Applied Mathematics & Information Sciences
An International Journal

The Parallel Theorem Proving Algorithm Based on Semi-Extension Rule

Zhang Li-Ming[1,2], **Ouyang Dan-Tong** [1,2], **Zhao Jian**[1,2] and **Bai Hong-Tao**[1,2,3]
[1]School of Computer Science and Technology, Jilin University, Changchun 130012, China
[2]Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University
[3]Center for Computer Fundamental Education, Jilin University, Changchun 130012, China
Email Address:ouyd@jlu.edu.cn; limingzhang@jlu.edu.cn

Received: Received Dec. 19, 2010; Revised Feb. 10, 2011; Accepted Apr. 5, 2011
Published online: 1 October 2012

Abstract: After a deep investigation on the maximum terms space of the clause set, the concept of the partial maximum terms space of the clause set, which the maximum terms of the clause set decomposed, is brought forward. By investigating the extension rule, this paper introduces the concept of the satisfiability and the unsatisfiability of the partial maximum terms space, and gives an algorithm determining the satisfiability of a partial space of the maximum terms - algorithm PSER (Partial Semi-Extension Rule). Then, the TP problem is decomposed into several sub-problems independent of each other, which can be solved by the given parallel computing method PPSER (Parallel Partial Semi-Extension Rule).

Keywords: Theorem Proving, Parallel Algoritm, Extension Rule.

1 Introduction

The classical NP-complete problem of TP has seen much interest in not just the theoretical computer science community, but also in areas where practical solutions to this problem which enable significant practical applications[1]. However, NP-Completeness does not exclude the possibility of finding algorithms that are efficient enough for solving many interesting satisfiability instances. These instances arise from many diverse areas - many practical problems in AI planning[2-5], circuit testing[6,7] and verification[8-10] for instance.

This research has resulted in the development of several TP algorithms that have seen practical success. These algorithms are based on various principles such as Resolution[11,12], Search[13], Binary Decision Diagrams[14], and Extension rules[15].

Extension-rule based TP method has commended considerable respect from many related researchers. For example, Murray[16,17] has applied the extension rule into the generation of the target language based on the knowledge compilation, and achieved good results. Besides, many researchers applied the extension rule to the model counting problem[18], and many amended it so as to applied it into the TP of modal logic[19] .Still some researchers improved the extension rule, and put forward series of algorithms such as NER, RIER, etc[20,21].

This paper is organized as follows. In section 2, the related extension-rule based TP methods are given. In section 3, the parallel TP method based on the Semi-extension rule is presented. The experimental results of comparing the algorithm proposed in this paper with other algorithms are also presented in section 4. Finally, our work of this paper is summarized in the last section.

2 Extension-Rule based Theorem Proving Method

We begin by specifying the notation that will be used in the rest of this paper. We use Ψ to denote a set of clauses in conjunctive normal form (CNF), C to denote a single clause, and M to denote the set of all the atoms that appear inΨ. The extension rule is defined as follows.

DEFINITION 1[15].Given a clause C, $C \in \Psi$, $D = \{ C \vee A, C \vee \neg A \mid$ "A" is an atom, $A \in M$, "A" and "$\neg A$" does not appear in $C\}$, we call the deduction process proceeding from C to D the extension rule of on C, and call D the result of applying the extension rule of on C.

THEOREM 1[15].A clause C is logically equivalent to the result of the extension rule D .

This theorem ensured the equivalence between the original clause set and the expanded clause set, thus extension rule can be regarded as an inference rule.

DEFINITION 2[22].A non-tautology clause is a maximum term on a set *M* iff it contains all the atoms in *M* in either positive form or negative form.

THEOREM 2[15].Given a set of clauses Ψ, let *M* be the set of all the atoms in it ($|M|= m$). If all the clauses in Ψ are maximum terms on *M*, then the clause set Ψ is unsatisfiable iff it contains 2^m clauses.

Apparently the set of all the maximum terms consist of m atoms is surely contains 2^m maximum terms. Therefore, it is only need to compute the number of distinct maximum terms can be deduced from the clause set that we can determine its satisfiability. In addition, when counting the number of the maximum terms that can be deduced from the clause set, we can use the inclusion-exclusion principle presented below.

THEOREM 3[22].(Inclusion–exclusion principle) The element number of the union of sets of set $A_1, A_2,\ldots, A_n$ can be compute using the formula below：

$$|A_1 \cup A_2 \cup \ldots \cup A_n| = \sum_{i=1}^{n}|A_i| - \sum_{1\le i<j\le n}|A_i \cap A_j| + \ldots + (-1)^{n+1}|A_1 \cap A_2 \cap \ldots \cap A_n|$$

THEOREM 4[15].The intersection of the sets that consist of the maximum terms expanded by two clauses respectively will be empty iff these two clauses contain complementary literals.

Given a set of clauses $\Psi=\{C_1, C_2, \ldots, C_n\}$, let *M* be the set of atoms that appear in it ($|M|= m$).Let P_i be the sets of all the maximum terms we can get from C_i by using the extension rule, and let *S* be the number of distinct maximum terms we can get from Ψ. By using the extension rule, we will have $S=|P_1 \cup P_2 \cup \ldots \cup P_n|$.

3 The Parallel Theorem Proving Algorithm Based on Semi-Extension Rule

The idea of the paralleled semi-extension rule based algorithm is as follows. Firstly, the algorithm decomposes the maximum terms space of the clause set into several partial maximum terms spaces, which convert the SAT problem of the clause set into the SAT problem of the partial maximum terms spaces. If there is a certain partial maximum terms space that is satisfiable, then the clause set is satisfiable. If all the partial maximum terms spaces are unsatisfiable, then the clause set is unsatisfiable. In other words, the clause set is satisfiable. In the following, the concept of the partial maximum terms space will be given.

DEFINITION 3. For the set $M=\{L_1,L_2,\ldots L_m\}$, the 2^m maximum terms corresponding to *M* is $\{\neg L_1 \vee \neg L_2 \vee \ldots \vee \neg L_{m-1} \vee \neg L_m, \neg L1 \vee \neg L2 \vee \ldots \vee \neg L_{m-1} \vee Lm, \ldots, L_1 \vee L_2 \vee \ldots \vee L_{m-1} \vee \neg L_m, L_1 \vee L_2 \vee \ldots \vee L_m\}$, and we number each maximum term as mi(0),mi(1),…,mi(2^{m-2}),mi(2^{m-1}).

DEFINITION 4.Given a clause set $\Psi=\{C_1, C_2,\ldots,C_n\}$, let *M* be the set of its literals, and $|M|=m$. We call maximum terms space of M as MI(*M*). Assuming that $1\le 2^k \le 2^m$,if we would like to decompose the maximum terms space into 2^k spaces, then each space is of this form MIS(*j*)={mi(*j*)|*j*∈{ mi($2^{m\times(j-1)}/ 2^k$), mi($2^{m\times(j-1)}/2^{k+1}$),…, mi($2^{m\times(j)}/ 2^{k-1}$)}}, $1\le j\le 2^k$.

DEFINITION 5. For the partial maximum terms space MIS(*j*), $1\le j\le 2^k\le 2^m$. If all the maximum terms in it can be expanded by the clauses of the clause set, then MIS(*j*) is said to be unsatisfiable. If there exist a certain maximum term that cannot be expanded by any clause of the clause set, then MIS(*j*) is said to be satisfiable.

THEOREM 5.If every partial maximum terms space is unsatisfiable, then the clause set is unsatisfiable. If there is a certain partial maximum terms space that is satisfiable, then the clause set is satisfiable.

In the following, the algorithm PSER which determines the satisfiability of the partial maximum terms space will be given.

DEFINITION 6. $M=\{L_1,L_2,\ldots,L_m\}$,$m=|M|$. Let clause $C= L_i \vee \ldots \vee L_j \ldots \vee L_d$, $1\le i\le j\le d\le m$, which *d* is referred as the degree of clause *C*. $\Psi=\{C\vee L_k, C\vee\neg L_k | d<k\le m\}$, we call the operation proceeding from *C* to the elements of Ψ the semi-extension rule, and the elements of Ψ the result of the semi-extension rule.

PROPOSITON 1.According to definition 6, when applying the semi-extension rule on *C*, the remaining m-d atoms could be positive or negative, therefore *C* can semi-expand 2^{m-d} clauses.

PROPOSITION 2.Let d_1 and d_2 be the degrees of clause C_1 and C_2 respectively, while $d_1<d_2$ and $C_1 \subseteq C_2$. According to proposition 1, the clause that C_1 or C_2 can semi-expand is obtained by compose the m-d_1 atoms or m-d_2 atoms in positive form or in negative form. Therefore, clauses that C_2 can semi-expand are a subset of the clauses that C_1 can expand.

According to proposition 2, when determining whether the maximum terms clause can be expanded by the clauses, we should determine whether it can be expanded by the clauses of smaller degree first. In the following, the algorithm determining the satisfiability of the partial maximum terms space will be given.

```
Function PSER(CNF:Ψ, INT:starti, INT:endi )
1   BEGIN
2       i←starti; Ψ=DegreeSort(Ψ);
3       While(i< endi)
4           BEGIN
5               T←getMaxTerm(i);
6   If Expand(Ψ, T)==false
7   Then Return SAT;
8   Else If (len<m) i=LastMi(C,M)
9   i++;
10          END
11      Return UNSAT;
12  END
```

In the following, the related theorem and algorithm of Expand will be given.

THEOREM 6[21].Given a clause set$\Psi=\{C_1,C_2,\ldots,C_n\}$, Let *M* be the set of its literals, and $|M|=m$. A maximum term $T=L_1 \vee L_2 \vee \ldots \vee L_m$ on M can be expanded by clause $C = L_i \vee \ldots \vee Lj\ldots \vee Ld$, $1 \leq i \leq j \leq d \leq m$, iff $\{L_i,\ldots,L_j,\ldots,L_d\} \subseteq \{L_1,L_2,\ldots,L_m\}$.

In the above, we gave the solving method of partial maximum terms space and the algorithm determining its satisfiability. The maximum terms space of a clause set can be decomposed into several partial maximum terms spaces. In doing so, the SAT problem of a clause set is converted into the SAT problems of several partial maximum terms spaces. If there is a partial maximum terms space that cannot be expanded, then the clause set is satisfiable. Or else, if all the partial maximum terms space is unsatisfiable, then the clause set is unsatisfiable. In the following, the parallel TP algorithm based on semi-extension rule will be given.

```
Function PPSER( CNF: Ψ, INT: threadnum)
    While i < threadnum  do
    BEGIN
            tid = creatthread ();
            If (tid = = 0 )
            BEGIN
            Result[i] = PSER(Ψ, start(i), end(i));
                    exit(0);
            END
            i++
    END
    While (1) do
    BEGIN
            int Count =0;
            While  j < threadnum  do
      BEGIN
                    if (Result[j]==SAT)
                            return  SAT;
                    if (Result[j]==UNSAT)
                            Count++;
      END
            if (Count==n)  return UNSAT;
    END
```

The concrete flow of the algorithm is as follows: The parent process distributes Threadnum sub-threads, and then these sub-threads are arranged to many cores of the processor by the Operating System, respectively. Each sub-thread calls the function PSER, and records the corresponding returned results using Result[j], while the parent process monitors running result of each sub-thread. If there is a sub-thread that its Result[j] is SAT, then the algorithm returns SAT. If the Result[j] of every sub-thread is UNSAT, then the algorithm returns UNSAT.

4 Experimental Results

On the basis of literature 21, in this section, we compare our algorithm PPSER with algorithm NER, algorithm IER and Directional Resolution algorithm 12 proposed by Dechter and et al, respectively. The experiments are carried out on a Dell Dimension C521, AMD Athlon(tm) 64 X2 Dual Core Processor 3600+, 1.9GHz 1022MB RAM with Windows XP. This experiment uses Uniform Random-3-SAT benchmark 23 lays on phase change zone and standard test cases of frb 24 as test cases. For the entire 1000 issues of uf20-90, table 1 only shows the experimental results of 10 issues randomly selected. The first column of the table shows the sample name, the latter 3 columns shows the runtime of three algorithms corresponding to each issue which the unit of data is Seconds (s). For each test case, we will test 10 times and take the mean value as the experimental results. As we can see from the test, our algorithm PPSER has a significant advantage on efficiency compared with the original algorithm, which is 8-20 times higher than the relatively fast algorithm NER.

Table.1. Experimental Results of Uniform Random-3-SAT Benchmark Instances.

PROBLEM	NER(s)	DR(s)	IER(s)	PPSER(s)
uf20-02	0.015	1.938	0.062	0.0008
uf20-05	0.015	1.359	0.109	0.0012
uf20-07	0.031	0.781	1.218	0.0018
uf20-09	0.078	4.797	0.609	0.0045
uf20-018	0.078	1.563	6.968	0.0041
uf20-023	0.046	1.203	0.484	0.0025
uf20-036	0.031	1.797	0.313	0.0018
uf20-040	0.046	1.390	0.453	0.0023
uf20-042	0.015	2.343	0.250	0.0012
uf20-069	0.078	4.000	0.484	0.0025

First, we select instances, which the parameters are $<N,20,10>$ and $<N, 30,10>$ respectively, for testing, where the parameter N is restricted as $60 \leq N \leq 160$. For each instance of different difficulty levels, they will randomly generate 10 samples for solving, and let the mean value as the final result. The experimental result is in Figure 1 and Figure 2.

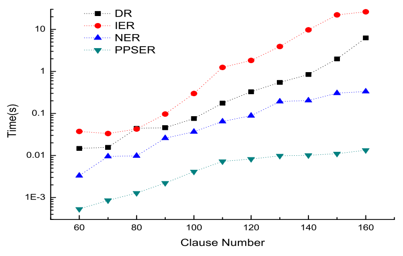

Fig.1. $<N, 20, 10>$

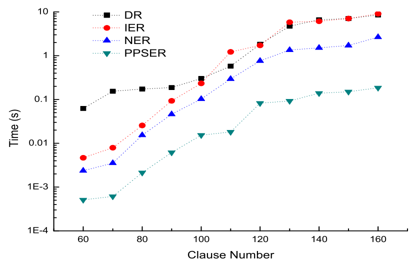

Fig.2. $<N, 30, 10>$

By testing the Random SAT problems, we can see from figure 1 and figure 2, that our algorithm PPSER has an obvious advantage on efficiency, which is 6-15 times higher than the relatively fast algorithm NER. Moreover, when the number of clauses increased to 130 above, the computing time of our algorithm PPSER is increased gently.

5 Conclusions

This method decompose the maximum terms space of the clause set into several partial maximum terms spaces, and determines the satisfiability of each partial maximum terms space. If all the partial maximum terms space are unsatisfiable, the clause set is unsatisfiable. Or else, if there is a certain partial maximum terms space that is satisfiable, then the clause set is satisfiable. Besides, when determining the satisfiability of the partial maximum terms space, if we found that a maximum term can be expanded by a certain clause, then all the maximum terms semi-expanded based on this clause have no need to determine their expandability, thus effectively reduced the number of the maximum terms to be determined. The experiment results show that our algorithm PPSER has a reasonable execution efficiency, which is superior to algorithm NER, DR, IER and et al.

Acknowledgments

This work was supported in part by NSFC under Grant Nos. 60973089, 60873148, 60773097,61003101; Jilin Province Science and Technology Development Plan under Grant Nos.20101501,20100185, 2009010 , 20 080107, 201101039; Doctoral Fund of Ministry of Education of China(20100061110031);Opening Fund of Top Key Discipline of Computer Software and Theory in Zhejiang Provincial Colleges at Zhejiang Normal University;Zhejiang Provincial Natural Science Foundation under Grant Nos.Y1100191;

References

[1]. S. A. Cook. The complexity of theorem-proving procedures, Proceedings of Third Annual ACM Symposium on Theory of Computing, (1971) 151-158.
[2]. H. Kautz, B. Selman. Planning as satisfiability, Proceedings of European Conference on Artificial Intelligence (ECAI-92), 1992.
[3]. J. Rintanen, K. Heljanko, I. Niemel. Parallel encodings of classical planning as satisfiability, Proceedings of JELIA, Lisbon, Portugal, (2004)307–319.
[4]. Henry Kautz, Bart Selman. Unifying sat-based and graph-based planning. Proceedings of IJCAI-99, (1999)318-325.
[5]. Henry A, Kautz. Deconstructing planning as satisfiability, Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06), 21(2)(2006)1524-1526.
[6]. P. Stephan, R. Brayton, and A. Sangiovanni. Combinational test generation using satisfiability, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 15(9)(1996) 1167-1176.
[7]. D. Jackson, M. Vaziri. Finding bugs with a constraint solver, Proceedings of International Symposium on Software Testing and Analysis, Portland,25(5)(2000)14-25.
[8]. A. Myla. TAME: using PVS strategies for special-purpose theorem proving, Annals Mathematics and Artificial Intelligence, 29(4) (2000)139-181.
[9]. Bruno Blanchet. An efficient cryptographic protocol verifier based on prolog rules, Proceedings of 14th IEEE Computer Security Foundations Workshop, (2001) 82-96.
[10]. Massart Thierry, Meuter Cedric, V B Laurent. On the complexity of partial order trace model checking, Inform. Process. Lett. 106(3)(2008) 120-126
[11]. M. Davis, H. Putnam. A computing procedure for quantification theory. Journal of ACM, 7(3)(1960) 201-215.
[12]. Dechter R, Rish I. Directional resolution: the davis-putnam procedure. Proceeding of 4th International Conference on Principles of KR&R, Bonn, Germany: Morgan Kaufmann, (1994) 134-145.
[13]. M. Davis, G. Logemann, D. Loveland, A machine program for theorem proving. Communications of the ACM, 5(7)(1962)394-397.
[14]. R. E. Bryant. Graph-based algorithms for boolean function manipulation , IEEE Transactions on Computers, 35(1986)677-691.
[15]. H Lin, JG Sun, YM Zhang. Theorem proving based on the extension rule, Journal of Automated Reasoning, 31(1)(2003) 11-21.
[16]. N. V. Murray, E. Rosenthal. Duality in knowledge compilation techniques, Proceedings of the International Symposium on Methodologies for Intelligent Systems, (2005)182-190.
[17]. H Lin, JG Sun. Knowledge compilation using extension rule, Journal of Automated Reasoning, 32(2)(2004)93-102.
[18]. MH Yin, H Lin, JG Sun. Counting models using extension rules, Proceedings of AAAI, (2007)1916-1917.
[19]. X Wu, JG Sun, H Lin, et al. Modal extension rule [J]. Progress in Natural Science, 15(6)(2005) 550-558.
[20]. X Wu, JG Sun, S Lu, et al. Improved propositional extension rule, Proceedings of the 1st International Conference on Rough Sets and Knowledge Technology, Chongqing, China, (2006)592-597.
[21]. LM Zhang HL Zeng, F Yangg, DT Ouyang. Dynamic Theorem Proving algorithm for Consistency-based diagnosis [J]. Expert Systems With Applications. 2011,38(6):7511-7516.
[22]. JG Sun, FJ Yang, DT Ouyang, et al. Discrete math. Beijing, High Education Press, 2002.
[23]. I P Gent, HV Maaren, T Walsh, et al. An online resource for research on sat, Proceedings of SAT 2000, (2000)283-292.
[24]. K Xu, F Boussemart, F Hemery, C Lecoutre. Random constraint satisfaction: easy generation of hard (satisfiable) instances , Artificial Intelligence, 171(2007)514-534.