Determinization of resolution by an algorithm operating on complete assignments

Eugene Goldberg

Cadence Berkeley Labs, USA, egold@cadence.com

Abstract. "Determinization" of resolution is usually done by a DPLL-like procedure that operates on partial assignments. We introduce a resolution-based SAT-solver operating on complete assignments and give a theoretical justification for determinizing resolution this way. This justification is based on the notion of a point image of resolution proof. We give experimental results confirming the viability of our approach. The complete version of this paper is given in [2].

1 Introduction

The resolution proof system has achieved an outstanding popularity in practical applications. Since resolution is a non-deterministic proof system, any SAT-solver based on resolution, one way or another, has to perform its "determinization". In the state-of-the-art SAT-solvers this determinization is based on using the DPLL procedure [1] that operates on partial assignments. The current partial assignment is extended until a clause is falsified. Then, the DPLL procedure backtracks to the last decision assignment and flips it. The search performed by the DPLL procedure can be simulated by so-called tree-like resolution (a special type of general resolution).

The reason for using partial rather than complete assignments is that by rejecting a partial assignment the DPLL procedure may "simultaneously" reject an exponential number of complete assignments. The premise of such an approach is that to prove that a CNF formula F is unsatisfiable one has to show that F evaluates to 0 for all complete assignments.

In this paper, we introduce the notion of a point image of a resolution proof that questions the premise above. A point image of a resolution proof can be viewed as an "encryption" of this resolution proof. Given a resolution proof R, one can always build its point image whose size is at most twice the size of R (measured in the number of resolution operations). On the other hand, given a set of points T and a CNF formula F one can use a simple procedure to test if T is a point image of a resolution proof. If it is, this procedure builds a resolution proof "specified" by T. This result implies that a resolution proof that F is unsatisfiable can be "guided" by testing the value of F in a sequence of points. Moreover, if a CNF formula F has a short resolution proof, the number of "guiding" points is negligible with respect to the size of the entire search space.

We introduce a SAT-solver operating on complete assignments that is inspired by the notion of a point image of resolution proof. The complete version of this paper is given in [2].

2 Main definitions

Let F be a CNF formula over a set X of Boolean variables. The satisfiability problem (SAT) is to find a complete assignment $\boldsymbol{p}$ (called a **satisfying assignment**) to the variables of X such that $F(\boldsymbol{p}) = 1$ or to prove that such an assignment does not exist. If F has a satisfying assignment, F is called **satisfiable**. Otherwise, F is **unsatisfiable**. A disjunction of literals is further referred to as a **clause**. A complete assignment to variables of X will be also called a **point** of the Boolean space $B^{|X|}$ where B={0,1}. A point $\boldsymbol{p}$ **satisfies** clause C if $C(\boldsymbol{p})$=1. If $C(\boldsymbol{p})$=0, $\boldsymbol{p}$ is said to **falsify** C.

Let C_1 and C_2 be two clauses that have opposite literals of a variable x_i. Then the clause consisting of all the literals of C_1,C_2 except those of x_i is called the **resolvent** of C_1,C_2. The resolvent of C_1,C_2 is said to be obtained by the **resolution operation**. Given an unsatisfiable CNF formula F, one can always generate a sequence of resolution operations resulting in an empty clause. This sequence of operations is called a **resolution proof**. The resolution proof system is very important from a practical of view because many successful SAT-algorithms for solving "industrial" CNF formulas are based on resolution.

3 Justification of our approach

In this section, we give a theoretical justification of our approach.

Let R be a resolution proof that a CNF formula F is unsatisfiable. Let T be a set of points that has the following property. For any resolvent C of R, obtained from parent clauses C' and C'' there are two points $\boldsymbol{p'}$ and $\boldsymbol{p''}$ of T such that

1. $C'(\boldsymbol{p'}) = 0$ and $C''(\boldsymbol{p''}) = 0$
2. Points $\boldsymbol{p'}$ and $\boldsymbol{p''}$ are different only in the variable in which clauses C' and C'' are resolved.

Then the set T is called a **point image of resolution proof $\boldsymbol{R}$**. The points $\boldsymbol{p'}$ and $\boldsymbol{p''}$ are called **a point image of the resolution operation** over clauses C' and C''.

Building a point image of a resolution proof. Given a resolution proof R that F is unsatisfiable, a point image T of R can be built as follows. We start with an empty set T. Then for every resolution operation from R over clauses C' and C'' we add to T two points $\boldsymbol{p'}$ and $\boldsymbol{p''}$ forming a point image of this operation (unless $\boldsymbol{p'}$ and/or $\boldsymbol{p''}$ have been added to T before). Clearly, the size of a set T built this way is at most twice the size of R.

Checking if a set of points is a point image. Given a set of points T and a CNF formula F, one can test if T is a point image of a resolution proof

by the following procedure. Let S be a set of clauses that initially consists of the clauses of F. At each step of the procedure, we pick a pair of clauses C' and C'' of S such that a point image of the resolution operation over C' and C'' is in T and add the resolvent to S unless it is subsumed by a clause of S. This procedure has three termination conditions. 1) If a point of T satisfies F, then clearly F is satisfiable and T is not a point image. 2) No new clause can be added to S at a step of the procedure. This means that T is not a point image of a resolution proof (because T does not have "enough" points) and so one can not say yet whether F is satisfiable or not. 3) An empty clause is derived at a step of the procedure. This means that T is a point image of a resolution proof that F is unsatisfiable.

The procedure above implies that one can use complete assignments to "guide" a resolution proof. The size of the "guiding" set T is at most twice as large as the size of the proof R the set T "guides".

A proof has a huge number of point images. Let *Res* be the resolution operation over clauses C' and C'' used in a proof that CNF formula F is unsatisfiable. The operation *Res*, in general, has a huge number of point images because points $\boldsymbol{p'}$ and $\boldsymbol{p''}$ forming a point image of *Res* are specified only for the variables of C' and C''. For the variables of F that are not in C' and C'', points $\boldsymbol{p'}$ and $\boldsymbol{p''}$ may have arbitrary (but identical) values (because, by definition, $\boldsymbol{p'}$ and $\boldsymbol{p''}$ are different only in the variable in which C' and C'' are resolved).

Since a point image of a resolution proof R is essentially the union of point images of resolution operations comprising R, the latter has a huge number of point images. However, not all point images of R are equivalent in the sense that some images are more regular and so can be more easily built by a deterministic algorithm. Resolution, being a non-deterministic proof system, does not distinguish between different point images of R. On the other hand, the fact that the "space" of images of R is very rich (and that some images are easier to find than others) implies that an algorithm operating on complete assignments can be used for finding resolution proofs.

4 Algorithm operating on complete assignments

In this section, we introduce a resolution-based SAT-solver called ***FI*** (Find Image). Although *FI* is inspired by the ideas of Section 3 it does not look for a point image of a resolution proof "directly". Instead, *FI* implements a DPLL-like procedure that operates on complete assignments. Here, we give only a very high-level picture of *FI*. A detailed description can be found in [2]. Besides, [2] explains the relation between the set of points "visited" by *FI* and the proof *FI* builds.

***FI* operation**. Operationally, *FI* can be viewed as a regular resolution-based SAT-solver that uses a complete assignment $\boldsymbol{p}$ as an "oracle". An initial assignment $\boldsymbol{p}$ can be generated randomly or using some heuristic. Let F be the current CNF formula (consisting of initial and conflict clauses) and $\boldsymbol{M(p)}$ be the set of clauses of F falsfiied by $\boldsymbol{p}$. Then only a variable of a clause from $\boldsymbol{M(p)}$

can be assigned a value during decision making. If a variable x_i is assigned a value $b \in \{0,1\}$ (either during decision making or BCP) and the value of x_i in $\boldsymbol{p}$ is $\overline{b}$, then value of x_i in $\boldsymbol{p}$ is flipped to b. In other words, one can view FI as a regular SAT-solver in which the choice of variables for decision making is controlled by a complete assignment that dynamically changes.

Interpretation in terms of complete assignments. The interpretation of FI above is convenient for "historical" reasons. However, we believe that a much more fruitful interpretation of FI is as follows. FI operates on complete assignments and so at any step of FI, every variable of the formula is assigned. Instead of making an assignment to a free variable x_i as in DPLL , FI fixes the assignment of x_i in $\boldsymbol{p}$. This fixing means that in all the points $\boldsymbol{p}$ visited later the value of x_i stays the same until the time it is "unfixed". Unfixing x_i in FI corresponds to unassigning x_i in a DPLL-like procedure and making it free again. Only variables of clauses from the set $\boldsymbol{M(p)}$ can be fixed. FI either fixes the value of a variable x_i that agrees with current point $\boldsymbol{p}$ or it first changes the value of x_i in $\boldsymbol{p}$ and only then fixes it. In the first case, the set $\boldsymbol{M(p)}$ of falsified clauses stays the same, in the second case it has to be recomputed.

Note, that while a DPLL-like procedure can reproduce the decision-making of FI, the opposite is not true. Namely, the "overwhelming majority" of search trees that can be built by DPLL are out of reach for FI because at every step, FI is limited in the choice of variables that can be used for decision-making. (In a sense, this extra power of DPLL is due to the fact that DPLL is not an algorithm, but rather a proof system still containing a lot of non-determinism.)

In [2] we list reasons why FI should be interpreted in terms of complete assignments. Here we mention only one of them. The underlying semantics of a DPLL-like procedure operating on partial assignments is that it covers all the points of the search space. If one considers FI just as a regular DPLL-like procedure with a particular decision-making "heuristic", it is hard to explain why this "heuristic" works. As we mentioned above, the decisions of FI are extremely limited being controlled by a complete assignment, that is by $1/2^n$ of the search space. On the other hand, such an explanation can be easily done in terms of point images of resolution proofs.

5 Advantages of using complete assignments

In Section 3 we gave a very "abstract" justification of using complete assignments in a resolution based algorithm. In this section, we list more concrete arguments (that are, in a sense, consequences of this abstract justification) in favor of our approach. In [2] we substantiate our claims experimentally.

Identifying small unsatisfiable sub-formulas. Current SAT-solvers are often used for solving huge CNF formulas e.g. in bounded model checking. The fact that SAT-solvers can efficiently prove the unsatisfiability of a CNF formula F of, say, 1 million variables usually means that there is an unsatisfiable sub-formula of a relatively small size (like 10-20 thousand variables). So identifying an unsatisfiable sub-formula is of great practical importance. Let G be an un-

satisfiable sub-formula of F. The advantage of using complete assignments is that any complete assignment $\boldsymbol{p}$ falsifies at least one clause of G. So at least one clause of G is always present in $M(\boldsymbol{p})$ and so clauses from G are always on the "radar" of *FI*. On the other hand, a resolution based SAT-solver operating on partial assignments may spend a lot of time trying to satisfy clauses of F that are not in G.

Finding clauses that can be resolved. Let C be a clause that contains variable x_i and is falsified by the current point $\boldsymbol{p}$. Let C' be a new clause falsified by the point $\boldsymbol{p'}$ obtained from $\boldsymbol{p}$ by flipping the value of x_i. Then clauses C and C' can be resolved in x_i. So, *FI* takes into account the "resolution nature" of the underlying proof system.

Efficient decision making. When solving large formulas it is very important to have a decision making procedure that is fast and at the same time manages to avoid branching on "irrelevant" variables. Making "redundant"decision assignments increases time spent on decision-making and may lead to the increase of redundant assignments made by the BCP procedure. Since the size of $M(\boldsymbol{p})$ is much smaller than that of the formula, picking the next variable to be fixed can be done efficiently. On the other hand, as we mentioned above, new clauses that appear in the set $M(\boldsymbol{p'})$ (where $\boldsymbol{p'}$ is obtained from $\boldsymbol{p}$ by flipping the value of a variable) can be resolved with clauses of $M(\boldsymbol{p})$ satisfied by $\boldsymbol{p'}$. So, by reducing our choice of variables to those of the clauses falsified by the current point, one reduces the probability of making assignments to "irrelevant" variables.

Successful use of frequent restarts. *FI* employs restarts. Instead of generating a new initial point randomly, *FI* starts with the last point visited in the previous iteration. This makes the resolution proof more "coherent" and allows one to use more frequent restarts successfully.

6 Experimental Results

In this section, we give results of experiments with an implementation of *FI*. We compare *FI*'s results (in terms of the number of conflicts i.e. backtracks) with those of *Forklift* and *Minisat*. (Many more experimental results can be found in [2].) Experiments show that although *FI* is extremely limited in its decision-making, it is competitive in the number of conflicts with *Forklift* and *Minisat*.

7 Conclusions

We introduce a resolution based SAT-solver *FI* that operates on complete assignments (points). *FI* is inspired by the fact that a resolution proof can be specified by a small set of points. Experimental results show the viability of our approach. Determinization of resolution by operating on complete assignments seems to be a promising way to design resolution-based SAT-solvers. In [2], we

Table 1. Some experimental results

Name	# formulas	*Forklift* #conflicts	*Minisat* #conflicts (#aborted)	*FI* #conflicts
Dimacs formulas				
aim	72	3,303	3,587	**3,256**
bf	4	774	383	**379**
dubois	13	**3,062**	4,904	3,260
hanoi	2	**26,156**	65,428	223,040
hole	5	227,102	1,538,350	**56,884**
ii	41	6,505	4,088	**1,254**
jnh	49	2,151	2,096	**2,069**
par16	10	**42,934**	47,568	70,915
par8	10	304	162	**83**
pret	8	4,342	6,892	**2,942**
ssa	8	744	367	**348**
Velev's formulas				
vliw-sat.1.0	100	679,827	1,413,027	**527,416**
fvp-unsat.1.0	4	101,991	180,240	**92,333**
3pipe	4	**24,738**	66,567	33,856
4pipe	5	**125,850**	538,932	154,321
5pipe	6	268,463	1,261,229	**231,975**
6pipe	2	218,461	>470,779(1)	**176,067**
Some other known formulas				
Beijing	16	494,534	> 721,258(1)	**106,896**
blocksworld	7	**2,116**	4,732	8,209
bmc	13	54,098	**44,195**	48,568
bmc1	31	**1,033,434**	1,326,812	1,568,729
planning	6	29,415	**17,153**	24,426

give the complete version of this paper that contains a more detailed description of *FI* and more experimental results.

References

1. M.Davis, G.Longemann, D.Loveland. *A Machine program for theorem proving.* Communications of the ACM. -1962. -V.5. -P.394-397.
2. E.Goldberg. *Determinization of resolution by an algorithm operating on complete assignments.* Technical report, CDNL-TR-2006-0110, January 2006, available at http://eigold.tripod.com/papers/fi.zip