人工知能学会研究会資料
SIG-FPAI-B404-13

節集合の簡単化によるMaxSATソルバーの高速化

Improving a MaxSAT solver by simplifying clause set.

上村　直輝[1*]　越村　三幸[2]　長谷川　隆三[2]
Naoki Uemura[1]　Miyuki Koshimura[2]　Ryuzo Hasegawa[2]

[1] 九州大学工学部
[1] School of Enginnering, Kyushu University
[2] 九州大学大学院システム情報科学研究院
[2] Faculty of Information Science and Electrical Enginnering, Kyushu University

Abstract: SATELITE is a preprocess of SAT solver. It eliminates variables and clauses, and decreases runtime of SAT solver. We use it in MaxSAT solver QMaxSAT which uses a normal SAT solver as an inference engine and CNF encodings of Boolean cardinality constraints. We compare QMaxSAT with SATELITE and without SATELITE by solving MaxSAT instances taken from the MaxSAT Evaluation 2014 while changing SAT solver. In this comparison, we use minisat2.0, minisat2.2.0, and glucose3.0 as an inference engine, and the CNF encoding "auto" which selects an appropriate CNF encoding form three encodings.

1　はじめに

充足可能判定問題（Boolean Satisfiability Problem：SAT）とは，与えられた命題論理式の充足可能性を判定する問題であり，最初に NP 完全性が証明された問題である [1]．人工知能や計算機工学における基本的な問題として，スケジュール問題，回路検証問題，制約充足問題などがあり，これらの問題を SAT 問題として解くことができる．SAT 問題を解くためのプログラムを SAT ソルバーと呼び，前述の問題をより多く解けるようにするために SAT ソルバーの性能向上は重要なことである．

SAT 問題の拡張として，MaxSAT 問題がある．SAT 問題が与えられた命題論理式を充足するような変数割り当てを探すのに対して，MaxSAT 問題はできるだけ多くの節を満たすような最適解を求めることを目的としている．MaxSAT 問題には，必ず満たさなければならない節（ハード節）とできるだけ多く満たしたい節（ソフト節）からなる部分 MaxSAT 問題や，各条件に重要度が重みとして付加された重み付き MaxSAT 問題などがある．MaxSAT 問題を解くためのソルバーの一つに QMaxSAT があり，推論エンジンとして SAT ソルバーを使用している．

また, SAT 問題と解くときに事前に問題を簡略化するプリプロセスという処理がある．プリプロセスを行うプログラム（プリプロセッサー）の一つに SATELITE というものがある．本論文では MaxSAT 問題を解くときに SATELITE[2] の有無による性能の比較を行い，MaxSAT 問題において各問題に対する SATELITE の有用性を調べる．

*連絡先：九州大学工学部
〒 819-0395 福岡市西区元岡 774
E-mail:1TE11086R@s.kyushu-u.ac.jp

2　QMaxSAT

QMaxSAT[3] とは，SAT ソルバーを推論エンジンとして利用する，SAT 解に基づく解法を用いた MaxSAT ソルバーである．QMaxSAT は PMS 問題を解くソルバーである．また，近年 WPMS 問題も解けるようになった．

与えられた PMS 問題（WPMS 問題）ϕ の MaxSAT 問題を求めることは，阻止変数 $b_i (i = 1, \ldots, n)$ をソフト節に加えた ϕ' に対して $\sum_{k=1}^{n} b_i < k(\sum_{i=1}^{n} w_i \cdot b_i < k)$ を満たす最小の k を見つけることと等価である．$\sum_{k=1}^{n} b_i < k(\sum_{i=1}^{n} w_i \cdot b_i < k)$ は基数制約と呼ばれ，この制約式を CNF 式へと符号化する手法がいくつも提案されている．QMaxSAT14.08-2.0 では符号化手法として，Warners による符号化手法 [4]，Bailleux らによる符号化手法 [5] ，Asin らによる符号化手法 [6]，Ogawa らによる符号化手法 [7] の４つから指定する手法を用いている [8].

QMaxSAT では，最初のモデルを求めた後に，１度だけ基数制約式の符号化を行う．その後のループで，前

回の k に応じて制約を強めていく．以下に，QMaxSAT の擬似コードを示す．

QMaxSAT の擬似コード

1. $\mathbf{A} = \phi^b$; //ϕ^b：阻止変数を付加した問題
2. $j = 1$; //SAT ソルバーの呼び出しカウンタ
3. **while** (**solve**($\mathbf{A}$)) { //モデル M_j の探索
4. $k_j = \sum_{i=1}^{n} w_i \cdot b_i$; //モデル M_j の下での計算
5. **if** ($j == 1$) $\mathbf{A} = \mathbf{A} \cup \mathbf{Card}$; //$\sum_{i=1}^{n} w_i \cdot b_i$ の SAT 符号化を $\mathbf{A}$ に加える
6. $\mathbf{A} = \mathbf{A} \cup \mathbf{Card}^{<k_j}$; //制約 $\sum_{i=1}^{n} w_i \cdot b_i < k$ の付加
7. $j = j + 1$;
8. }
9. **if** ($j > 1$) **return** M_{j-1};
10. **else return** *unsatisfiable*

以下に，擬似コードの簡単な説明を記載する．

- 阻止変数を付加した問題を用意する（１行目）
- SAT ソルバーを用いてモデルを探す（３行目）
- j 回目に得られたモデルにおいて，阻止変数に１が割り当てられたソフト節の重みの合計 $\sum_{i=1}^{n} w_i \cdot b_i$ を計算し，k_j とする（４行目）
- ループの１度目で基数制約の論理式を符号化し，問題に付加する．符号化の方式は選択した方式によって異なる（５行目）
- 問題に制約 $\sum_{i=1}^{n} w_i \cdot b_i < k$ を付加する．（６行目）
- 擬似コードの３〜７行目を，モデルが得られなくなるまで繰り返す．
- モデルが得られなくなった場合，最後のモデルが最適解となる（９行目）
- もし最初のモデルが得られなかった場合，その問題はハード節を満たすことができないため充足不能と判断する (１０行目)

現在，QMaxSAT では基数制約の CNF 符号化の方式が 4 つ組み込まれている．以下に符号化方式を示す．

- Bailleux らによって提案された基数制約の CNF 符号化で Totalizer と呼ばれる．重み w_i を一進数で符号化，つまり w_i のビット列で符号化し，それぞれを一進加算して総和を求める機構を CNF 符号化している．
- Ogawa らによる符号化で，Module Totalizer と呼ばれる．重み w_i を正整数 p を基数とする modulo 数で符号化し，それぞれトーナメント方式で modulo 加算して総和を求める機構を CNF 符号化している．modulo の商と剰余の加算の符号化に，Bailleux の符号化が用いられている．
- Asín らによる符号化法で，Cardinality Network と呼ばれる．重み w_i を一進数で符号化し，トーナメント方式で併合ソートしていく機構を CNF 符号化している．ここで，併合は基数番目の要素の併合結果と偶数番目の要素の併合結果を，下位の方から順次比較することにより行われる．
- Warners による符号化方式で，Binary Adder と呼ばれる．重み w_i を二進数で符号化し，それぞれをトーナメント方式で二進加算して総和を求める機構を CNF 符号化している．

2.1 生成される変数と節

前述のそれぞれの方法で生成される変数と節の数は次のようになる．

表 1: 変数と節の数

	変数	節
bail	$O(m \cdot \log m)$	$O(m^2)$
ogaw	$O(m \cdot \log m)$	$O(m^{3/2})$
asin	$O(m \cdot \log^2 m)$	$O(m \cdot \log^2 m)$
warn	$O(n \cdot \log w)$	$O(n \cdot logw)$

n：ソフト節の数
m：ソフト節の重さの総和
w：ソフト節の重さの最大値
*ogaw では $p = m^{1/2}$ で計算している

これによると生成される変数の数は $warn \leq ogaw \leq bail \leq asin$ の順であり，生成される節の数は $warn \leq asin \leq ogaw \leq bail$ である．

PMS 問題は各ソフト節の重みが 1 の WPM 問題として考えることができるので，PMS 問題も WPM 問題も同様のソルバーで解くことができる．

2.2 符号化方式の自動選択

QMaxSAT には問題に応じて符号化方式を自動で選択する auto という方式がある．最初に見つけた制約を k，問題の重さの合計を w としたとき，auto では以下の方法で符号化方式を決定する．

1. *if* $(\lceil \log_2 k \rceil + \lceil \log_2 w \rceil < 15)$
2. $card = bail$;
3. *else if* $(k < 3)$
4. $card = warn$;
5. *else if* $(\lceil \log_2 w \rceil < 17)$
6. $card = ogaw$;
7. *else*
8. $card = warn$;

3 SATELITE

SATELITE[2] とは，SAT ソルバーのプリプロセスにおいて処理を行うのプリプロセッサーの一つで，問題を解くときに 1 度だけ処理を行う．SATELITE は与えられた問題に対して，変数や節の数を減らすことで問題を簡単にし，全体の計算時間を短縮することを目的としている．ここでは，SATELITE の特徴について説明していく．

3.1 Variable Elimination

Variable Elimination とは，SATELITE において変数を削除するプロセスのことである．

与えられた CNF 式に x を含む節 $C_1 = \{x, a_1, \ldots, a_n\}$ と $\neg x$ を含む節 $C_2 = \{\neg x, b_1, \ldots, b_m\}$ があるとする．このとき，C_1 と C_2 から導出節 (resolvent) $C = \{a_1, \ldots, a_n, b_1, \ldots, b_m\}$ を導くことができる．この論文では，この導出関係を $C = C_1 \otimes C_2$ と表すことにする．この導出関係は節の集合同士でも適用できる．S_1 を x を含む節の集合とし，S_2 を $\neg x$ を含む節の集合とする．このとき，$S_1 \otimes S_2$ は次の 1 式のようになる．

$$S_1 \otimes S_2 = \{C_1 \otimes C_2 | C_1 \in S_1, C_2 \in S_2\} \qquad (1)$$

この関係式を用いることで，CNF 式に含まれる変数を削減することができる．ある CNF 式が与えられたとき，x を含む節の集合を S_x とし，$\neg x$ を含む節を $S_{\neg x}$ とし，$S = S_x \cup S_{\neg x}$ とする．このとき，$S' = S_x \otimes S_{\neg x}$ とすると，S と S' を入れ替えて推論を行うことができる．こうすることで，CNF 式に含まれる変数 x を削除することができる．

3.2 Subsumption

Subsumption とは，SATELITE において節を削減するプロセスの一つである．

ある CNF 式の節 C_1, C_2 が $C_1 \subseteq C_2$ の関係であるとき，C_1 が *True* となるときは必ず C_2 も *True* となる．そのため，すべての節を *True* にする SAT の推論において C_1 を考えていれば C_2 を考えなくてもよくなる．このとき，C_1 は C_2 を包摂する (subsume) と言い，CNF 式から C_2 を取り除くことができる．

3.3 Self-Subsumption

Self-Subsumption も Subsumption と同様に節を削減するプロセスの一つである．この処理は Variable Elimination と Subsumption を組み合わせることで起こる．

例えば，$C_1 = \{x, a, b\}$，$C_2 = \{\neg x, a\}$ の二つの節があるとする．この二つの節の resolvent をとると，$C_1{}' = \{a, b\}$ を導くことができる．このとき，C_1 と $C_1{}'$ に着目すると，$C_1{}' \subseteq C_1$ であるため $C_1{}'$ は C_1 を subsume することができる．そのため，CNF 式に $C_1{}'$ を加えると C_1 を取り除くことができる．このように，他の節との resolvent をとった節を使って自身を subsume するような処理を Self-Subsumption と呼び，C_1 は C_2 を使って Self-Subsumption を行って強化された，という．

3.4 論理ゲートによる節削除

CNF 式の中に含まれる節を組み合わせて論理ゲートを作れる場合がある．例えば，CNF 式に（2）式のような節があるとする．

$$\ldots\{x, \neg a, \neg b\}, \{\neg x, a\}, \{\neg x, b\}\ldots \qquad (2)$$

これらの節から AND ゲート $x = (a \wedge b)$ を取り出すことができる．3.1 節，3.2 節，3.3 節で述べたプロセスにこの論理ゲートの特徴を組み合わせることにより，さらに CNF 式の節を削減することができる．

G を論理ゲートを作れる節の集合とする．その中で x を含む節を G_x とし，$\neg x$ を含む節を $G_{\neg x}$ とする．$G = G_x \cup G_{\neg x}$ となる．また，S を変数 x を含む節の集合とし，R を S から G を取り除いた部分であるとする．つまり，$S = G \cup R$ となる．この G と R を x を含む部分と $\neg x$ を含む部分に分けると $S = (G_x \cup R_x) \cup (G_{\neg x} \cup R_{\neg x})$ となる．3.1 節で述べたように $S = S_x \cup S_{\neg x}$ は $S' = S_x \otimes S_{\neg x}$ と置き換えることができることにより，S' について考えればよい．$S' = S_x \otimes S_{\neg x}$ を整理すると，$S' = S'' \cup G' \cup R'$ となる．ただし，S'', G', R' はそれ

表 2: 実験環境

CPU	Intel Xeon CPU E5-2680 v2 @2.80GHz
メモリ	6.2GB
OS	linux
制限時間	1800 秒（1 問あたり）

表 3: 実験に使用した問題（MaxSAT Evaluation 2014[12] より）

種類（略称）	問題数
部分 MaxSAT-crafted（pms_crafted）	421
部分 MaxSAT-industrial（pms_industrial）	568
重み付き部分 MaxSAT-crafted（wpms_crafted）	310
重み付き部分 MaxSAT-industrial（wpms_industrial）	410

ぞれ以下のようになる．

$$S^{''} = (R_x \otimes G_x) \cup (G_{\neg x} \otimes R_{\neg x}),$$
$$G^{'} = G_x \otimes G_{\neg x}, R^{'} = R_x \otimes R_{\neg x} \quad (3)$$

さらに，定理 $S^{''} \models G^{'} \cup R^{'}$[9] を用いることにより，$S^{'}$ は $S^{''}$ と置き換えることができる．$|S^{'}| > |S^{''}|$ であるため，単純に resolvent をとった $S^{'}$ より節の数を削減できる．

4 実験

本章では SATELITE を MaxSAT ソルバーで使用した場合と使用していない場合での性能比較を行う．使用した MaxSAT ソルバーは QMaxSAT であり，推論エンジンとして minisat2.0[10], minisat2.2.0[10], glucose3.0 [11] を使用している．表 2 に実験環境を，表 3 に使用した問題を示す．

表 4 に符号化方式に auto を使用した時の実験の結果を示す．上から順に minisat2.0，minisat2.2.0，glucose3.0 を推論エンジンとしたときの結果を表す．none の列は SATELITE を使用していない時の結果を，once の列は SATELITE を使用した時の結果を表す．また，表の中の数字はそれぞれの方式で解けた問題数を表す．minisat2.0 での pms_industrial では 11 問, minisat2.2.0 の pms_crafted では 4 問多くの問題を解いている．全体的に解答数が少なくなっているほうが多い．

表 4: 実験結果

問題の種類	none	once
（minisat2.0）		
pms_crafted	319	319
pms_industrial	367	378
wpms_crafted	147	146
wpms_industrial	159	158
（minisat2.2.0）		
pms_crafted	310	314
pms_industrial	443	433
wpms_crafted	156	154
wpms_industrial	296	297
（glucose3.0）		
pms_crafted	320	319
pms_industrial	456	456
wpms_crafted	181	182
wpms_industrial	308	300

表 5: SATELITE により節の数が減った問題の数

pms_crafted	161
pms_industrial	500
wpms_crafted	120
wpms_industrial	410

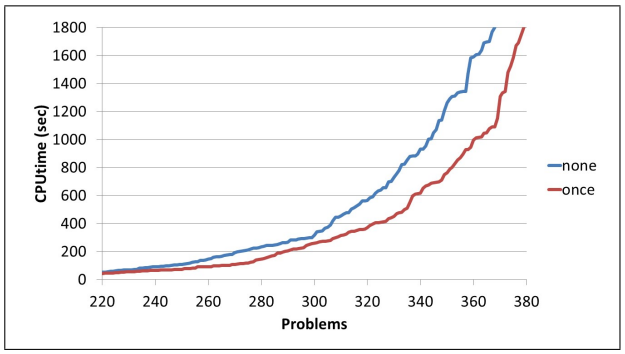

図 1: minisat2.0 での pms_industrial の推論時間と解けた問題数

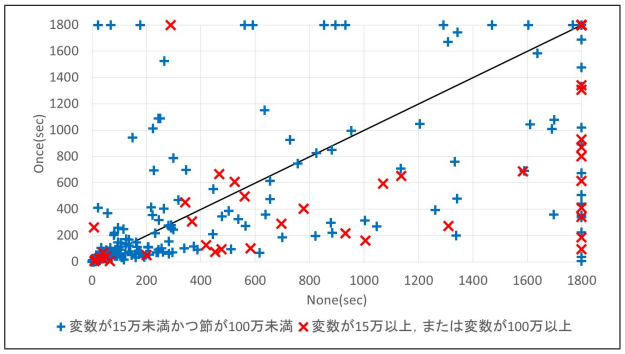

図 2: minisat2.0 での pms_industrial の同じ問題に対する推論時間の比較

問題によっては SATELITE を使用しても変化がないものがある．表 5 に SATELITE によって節の数が減少した問題の数を示す．pms_crafted や wpms_crafted では 3 割〜4 割の問題でしか簡単化を行っていない．pms_industrial では 9 割近くの問題を，wpms_industrial ではすべての問題で簡単化を行っている．この結果から crafted 分野よりも industrial 分野のほうが SATELITE による影響が大きくなることがわかる．

次に実験結果のグラフを示す．グラフの特徴はすべての問題の種類に対して同様であるため，結果の良かった minisat2.0 での pms_industrial のグラフだけを示す．図 1 は minisat2.0 での pms_industrial の問題数と推論時間の関係をグラフにしたものである．グラフは推論時間でソートしてある．横軸は問題数を表し，縦軸は推論時間（単位は秒）を表す．例えば 400 秒以内に解いた問題数を見てみると，SATELITE を使わない場合は約 300 問の問題を解いたのに対して，SATELITE を使った場合は約 320 問の問題を解いている．このグラフでは右側に来ているもののほうが性能が良いことを表すので，SATELITE を使用した場合のほうが結果が良かったことを示している．

図 2 は同じ問題に対する推論時間の比較のグラフである．横軸は SATELITE を使わなかった場合の推論時間（単位は秒）を表し，縦軸は SATELITE を使った場合の推論時間（単位は秒）を表す．グラフの右端にある点は，SATELITE を使わなかった場合は制限時間以内に解けなかった問題であるが，SATELITE を使った場合は解けるようになった問題のことを表す．逆に，グラフの上端にある点は SATELITE を使わなかった場合は制限時間以内に解けた問題であるが，SATELITE を使うことにより制限時間以内に解けなくなった問題のことを表す．グラフの右下に来ているものが多いほど良い結果となったことを表す．また，グラフの赤い点は問題の変数の数が 15 万以上または節の数が 100 万以上のものを表し，グラフの青い点は問題の変数の数が 15 万未満かつ節の数が 100 万未満の問題を表す．

グラフの上端にある問題の数は 13 問であるのに対して，グラフの右端に来ている問題の数は 24 問であり，結果として 11 問多くの問題を解いたことになる．また，グラフの右下にある問題の数のほうが多いので全体的に推論時間が早くなっていることがわかる．一方，問題の変数や節の数に着目すると変数の数が 15 万以上または節の数が 100 万以上のものはグラフの右下に多く集まっており，またグラフの上端にある問題は 13 問中 1 問となっている．この結果から，変数や節の数によって SATELITE が有効に働く場合と悪影響を与える場合があることがわかる．

5 おわりに

SATELITE により問題の簡単化を行えたものは industrial 分野に多くあった．改良を加えることにより industral 分野での性能向上が期待できる．SATELITE による効果が一番大きかったのは minisat2.0 での pms_industrial であった．minisat2.2.0 や glucose3.0 では良い結果が得られなかったが，改良を加えることにより高速化を図ることができる．問題によって SATELITE が有効に働く場合と悪影響を与える場合があるので，問題によって SATELITE を使い分けるソルバーを開発すればさらに良い結果が得られると考えられる．今後の研究課題としては，問題の特徴を解析し SATELITE を使い分けるソルバーの開発を行うことである．

謝辞

本研究は JSPS 科研費 25330085 の助成を受けたものです．

参考文献

[1] Stephen A.Cook.“The complexity of theorem-proving procedures”, In proceedings of the Third IEEE Symposium on the Foundations of Computer Science, page 151-158, 1971.

[2] Niklas Eén and Armin Biere. Effective Preprocessing in SAT through Variable and Clause Elimination.Theory and Applications of Satisfiability Testing Lecture Notes in Computer Science Volume 3569, 2005, pp 61-75.

[3] Miyuki Koshimura, Tong Zhang, Hiroshi Fujita, and Ryuzo Hasegawa. QMaxSAT: A Partial Max-SAT Solver: JSAT, Vol.8(2012), system description, pages 95-100.

[4] Joost P. Warners. A linear-time transformation of linear inequalities into conjunctive normal form. *Information Processing Letters*, 68:63-69, 1998.

[5] Olivier Bailleux and Yacine Boufkhad. Efficient CNF Encoding of Boolean Cardinality Constraints. In *Proceedings of 9th International Conference on Principles and Practice of Constraint Programming (CP 2003)*, pages 108-122, 2003.

[6] Roberto Asín, Robert Nieuwenhuis, and Albert Oliveras. Cardinality Networks: a theoretical and empirical study. *Constraints*, 16:195-221, 2011.

[7] Toru Ogawa, YangYang Liu, Ryuzo Hasegawa, Miyuki Koshimura, and Hiroshi Fujita. Modulo Based CNF Encoding of Cardinality Constraints and Its Application to MaxSAT Solvers. In *Proceedings of IEEE 25th International Conference on Tools with Artificial Intelligence (ICTAI 2013)*, pages 9-17, 2013.

[8] 越村三幸, 有村寿高. 基数制約の SAT 符号化を用いた MaxSAT ソルバーの試作. 人工知能学会全校大会論文集 28, 1-4, 2014.

[9] Éric Grégoire, Richard Ostrowski, Bertrand Mazure, and Lakhdar Saïs. Automatic extraction of functional dependencies. In Theory and Applications of Satisfiability Testing Lecture Notes in Computer Science Volume 3542, 2005, pp 122-132.

[10] The MiniSat page. http://minisat.se/Main.html

[11] Audemard, G. Simon, L.: GLUCOSE: a solver that predicts learnt clauses quality, SAT 2009 competitive events booklet, preliminary version, pp.7-8 (2009)

[12] Max-SAT 2014 Ninth Max-SAT Evaluation. http://www.maxsat.udl.cat/14/index.html