SAT Techniques and Knowledge Base Verification

OLGA TVERETINA, HANS ZANTEMA
Department of computer science
Technical University of Eindhoven
P.O. Box 513, 5600 MB Eindhoven
THE NETHERLANDS

Abstrract:- For many application domains the best way to produce intelligent behavior is to encode knowledge about this domain to knowledge base. Verification of KB is one of the crucial issues in developing reliable knowledge-based systems. SAT techniques as resolution and DPLL can be used for verifying knowledge bases. We give a transformation of a DPLL refutation to a resolution refutation of a number of steps which is essentially less than the number of unit resolution steps applied in the DPLL refutation.

Key-words:- Artificial intelligence, knowledge base, verification, propositional logic, satisfiability, DPLL, resolution.

1 Introduction

Last years there has been an explosion of research in artificial intelligence (AI) into propositional satisfiability (SAT), because we can often solve real world problems by encoding them into SAT. SAT problem is to determine for a given propositional formula whether there exists a satisfying assignment, i.e., whether it is not equivalent to false.

AI focuses on developing intelligent software and hardware systems. In many application domains the best way to produce intelligent behavior of such a system is encoding knowledge about the domain into a Knowledge Base (KB). The KB needs to have three major properties: soundness, confidence that a conclusion is true; completeness, the system has the knowledge to be able to reach a conclusion; and tractability, it is realistic that a conclusion can be reached.

Verification of KB is one of the crucial issues in developing reliable knowledge-based systems. KB is a set of if-then rules. The problem of verifying of some KBs can be naturally transformed to SAT.

Propositional resolution proof system is a very important one in AI, and it constitutes an essential part of many automated theorem provers.

Resolution proof systems [4] is a refutation system. It means that we are refuting initial set of clauses by deriving a contradiction represented as the empty clause. In this proof system we deal with formulas in conjunctive normal form (CNFs). Many natural statements either can be represented as CNFs or transformed to CNFs.

To prove that we can conclude a formula V from a set of axioms A we have to negate V and add it to A and convert all formulas to CNF.

A related method is the DPLL procedure [1]. It consists of a combination of unit resolution and doing case analysis upon p and $\neg p$ and going on recursively.

The paper is organized as follows. In section 2 and section 3 we give some definitions, preliminary lemmas and the basic theorem. In section 4 we give the definition of a DPLL tree such that the nodes correspond to recursive calls in the DPLL procedure. Section 5 contains the main results about the length of the resolution refutation which are presented in two theorems. In section 6 we show that the upper bound is tight. Section 7 contains some conclusions.

2 Knowledge representation and inference engine in KBs

There are three main parts of knowledge-based system: a KB, a database of facts, an inference engine. A popular approach for knowledge representation in a KB is to use IF-THEN rules.

At the logical level rule IF A THEN B can be represented like $A \to B$.

An inference engine is a system that allows to derive new rules and facts from currently-known rules and facts.

One of the inference engines is resolution. To prove with resolution that P can be derived from a KB we have to convert $\neg P$ and rules in KB to CNF and to apply a resolution rule.

3 Basic definitions and preliminaries

This section contains notations and definitions used in the paper.

Let A be a set of atoms. The symbols p, q and r are reserved to denote atoms.

A literal is an atom or a negated atom, as $\neg p$. The typical elements of the literal set L are denoted as l and k.

A *clause* is a finite, possibly empty, set of literals. The letters C and D are reserved to denote clauses. The number of literals in the clause is the *length* of this clause. Clauses of length one are called *unit clauses*.The empty clause is denoted by $\perp$.

A *CNF* is a finite, possibly empty, set of clauses. The letters V and W are used to denote CNFs.

Suppose V, W are CNFs; $C \cup \{p\}$, $D \cup \{\neg p\} \in V$, where $p \in A$. Then the transition from V to W, where $W = V \cup \{C \cup D\}$ is called a resolution step and denoted as $V \overset{CDp}{\longmapsto} W$. C, D, p can be omitted in the context where they are not relevant.

If $V_0 \longmapsto ... \longmapsto V_n$ then $V_0, ..., V_n$ is called a resolution sequence of $length\ n$. If $V_0 \overset{l_1}{\underset{u}{\mapsto}} ... \overset{l_n}{\underset{u}{\mapsto}} V_n$ then $V_0, ..., V_n$ is called a unit resolution sequence.

Suppose V_0 is a CNF and C is a clause. We say that C is *derived* from V_0 in $n > 0$ resolution steps if there is a resolution sequence $V_0, ..., V_n$ such that $V_n = V_{n-1} \cup C$. And we say that C is derived from V_0 in 0 resolution steps if $C \in V_0$.

An assignment is a mapping that assigns *false* or *true* to atoms. If the atom p is assigned to true then $\neg p$ is assigned to false and vice versa. An assignment *satisfies* a clause if it maps at least one of its literals to true. An assignment satisfies a CNF V if and only if it *satisfies* each of its clauses, and V is called *satisfiable*. If there is no assignment that maps a CNF V to true then V is called *unsatisfiable*.

A resolution sequence $V_0, ..., V_n$ such that $\perp \in V_n$ is called a *resolution refutation* and n is called a *length of the resolution refutation*.

In the following we use the well-known fact that the *CNF V* is unsatisfiable if there is a resolution refutation starting from V.

In our paper some proofs are omitted, they can be found in [6].

Lemma 1 *If $\perp$ can be derived in m resolution steps from $V \cup \{\{l\}\}$ then either $\perp$ can be derived from V in at most m resolution steps or $\neg l$ can be derived from V in at most $m-1$ resolution steps.*

The following theorem follows from Lemma 1.

Theorem 2 *If $\perp$ can be derived from $V \cup \{\{l\}\}$ in $m > 0$ resolution steps, and $\perp$ can be derived from $V \cup \{\{\neg l\}\}$ in $n > 0$ resolution steps then $\perp$ can be derived from V in at most $m + n - 1$ resolution steps.*

Proof. If $\perp$ can be derived from $V \cup \{\{l\}\}$ in $m > 0$ resolution steps then by Lemma 1 one of the following holds

1. $\perp$ can be derived from V in at most m resolution steps.
2. $\neg l$ can be derived from V in at most $m-1$ resolution steps.

   If $\perp$ can be derived from $V \cup \{\{\neg l\}\}$ in $n > 0$ resolution steps then by Lemma 1 one of the following holds

3. $\perp$ can be derived from V in at most n resolution steps.
4. l can be derived from V in at most $m-1$ resolution steps.

In cases 1 or 3 $\bot$ can be derived from V in $\min(m,n)$ resolution steps. In case of combination of cases 2 and 4 we need extra resolution step to get $\bot$. And it can be derived in $m-1+n-1+1=m+n-1$ resolution steps. □

This theorem is basic for result described in the following.

4 A DPLL procedure

Given a CNF V and a literal l. Let $V|l$ denotes the formula obtained from V by removing all the clauses that contain l and deleting all $\neg l$ from all clauses that contain $\neg l$.

A literal l in the CNF V is called monotone if $\neg l$ does not appear in V.

The procedure of deleting monotone literals from V is denoted as $mon\_lit(V)$.

A *DPLL tree* T on V is a binary tree , where every node is labelled with a unit resolution sequence. The root is labelled with a unit resolution sequence starting from V. If a node is labelled with the unit resolution sequence $V_1, ..., V_n$ then the left child is labelled with a unit resolution sequence starting from $mon\_lit(V_n|p)$ and the right child is labelled with a unit resolution sequence starting from $mon\_lit(V_n|\neg p)$. A node is a leaf if either $V_n = \emptyset$ or $\bot \in V_n$.

A DPLL tree is nothing else as a static representation of the recursive calls in the executing of the usual DPLL procedure.

If T is a DPLL tree on V then V is satisfiable if and only if there exists a leaf in T labelled with a unit resolution sequence $V_1, ..., V_n$ such that $V_n = \emptyset$.

Hence building a DPLL tree implies decision procedure for satisfiability, therefore we will speak about DPLL proof rather than DPLL tree.

A DPLL tree on unsatisfiable formula is called a DPLL refutation.

Suppose a node is labelled with $mon\_lit(V|l) \longmapsto V_1 \longmapsto ... \longmapsto V_n$. Then the number of unit resolution steps corresponding to the node is defined to include the length of the unit resolution sequence n and the number of $\neg l$ in V.

The total number of unit resolution steps for the DPLL tree is called the length of the DPLL proof.

5 Upper bounds on resolution refutation length

In this section we give two upper bounds on resolution refutation length measured in the length of the DPLL refutation and the number of its nodes. The first one is a direct analysis of a DPLL refutation. The second one has an extra restriction on a resolution sequence used in the first result. The second bound is stronger, we even show that it will be tight.

Theorem 3 *Suppose V is an unsatisfiable CNF; a DPLL refutation on V has length s and the number of its nodes is r. Then there exists a resolution refutation on V of length less or equal $s-(r-1)/2$.*

Proof. Induction on r.

Base case. Let $r=1$. Then $s-(1-1)/2 = s$. The lemma holds.

Inductive step. Assume that the Lemma holds for $r-2$. By induction hypothesis the lemma holds for the subtrees rooted at children nodes of the root.

Let one subtree have a DPLL refutation of length s_1 and the number of its nodes be r_1. Let another subtree have a DPLL refutation of length s_2 and the number of its nodes be r_2. And s_0 be a number of unit resolution steps corresponding to the root.

Then by Theorem 2 the length of a resolution refutation on V is $s_0 + ((s_1-(r_1-1)/2) + (s_2-(r_2-1)/2)-1) = s-(r-1)/2$, where $s = s_0+s_1+s_2$, $r = r_1+r_2+1$. □

We can improve the upper bound by making a restriction on the unit resolution sequences associated with nodes of the DPLL refutation.

We say that a unit resolution sequence is *complete* if no unit resolution steps can be applied at the last CNF of the sequence.

For the proof of the theorem we use the following lemmas.

Lemma 4 *Let V be a CNF such that it contains no monotone literals, and no unit resolution can be applied. Then V contains no unit clauses.*

Proof. By contradiction.

Let $V = \{\{l\}\} \cup V'$, where V' is a CNF. By the lemma assumption V contains no monotone literals so $\exists C \in V' : \neg l \in C$. And at least one unit resolution step can be applied. We have a contradiction. And V contains no unit clauses. $\square$

Now we arrive at the crucial observation mentioned in the introduction.

Lemma 5 *If $\perp$ can be derived from $V \cup \{\{l\}\}$ in $n \geq 2$ unit resolution steps, and V contains no unit clauses then either $\neg l$ or $\perp$ can be derived from V in at most $n-2$ resolution steps.*

Proof. Induction on n.

Base case. $n = 2$.

As V does not contain monotone literals so $\perp$ cannot be derived from V in two unit resolution steps . And the lemma holds for $n = 2$.

Inductive step. Let the lemma hold for $n-1$.

Suppose $\perp$ can be derived from $V \cup \{\{l\}\}$ in $n \geq 2$ unit resolution steps.

If the unit resolution sequence contains more than one l-step then the lemma holds.

As V contains no unit clauses the first unit resolution step is an l-step. So the remaining case if this first step is the only l-step.

Then $V \cup \{\{l\}\} \overset{l}{\underset{u}{\longmapsto}} V \cup \{\{l\}\} \cup \{\{l'\}\}$. And $\perp$ can be derived from $V \cup \{\{l\}\} \underset{u}{\longmapsto}$ V$\cup\{\{l\}\} \cup \{\{l'\}\}$ in $n-1$ unit resolution step.

As the resolution sequence contains only one l-step then $\perp$ can be derived from $V \cup \{\{l'\}\}$ in $n-1$ unit resolution step.

By induction hypothesis either $\neg l'$ or $\perp$ can be derived from $V \cup \{\{l'\}\}$ in no more than $n-3$ resolution steps. As $\{\neg l, l'\} \in V$ then we need one extra resolution step in case if we derived $\neg l'$. And either $\neg l$ or $\perp$ can be derived from V in $n-2$ resolution steps. $\square$

Just like Theorem 2 was needed to prove Theorem 3 we now state Theorem 6 to use it for proving Theorem 7.

Theorem 6 *Suppose V contains no unit clauses. If $\perp$ can be derived from $V \cup \{\{l\}\}$ in $m > 2$ unit resolution steps, and $\perp$ can be derived from $V \cup \{\{\neg l\}\}$ in $n > 2$ unit resolution steps then $\perp$ can be derived from V in $m+n-3$ resolution steps.*

Proof. If $\perp$ can be derived from $V \cup \{\{l\}\}$ in $m > 2$ resolution steps then by Lemma 5 one of the following holds

1. $\perp$ can be derived from V in at most $m-2$ resolution steps.
2. $\neg l$ can be derived from V in at most $m-2$ resolution steps.

   If $\perp$ can be derived from $V \cup \{\{\neg l\}\}$ in $n > 2$ resolution steps then by Lemma 5 one of the following holds
3. $\perp$ can be derived from V in at most $n-2$ resolution steps.
4. l can be derived from V in at most $m-2$ resolution steps.

In cases 1 or 3 $\perp$ can be derived from V in $\min(m-2, n-2)$ resolution steps. For $m > 2$ and $n > 2$ $\min(m-2, n-2) \leq m+n-3$. In case of combination of cases 2 and 4 we need extra resolution step to get $\perp$. And it can be derived in $m-2+n-2+1 = m+n-3$ resolution steps. $\square$

Theorem 7 *Suppose V is an unsatisfiable CNF; a DPLL refutation on V has length s, the number of its nodes is $r \geq 3$ and every unit resolution sequence associated with a node is complete. Then there exists a resolution refutation on V of length less or equal $s-r$.*

Proof. Induction on r.

Base case. Let $r = 3$. Then the Lemma holds by Lemma 4 and Theorem 6.

Inductive step. Assume that the Lemma holds for $r-2$. By induction hypothesis the lemma holds for the subtrees rooted at children nodes of the root.

Let one subtree have a DPLL refutation of length s_1 and the number of its nodes be r_1. Let another subtree have a DPLL refutation of length s_2 and the number of its nodes be r_2. And s_0 be a number of unit resolution steps corresponding to the root.

Then by Theorem 2 the length of a resolution refutation on V is $s_0 + ((s_1 - r_1) + (s_2 - r_2) - 1) = s - r$, where $s = s_0 + s_1 + s_2$, $r = r_1 + r_2 + 1$. $\square$

6 Tightness of the upper bound

A CNF is *minimally unsatisfiable* if it is unsatisfiable and each of its subsets is satisfiable.

Let $V_0, ..., V_n$ be a resolution sequence, where $V_n = \{C_1, ..., C_m\}$. Then a directed graph $G(V, E)$, where V is a set of vertices and E is a set of edges is called a *resolution graph* if $V = \{C_1, ..., C_m\}$ and $E = \{(C_i, C_j) : \exists r \in \{0, ..., n-1\}$, $l \in L$, $C_k \in V_r$ such that $V_{r+1} = V_r \cup C_j$, where $C_j = (C_i \cup C_k) \backslash \{l, \neg l\}\}$.

Lemma 8 *Suppose minimally unsatisfiable CNF V contains n clauses. Then there is no resolution refutation on V with length less than $n - 1$.*

We introduce some class of CNFs to prove that the upper bound presented by Theorem 7 is tight.

Let $n \geq 1$. We define $V_n = \{\{\neg p_1, q_1\}, \{\neg p_2, \neg q_1\}, q_2\}, ..., \{\neg p_n, \neg q_{n-1}, q_n\}, \{\neg q_n, q_{n+1}\}, \{\neg q_n, q_{n+2}\}, \{\neg q_{n+1}, \neg q_{n+2}\}, \{p_1, r_{11}\}, \{\neg r_{11}, r_{21}\}, \{\neg r_{11}, r_{31}\}, \{\neg r_{31}, \neg r_{21}\}, ..., \{p_n, r_{1n}\}, \{\neg r_{1n}, r_{2n}\}, \{\neg r_{1n}, r_{3n}\}, \{\neg r_{2n}, \neg r_{3n}\}\}$.

Lemma 9 *For $n \geq 1$ V_n is minimally unsatisfiable.*

Lemma 10 *For $n \geq 1$ V_n has a DPLL refutation of length $7n + 3$ and the number of its nodes is $2n + 1$.*

Now we are ready to prove tightness of our main result.

Theorem 11 *For $n \geq 1$ V_n has a DPLL refutation such that it has length s, the number of its nodes is r and there is no resolution refutation on V_n of length less than $s - r$.*

Proof. By Lemma 10 $\forall n \geq 1$ V_n has a DPLL refutation of length $7n + 3$ and the number of its nodes is $2n + 1$. By Theorem 7 there exists a resolution refutation on V of length less or equal $5n + 2$.

For $n \geq 1$ V_n has $5n + 3$ clauses. By Lemma 8 and Lemma 9 there is no resolution refutation on V of length less than $5n + 2$. $\square$

7 Conclusions

In some application domains real time knowledge-based systems are used. New rules can be added to a KB during the work of such a system. And fast verification of the KB is a very crucial issue. Our approach can be implemented for verifying of a propositional part of a KB. From our experiments we found out that for many formulas, including pigeon hole formulas, the constructed resolution refutation had a length that was much less than $s - r$.

References:

[1] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem proving. C. ACM 5 (1962). 394-397.

[2] M. Davis, and H. Putnam. A computing procedure for quantification theory. J. ACM 7 (1960). 201-215.

[3] D.W. Loveland. Automated theorem proving: a logic basis. North-Holland, Amsterdam, 1978.

[4] J.A. Robinson. A machine oriented logic based on the resolution principle. J. ACM 12 (1965). 23-41.

[5] J. P. Marques Silva. An overview of backtrack search satisfiability algorithms. Fifth International Symposium on Artificial Intelligence and Mathematics, January 1998.

[6] O.Tveretina, H. Zantema, Transforming DPLL to Resolution. Technical report, Technical University of Eindhoven, http://www.win.tue.nl/ hzantema/other.html.

[7] N. Yugami. Theoretical analysis of Davis-Putnam procedure and propositional satisfiability. In Proceedings of IJCAI-95. 282-288, 1995.