International Journal of Computer Science & Applications *© Technomathematics Research Foundation*
Vol. I, No. 1, pp. 1-11

Prime Implicates of First Order Formulas

Manoj K. Raut and Arindama Singh[1]
Department of Mathematics
Indian Institute of Technology Madras
Chennai-600036, India

Abstract

This paper extends the notion of prime implicates to first order logic formulas without equality which are assumed to be in Skolem Conjunctive Normal Form. Using the extended notions of consensus and subsumption it is shown that the consensus-subsumption algorithm for computing prime implicates well known for propositional formulas can be conditionally lifted to first order formulas.

Keywords: First order logic, prime implicates, consensus, subsumption

1. Introduction

Several frameworks such as assumption based truth maintenance systems [7, 8, 14] and etc. have been devised to handle incomplete and uncertain information. The main idea in these frameworks is to execute all possible reasoning processes with respect to a knowledge base before a goal (a query) is presented to the system. In clause maintenance system-like frame works this involves computing the set of all prime implicates of the propositional knowledge base. There have been several algorithms for generating the set of prime implicates for propositional knowledge base; see for example [4, 6, 9, 11, 12, 17, 18].

Computing prime implicates is one type of exact knowledge compilation in propositional theory. Knowledge compilation is a technique in which on-line inference with respect to a knowledge base KB being intractable is compiled into an approximate or equivalent theory with respect to which online inference is tractable. So a clausal propositional theory is mapped into a set of prime implicates of the theory in order to make query answering efficient. If $\pi_1, \ldots, \pi_k$ are prime implicates of the propositional knowledge base KB, then $\pi_1 \wedge \ldots \wedge \pi_k$ is equivalent to KB. So for any query Q assumed to be a disjunctive clause, $KB \models Q$ iff for every prime implicate π_i, $\pi_i \models Q$, i.e, $\pi_i \subseteq Q$, which can be checked in polynomial time from the set of prime implicates.

Most of the research works have been confined to propositional theory [2, 3, 10] despite the higher expressing capacity of first order theory. Since expressive power is restricted in propositional theory, first order theory is needed to represent knowledge in many problems. In such

[1]Corresponding Author, email: asingh@iitm.ac.in

systems we use some restrictive version, i.e., clausal form of first order logic to store knowledge. The clausal form or free variable forms are obtained using by skolemization and converting the formula to one of disjunctive or conjunctive normal forms. In this paper we assume that first order formulas are in SCNF (Skolem Conjunctive Normal Form) formulas and use the mgu (most general unifier) to derive the set of prime implicates of such formulas and then explore the properties of primeness. In order to use the consensus-subsumption method [6, 13, 19] we also extend these notions to first order logic [15]. It is shown that the consensus-subsumption algorithm for computing prime implicates of a first order theory is partially correct.

This method of computation of prime implicates is useful for efficient abductive hypothetical reasoning to prove a given query. It is also useful in Truth Maintenance Systems and non-monotonic reasoning. But first order knowledge compilation encounters a number of issues such as semi-decidability of entailment and of termination of the compilation algorithm which are not visible in propositional knowledge compilation. In our approach, termination is established due to well defined composition of substitution associated with implicates.

This paper is organized as follows. The required extension of notions are carried out in Section 2. The properties of primeness and query entailment problem are explored in section 3. The algorithm of consensus-subsumption method and its correctness are presented in Section 4. It also includes examples illustrating the algorithm. Section 5 concludes the paper.

2. Preliminaries

We briefly scan through the syntactic machinery of first order logic without equality for the purpose of establishing the notation and notion used in the paper.

Terms and formulas, free and bound variables, scopes of occurrences of quantifiers, and etc. are defined as usual. Formulas are denoted by upper case letters. If A is a formula and $x_1, \ldots, x_n$ are free variables in A, then the result of substituting the terms $t_i, 1 \leq i \leq n$ for every free occurrence of x_i, in A is denoted by $A[x_1/t_1, \ldots, x_n/t_n]$. The empty substitution is denoted by ϵ. For an interpretation i, we write $i \models X$ if i is a model of the formula X. For a formula (or a set of formulas) Σ and any formula Y, we write $\Sigma \models Y$ to denote the fact that for every interpretation i if i is a model of every formula in Σ then i is a model of Y.

A literal is an atomic formula or negation of an atomic formula. A (disjunctive) clause is a finite disjunction of literals, also represented as a set of literals. A quantifier free formula is in conjunctive normal form(CNF) if it is a conjunction of disjunctive clauses, also represented as a set of clauses, a set of sets of literals. Each formula is converted to a quantifier free formula by Skolemization before represented as a CNF. Thus, we consider only Skolem Conjunctive Normal Form Formulas (SCNF).

Two literals r and s are *complementary* to each other iff the set $\{r, \neg s\}$ is unifiable with respect to a most general unifier ζ. In such a case the most general unifier ζ of the set $\{r, \neg s\}$ is called a complementary substitution. For example, Qxb and $\neg Qf(a)y$ are complementary to each other with respect to the complementary substitution (a most general unifier) $[x/f(a), y/b]$.

A clause is called *fundamental* if it doesn't contain a literal and it's negation. So a non-fundamental clause is valid. We avoid taking non-fundamental clauses in a formula in SCNF because the universal quantifiers appearing in the beginning of the formula can appear before each conjunct of the SCNF. A clause is therefore always interpreted as its universal closure, i.e, all the free variables are assumed to be universally quantified. A clause containing a literal and its negation is always valid and can be discarded from the formula without changing the truth value of the formula. So each clause in a formula of the knowledge base is assumed to be fundamental.

Let C_1 and C_2 be two clauses. Then C_1 *subsumes* C_2 iff there is a substitution σ such that $C_1\sigma \subseteq C_2$. For example, $\{\neg Rxf(a), \neg Py\}$ subsumes $\{\neg Rg(a)f(a), \neg Py, Qz\}$ with $\sigma = [x/g(a)]$. A fundamental conjunctive clause C is an *implicate* of a formula X (assumed to be in SCNF) iff $X\sigma \models C$ for a substitution σ. $\Psi(X)$ denotes the set of all implicates of X. An implicate C is a *prime implicate* of X iff there is no other implicate $C^{'}$ of X such that $C^{'}$ subsumes C. We denote the set of prime implicates of a formula X by $\Pi(X)$. Note that the notion of prime implicate is well defined as the knowledge base contains clauses unique up to subsumption. Let Y be a set of fundamental clauses. The residue of subsumption of Y, denoted by $Res(Y)$ is a subset of Y such that for every clause $C \in Y$, there is a clause $D \in Res(Y)$ where D subsumes C; and no clause in $Res(Y)$ subsumes any other clause in $Res(Y)$.

Let D_1 and D_2 be two clauses in propositional logic. Let l be a literal such that $l \in D_1$ and $\neg l \in D_2$. Then propositional consensus of D_1 and D_2 is the clause $PCON(D_1, D_2) = D = (D_1 - \{l\}) \cup (D_2 - \{\neg l\})$ provided the new clause D is fundamental. Thus $PCON(D_1, D_2)$ is simply the propositional resolvent with the constraint that it should be fundamental. Note that a non-fundamental clause is tautology, being disjunctive, and thus it need not be kept in the compiled knowledge base. In contrast, we use $CON(C_1, C_2)$ for consensus of two clauses in first order logic. Let C_1 and C_2 be two clauses in X and $r \in C_1$ and $s \in C_2$ be two complementary literals with respect to a most general unifier σ. Then the consensus of C_1 and C_2 is $C = CON(C_1, C_2) = \{(C_1 - \{r\}) \cup (C_2 - \{s\})\}\sigma$ provided that C is fundamental. In that case, C is also equal to $\{(C_1\sigma - \{t\}) \cup (C_2\sigma - \{\neg t\})\}$ which is, in turn, equal to $PCON(C_1\sigma, C_2\sigma)$ where $r\sigma = t$ and $s\sigma = \neg t$ and t is a literal. We write $CON(C_1, C_2)$ as the consensus of C_1 and C_2 and $PCON(C_1\sigma, C_2\sigma)$ as the propositional consensus of $C_1\sigma$ and $C_2\sigma$. For example, if $C_1 = \{Rbx, \neg Qg(a)\}$ and $C_2 = \{Rab, Qz\}$ then $CON(C_1, C_2) = \{Rbx, Rab\} = PCON(C_1[z/g(a)], C_2[z/g(a)])$. If C is a consensus of two clauses C_1 and C_2 with respect to a substitution σ then we say that the consensus C is associated with σ. Obviously $X\sigma \models C$. Let $X = \{C_1, \ldots, C_n\}$. So each C_i is associated with ϵ by convention. Let C and D be two clauses (implicates) associated with σ_1 and σ_2 respectively. Then their consensus with respect to a substitution σ is defined provided $\sigma_1\sigma = \sigma_2\sigma$. In that case the consensus is $PCON(C\sigma, D\sigma)$ and the consensus is associated with the substitution $\sigma_3 = \sigma_1\sigma = \sigma_2\sigma$. Our problem is to compute the set of all prime implicates of a given formula X in SCNF.

3. Properties of Primeness

Let $X = \{C_1, \ldots, C_n\}$ be a formula where each clause C_i is fundamental. Then each C_i is an implicate of X with respect to the empty substitution, but each may not be a prime implicate. The key is the subsumption of implicates of X. Since clauses are disjunctive, we observe that: If C_1 subsumes C_2 then there is a substitution σ such that $C_1\sigma \models C_2$. Our aim is to see how deletion of subsumed clauses lead to the computation of prime implicates.

Lemma 3.1 A clause C is an implicate of X if and only if there is a prime implicate $C^{'}$ of X such that $C^{'}$ subsumes C.

Proof If the implicate C is not prime, then there is an implicate $D \neq C$ and a substitution σ such that $D\sigma \subseteq C$. Let $\mathcal{D}$ be the set of all such clauses D, *i.e.*, $\mathcal{D} = \{D$: D is an implicate of X and D$\sigma \subseteq$ C, for some substitution $\sigma\}$. Then there exists a finite subset $\mathcal{D}^{'} = \{D_1, \ldots, D_n\}$ of $\mathcal{D}$ such that for each $D \in \mathcal{D}$ there is $D_i \in \mathcal{D}^{'}$ and a substitution τ such that $D = D_i\tau$ since there are only a finite number of variables in D and C is finite. Now $\mathcal{D}^{'}$ is a finite set having a strict partial order as subsumption. Each element of $\mathcal{D}^{'}$ is an implicate of X. Any minimal element of $\mathcal{D}^{'}$ is a prime implicate of X.

Conversely, being a prime implicate of X, $C^{'}$ is also an implicate of X. So $X\sigma_1 \models C^{'}$ which implies $X\sigma_1 \models C^{'}\sigma_2$. As $C^{'}$ subsumes C, $C^{'}\sigma_2 \models C$. So $X\sigma_1 \models C$, i.e, C is an implicate of X. □

Lemma 3.2 $X \equiv \Psi(X)$.

Proof Since every clause of X is an implicate, $X \subseteq \Psi(X)$. As both X and $\Psi(X)$ are conjunctive, $\Psi(X) \models X$.

Conversely, let i be a model of X. If C is an implicate of X, then by definition, $i \models C$. As, $\Psi(X)$ is the set of all such implicates C and $\Psi(X)$ is, in fact, a conjunction of such clauses, $i \models \Psi(X)$. Therefore, $X \models \Psi(X)$. □

Lemma 3.3 $\Psi(X) \equiv Res(\Psi(X))$.

Proof As $Res(\Psi(X)) \subseteq \Psi(X)$ and both are conjunction of clauses, $\Psi(X) \models Res(\Psi(X))$.

Conversely, let i be a model of $Res(\Psi(X))$. Then, $i \models C$ for every $C \in Res(\Psi(X))$. If all the clauses of $\Psi(X)$ are in $Res(\Psi(X))$ then it is through. If there exists at least one clause $D \in \Psi(X)$ such that $D \notin Res(\Psi(X))$ then there exists a clause $C^{'} \in Res(\Psi(X))$ such that $C^{'}$ subsumes D. Then, $C^{'}\sigma \models D$. As i is a model of $Res(\Psi(X))$, $i \models C^{'}$ and $i \models D$. This shows that $i \models \Psi(X)$. □

Lemma 3.4 $Res(\Psi(X)) = \Pi(X)$.

Proof If $C \notin \Pi(X)$, then there is an implicate D, a clause in $\Psi(X)$ such that D subsumes C. Then $C \notin Res(\Psi(X))$, i.e, $Res(\Psi(X)) \subseteq \Pi(X)$.

For the converse, let $C \in \Pi(X)$. Then, as an implicate of X, $C \in \Psi(X)$. As C is prime there does not exist any implicate D of X such that D subsumes C, i.e, $C \in Res(\Psi(X))$. □

From the above lemmas we observe that $X \equiv \Pi(X)$. The following results show the relation between consensus closure and prime implicates. Note that if C is an implicate of X and there exist two substitutions σ and τ such that $X\sigma \models C$ and $X\tau \models C$, then due to unification, there exists a substitution δ such that $\sigma = \delta\sigma'$ and $\tau = \delta\tau'$ for some substitutions σ' and τ'. As a consequence, if C, D are implicates of X, with $X\sigma \models C, X\tau \models D$, and $PCON(C\delta, D\delta)$ exists, then $\sigma\delta = \tau\delta$ holds. This observation is used in the proof of the following theorem.

Theorem 3.5 Consensus of two implicates of a formula is an implicate of the formula.

Proof Let C_1 and C_2 be two implicates of a formula X associated with σ_1 and σ_2, respectively. $CON(C_1, C_2) = PCON(C_1\sigma, C_2\sigma)$ provided $\sigma_1\sigma = \sigma_2\sigma$ for some substitution σ. So $C = CON(C_1, C_2) = ((C_1\sigma - \{t\}) \cup (C_2\sigma - \{\neg t\}))$. Let i be a model of $X\sigma_1\sigma$. Since C_1 is an implicate of X associated with σ_1, $X\sigma_1 \models C_1$. This implies $i \models C_1\sigma$. As $X\sigma_1\sigma = X\sigma_2\sigma$, we have similarly, $i \models C_2\sigma$. Now, if $i \models t$, then $i \not\models \neg t$. Since $i \models C_2\sigma$, we have $i \models C_2\sigma - \{\neg t\}$. On the other hand if $i \not\models t$, $i \models C_1\sigma$ implies that $i \models C_1\sigma - \{t\}$. In any case, $i \models ((C_1\sigma - \{t\}) \cup (C_2\sigma - \{\neg t\}))$. This proves that C is an implicate of X. □

For a clausal knowledge base X, let $M(X) = X \cup \{S : S$ is a propositional consensus of a pair of clauses in $X\}$. The *propositional consensus closure* of X is $\overline{M}(X) = \cup\{M^i(X) : i \in \mathbb{N}\}$. Note that propositional consensus closure treats all the literals as propositional variables in contrast to the consensus closure defined below where substitutions can change the literals. It is well known that $\overline{M}(X)$ can be constructed since the sequence $M^i(X)$ terminates. This notion of consensus closure is extended to first order case as follows.

For a set of clauses X, let $L(X)$ be the set of all consensus of clauses in X along with the clauses of X, *i.e.*, $L(X) = X \cup \{S : S$ is a consensus of each possible pair of clauses in $X\}$. We construct the sequence $X, L(X), L(L(X)), \ldots$, i.e, $L^{n+1}(X) = L(L^n(X))$ for $n \geq 0$, and $L^0(X) = X$. We write the *consensus closure* of X as $\overline{L}(X) = \cup\{L^i(X) : i \in \mathbb{N}\}$. From Theorem 3.5, it follows that $\overline{L}(X) \subseteq \Psi(X)$.

Example 3.1 Let $X = (Px \vee \neg Qg(a)) \wedge (Rby \vee Qy) \wedge (\neg Pg(w) \vee \neg Rxf(a)) = C_1 \wedge C_2 \wedge C_3$.
Note that C_1, C_2 and C_3 are associated with empty substitution ϵ. The consensus between C_1 and C_2 with respect to the substitution $[y/g(a)]$ is $C_4 = \{Px, Rbg(a)\}$, between C_1 and C_3 with respect to the substitution $[x/g(w)]$ is $C_5 = \{\neg Qg(a), \neg Rg(w)f(a)\}$, between C_2 and C_3 with respect to the substitution $[x/b, y/f(a)]$ is $C_6 = \{\neg Pg(w), Qf(a)\}$. Note that C_4 is associated with $[y/g(a)]$, C_5 is associated with $[x/g(w)]$, and C_6 is associated with $[x/b, y/f(a)]$. These are all possible consensus with suitable most general unifiers taken as substitutions.
Adding these three new clauses to X, we see that

$$\begin{aligned} L^1(X) = \{&\{Px, \neg Qg(a)\}, \{Rby, Qy\}, \{\neg Pg(w), \neg Rxf(a)\}, \{Px, Rbg(a)\},\\ &\{\neg Pg(w), Qf(a)\}, \{\neg Qg(a), \neg Rg(w)f(a)\}\} = \{C_1, \ldots, C_6\}. \end{aligned}$$

While computing $L^2(X)$, it is enough to compute possible consensus of clauses C_1,C_2,C_3 with

5

any of C_4,C_5,C_6 and consensus among the clauses C_4, C_5, C_6. The consensus between C_1 and C_4 is not possible because there are no complementary pair of literals in them. The consensus between C_1 (associated with the empty substitution ϵ) and C_5 (associated with $[x/b, y/f(a)]$) with respect to the substitution $[x/g(w)]$ is not possible as composition of substitution is not properly defined, i.e, as $[\epsilon][x/g(w)] \neq [x/b, y/f(a)][x/g(w)]$. Similarly the consensus between C_4 (associated with $[y/g(a)]$) and C_5 (associated with $[x/b, y/f(a)]$) is not possible with respect to the substitution $[x/g(w)]$ as composition of substitution is not properly defined, i.e, as $[y/g(a)][x/g(w)] \neq [x/b, y/f(a)][x/g(w)]$. Similarly no more consensus can be taken place between clauses of $L^1(X)$. This implies $L^1(X) = L^2(X)$. Therefore, $\overline{L}(X) = L^1(X)$.

The following result connects $\Psi(X)$, the set of all implicates with $\overline{L}(X)$, the consensus closure of X.

Theorem 3.6 C is an implicate of X if and only if there is $D \in \overline{L}(X)$ such that D subsumes C.

Proof As C is an implicate of X, $X\sigma_1 \models C$ for some substitution σ_1. Then $X\sigma_1\sigma_2 \models C$ propositionally for possibly another substitution σ_2. Write $\sigma = \sigma_1\sigma_2$. This implies that there is a propositional prime implicate D of $X\sigma$ such that $D \subseteq C$. By consensus subsumption theorem [6], D is in the propositional consensus closure of $X\sigma$, *i.e*, $D \in \overline{M}(X\sigma) = \cup_{k\in\mathbb{N}} M^k(X\sigma)$. We show that if $D \in M^k(X\sigma)$ then there is $\overline{D} \in \overline{L}(X)$ such that $\overline{D}\sigma \subseteq D$. This is accomplished by induction on k.

If $D \in M^0(X\sigma) = X\sigma$ then there is $\overline{D} \in X \subseteq \overline{L}(X)$ such that $\overline{D}\sigma = D$.

Lay out the induction hypothesis that for every $D_0 \in M^n(X\sigma)$, there is $\overline{D}_0 \in \overline{L}(X)$ such that $\overline{D}_0\sigma \subseteq D_0$. Let $D \in M^{n+1}(X\sigma)$. D is generated by taking propositional consensus from $M^n(X\sigma)$. Then there are $C_1, C_2 \in M^n(X\sigma)$ such that $D = (C_1 - \{l\}) \cup (C_2 - \{\neg l\})$. It is clear that $l = m\sigma$ for some literal m. By induction hypothesis, there are $\overline{C}_1, \overline{C}_2 \in \overline{L}(X)$ such that $\overline{C_1}\sigma \subseteq C_1$, $\overline{C_2}\sigma \subseteq C_2$. Then $\overline{D}\sigma = ((\overline{C}_1 - \{m\}) \cup (\overline{C}_2 - \{\neg m\}))\sigma \subseteq D$. Hence, $\overline{D} \in \overline{L}(X)$.

Conversely, let i be a model of X. Since $D \in \overline{L}(X)$, D is an implicate of X associated with some substitution σ_1. $X\sigma_1 \models D$ which implies $i \models D$. As D subsumes C, $D\sigma_2 \models C$ for some substitution σ_2. This gives $i \models C$. Then C is an implicate of X. □

Theorem 3.6 does not say that $\Psi(X) \subseteq \overline{L}(X)$, in general. However, such a relation exists between $\Pi(X)$ and $\overline{L}(X)$ as the following statement says.

Theorem 3.7 The set of all prime implicates is a subset of the consensus closure of X, i.e, $\Pi(X) \subseteq \overline{L}(X)$. Moreover, $\Pi(X) = Res(\overline{L}(X))$.

Proof Let $C \in \Pi(X)$. Since C is also an implicate of X, by Theorem 3.6, there is $D \in \overline{L}(X)$ such that D subsumes C. If $C \notin \overline{L}(X)$, then D, an implicate of X, subsumes C and $D \neq C$. This contradicts that C is prime. Hence $C \in \overline{L}(X)$. This proves that $\Pi(X) \subseteq \overline{L}(X)$. Due to Theorem 3.5, $\overline{L}(X) \subseteq \Psi(X)$. If $D \in \Psi(X) - \overline{L}(X)$, then by Theorem 3.6, D is subsumed by some clause, say, $C \in \overline{L}(X)$. Therefore, $Res(\overline{L}(X)) = Res(\Psi(X))$. Due to Lemma 3.4, we obtain $Res(\overline{L}(X)) = \Pi(X)$. □

4. Consensus Subsumption Method

The results in Section 3 suggest the following method for computing the set of prime implicates $\Pi(X)$ of an SCNF X. In the "algorithm" $CONSUM$ below, we use consensus followed by subsumption for computing prime implicates, as in propositional case, though with extended meanings of the operations that use substitutions. Recall that for a set of clauses A, $L(A)$ denotes the set of clauses of A along with the consensus of each possible pair of clauses of A. Once $L(A)$ has been obtained, computation of $L(L(A))$ will be wasteful if one starts fresh with $L(A)$ for taking consensus. It is enough to take a clause C form $L(A)-A$ and a clause D from $L(A)$ to compute $CON(C, D)$. Then $L(L(A))$ will be computed by $L(L(A)) = L(A) \cup \{CON(C, D) : C \in L(A) - A, D \in L(A)\}$. The algorithm applies subsumption on $L(A)$ and keeps the residue $Res(L(A))$ and then repeats the steps till two iteration steps produce the same result. The working out of the algorithm $CONSUM$ is further explained in the proof of Theorem 4.1.

Algorithm: $CONSUM$
Input: R, the set of clauses (in SCNF).
Output: $\Pi(R)$, the set of prime implicates of R.

```
begin
    update R by removing all non-fundamental clauses from it
    if $R = \phi$
        $\pi(R) = \phi$
    else
        $Q_0 := \phi$
        $i := 1$
        $Q_i := R$
        while $Q_i \neq Q_{i-1}$
        do
            compute $L(Q_i)$
            $Q_{i+1} := Res(L(Q_i))$
            $i := i + 1$
        enddo
        $\Pi(R) := Q_i$
    endif
    return $\Pi(R)$
end
```

The method of consensus-subsumption written as an algorithm is, in fact, not an algorithm. This is because, the 'algorithm' may not terminate for some inputs. This goes along well with the undecidability of first order logic and the best we may hope is a partial correctness of the algorithm. See Example 4.1 below concerning transitivity axiom, for illustration.

Example 4.1 Let $X = \{\neg Pxy \vee \neg Pyz \vee Pxz,\ \neg Pst \vee \neg Ptu \vee \neg Puw \vee Psw\}$.
$L(X) = \{\neg Pxy \vee \neg Pyz \vee Pxz,\ \neg Pst \vee \neg Ptu \vee \neg Puw \vee Psw,$
$\neg Pwz \vee Psz \vee \neg Pst \vee \neg Ptu \vee \neg Puw\}$.
Since in $L(X)$, none of the clauses subsumes any other, $Res(L(X)) = L(X)$. Similarly,
$L^2(X) = \{\neg Pxy \vee \neg Pyz \vee Pxz,\ \neg Pst \vee \neg Ptu \vee \neg Puw \vee Psw,\ \neg Pwz \vee Psz \vee \neg Pst$
$\vee \neg Ptu \vee \neg Puw,\ \neg Pwy \vee Psz \vee \neg Pst \vee \neg Ptu \vee \neg Puw,$
$\neg Pzu \vee \neg Puw \vee Psw \vee \neg Pwz \vee \neg Psz \vee \neg Pzu \vee \neg Puw\}$.
Since none of the clauses in $L^2(X)$ subsumes any other clauses in it, $Res(L^2(X)) = L^2(X)$ and this property is satisfied for each $L^n(X)$, $n \in \mathbb{N}$. Moreover, $L^{n+1}(X)$ has more clauses than $L^n(X)$. Therefore, $\overline{L}(X)$ is infinite. Hence the algorithm does not terminate in this case.

Theorem 4.1 If the algorithm $CONSUM$ terminates, then it correctly computes the set of prime implicates of an SCNF formula.

Proof Let R be the given set of clauses. If any clause of R contains complementary literals then the clause is valid and hence it is discarded as the truth value does not change. The remaining set contains all and only fundamental clauses. If this set is empty then obviously, there are no prime implicates. This justifies the first block of the algorithm. Suppose the number of clauses in the updated clause set is nonzero (but finite). Call this set Q_1. The algorithm computes $L(Q_1)$ and then applies subsumption on $L(Q_1)$ to obtain the residue as Q_2. By Theorem 3.7, $\pi(R) \subseteq \overline{L}(Q_1)$. However if a clause in $L(Q_1)$ is subsumed by another clause in $L(Q_1)$, then neither this clause nor any further consensus of this (taken with others) is a prime implicate. Hence $\Pi(R) \subseteq \overline{Q}_2$ as $Q_2 = Res(L(Q_1)) \subseteq \overline{L}(Q_1)$. Thus we see that for each $i \geq 2$, $\Pi(R) \subseteq \overline{Q}_i \subseteq \overline{L}(Q_{i-1}) \subseteq \overline{L}(Q_1)$. Then $\Pi(R) \subseteq Res(\overline{Q}_i) \subseteq Res(\overline{L}(Q_i)) \subseteq L(\overline{Q}_i)$. When the algorithm terminates, there is $m \in \mathbb{N}$ such that $\overline{L}(Q_m) = \overline{L}(Q_{m+1})$. Hence $Res(\overline{L}(Q_m)) = Res(\overline{L}(Q_{m+1}))$. Together they yield, $Res(\overline{L}(Q_m)) = Res(\overline{L}(Q_1))$. By Theorem 3.7, $\Pi(X) = Res(L(Q_m))$. Since the algorithm computes $Res(\overline{L}(Q_m))$, the proof is complete. □

Note that instead of taking $Res(\overline{L}(R))$, the algorithm uses subsumption test at each step of taking consensus closure. This is efficient compared to generating unnecessary consensus in $\overline{L}(R)$ and then taking subsumption at the end.

Example 4.2 Let $X = \{Pxa \vee Ryf(x), \neg Rbz, \neg Pbz \vee Qz\}$.
Now $Q_0 := \phi$, $Q_1 := X$, $Q_0 \neq Q_1$.
The literals $Ryf(x)$ in $\{Pxa \vee Ryf(x)\}$ and $\neg Rbz$ in $\{\neg Rbz\}$ form a pair of complementary literals with respect to the substitution $[y/b, z/f(x)]$. So, the consensus of $\{Pxa \vee Ryf(x)\}$ and $\{\neg Rbz\}$ with respect to the substitution $[y/b, z/f(x)]$ is $\{Pxa\}$. Similarly, the consensus between $\{Pxa \vee Ryf(x)\}$ and $\{\neg Pbz \vee Qz\}$ with respect to the substitution $[x/b, z/a]$ is $\{Ryf(b), Qa\}$. The substitutions are the most general ones. This two clauses are added to Q_1 to get $L(Q_1)$.

$L(Q_1) = \{\{Pxa, Ryf(x)\}, \{\neg Rbz\}, \{\neg Pbz, Qz\}, \{Pxa\}, \{Ryf(b), Qa\}\}$.

8

In $L(Q_1)$, as $\{Pxa\}$ subsumes $\{Pxa, Ryf(x)\}$ with respect to the empty substitution ϵ, we have the residue as $Q_2 = Res(L(Q_1)) = \{\{Pxa\}, \{\neg Rbz\}, \{\neg Pbz, Qz\}, \{Ryf(b), Qa\}\}$. As $\{\neg Rbz\}$ is associated with $\sigma_1 = \epsilon$ and $\{Ryf(b), Qa\}$ is associated with $\sigma_2 = [x/b, z/a]$ the consensus between them with respect to $\sigma = [y/b, z/f(b)]$ cannot be computed because $\sigma_1\sigma \neq \sigma_2\sigma$. Similarly the consensus between $\{Pxa\}$ and $\{\neg Pbz, Qz\}$ cannot be computed with respect to $[x/b, z/a]$. Now, consensus cannot be taken any more in Q_2 and no subsumption takes place too. $Q_2 = L(Q_2) = Q_3$. So, Q_2 is the desired set of prime implicates; $\Pi(X) = Q_2$.

Example 4.3 Let $X = Q_1 = \{\{Qy\}, \{\neg Rf(x)b\}, \{Px, Ryb, \neg Qz\}\}$.
Now, $\phi = Q_0 \neq Q_1 = X$. As $\{Qy\}$ and $\{Px, Ryb, \neg Qz\}$ contains a pair of complementary literals, the consensus between the two clauses with respect to the substitution $[y/z]$ is $\{Px, Rzb\}$. Similarly the consensus between $\{\neg Rf(x)b\}$ and $\{Px, Ryb, \neg Qz\}$ with respect to a substitution $[y/f(x)]$ is $\{Px, \neg Qz\}$. The two derived clauses are added to Q_1 to get $L(Q_1)$.

$$L(Q_1) = \{\{Qy\}, \{\neg Rf(x)b\}, \{Px, Ryb, \neg Qz\}, \{Px, Rzb\}, \{Px, \neg Qz\}\}.$$

Since $\{Px, Rzb\}$ subsumes $\{Px, Ryb, \neg Qz\}$ in $L(Q_1)$ with the substitution $\sigma = [z/y]$, the clause $\{Px, Ryb, \neg Qz\}$ is discarded from $L(Q_1)$ to obtain Q_2.

$$Q_2 = Res(L(Q_1)) = \{\{Qy\}, \{\neg Rf(x)b\}, \{Px, Rzb\}, \{Px, \neg Qz\}\} \neq Q_1.$$

In Q_2, the consensus between $\{Qy\}$ associated with ϵ and $\{Px, \neg Qz\}$ associated with $[y/f(x)]$ with respect to a substitution $[y/z]$ is not possible as the composition of substitution is not defined. With respect to the substitution $[z/f(x), y/f(x)]$ consensus cannot be taken into consideration as $[z/f(x), y/f(x)]$ is a unifier but not a most general unifier. Moreover, the consensus between $\{\neg Rf(x)b\}$ and $\{Px, Rzb\}$ cannot be computed due to undefined composition of substitution. No consensus takes place between clauses of Q_2. Further, no clause in Q_2 subsumes any other clause. Therefore, $Q_2 = L(Q_2) = Res(L(Q_2))$. Q_2 is the set of prime implicates; $\Pi(X) = Q_2$.

Example 4.4 Consider the formula in Example 3.1. Since none of the clauses of $L^1(X)$ are subsumed by others and $L^1(X)$ was the consensus closure of X, $L^1(X) = Res(L^1(X))$. The set of prime implicates is given by

$$\begin{aligned}\Pi(X) = L^1(X) = \{&\{Px, \neg Qg(a)\}, \{Rby, Qy\}, \{\neg Pg(w), \neg Rxf(a)\}, \{Px, Rbg(a)\},\\ &\{\neg Pg(w), Qf(a)\}, \{\neg Qg(a), \neg Rg(w)f(a)\}\}.\end{aligned}$$

5. Conclusion

We note that instead of SCNF, validity invariant skolemized formulas for knowledge representation will not be of much help as the underlying theory is not decidable. Also, since reasoning processes use unsatisfiability, the extension carried out here is appropriate. Note that consensus corresponds to binary resolvent and subsumption relates to the factor rule in the resolution procedure for first order logic without equality. Since binary resolution and factor rule together form a complete deduction mechanism for first order logic, the method of consensus-subsumption succeeds in computing prime implicates provided it terminates. For query answering, the factor rule is left for on-line reasoning. It requires further heuristic based on the problem domains so that appropriate substitutions might be chosen to make on-line reasoning efficient. We also remark that further heuristic may be required for the cases of non-termination of the method.

It is also easy to see that other consensus based methods such as Quine-McClusky algorithm [1, 13, 19] and etc. may also be adopted. It will also be interesting if transversal clauses method [16] can be extended to first order logic formulas. We note that the algorithm *CONSUM* computes efficiently the set of prime implicates by taking residue of subsumption at each step of the iteration. However, this is not final, as it still generates some similar clauses in subsequent iterations repeatedly. It is of further interest how to curtail taking wasteful consensus. We must also note that the knowledge compilation technique discussed here is applicable only for those expressed in SCNF. It does not, for example, include the Prolog knowledge bases. Attempts must be made for more general types of first order knowledge bases, since this technique is comparatively more exact than the approximate knowledge compilation as discussed in [5].

References

[1] Biswas, N. N. (1975), *Introduction to Logic and Switching Theory*, Gordon & Breach Science Pub.

[2] Bylander, T. (1991), A simple model of knowledge compilation, *IEEE Expert*, **6(2)**.

[3] Cadoli, M. and F.M. Donini, F. M. (1998), A survey on knowledge compilation. *AI Communications-The European Journal for Articial Intelligence*, **10**, 137–150.

[4] Castell, T. and Cayrol, M. (1996), Computation of prime implicates and prime implicants by the Davis-Putnam procedure, *Workshop on "Advances on Propositional Deduction"*, **ECAI-96**, 61–64.

[5] del Val, A., (1996), Approximate Knowledge Compilation: The First Order Case, AAAI'96, In *Proceedings of AAAI-96*, Portland, Oregon, 498-503.

[6] Kean, A., Tsiknis, G. (1990), An Incremental Method for Generating Prime Implicants / Implicates, *J. of Symbolic Computation*. **9**, 185-206.

[7] Kean, A. and Tsiknis, G. (1992), Assumption based reasoning and clause management systems. *Computational Intelligence* **8**,1 , 1–24.

[8] de Kleer J. (1986), An Assumption Based TMS, *Artificial Intelligence*, **28**, 127-162.

[9] de Kleer, J. (1992), An improved incremental algorithm for computing prime implicants. In *Proceedings of AAAI-92*, San Jose, CA, 780–785.

[10] Marquis, P. (1995), Knowledge compilation using theory prime implicates. In *Proceedings of IJCAI*, 837–843.

[11] Marquis, P. and Sadaoui, S. (1996), A new algorithm for computing theory prime implicates computation, In *Proceedings of AAAI-96*, 504–509.

[12] Ngair, T. (1993), A new algorithm for incremental prime implicate generation. In *Proceedings of IJCAI-93*, Chambery, France.

[13] Quine, W. V. (1959), On Cores and Prime Implicates of Truth Functions, *Amer. Math. Monthly*, **66**, 755-760.

[14] Reiter, R., de Kleer, J. (1987), Foundations of Assumption-Based Truth Maintenance System(ATMS): Preliminary Report, In *Proceedings of AAAI-87*, 183-188.

[15] Shoenfield, J. R. (1967), *Mathematical Logic*, Addition-Wisley, Reading, MA.

[16] Singh,A. (1999), Computing Prime Implicants Via Transversal Clauses, *Int. J. Computer Math.*, **70**, 417-427.

[17] Slagle, J. R., Chang C. L., Lee, R. C. T. (1970), A New algorithm for Generating Prime Implicants, *IEEE trans. on Comp.*, **C-19** (4), 304-310.

[18] Strzemecki, T. (1992), Polynomial-time algorithm for generation of prime implicants. *Journal of Complexity* **8** , 37-63.

[19] Tison, P. (1967), Generalized Consensus Theory and Application to the Minimisation of Boolean Functions, *IEEE Trans. on Elec. Comp*, **EC-16** (4), 446-456.