1E3-OS-4-7

高速充足可能性判定器を用いた命題論理の結論発見器の実装

A propositional concequecnce finding useing modern SAT solver

村松 匠[*1] 鈴木 健士郎[*2] 鍋島 英知[*3] 岩沼 宏治[*3]
Takumi Muramatsu Kenshirou Suzuki Hidetomo Nabeshima Kouji Iwanuma

[*1]山梨大学大学院医学工学総合教育部コンピュータ・メディア工学専攻
Computer Science and Media Engineering, Department of Education Interdisciplinary Graduate School of Medicine and Engineering, University of Yamanashi

[*2]ＮＥＣ ソフト株式会社
NEC Soft, Ltd.

[*3]山梨大学大学院医学工学総合研究部
Department of Research Interdisciplinary Graduate School of Medicine and Engineering, University of Yamanashi

In this paper, we propose a new propositional consequence finding system based on s satisfiability testing program (called a SAT solver). Propositional or boolean satisfiablility testing (SAT) is a specialization of propositional consequence finding, and the performance of modern SAT solvers has improved dramatically in recent years. For finding consequences from a given propositional formula, we use a fast SAT solver as an inference engine. Our approach is based on the refutation theorem. A SAT solver checks the satisfiability of the given formula and the negation of hypothesis which is a candidate of consequences. We propose several pruning techniques to reduce the search space of hypotheses. We show the performance of our system by experiment results.

1. はじめに

本研究の目的は，高速な命題論理の充足可能性判定器を用いた結論発見器の実装である．

公理集合から自明ではない興味深い結論を導出することを**結論発見** (consequence finding) と呼ぶ．結論発見は，帰納推論や発想推論を実現する手段として，多くの応用分野に適用できる有用な枠組みである．本稿では命題論理における結論発見問題を対象とする．ある命題論理式から導出可能な結論の数は有限であるが，大規模な問題においては膨大な数の結論が生成される可能性がある．そこで本研究では，ユーザが指定した特定の語彙に属し，包摂に関して極小な結論の発見を考える．もし言語全体を対象としたならば，これは**主節** (prime implicates) の発見に等しい．

この目的のために近年性能向上が著しい命題論理の充足可能性判定器を利用する．命題論理における充足可能性判定問題（satisfiability testing problem; **SAT 問題**）を解く最新の充足可能性判定器（以降，SAT ソルバーと呼ぶ）は，数百万の命題変数から構成される SAT 問題を求解することが可能になってきている．我々の結論発見手法は，反駁定理に基づく．結論の候補となる仮説の否定を命題論理式に加え，その充足可能性を判定し，もし充足不能ならば仮説を結論として出力する．この手法では，いかにテストすべき仮説集合を抑え込むかが重要となる．この目的のために，我々は最新の SAT 技術に基づく種々の枝刈り手法を導入する．例えば，矛盾原因解析技術を利用した仮説空間の枝刈りや，インクリメンタル探索を利用した冗長な探索の枝刈りなどである．

関連研究の 1 つに，Simon らにより提案された結論発見器 Zres[7] がある．Zres では，ZBDD 上での kernel resolution を利用して複数の結論を効率的に導出する．Zres は，比較的小規模な問題においては膨大な数の結論を高速に列挙可能な非常に優れたシステムである．しかし，最新の SAT ソルバーが対象とするような数百万変数からなる大規模問題においては，メモリ上の制約や巨大な ZBDD 上での演算が時間を要することから，現実的な時間内に結論を求めることが困難である．一方，我々のシステムでは，全解列挙は困難であったとしても，結論集合の一部は導出できる可能性が高く，現実的な結論列挙手法として利用可能である．

本論文の構成は次の通りである．まず 2. 節において命題論理の結論発見問題を定義する．3. 節において最新 SAT ソルバーにおける要素技術を紹介し，4. 節において提案手法を示す．5. 節は実験結果であり，6. 節で本研究をまとめ，今後の課題について述べる．

連絡先: 山梨大学大学院医学工学総合教育部 コンピュータ・メディア工学専攻，〒 400 - 8511 山梨県甲府市武田 4 - 3 - 11，E-mail : muramatsu@nabelab.org

2. 命題論理の結論発見問題

ある命題論理式から導出可能な結論の数は有限ではあるが，大規模問題においては膨大な数の結論が導出される可能性がある．そこで一階述語論理における結論発見手続き SOL [1] に倣って，求めたい結論の語彙を表す**生成領域**を定義する．まず用語を定義する．命題変数またはその否定を**リテラル**といい，リテラルの選言を**節**という．生成領域 $\mathcal{P}$ はリテラルの集合 L と結論が満たすべき条件 $cond$ からなる 2 つ組で定義される：

$$\mathcal{P} = \langle L, cond \rangle$$

ある節 C が $\mathcal{P}$ に属するとは，C に含まれるすべてのリテラルが L に含まれ，かつ C が条件 $cond$ を満たしている場合をいう．$cond$ の例としては，結論の長さ制限などが挙げられる．ただし本論文では，生成領域の条件は考慮せず，今後の課題とする．

節集合 Σ に対して，Σ の論理的帰結で生成領域 $\mathcal{P}$ に属する節の集合を $Th_{\mathcal{P}}(\Sigma)$ と表す．また節集合 Σ 中の任意の節により真に包摂されない節集合 $\mu\Sigma$ と表す．**命題論理の結論発**

見問題とは，$\mu Th_{\mathcal{P}}(\Sigma)$ を求めることである．もし生成領域のリテラル集合が空の場合は，空節のみが結論となるため，充足可能性の判定問題と等しくなる．一方，すべてのリテラルを対象とした場合は，主節 (prime implicates) の列挙に等しい．

3. 最新 SAT ソルバーの要素技術

SAT は，最初に NP 完全性が証明された問題であり，計算機科学において重要な基本問題の 1 つである．SAT 問題を解くソルバーの性能は，古典的な求解手続きである DPLL[2] に，矛盾からの節学習とバックジャンプ法 [6]，監視リテラルによる高速単位伝搬 [3]，優れた変数選択ヒューリスティックス [3]，リスタート戦略 [5]，部分解キャッシング [12]，有用な学習節の評価尺度 [10] などの技術の導入によって飛躍的に性能を伸ばしている．本節では，本研究に関連するインクリメンタル探索と仮定に基づく充足可能性判定とその原因解析技術について紹介する．その他の技術に関しては，例えば文献 [11] などを参照されたい．

古典的な DPLL 手続きに矛盾からの節学習 (conflict driven clause learning)[6] を導入した SAT ソルバーを **CDCL ソルバー**と呼ぶ．CDCL ソルバーでは，探索の過程で矛盾（ある（部分）真偽値割り当てにおいて論理式が充足不能になること）に遭遇した場合，その矛盾を引き起こした原因を解析し，その否定を節として学習することで，再び同様の探索失敗に陥ることを防ぐ．学習節は，元の論理式から導出された帰結でもある．インクリメンタル探索では，この学習節集合を，別の問題を解くために再利用する．

SAT は判定問題であるため，例えば最適化問題の SAT 解法などでは，複数の SAT 問題を解くことが本質的に必要となる．このような場合，しばしば SAT 問題に節を追加しながら何度も解くことが行われる．2 つの SAT 問題 P, Q に対し，$P \subseteq Q$ であるならば，P を解く過程で得られた学習節集合は，Q の帰結でもあるため，Q を解く際に再利用することが可能である．すなわち，P を解く際に失敗した探索空間を再び探索することを防ぎながら，Q を効率よく解くことが可能である．これを SAT ソルバーの**インクリメンタル探索**という．

また，最新 CDCL ソルバーの多くでは，いくつかのリテラルを真と仮定した上で，充足可能性の判定を行うことができる．すなわち，SAT 問題を P，リテラル集合を $A = \{L_1, \cdots, L_n\}$ とすると，$P \wedge L_1 \wedge \cdots \wedge L_n$ の充足可能性を判定する．この過程で獲得される学習節は，ソルバー内部において $L_1, \cdots, L_n$ との融合を抑止するため，P の帰結となる．従って，ある 2 つの SAT 問題 P,Q について，それらが単位節 $L_1, \cdots, L_n$ を除き $P \subseteq Q$ であるならば，やはり Q を解くために，P から獲得された学習節集合を再利用可能である．

さらに，もし P が充足可能で，$P \wedge L_1 \wedge \cdots \wedge L_n$ が充足不能であった場合，CDCL ソルバーは先に述べた矛盾解析技術を利用して，充足不能の原因 $L'_1, \cdots, L'_m$ $(L'_i \in A, m \leq n)$ を求めることが可能である．すなわち，$A \setminus \{L'_1, \cdots, L'_m\} = \{L''_1, \cdots, L''_k\}$ $(m + k = n)$ とすると，$P \wedge L'_1 \wedge \cdots \wedge L'_m$ は充足不能であり，$P \wedge L''_1 \wedge \cdots \wedge L''_k$ は充足可能である．本稿ではこれを**仮定に基づく充足可能性判定とその原因解析技術**と呼ぶ．

4. SAT ソルバーによる結論発見手続き

我々の提案する結論発見手法は，生成領域から結論の候補である節（これを**仮説**と呼ぶ）を生成し，よく知られた次の反駁定理を用いて結論かどうかを判定する手法である．

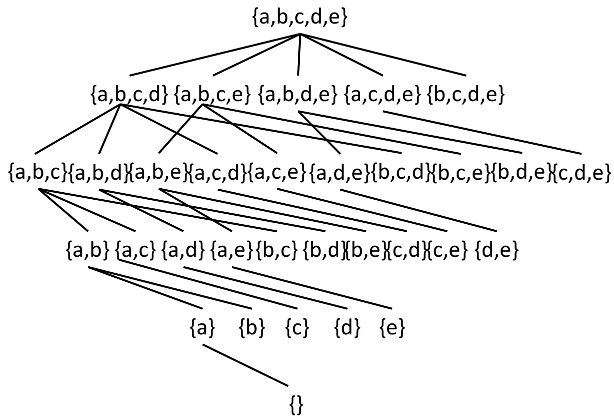

図 1: 集合列挙木の例

定理 1 反駁定理 Σ を節集合，H を節としたとき，$\Sigma \wedge \neg H$ が矛盾するなら，H は Σ から導出可能である．

もし仮説の否定を追加した問題が充足不能ならば，その仮説を結論として出力する．

テストすべき仮説は生成領域から生成する．生成領域を $\langle L, cond \rangle$ とすると，仮説集合は $\{H \in 2^L \mid H \text{ satisfies } cond\}$ となる．本稿では $cond$ は考慮しないため，L のサイズを n とすると，2^n 個の膨大な数の仮説を判定する必要がある．そこで，以下に示す種々の仮説空間の枝刈り手法を利用し，効率化を図る．

1. 仮説列挙法
2. インクリメンタル探索による学習節の再利用
3. 原因解析に基づく枝刈り
4. SAT モデルによる枝刈り
5. 結論のシンメトリーに基づく枝刈り

4.1 仮説列挙法

生成領域に含まれるリテラル集合を L とする．本手法では，以下の手順で構成される集合列挙木を探索する．まず L 上に全順序関係 $\prec$ を仮定し，根ノードを L とする．あるノード P の子ノード C は，P からあるリテラル $e \in P$ を取り除いた集合 $C = P \setminus \{e\}$ として定義される．ここで $\forall e' \in (L \setminus P)(e \prec e')$ である．図 1 に，$L = \{a, b, c, d, e\}$，$\prec$ をリテラル名の辞書式順序とした場合の集合列挙木の例を示す．本手法では，集合列挙木を構成して，これを深さ優先で探索することで，重複なく仮説の探索を行う．

4.2 インクリメンタル探索による学習節の再利用

生成領域に含まれるリテラル集合を L とする．ある仮説 $H \in 2^L$ が論理式 Σ の結論となるかどうかを判定するため，本手法では SAT ソルバーにより $\Sigma \wedge \neg H$ の充足可能性を判定する．もし充足不能ならば H は結論である．ここで $\neg H$ はリテラルの連言となるため，3. 節で述べたように，学習節を再利用することが可能である．すなわち，以前の SAT ソルバーの実行において探索済みの空間を何度も繰り返し探索することを避けることが可能となる．

4.3 原因解析に基づく枝刈り

本手法では，3. 節で述べた最新 SAT ソルバーの “仮定に対する原因解析技術” を利用して仮説空間の枝刈りを行う．仮説を H とし，仮説の否定を $\overline{H} = \{\neg e \mid e \in H\}$ で表し，これをリテラルの連言と同一視する．もし $\Sigma \wedge \overline{H}$ が充足不能の場合，SAT ソルバーは充足不能の原因 $\overline{H'} \subseteq \overline{H}$ を出力する（もし $\overline{H'} = \emptyset$ ならば，充足不能の原因は Σ にあり，Σ そのものが矛盾していることになる）．本研究では，包摂に関して極小の結論集合を求めるため，H' が包摂する仮説を調べる必要はない．従って，先に示した集合列挙木から H' に包摂されるノードを除去することが可能である．ただし単純にノードを取り除いてしまうと，親子関係が失われる場合があるため，仮説の集合列挙木の構成法を拡張する．

あるノード P について，$\Sigma \wedge \overline{P}$ が充足不能の場合に，充足不能の原因解析技術より得られる部分結論を $P^s \subseteq P$ で表す．あるノード P の子ノード C は，P からあるリテラル $e \in P^s$ を取り除いた集合 $C = P \setminus \{e\}$ として定義される．ここで $\forall e' \in (L \setminus P)(e \prec e')$ である．

例として，生成領域が $\mathcal{P} = \langle\{a, b, c, d, e\}\rangle$ の結論発見問題を考える．根ノード $R = \{a, b, c, d, e\}$ に対し，部分結論 $\{a, b, c\}$ が求まったとする．この場合，R の子ノードは $\{a, b, d, e\}$, $\{a, c, d, e\}$, $\{b, c, d, e\}$ の 3 つとなる．このとき，$\{a, c, d, e\}$ の子孫ノードでは，c を含む仮説のみ生成し，$\{b, c, d, e\}$ では，b, c を含む仮説のみを生成する．さらにノード，$\{a, c, d, e\}$ について部分結論 $\{c, d\}$ が求まったとすると，その子ノードは $\{a, c, e\}$ のみとなる．

このように，SAT ソルバーの “仮定に対する原因解析技術” を利用することで，部分結論に包摂される仮説を試すことなく取り除くことが可能となる．

4.4 モデルによる枝刈り

ある仮説 H に対し，$\Sigma \wedge \overline{H}$ が充足可能であるなら，H に包摂されるノードが表す仮説も充足可能である．よってそれらは枝刈り可能である．また，$\Sigma \wedge \overline{H}$ を充足するモデル（真偽値割り当て）が包摂するノードが表す仮説も充足可能である．これも同様に枝刈り可能である．

4.5 シンメトリーに基づく枝刈り

SAT 問題にはしばしば対称性のある構造（シンメトリー）が含まれる．本節では，シンメトリーを利用した効率的なモデル生成と結論生成手法を示す．本稿では構文的シンメトリーを対象とする．Ω を整数集合 $\{1, 2, \cdots, N\}$ としたとき，Ω から Ω への全単射写像を**置換**と呼ぶ．**構文的シンメトリー**とは，式の構造を変化させないリテラルの置換である．例えば以下の式 F に対し，

$$F = (a \vee b \vee c) \vee (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee a) \wedge (\neg a \vee \neg b \vee \neg c)$$

a を b, b を a, $\neg a$ を $\neg c$, $\neg c$ を $\neg a$ に置換した式 F' は,

$$F' = (b \vee a \vee c) \vee (\neg c \vee a) \wedge (\neg b \vee c) \wedge (\neg a \vee b) \wedge (\neg c \vee \neg b \vee \neg a)$$

となり，F と同値である．このような置換が構文的シンメトリーである．

論理式 Σ から抽出した構文的シンメトリーの集合を Δ とする．ある仮説 H に対し，$\Sigma \wedge \overline{H}$ が充足不能の場合，4.3 節で述べた原因解析技術より，部分結論 $H' \subseteq H$ が求まる．このとき，H' と構文的シンメトリーの関係にある $\delta(H')$ $(\delta \in \Delta)$ もまた結論である．よって，$\delta(H')$ に包摂される仮説は，試すことなく取り除くことが可能である．

表 1: Zres との比較実験（鳩の巣問題，全解探索）

問題名	#V	#C	Zres		提案手法	
			time	結論数	time	結論数
php2-2	4	4	0.0	12	0.0	12
php3-3	9	12	0.0	87	0.0	87
php4-4	16	28	0.1	728	14.9	728
php5-5	25	55	12.2	7710	T.O.	7705
php5-6	30	65	153.9	69010	T.O.	58785
php6-6	36	96	1818.0	95652	T.O.	69576
php6-7	42	111	T.O.		T.O.	79065
php7-7	49	154	T.O.		T.O.	141414
php7-8	56	175	T.O.		T.O.	

さらに，もし $\Sigma \wedge \overline{H}$ が充足不能の場合に，SAT ソルバーが**充足不能な極小の部分論理式** (minimally-unsatisfiable subformula; MUS) を出力可能ならば，その部分論理式から抽出した構文的シンメトリー集合 Δ'（局所シンメトリーに相当する）を利用して，H' とシンメトリー関係にある結論 $\delta'(H')$ $(\delta' \in \Delta')$ を求めることが可能である．通常，局所シンメトリーは,（論理式 Σ から抽出した）大域的シンメトリーと比べて多数存在するため，より多くの枝刈りが期待できる．

逆に，$\Sigma \wedge \overline{H}$ が充足可能の場合，そのモデル M と構文的シンメトリー関係にある $\delta(M)$ $(\delta \in \Delta)$ もまたモデルとなる．よって，4.4 節で述べたモデルによる枝刈りが適用可能である．

5. 評価実験

提案手法の性能を評価するため，命題論理の結論発見器 Zres[7] と，一階述語論理の結論発見器 SOLAR[4] と比較実験を行った．なお，提案手法では 4.5 節で述べたシンメトリーに基づく枝刈りは現在未実装である．

Zres との比較実験では，Zres が大規模な問題を扱うことができないため，比較的小規模な鳩の巣問題 9 問で評価を行った．実験環境は Intel Xeon 2.80GHz, 1GB RAM である．Zres との比較結果を表 1 に示す．問題名 php$n$-m は，鳩が n 羽，巣箱が m 個の問題を表す．すべて充足可能な問題である．#V および #C は，それぞれ各問題に含まれる命題変数の数と節の数を表す．結論数は各結論発見器が時間内に発見した結論数, time は CPU 秒である．“T.O.” は制限時間 3600 CPU 秒以内にすべての結論を列挙できなかったことを表す．Zres は全解探索完了後に結論を出力するため，制限時間内に全解探索が完了しなかった問題の結論数は表記していない．

表 1 より，Zres は変数数が 40 程度の問題であれば 1 時間以内にすべての結論を求めることができている．また結論数においても Zres は膨大な数の結論を求めている．提案手法は Zres と比べると性能は劣るが，問題の規模が大きくなっても，結論の列挙ができている．鳩の巣問題には，4.5 節で述べた構文的シンメトリーが多数存在するため，シンメトリーを利用した枝刈り手法を実装することで性能の改善が期待できる．

次に一階述語論理の結論発見器 SOLAR との比較を行った．実験では，鳩の巣問題 34 問と SAT Competition 2009 の Application 部門から充足可能な問題 43 問を利用した．実験環境は Core 2 Duo, 1.66GHz,2 GB RAM で，制限時間を 1 問あたり 600 CPU 秒とした．実験では，3 種類の生成領域（high, middle, low）を用いた．high は，各問題において頻出するリテラルの上位 10%， middle は中間 10%， low は下位 10%とした．頻出するリテラルは，真となる可能性が高いので，high は短い結論が多いと考えられる．また， low は長

表 2: SOLAR との比較実験 (鳩の巣問題，全解探索)

	SOLAR			提案手法		
生成領域	low	middle	high	low	middle	high
求解数	6	4	4	36	15	15
平均結論長	0.00	0.00	2.58	45.00	2.51	3.51
求解時間	68.1	8.0	7.7	8.0	11.4	53.4

表 3: SOLAR との比較実験 (SAT Competition 2009，全解探索)

	SOLAR			提案手法		
生成領域	low	middle	high	low	middle	high
求解数	-	-	-	3	1	1
平均結論長	-	-	-	1.00	1.00	1.00
求解時間	-	-	-	186.5	413.1	378.5

表 4: SOLAR との比較実験 (SAT Competition 2009，制限時間内に発見できた結論数)

	SOLAR			提案手法		
求解数	5	5	5	43	43	43
平均結論長	9610	8728	14683	2751	5190	4457
求解時間	129.98	22.90	22.75	71.17	7.65	11.69

い結論が多いと考えられ，middle は high と low の結論の長さの中間の結論が多いと考えられる．

鳩の巣問題での SOLAR との比較結果を表 2 に示す．表中の求解数は時間内に全解探索ができた問題数，平均結論長は求めた結論の平均の長さ，時間は全解探索ができた問題の合計時間（秒）を表す．結果から，SOLAR よりも提案手法の方が圧倒的に多くの問題を求解に成功している．SAT Competition の問題での比較結果を表 3 に示す．SOLAR はほとんどの問題において，問題の読み込みが処理時間の大半を占めてしまい，全解探索できた問題はなかった．それに対し，提案手法では，各生成領域において，少なくとも一つは全解探索に成功している．また，制限時間内に発見できた結論数について，表 4 に示す．求解数は時間内に少なくとも一つ以上の結論を導出できた問題の数，平均結論数は求めた結論の平均の長さ，求解時間は各問題において，一つ目の結論を導出できた合計時間 (秒) を表す．結果から提案手法は全問題において，少なくとも一つ以上の結論を発見することに成功している．また，導出した結論も SOLAR よりも短い結論を導出できたことがわかる．

これらの実験結果から，命題論理に特化した結論発見手法を開発することは有用であるといえる．

6. まとめと今後の課題

本論文では，命題論理の結論発見器の実現を目標とし，最新の SAT 技術に基づく命題論理の結論発見器を提案した．我々の手法は，全解列挙が困難な大規模問題においても，結論集合の一部を導出できる可能性が高いため，現実的な結論列挙手法として利用可能である．

今後の課題としては，4.5 節で述べたシンメトリーに基づく枝刈り手法の実装や膨大な結論集合の効率的な表現方法の検討などが挙げられる．また，現システムでは SAT ソルバーとして MiniSat[8] を利用しているため， より充足不能の証明に強いソルバーを導入し比較検討することも課題の 1 つである．

謝辞

本研究の一部は，科学研究費補助金若手研究 (B)(No. 23700164) による．

参考文献

[1] Katumi Inoue. Linear resolution for consequence finding. Artificial Interlligence, Vol. 56, pages 301-353, 1992.

[2] Davis M., Logemann G., and Loveland D. A mashine program for theorem proving. Communicatons of the ACM , 5(7):394-397, 1962.

[3] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an Efficient SAT Solver. In Proceedings of DAC-01, pages 530-535, 2001.

[4] Hidetomo Nabeshima, Koji Iwanuma, Katsumi Inoue, and Oliver Ray. Solar : An automated deduction sysytem for consequence finding. Journal of AI Communications, Vol. 23, No. 2-3, pp183 - 203, 2010

[5] Gomes C. P., Selman B., and Kautz H. A. Boosting combinatorial search through randomiztion. In AAAI-98, pages 431-437, 1998.

[6] J. P. M. Silva and K. A. Sakallash. Grasp-a search algorithm for propositional satisfiability. IEEE Transactions on Computers, 48:506-521,1999.

[7] Laurent Simon and Alvaro del Val. Efficient consequence finding. In Proceedings of IJCAI-01, pages 359-370, 2001.

[8] Niklas Een., Niklas Sorensson. An extensible sat Solver, Proceeding of SAT 2003, pages 502-518, 2003.

[9] Belaid Benhamou. Study of symmetry in constraint satisfaction problems. In Proceedings of CP-94, pages 246-254, 1994.

[10] Giiles Audemard, Laurent Simon. Predicting Learnt Clause Quality in Modern SAT Solvers, IJCAI, Boutilier, C.(ed), pages 399-404, 2009.

[11] 鍋島 英知, 宋 剛秀. 高速 SAT ソルバーの原理, 人工知能学会誌, VOL.25, No.1, pages 68-76, 2010.

[12] Pipatsrisawat, K. and Darwiche, A. A Lightweight Component Cahing Scheme for Satisfiability Solvers, SAT, Marques-Silva, J. and Sakallah, K. A.(eds.), Lecture Notes in Computer Science, Vol. 4501, Springer, pages 294-299, 2007.