Improved Resolution-Based Method for Satisfiability Checking Formulas of the Language L

Anatoly Chebotarev and Sergey Krivoi

Glushkov Institute of Cybernetics Ukrainian Academy of Sciences
Glushkov's pr., 40, Kiev, 03187, Ukraine
ancheb@gmail.com, krivoi@i.com.ua

Abstract. The language L is used for specifying finite automata, and is a fragment of a first order language with monadic predicates. Checking specification for satisfiability plays an important role in the development of reactive algorithms. Restricted syntax of this language and interpreting it over the integers make it possible to substantially improve resolution-based methods for satisfiability checking. This has been done in previous papers devoted to R- and S-resolution. In this paper, we present yet another improvement based on the restriction of the type of atoms upon which the resolution is allowed.

1 Introduction

The language L is used as a specification language in the system for provably correct design of reactive algorithms from their logical specifications [1]. This language is a subset of a first-order language with monadic predicates interpreted over the set of integers. Checking specifications for satisfiability plays an important role in the design process. The corresponding procedure is used in almost all specification transformations not only to detect the internal inconsistency of the specification but also to verify the designer's decisions (changes in the specification) in the course of interactive development of the algorithm [2,3]. So, a rather high efficiency of satisfiability checking algorithm is required. The algorithm we propose here is based on the resolution inference search procedure. The main reason of inefficiency of resolution-based methods is generating a large number of redundant clauses during the inference search process. A reduction in the number of generated clauses is attained by imposing various restrictions on the application of the resolution rule. An efficient method for checking satisfiability of a language L formula was suggested in [4], where the simplification of the corresponding procedure was achieved by means of restricting the type of atoms upon which the resolving is allowed. The resolution rule in this method was called R-resolution. Additional improvements to this method were made in [5], where a set of clauses was partitioned into several classes and resolving was only allowed between clauses belonging to the same class. This results in significant reduction in the amount of the generated clauses as compared with

I. Virbitskaite and A. Voronkov (Eds.): PSI 2006, LNCS 4378, pp. 438–442, 2007.

the R-resolution method. In this paper, we present one more improvement to this method connected with the additional restriction on the type of atoms upon which the resolving is allowed.

2 Basic Notions

First, we recapitulate briefly the basic notions concerning the language L and R-resolution. For more details the reader is referred to [4]. Let T be a class of formulas constructed by means of logical connectives from atoms of the form $p(t+k)$, where p is a monadic predicate symbol, t is a variable ranging over the set of integers $\mathbf{Z}$, and k is an integer constant called the rank of the atom. The language L consists of the formulas of the form $\forall tF(t)$, where $F(t) \in T$ and is interpreted on $\mathbf{Z}$. The example of such a formula is $\forall t(y(t-1)\&x(t) \rightarrow y(t))$, where y and x are predicate symbols and $(t-1)$ is an abbreviation for $(t+(-1))$.

A formula $\forall tF(t)$ is called satisfiable if it has a model, i. e. the interpretation in which it evaluates to true. Since $F(t)$ is interpreted over the set of integers the equivalence $\forall tF(t) \leftrightarrow \forall tF(t+k)$, where $F(t+k)$ denotes the formula obtained from $F(t)$ by adding k to the ranks of all its atoms, holds for any integer k. So, we may assume that the maximum rank of atoms occurring in any formula is equal to 0. Such formulas will be referred to as right-normalized. The formula $F(t)$ in the specification is assumed to be represented in the conjunctive normal form which is viewed as a set of clauses, i. e. disjunctions of literals, where a literal is an atom or its negation. A clause containing no literals is called an empty clause (denoted by $\Box$). A set of clauses is called right-normalized if each its clause is right-normalized.

Definition 1. *Let $c_1 = c \vee p(t)$, $c_2 = c' \vee \neg p(t)$ be right-normalized clauses, where $p(t)$ is an atom of rank 0. The clause $c \vee c'$ is called an R-resolvent of c_1 and c_2 upon the atom $p(t)$.*

R-resolution (restricted resolution) is an inference rule which only admits resolving upon atoms of rank 0.

Definition 2. *An R-deduction of a clause c from a set of clauses C is a finite sequence of clauses $c_1, \ldots, c_m$ such that $c_m = c$ and each c_i $(i = 1, \ldots, m)$ either belongs to C, or is an R-resolvent of c_j and c_k for j, $k < i$, or is a result of right-normalization of c_{i-1}.*

Definition 3. *A clause $c_1(t)$ subsumes a clause $c_2(t)$ if there exists $k \in \mathbf{Z}$ such that $c_1(t+k)$ is a subset of $c_2(t)$.*

A clause set C is called unsatisfiable if it specifies an unsatisfiable formula $\forall tF(t)$. The following proposition has been proved in [4].

Proposition 1. *A set C of right-normalized clauses is unsatisfiable if and only if there exists an R-deduction of the empty clause from C.*

The corresponding procedure checking a set of clauses for satisfiability is called an R-completion procedure. In this procedure, the clauses subsumed by other clauses are removed after adding each new clause to the current set of clauses.

3 Separate Resolution Method

Let $p_1 < p_2 < \ldots < p_n$ be an ordering of predicate symbols occurring in the right-normalized set of clauses C. This ordering of predicate symbols is associated with the partition of C into subsets C_i $(i = 1, \ldots, n)$, where C_n consists of all the clauses which contain the atom $p_n(t)$ (the literal $p_n(t)$ or $\neg p_n(t)$) and C_i $(i = 1, \ldots, n-1)$ consists of all the clauses that do not belong to any C_j $(j > i)$ and contain the atom $p_i(t)$.

Definition 4. *An S-deduction of a clause c from the set of clauses C is such an R-deduction of c from C, where the R-resolution rule is only applied to the clauses in the same subset C_i and the only atom the clauses from C_i are resolved upon is $p_i(t)$.*

The method is based on the following theorem.

Theorem 1. *If C is an unsatisfiable set of right-normalized clauses, then for any ordering of the predicate symbols occurring in C, there exists an S-deduction of the empty clause from c.*

First we prove the following proposition.

Proposition 2. *For any right-normalized clause set C with ordered predicate symbols and a clause c containing no atoms of rank grater than -1, the existence of an R-deduction of c from C implies the existence of its S-deduction.*

The validity of Theorem 1 immediately follows from this proposition since the empty clause does not contain any atoms.

To prove Proposition 2, it suffices to consider an R-deduction that does not contain clauses obtained by application of the right-normalization operation. We shall refer to such an R-deduction as a simple R-deduction. Indeed, if we define appropriately a notion of the depth of a deduction (for example, the number of successive applications of the right-normalization operation in the deduction tree) we can easily show by induction on the depth of the deduction that if Proposition 2 holds for a simple R-deduction it also holds for any other R-deduction.

Let c be a clause that does not contain atoms of rank grater than -1. Consider a simple R-deduction of c from the clause set $C = \{c_1, \ldots, c_n\}$. In such a deduction, all clauses, except the last, contain atoms of rank 0. With every clause c_i of this deduction we associate a clause c'_i consisting of all literals of rank 0 contained in the clause c_i. The sequence of clauses c'_i corresponding to the R-deduction of clause c is an R-deduction of $\Box$ from the clause set $C' = \{c'_1, \ldots, c'_n\}$. It is easy to show that if there exists an S-deduction of $\Box$ from C', then there also exists an S-deduction of c from C. Thus the problem is reduced to the propositional case.

We now consider an unsatisfiable set of clauses C' whose atoms are propositional variables ordered in the following way: $p_1 < p_2 < \ldots < p_q$. The existence of an S-deduction of $\Box$ from C' follows from the Davis-Putnam method [6] which can be reformulated as follows.

Proposition 3. *Let C' be a set of propositional clauses and p any propositional variable occurring in C'. If all the resolvents upon variable p which are not tautologies are added to C' and all the clauses containing p or ¬p are removed, then the resulting set of clauses is unsatisfiable if and only if the original set is unsatisfiable.*

Let $W_1, W_2, \ldots, W_q$ be the partition of C' corresponding to the above ordering of the variables. The variable p_q is contained only in the clauses of W_q. Applying the rule of Proposition 3 to W_q and p_q we eliminate the variable p_q from the set of variables occurring in the resulting clause set. Next, we eliminate the variable p_{q-1} applying this rule to the clauses in W_{q-1}. Proceeding in this manner, we obtain an S-deduction of $\Box$ from C'. Thus, if there exists a simple R-deduction of a clause c containing no atoms of rank grater than -1 from the clause set C, then there exists an S-deduction of $\Box$ from C' and hence there exist an S-deduction of c from C. This completes the proof of Proposition 2 as well as Theorem 1.

We now can summarize the main features of the separate resolution method.

1. R-resolving is only allowed between clauses belonging to the same subset of the partition corresponding to the chosen ordering of predicate symbols.

2. In every subset C_i of the clause set partition, resolving is only allowed upon the atom $p_i(t)$.

3. The order in which the subsets of clauses are handled is not essential because the subsets are not removed after generating all resolvents upon the corresponding atom.

4. An S-resolvent c that is not a tautology and not subsumed by any other of the existing clauses is added to the corresponding subset (according to the partitioning rule) in the right-normalized form, and all the clauses in the current set of clauses that are subsumed by c are removed.

4 Example

Consider the partition of the clause set corresponding to the following ordering of its predicate symbols: $x < u < y < z$.

The subset C_4 (corresponds to z):
$(y(t-2) \vee \neg y(t-1) \vee z(t) \vee \neg u(t))$ 1,
$(\neg z(t-2) \vee \neg y(t-1) \vee \neg z(t) \vee y(t))$ 2,
The subset C_3 (corresponds to y):
$(z(t-1) \vee y(t) \vee \neg u(t))$ 3,
$(y(t-2) \vee \neg y(t-1) \vee \neg y(t) \vee u(t))$ 4,
$(z(t-1) \vee \neg y(t-1) \vee y(t) \vee \neg x(t))$ 5,
$(z(t-1) \vee y(t) \vee x(t))$ 6.
The subset C_2 (corresponds to u):
$(z(t-1) \vee \neg u(t-1) \vee \neg u(t) \vee x(t))$ 7,
$(z(t-1) \vee \neg y(t-1) \vee u(t-1) \vee u(t) \vee \neg x(t))$ 8.
The subset C_1 (corresponds to x):
$(\neg z(t-2) \vee \neg y(t-2) \vee u(t-1) \vee x(t))$ 9.

The number of the clause is written to the right of it, and pairs of numbers written to the left of the resolvents indicate the numbers of clauses being resolved.

The process of S-completion proceeds as follows.

(1, 2) $(\neg z(t-2) \vee y(t-2) \vee \neg y(t-1) \vee \neg u(t) \vee y(t))$ 10, is added to C_3.

(4, 5) $(y(t-2) \vee z(t-1) \vee \neg y(t-1) \vee \neg x(t) \vee u(t))$ 11, is added to C_2.

(4, 6) $(y(t-2) \vee z(t-1) \vee \neg y(t-1) \vee x(t) \vee u(t))$ 12, is added to C_2.

(7, 12) $(y(t-2) \vee z(t-1) \vee \neg u(t-1) \vee \neg y(t-1) \vee x(t))$ 13, is added to C_1.

The process terminates after generating four resolvents while in the process of R-completion 35 clauses are generated.

5 Conclusion

We have proposed an efficient resolution based method for satisfiability checking specifications in the language L. This method leads to significant reduction in the number of clauses generating during the satisfiability checking in comparison with the method of R-resolution. One more factor ensuring the efficiency of the method is reduction in the number of clause pairs checking for the possibility to be resolved. The result proved in the paper may be regarded as the proof of completeness of the strategy combining a predicate symbols ordering with R-resolution.

It should be noted that different orderings of predicate symbols lead to different partitions of the clause set that may result in different numbers of clauses generated in the process of satisfiability checking. In turn, different orders of subsets handling may lead to different run times of the procedure. As an appropriate heuristics we recommend to handle subsets of clauses C_i in the decreasing order of their subscripts.

References

1. A. Chebotarev. Provably-correct development of reactive algorithms. *Proc. Int. Workshop "Rewriting Techniques and Efficient Theorem Proving" (RTETP-2000), P. 117 - 133 (2000).*
2. A. Chebotarev. Determinisations of logical specifications of automata. *Cybernetics and Systems Analysis (translated from Russian), v.31, N1, P. 1 - 7 (1995).*
3. A. N. Chebotarev, M.K. Morokhovets. Resolution-based approach to compatibility analysis of interacting automata. *Theoretical Computer Science, 194, P. 183 - 205 (1998).*
4. A. N. Chebotarev, M.K. Morokhovets. Consistency checking of automata functional specifications. *Proc. LPAR'93, Lecture Notes in Artificial Intelligence, v. 698, Springer, Berlin, P. 76 - 85.(1993).*
5. A. Chebotarev. Separate Resolution Method for Checking the Satisfiability of Formulas in the Language L. *Cybernet. Systems Analysis (translated from Russian) v.34, N6, P. 794 - 799. (1998).*
6. C.L. Chang, R.C.T. Lee. Symbolic Logic and mechanical theorem proving. *Academic press. (1973).*