shall sometimes write in the sequel an elementary pure Horn clause $\bigvee_{x \in S} \overline{x} \vee y$ simply as $S \vee y$. Also following the logical programming terminology we shall sometimes refer to the proposition letters in S as *subgoals* and y as the *head* of the clause.

For two Boolean functions f and g we write $f \leq g$ if for every 0-1 vector $f(x_1, \ldots, x_n) = 1$ implies $g(x_1, \ldots, x_n) = 1$. We say that a clause C_1 *subsumes* another clause C_2 if $C_1 \leq C_2$ (e.g. the clause $\overline{x} \vee z$ subsumes the clause $\overline{x} \vee \overline{y} \vee z$). A clause C is called an *implicate* of the function f if $f \leq C$. An implicate C of a function is called *prime* iff there is no distinct implicate C' subsuming C, or in other words, an implicate of a function is prime iff dropping any literal from it produces a clause which is not an implicate of the function.

A CNF $\mathcal{F}$ representing a function f is called *prime* iff each clause of $\mathcal{F}$ is a prime implicate of the function. For the cases of Horn and pure Horn functions, it is well-known (see e.g. [7]) that each prime implicate of a

(pure) Horn function is (pure) Horn. Thus, in particular, any prime CNF representing a pure Horn function is pure Horn.

Two clauses C_1 and C_2 are said to have a *consensus* if they contain exactly one complementary pair of literals, i.e. if there exists exactly one proposition letter 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 proposition letter 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 conjunction of $\tilde{C}_1$ and $\tilde{C}_2$ is called the consensus of the parent clauses C_1 and C_2. Note that this consensus is a clause (does not contain a proposition letter and its negation). Moreover it is easy to see that if both C_1 and C_2 are implicates of some Boolean function f then so is their consensus $\tilde{C}_1\tilde{C}_2$.

In the rest of this paper we will deal only with pure Horn functions. We will restrict our attention to functions which do not have any unit (degree one) implicates. Such a restriction can be made without a loss of generality because any pure Horn function f given by a Horn CNF can be decomposed (in quadratic time) into a conjunction of literals (unit implicates) f' and a pure Horn function f'' having no unit prime implicates and such that f' and f'' depend on disjoint sets of proposition letters. See e.g. [10] for details. This means in particular that from now on any quadratic implicate will be assumed to be prime.

3 Forward Chaining.

Since our focus here is to decide the complexity of finding a minimum CNF of a given pure Horn function (recall that for every graph G the function f_G is pure Horn), and since a prime CNF can be obtained from any pure Horn CNF representation of a pure Horn function in polynomial time (see e.g. [7]) without increasing the number of clauses, we may restrict our attention solely to prime implicates and prime CNFs of pure Horn functions without any loss of generality.

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

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

In a relational database terminology the variables in $R_{\mathcal{F}}(Q)$ are said to be "chained" to the subset Q (see e.g. [3]). In an expert system terminology the usage of the clause $S \vee y$ is called "firing the rule" $\bigwedge_{x \in S} x \rightarrow y$ (see e.g. [8]).

The following lemma (proved in [9]) will be frequently used in the rest of this paper.

Lemma 3.1 *A pure Horn clause $Q \vee y$ is an implicate of a pure Horn function f given by a pure Horn CNF $\mathcal{F}$ if and only if $y \in R_{\mathcal{F}}(Q)$.*

It was also proven in [10] that the forward chaining procedure on a CNF $\mathcal{F}$ can be carried out in $O(|\mathcal{F}|)$ time.

4 The Restricted Consensus for Pure Horn Functions.

In this section we shall show that for the class of pure Horn functions we can state a special form of the well-known Quine theorem (se e.g. [12, 13]) and that this form posseses some desirable properties.

The Quine theorem claims that given a CNF $\mathcal{F}$ representing a (general) Boolean function f, any prime implicate of f can be derived from $\mathcal{F}$ by a chain (sequence) of consensuses. We shall show that in the case of pure Horn functions it is possible to be more restrictive in a sense that not all possible consensuses will be allowed to appear in the deriving chain.

Let C_1, C_2 be two clauses that have a consensus. Their consensus is called *k-restricted* iff at least one of the clauses C_1, C_2 has degree at most k, i.e. iff $\min\{\deg(C_1), \deg(C_2)\} \leq k$.

Proposition 4.1 *Let $\mathcal{F}$ be an arbitrary pure Horn CNF of a pure Horn function f and let C be an arbitrary prime implicate of f of degree k. Then C can be derived from $\mathcal{F}$ by a sequence of k-restricted consensuses.*

Proof. Let $C = \overline{x}_1 \vee \overline{x}_2 \vee \ldots \vee \overline{x}_{k-1} \vee y$. Since C is an implicate of f, the forward chaining procedure on $\mathcal{F}$ starting with the set $S = \{x_1, x_2, \ldots, x_{k-1}\}$ eventually includes y into $R_{\mathcal{F}}(S)$. Let us denote $z_1, z_2, \ldots, z_t$ the variables that are subsequently included into $R_{\mathcal{F}}(S)$ before y is included (we assume that they are included in the same order in which they are numbered), in particular let z_1 be the first variable included and let $y = z_t$.

We shall show by induction on i that for every z_i, $1 \leq i \leq t$ there exists a set of indices $I_i \subseteq \{1, 2, \ldots, k-1\}$ such that a clause

$$C_i = \bigvee_{j \in I_i} \overline{x}_j \vee z_i$$

can be derived from $\mathcal{F}$ by a sequence of k-restricted consensuses.

- Base case $i = 1$.
  By the definition of forward chaining there exists a set I_1 such that the clause C_1 is directly present in $\mathcal{F}$ (i.e. no consensus is necessary).

- Induction step.
  Let us assume that the induction hypothesis is true for all $i < \ell$ (where $\ell \leq t$). Since z_ℓ was included into $R_{\mathcal{F}}(S)$ there must exist subsets of indices $J \subseteq \{1, 2, \ldots, k-1\}$ and $K \subseteq \{1, 2, \ldots, \ell - 1\}$ such that the clause

  $$\tilde{C} = \bigvee_{j \in J} \overline{x}_j \vee \bigvee_{i \in K} \overline{z}_i \vee z_\ell$$

  is in $\mathcal{F}$. By the induction hypothesis each of the clauses C_i can be derived from $\mathcal{F}$ by k-restricted consensuses for every $i < \ell$, so in particular for every $i \in K$. But now the chain of k-restricted consensuses starting with $\tilde{C}$ and using one by one (in an arbitrary order) the clauses C_i, $i \in K$ for subsequent consensuses (where the result of each consensus is immediately used as one of the parent clauses in the next consensus) eliminates from $\tilde{C}$ all the literals $\overline{z}_i$, $i \in K$ and possibly adds some literals $\overline{x}_j$, $j \in L \subseteq \{1, 2, \ldots, k-1\} \setminus J$. Therefore the above consensus chain derives the clause C_ℓ where $I_\ell = J \cup L$.

So now we know that there exists a subset of indices $I_t \subseteq \{1, 2, \ldots, k-1\}$ such that the clause

$$C_t = \bigvee_{j \in I_t} \overline{x}_j \vee z_t = \bigvee_{j \in I_t} \overline{x}_j \vee y$$

can be derived from $\mathcal{F}$ by a sequence of k-restricted consensuses. That means that C_t is an implicate of the function f. However, we assumed that the clause C is a prime implicate of f and therefore necessarily $I_t = \{1, 2, \ldots, k-1\}$ and $C_t = C$ which concludes the proof.

□

An immediate consequence of the theorem is that any quadratic implicate can be derived by a consensus chain where the degree of each clause resulting from a consensus is less or equal than the bigger degree among the parent clauses. This follows from the fact that in each 2-restricted consensus (at least) one of the parent clauses is quadratic and since one literal from each parent clause is eliminated by the consensus the resulting clause cannot have a higher degree than the other parent clause. This means in particular that for each quadratic implicate of a pure Horn function represented by a CNF $\mathcal{F}$ there exists a consensus chain which derives the implicate and in which the degree of every intermediate clause is dominated by the maximum degree of a clause in $\mathcal{F}$.

The next theorem will generalize the above observation to get a bound on the degree of intermediate clauses in a consensus chain deriving an implicate of degree k. The bound will be dependent only on the maximum degree of a clause in the given CNF and on the number k. This is in a contrast to the case of general Boolean functions where no bound for the degree of intermediate consensuses given in terms of the degree of the implicate to be derived and/or in terms of the degrees of clauses in the given CNF exists. The only (trivial) bound in this case is the number of variables.

Corollary 4.2 *Let $\mathcal{F}$ be an arbitrary pure Horn CNF of a pure Horn function f, let C be an arbitrary prime implicate of f of degree k, and let q be the maximum degree of a clause in $\mathcal{F}$. Then C can be derived from $\mathcal{F}$ by a sequence of k-restricted consensuses where every intermediate clause has degree at most $q + k - 2$.*

Proof. If we follow the proof of the previous theorem all we need to show is that the consensus chain starting with the clause $\tilde{C}$ and using the clauses C_i, $i \in K$ to derive the clause C_ℓ has the desired property, i.e. that no intermediate clause has degree bigger than $q + k - 2$. But this is more or less obvious since each intermediate clause contains at most $k - 1$ x-literals and at most one less z-literals than the clause $\tilde{C}$ (since in every consensus one of the z-variables is eliminated). Therefore each intermediate clause contains at most $(k - 1) + (q - 1) = q + k - 2$ literals.

□

5 Quadratic Implicates Of a Pure Horn Function.

In this section we shall investigate quadratic implicates of a given pure Horn function. Every such implicate consists of exactly one negative and exactly one positive literal, i.e. has a form $\overline{x} \vee y$ which is logically equivalent to $x \longrightarrow y$ (x implies y). Therefore if we consider the set of all proposition letters used in the given function to be a set of nodes, every quadratic implicate can be represented by a directed arc and the set of all quadratic implicates by a directed graph. It is easy to see that the strong components of this directed graph correspond to classes of logically equivalent proposition letters. Discovering these equivalence classes is important for two reasons :

- they give an additional insight into the logical structure of the given function which may not be apparent from the given CNF
- they can be used for a heuristic minimization of the given function

The latter reason we be studied in more detail in the next subsection.

5.1 Heuristic Minimization.

The problem of finding a minimal representation of a given pure Horn function arises and constitutes an important problem in several application areas. One such area is the design of propositional expert systems (also known as "knowledge bases" or "rule bases") [8, 9]. The underlying logical structure of many rule bases used in practice corresponds to a pure Horn function and finding the most compact representation of such a rule base (which is desired for quick query answering and low memory requirements) is tantamount

to minimizing the corresponding pure Horn function. Another area in which the problem of pure Horn minimization plays an important role is the area of relational databases, namely the question how to optimally utilize functional dependencies among attributes [3, 11] (the Boolean function representing the functional dependencies is always pure Horn).

The problem can be stated as follows. Given a pure Horn CNF find an equivalent CNF (i.e. a CNF representing the same function) which is minimum with respect to a certain criterium, e.g. which consists of a minimum number of clauses or a minimum number of literals. It was proven in [11] and [1] that this problem is NP-complete with respect to both of the above criteria.

For this reason good minimization heuristics are sought. One simple heuristic makes use of the classes of equivalent proposition letters. Given a pure Horn CNF $\mathcal{F}$ the heuristic is :

1. Construct the directed graph corresponding to all quadratic implicates of $\mathcal{F}$.
2. Detect the strong components of the graph. Those correspond to the classes of logically equivalent proposition letters.
3. Remove from $\mathcal{F}$ all clauses that have all literals within a single equivalence class and replace them by a simple cycles of quadratic clauses over all proposition letters in a class (one cycle per each class). The idea is that for each equivalence class the added clauses constitute a minimum expression that implies the equivalence of the proposition letters in the class.

As an example consider the pure Horn CNF

$$\mathcal{F} = (\overline{x}_1 \vee \overline{x}_2 \vee x_4)(\overline{x}_4 \vee x_6)(\overline{x}_8 \vee x_6)(\overline{x}_6 \vee x_5)(\overline{x}_5 \vee x_7)(\overline{x}_7 \vee x_6)(\overline{x}_5 \vee \overline{x}_6 \vee x_4)$$

$$(\overline{x}_5 \vee \overline{x}_7 \vee x_8)(\overline{x}_4 \vee \overline{x}_8 \vee x_7)(\overline{x}_8 \vee x_9)(\overline{x}_8 \vee x_{10})(\overline{x}_9 \vee \overline{x}_{10} \vee x_7)(\overline{x}_5 \vee x_3)(\overline{x}_3 \vee x_1)(\overline{x}_3 \vee x_2)$$

which has quadratic implicates $\overline{x}_3 \vee x_1, \overline{x}_3 \vee x_2, \overline{x}_5 \vee x_3, \overline{x}_8 \vee x_9, \overline{x}_8 \vee x_{10}, \overline{x}_4 \vee x_5, \overline{x}_4 \vee x_6, \overline{x}_4 \vee x_7, \overline{x}_4 \vee x_8, \overline{x}_5 \vee x_4, \overline{x}_5 \vee x_6, \overline{x}_5 \vee x_7, \overline{x}_5 \vee x_8, \overline{x}_6 \vee x_4, \overline{x}_6 \vee x_5, \overline{x}_6 \vee x_7, \overline{x}_6 \vee x_8, \overline{x}_7 \vee x_4, \overline{x}_7 \vee x_5, \overline{x}_7 \vee x_6, \overline{x}_7 \vee x_8, \overline{x}_8 \vee x_4, \overline{x}_8 \vee x_5, \overline{x}_8 \vee x_6, \overline{x}_8 \vee x_7$. These implicates can be represented by a directed graph shown in Figure 1. The only

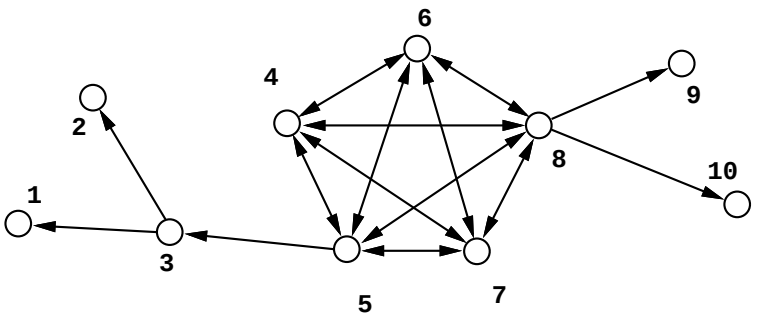

Figure 1: directed graph representing quadratic implicates

strong component of the graph consisting of more than one node is $C = \{4, 5, 6, 7, 8\}$. After deleting from $\mathcal{F}$ all clauses that have all literals within C and adding a cycle of quadratic clauses on C we get the following CNF

$$\mathcal{F}' = (\overline{x}_1 \vee \overline{x}_2 \vee x_4)(\overline{x}_4 \vee x_6)(\overline{x}_6 \vee x_8)(\overline{x}_8 \vee x_7)(\overline{x}_7 \vee x_5)(\overline{x}_5 \vee x_4)$$

$$(\overline{x}_8 \vee x_9)(\overline{x}_8 \vee x_{10})(\overline{x}_9\overline{x}_{10} \vee x_7)(\overline{x}_5 \vee x_3)(\overline{x}_3 \vee x_1)(\overline{x}_3 \vee x_2)$$

The improvement is from 15 clauses and 35 literals to 12 clauses and 26 literals. However $\mathcal{F}'$ is not a minimum CNF of the given function. One of the minimum CNFs (not unique) is

$$\mathcal{F}'' = (\overline{x}_1 \vee \overline{x}_2 \vee x_4)(\overline{x}_4 \vee x_6)(\overline{x}_6 \vee x_8)(\overline{x}_7 \vee x_5)(\overline{x}_8 \vee x_9)$$

$$(\overline{x}_8 \vee x_{10})(\overline{x}_9\overline{x}_{10} \vee x_7)(\overline{x}_5 \vee x_3)(\overline{x}_3 \vee x_1)(\overline{x}_3 \vee x_2)$$

consisting of 10 clauses and 22 literals ($\mathcal{F}''$ is minimum with respect to both the number of clauses and the number of literals).

Another possibility is to pick a "representative" from each equivalence class and instead of adding the cycles of quadratic clauses simply replace each literal by its corresponding representative and keep the lists of equivalent propositional letters separately. This further simplifies the formula. In the above case we would get

$$\mathcal{F}''' = (\overline{x}_1 \vee \overline{x}_2 \vee x_4)(\overline{x}_4 \vee x_9)(\overline{x}_4 \vee x_{10})(\overline{x}_9\overline{x}_{10} \vee x_4)(\overline{x}_4 \vee x_3)(\overline{x}_3 \vee x_1)(\overline{x}_3 \vee x_2)$$

where x_4 was picked as the representative of the only nontrivial equivalence class. This is really not a minimization in the sense defined earlier in this section ($\mathcal{F}'''$ is not equivalent to $\mathcal{F}$) but such an approach may be suitable for certain applications.

One question, however, remains unaswered. That is how to generate all the quadratic implicates in the first step of the heuristic. The next subsection will answer this question.

5.2 Generating Quadratic Implicates.

Proposition 4.1 says that every qudratic implicate can be derived from the given CNF by a chain of 2-restricted consensuses. The fact that these consensuses are "nonexpanding" (the result cannot have a higher degree than both parent clauses) as proved in corollary 4.2 suggests an existence of a polynomial time procedure for generating the quadratic implicates.

Indeed such procedures exist and perhaps the simplest one is the following application of forward chaining. By lemma 3.1 $\overline{x}_i \vee x_j$ is an implicate of a pure Horn CNF $\mathcal{F}$ iff $x_j \in R_{\mathcal{F}}(\{x_i\})$. Therefore if we want to find out for which j say $\overline{x}_1 \vee x_j$ is an implicate, all we have to do is run the forward chaining procedure on $\mathcal{F}$ with $Q = \{x_1\}$. This can be done in $O(|\mathcal{F}|)$ time. Obviously if we want to generate all quadratic implicates it is enough to run the forward chaining procedure on $\mathcal{F}$ subsequently for all proposition letters, i.e. with $Q = \{x_1\}$, $Q = \{x_2\}$ up to $Q = \{x_n\}$. This procedure has a complexity $O(n|\mathcal{F}|)$.

What may be a little bit surprising is that this very simple procedure does the best possible job if we measure the performance by a worst case complexity. Just consider the following CNF

$$\mathcal{F} = (\overline{x}_1 \vee x_2)(\overline{x}_2 \vee x_3)\ldots(\overline{x}_{n-1} \vee x_n).$$

It has a size only $|\mathcal{F}| = 2(n-1) = O(n)$ but nevertheless any quadratic clause $\overline{x}_i \vee x_j$ with $i < j$ is its implicate. Since there is $O(n^2)$ such clauses, any procedure generating all quadratic implicates of $\mathcal{F}$ must take at least $O(n^2) = O(n|\mathcal{F}|)$ steps.

Unfortunately this argument has one weak point. If we think of the set of all quadratic implicates as of a directed graph, it is obvious that this graph is transitively closed. However, in most application we are really interested only in the structure of the directed graph (e.g. for the minimization heuristic described in the previous section we only need to detect the strong components of the graph) and therefore we do not need to generate the entire transitive closure. What we are interested in is generating a *skeleton graph* i.e. a minimal (under inclusion) graph such that its transitive closure represents the set of all quadratic implicates.

In this sense the previous example fails because the CNF $\mathcal{F}$ itself represents the skeleton graph of size $O(n)$ while the "blind" forward chaining procedure will still work $O(n|\mathcal{F}|)$ time. The natural question to ask here is whether the above is a typical case, and if not then how big can the skeleton graph get in terms of the input CNF.

Once again the answer may be surprising. There are CNFs of size $O(n)$ for which even the skeleton graph of the quadratic implicants has size $O(n^2) = O(n|\mathcal{F}|)$. Therefore on these CNFs even every procedure that generates just the skeleton graph cannot beat the complexity of the simple forward chaining procedure.

The next subsection is devoted to constructing such CNFs.

5.3 CNFs With Large Skeleton Graphs.

The goal of this subsection is to construct CNFs of size $O(n)$ which have skeleton graphs of size $O(n^2) = O(n|\mathcal{F}|)$. This will show that the simple forward chaining procedure that generates all quadratic implicates has the best possible worst case complexity even when compared to procedures that generate only quadratic implicates corresponding to a skeleton graph.

Let us first construct a formula $\mathcal{F}$ that will not yet achieve our goal but which will help in understanding the subsequent more complicated construction.

Let us divide the set of all n proposition letters into 3 subsets of sizes $n_1 = \frac{1}{2}(n - \sqrt{n})$ (twice) and $n_2 = \sqrt{n}$ (to avoid problems with rounding we may assume for now that n is a square of an even number). Let us denote the proposition letters by $s_1, \ldots, s_{n_1}; x_1, \ldots, x_{n_2}$ and $h_1, \ldots, h_{n_1}$. Now let $\mathcal{F}$ consists of the following clauses :

- $\overline{s_i} \vee x_j \quad \forall i \in \{1, \ldots, n_1\}\ \forall j \in \{1, \ldots, n_2\}$
- $\overline{x_i} \vee \overline{x_j} \vee h_{\varphi(i,j)} \quad \forall i \neq j \in \{1, \ldots, n_2\}$ where φ is an arbitrary one to one mapping of pairs from $\{1, \ldots, n_2\}$ onto elements of $\{1, \ldots, n_1\}$

or more informally

- a quadratic clause (i.e. an implication) from each $s-$variable to each $x-$variable
- a cubic clause for each pair of $x-$variables as subgoals having a unique $h-$variable as the head of the clause. Note that $\binom{n_2}{2} = n_1$ and so φ exists.

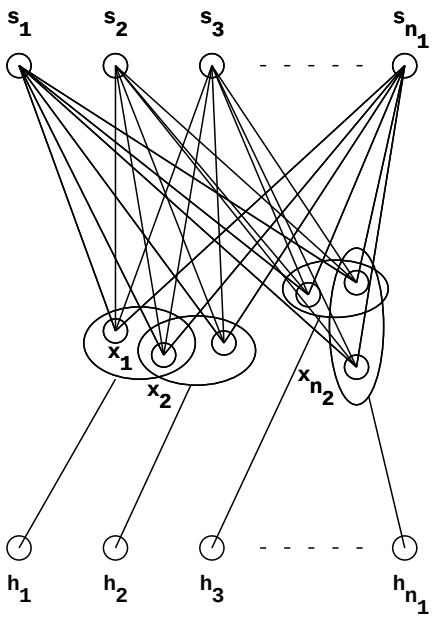

Figure 2:

Figure 2 shows the above relations. There is $O(n^{\frac{3}{2}})$ quadratic clauses (first type) and $O(n)$ cubic clauses (second type). Hence $|\mathcal{F}| \sim O(n^{\frac{3}{2}})$. However, every clause of a type

- $\overline{s_i} \vee h_j \quad i, j \in \{1, \ldots, n_1\}$

(i.e. an implication from s_i to h_j) is a quadratic implicate of $\mathcal{F}$. There is $O(n^2)$ such clauses and so $\mathcal{F}$ has an $O(n^{\frac{1}{2}})$ "expansion" between the input size and the skeleton graph size.

The reason why $\mathcal{F}$ is not yet "good enough" is that there are too many implications between $s-$variables and $x-$variables. However, their number can be reduced as follows. Let n_3 be a number such that $\binom{n_3}{2} = n_2$. Clearly $n_3 \sim O(n^{\frac{1}{4}})$. Take the last n_3 $s-$variables, rename them $y_1, \ldots, y_{n_3}$ and construct clauses

- $\overline{y_i} \vee \overline{y_j} \vee x_{\varphi(i,j)} \quad \forall i \neq j \in \{1, \ldots, n_2\}$ where φ is an arbitrary one to one mapping of pairs from $\{1, \ldots, n_3\}$ onto elements of $\{1, \ldots, n_2\}$

There is only $O(n^{\frac{1}{2}})$ such clauses. Now remove all implications between $s-$variables and $x-$variables and replace them by all implications between $s-$variables and $y-$variables (see figure 3). More formaly, clauses

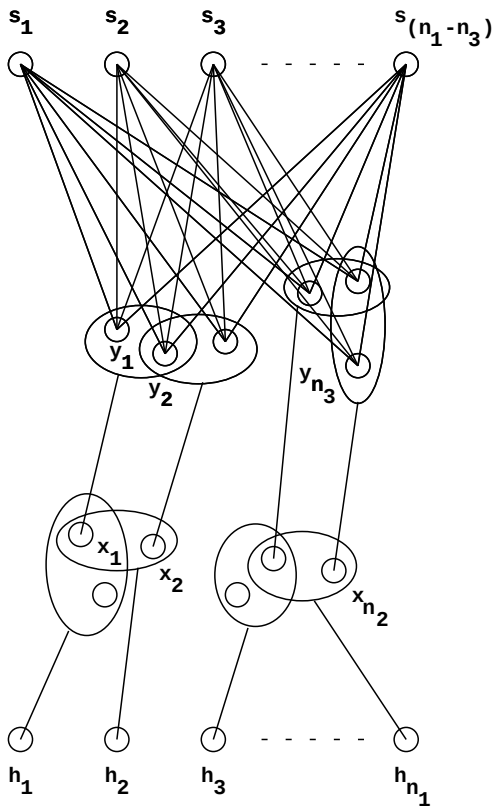

Figure 3:

- $\overline{s_i} \vee x_j \quad \forall i \in \{1, \ldots, n_1\}\ \forall j \in \{1, \ldots, n_2\}$

are removed and are replaced by clauses

- $\overline{s_i} \vee y_j \quad \forall i \in \{1, \ldots, n_1\}\ \forall j \in \{1, \ldots, n_3\}$

Since there are only $O(n^{\frac{5}{4}})$ such clauses, we managed to reduce $|\mathcal{F}|$ to $O(n^{\frac{5}{4}})$ while maintaining the $O(n^2)$ size of the output.

Now it is more or less obvious that iterating the above procedure would eventually yield a CNF that we are looking for. However, to avoid problems with rounding (note that in the above construction there is no guarantee that a number n_3 such that $\binom{n_3}{2} = n_2$ exists) we shall start building the formula from "the top" rather than from "the bottom".

Let $n_1 = 6$ be the size of the "topmost" subset of proposition letters that will serve as subgoals of the first level of cubic clauses and denote these proposition letters $x_{11}, \ldots, x_{1n_1}$. Define $n_2 = \binom{n_1}{2}$, pick a set of yet unlabeled proposition letters and denote them $x_{21}, \ldots, x_{2n_2}$. Keep iterating the last step until $n_l \geq \frac{n}{4}$, where l is the first index satisfying the inequality. In this $l-$th level pick only $\frac{n}{4}$ (rather than n_l) proposition letters and denote them $x_{l1}, \ldots, x_{l\frac{n}{4}}$. Let $\hat{n} = \sum_{i=1}^{l-1} n_i + \frac{n}{4}$ and $\tilde{n} = n - \hat{n}$. Denote the remaining unlabeled proposition letters by $s_1, \ldots, s_{\tilde{n}}$. Before defining the clauses of G let us first derive an upper bound for $\hat{n}$.

We know that $n_{i+1} = \binom{n_i}{2} = \frac{1}{2} n_i(n_i - 1)$ for all $i \in \{1, \ldots, l-1\}$. Using the inequalities $\frac{1}{3}k^2 \leq \frac{1}{2}k_i(k_i - 1) \leq \frac{1}{2}k^2$ which holds $\forall k \geq 3$ we can write

$$\frac{1}{3}n_i^2 \leq n_{i+1} \leq \frac{1}{2}n_i^2 \quad \forall i \in \{1, \ldots, l-1\}$$

and therefore

$$3\left(\frac{n_1}{3}\right)^{2^i} \leq n_{i+1} \leq 2\left(\frac{n_1}{2}\right)^{2^i} \quad \forall i \in \{1, \ldots, l-1\}$$

or equivalently

$$3 \cdot 2^{2^i} \leq n_{i+1} \leq 2 \cdot 3^{2^i} \quad \forall i \in \{1, \ldots, l-1\}$$

Now the inequality $2\sum_{i=0}^{j} 3^{2^i} \leq 3 \cdot 2^{2^{j+1}}$ which can be easily proved by induction on j gives us

$$\sum_{i=1}^{l-2} n_i \leq 2\sum_{i=0}^{l-3} 3^{2^i} \leq 3 \cdot 2^{2^{l-2}} \leq n_{l-1} < \frac{n}{4}$$

which implies

$$\hat{n} = \sum_{i=1}^{l-2} n_i + n_{l-1} + \frac{n}{4} < \frac{3n}{4}$$

and therefore both $\hat{n}$ and $\tilde{n}$ lie within the interval $(\frac{n}{4}, \frac{3n}{4})$. Now the CNF $\mathcal{F}$ will consist of the following clauses :

- $\overline{s_j} \vee x_{1k} \quad \forall j \in \{1, \ldots, \tilde{n}\}\ \forall k \in \{1, \ldots, 6\}$
- $\overline{x_{ij}} \vee \overline{x_{ik}} \vee x_{i+1,\varphi_i(j,k)} \quad \forall i \in \{1, \ldots, l-2\}\ \forall j \neq k \in \{1, \ldots, n_i\}$ where φ_i ($i \in \{1, \ldots, l-2\}$) is an arbitrary one to one mapping of pairs from $\{1, \ldots, n_i\}$ onto elements of $\{1, \ldots, n_{i+1}\}$
- $\overline{x_{l-1,j}} \vee \overline{x_{l-1,k}} \vee x_{l,\varphi_{l-1}(j,k)}$ for arbitrarily chosen $\frac{n}{4}$ pairs (j,k) , $j \neq k \in \{1, \ldots, n_{l-1}\}$ (out of the total number of n_l pairs) where φ_{l-1} is an arbitrary one to one mapping of the chosen pairs onto elements of $\{1, \ldots, \frac{n}{4}\}$

Figure 4 shows the above relations. There is $6\tilde{n}$ quadratic clauses (first type) and $\hat{n} - 6$ cubic clauses (second and third types) in $\mathcal{F}$. The latter number comes from the fact that every $x-$variable except $x_{11}, \ldots, x_{16}$ is a head of exactly one cubic clause. So $|\mathcal{F}| = 2 \cdot 6\tilde{n} + 3(\hat{n} - 6) \sim O(n)$. On the other hand every clause of a type

- $\overline{s_j} \vee x_{lk} \quad j \in \{1, \ldots, \tilde{n}\}\ k \in \{1, \ldots, \frac{n}{4}\}$

is a quadratic implicate of $\mathcal{F}$. There is $\tilde{n}\frac{n}{4} > \frac{1}{16}n^2$ such clauses which finishes the construction.

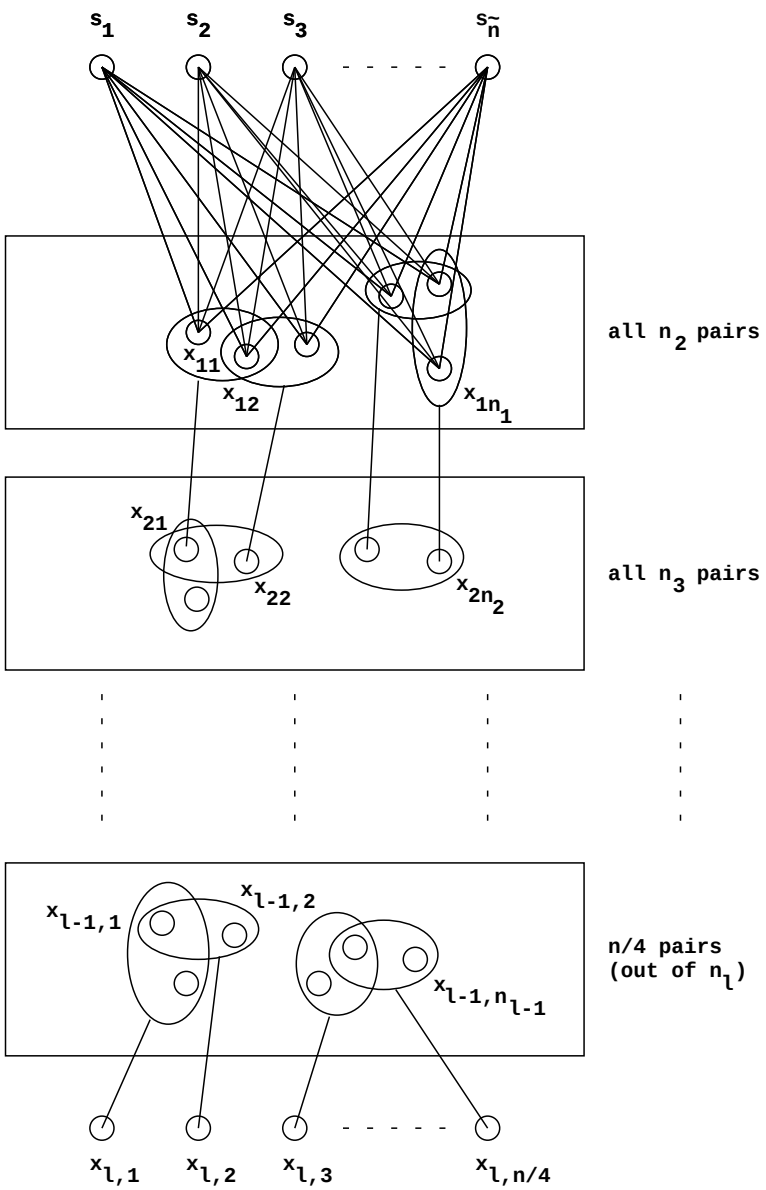
s1
s2
s3
s~n
x11
x12
x1n1
all n2 pairs
x21
x22
x2n2
all n3 pairs
xl-1,1
xl-1,2
xl-1,nl-1
n/4 pairs
(out of nl)
xl,1
xl,2
xl,3
xl,n/4

Figure 4:

References

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

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

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

[4] 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.

[5] 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.

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

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

[8] 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.

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

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

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

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

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