成蹊大学理工学研究報告
J. Fac. Sci.Tech., Seikei Univ.
Vol.51 No.2 (2014)pp.19-22

最大充足可能性問題の疎な例題に対する厳密アルゴリズム

脊戸　和寿*

Solving Sparse Instances of Max SAT via Width Reduction and Greedy Restriction

Kazuhisa SETO*[1]

ABSTRACT：We present a moderately exponential time polynomial space algorithm for sparse instances of Max SAT. For instances with n variables and cn clauses, our algorithm runs in time $O(2^{(1-\mu(c))n})$, where $\mu(c) = O(1/c^2\log^2 c)$. Previously, an exponential space algorithm with $\mu(c) = O(1/c\log c)$ was shown by Dantsin and Wolpert [SAT 2006] and a polynomial space algorithm with $\mu(c) = O(1/2^{O(c)})$ was shown by Kulikov and Kutzkov [CSR 2007]. Our algorithm is based on the combination of two techniques, width reduction of Schuler and greedy restriction of Santhanam.

Keywords：exponential time algorithm, polynomial space, satisfiability

(Received March 25, 2005)

1．はじめに

最大充足可能性問題(Max SAT)とは, 入力として与えられる節集合に対し，充足可能な節の最大数を求める問題である．

節とはリテラルの論理和であり，リテラルとはブール変数とその否定である．また各節が高々k個のリテラルしか含まないものをMax k-SATと呼ぶ．これらの問題は代表的なNP困難問題の 1 つである．n変数，m節からなるMax SATのインスタンスが与えられたとき，自明に$O(m2^n)$時間で解ける．我々の目標は，Max SATをある絶対定数$\mu>0$に対して, $O(\mathrm{poly}(m)2^{(1-\mu)n})$時間で解くことである．しかし，この目標達成はmに制限がないと非常に困難である．そこで我々はm=cnに制限した疎な例題に焦点をあてる．Max SATの疎な例題に対しては既に, DantsinとWolpertによって$\mu(c)= \Omega(1/(c\log c))$の指数領域アルゴリズムが示されている [DW06]．また，KulikovとKutzkovによって$\mu(c)= \Omega(1/2^{O(c)})$の多項式領域アルゴリズムが示されている[KK07]．本稿では，これらの結果を改良し，$\mu(c)= \Omega(1/c^2\log^2 c)$の多項式領域アルゴリズムを与える．

*： 理工学部情報科学科助教 (seto@st.seikei.ac.jp)

1．1　アルゴリズムに用いる手法

まず，Santhanamのformula SATに対するgreedy restriction algorithmを用いて，Max k-SATが$\mu(c)= \Omega(1/k^2c^2)$時間で解けることを示す．次に，Max SATのインスタンスをMax k-SATのインスタンス集合に変形する．このとき，元のインスタンスの最適解はいずれかのMax k-SATのインスタンスの最適解に保存される必要がある．そのために，Schulerのwidth reductionを用いる [Sch05]．これらを組み合わせることで，所望の結果を得る．

2．準備

$V = \{x_1,\ldots,x_n\}$をブール変数集合とする．ブール値の`true'を 1，`false'を 0 で表現する．変数$x \in V$の否定をx'と表記する．リテラルは変数またはその否定である．

k-constraintはブール関数$\phi:\{0,1\}^k \to \{0,1\}$であり, Vのk変数に依存する．0-constraintは`0'または`1'の定数関数を表す．Max CSPのインスタンスΦは制約と重みで構成される．各ϕ_iがk_i-constraintでw_iが正の数として，$\Phi=\{(\phi_1, w_1),\ldots,(\phi_m, w_m)\}$と表す．$\Phi$の幅を$\max_i k_i$とする．$\mathrm{Val}(\Phi, a):=\Sigma_{i\in[m]} w_i \cdot \phi_i(a)$とし，$\mathrm{Opt}(\Phi):=\max_{\{a \in \{0,1\}n\}}\mathrm{Val}(\Phi, a)$とする．$y_1,\ldots,y_l$を任意のリテラルとすると，Max SATは, 各制約が$y_1 \vee \cdots \vee y_l$の形で表される．また，インスタンスには，0-constraintsが存在することを許す．

３．Max formula SATに対するGreedy Restriction Algorithm

本章では，各制約がド・モルガン式で与えられるMax CSPに対するアルゴリズムを導入する．これは，Max k-SATも解くことができる．

3．1　ド・モルガン式とその単純化規則

ド・モルガン式は，葉がリテラル$\{0,1, x_1,\ldots,x_n, x_1',\ldots,x_n'\}$により，内部ノードが∨または ∧によりラベル付けされた 2 分木で表現される．ド・モルガン式 ϕが与えられたとき，ϕの部分式とはϕの部分木をいう．ϕのサイズは，葉の数で定義し，$L(\phi)$と表記する．$var(\phi)$をϕにリテラルとして存在する変数の集合とする．ϕに存在する変数xの頻度を，xまたはx'でラベル付けされた葉の数で定義し，$freq_\phi(x)$と表記する．

式ϕ，変数集合$\{x_{i1},\ldots,x_{ik}\}$，定数$a_1,\ldots,a_k \in \{0,1\}$に対し，$x_{ij}$，$x_{ij}'$にそれぞれ，$a_j$， a_j'を割り当て，さらに図 1 の手続きSimplifyを適用することにより，ϕから得られる式を$\phi[x_{i1}= a_1,\ldots, x_{ik} = a_k]$と表記する．この手続きにより，式のサイズは小さくなる．これは，[Has98, San10]で用いられた単純化規則と同じである．Simplifyは，ϕのサイズの多項式時間で動作し，単純化により得られる式はϕと同じ関数を計算する．

```
Simplify(φ : formula)
Repeat the following until there is no decrease in size of φ.
(a) If 0 ∧ φ (0 ∨ φ) occurs as a subformula, where
    φ is any formula, replace this subformula by 0 (φ).
(b) If 1 ∧ φ (1 ∨ φ) occurs as a subformula, where
    φ is any formula, replace this subformula by φ (1).
(c) If y ∨ φ (y ∧ φ) occurs as a subformula, where
    φ is a formula and y is a literal, then replace all occurrences
    of y in φ by 0 (1) and all occurrence of y' by 1 (0).
```

図１　単純化規則

3．2　Max formula SATアルゴリズム

Max formula SATは各制約がド・モルガン式であるMax CSPである．つまり，各ϕ_iがド・モルガン式で与えられる．$var(\Phi):=\cup_{i\in[m]} var(\phi_i)$, $L'(\Phi):=\Sigma_{\{i:L(\phi i)\geqq 2\}} L(\phi_i)$，そして$freq'_\Phi(x):= \Sigma_{\{i:L(\phi i)\geqq 2\}} freq'_{\Phi i}(x)$と定義する．
任意のインスタンスΦ，任意の変数集合$\{x_{i1},\ldots,x_{ik}\}$，任意の割当$a_1,\ldots,a_k \in \{0,1\}$に対して，$\Phi[x_{i1}=a_1,\ldots,x_{ik}=a_k]:=\{(\phi_{1'}, w_1),\ldots,(\phi_{m'}, w_m)\}$と定義する．ここで，$\phi_{i'}= \phi_i[x_{i1}=a_1,\ldots,x_{ik}=a_k]$である．図 2 はMax formula SATアルゴリズムである．

```
EvalFormula(Φ={(φ1, w1),…,(φm, wm)}):instance, n: integer)
if  L'(Φ)=cn < 3n/4,
    compute Opt(Φ) and return Opt(Φ).
else
    x=argmax_{x∈var(Φ)} freq'_Φ(x).
    K0 ← EvalFormula(Φ[x=0], n-1).
    K1 ← EvalFormula(Φ[x=1], n-1).
    return max{ K0, K1}.
```

図２　Max formula SATアルゴリズム

アルゴリズムの計算時間は以下の通りである．
定理１．n変数，$L'(\Phi)$=cnのMax formula SATのインスタンスΦが与えられたとき，EvalFormulaは，$Opt(\Phi)$を$O(poly(n)2^{(1-\mu(c))n})$時間で計算する．ここで，$\mu(c)=1/32c^2$ である．また，n変数，cn節のMax k-SATのインスタンスΦの最適解$Opt(\Phi)$を$O(poly(n)2^{(1-\mu(c))n})$時間で計算する．ここで，$\mu(c)=1/32k^2c^2$ ある．

４．Max SATアルゴリズム

この章では，主結果であるMax SATの疎なインスタンスに対するアルゴリズムを示す．
アルゴリズムは以下である．

```
MaxSAT(Φ={(φ1, w1),…,(φm, wm)}: instance, k, n: integer)
if ∀ φi ∈Φ, |var(φi)| ≦ k,
    return  EvalFormula(Φ, n).
else
   Pick arbitrary Φi=(l1 ∨ … ∨ lk') such that k'>k.
   ΦL ← {Φ\{(φi, wi)}}∪{(l1 ∨ … ∨ lk),w_i}}.
    KL ← MaxSAT(ΦL, k, n).
   ΦR ← Φ[ l1 =…= lk=0].
    KR ← MaxSAT(ΦR, k, n).
    return max{ KL, KR }.
```

図３　Max SATアルゴリズム

以下は本稿における主定理である．
定理２．n変数，cn節のMax SATのインスタンスΦが与えられたとき，MaxSATは，$Opt(\Phi)$を$O(poly(n)2^{(1-\mu(c))n})$時間で計算する．ここで，$\mu(c)=1/c^2 \log^2 c$である．

参考文献

[DW06] E. Dantsin, A. Wolpert : “MAX-SAT for formulas with constant clause density can be solved faster than in O(2^n) time”, In Proc. of SAT, LNCS 4121, pp. 266-276, 2006

[Has98] J. Hastad: “The Shrinkage Exponent of De Morgan Formulas is 2”, SIAM J. Comput”, 27(1):48--64, 1998

[KK07] A. S. Kulikov and K. Kutzkov: “New Bounds for MAX-SAT by Clause Learning”, In Proc. of CSR, LNCS 4649, pp.194-204, 2007

[San10] R. Santhanam: “Fighting Perebor: New and Improved Algorithms for Formula and QBF Satisfiability”, In Proc. of FOCS, pp.183-192, 2010.

[Sch05] R. Schuler: “An algorithm for the satisfiability problem of formulas in conjunctive normal form”, Jornal of Algorithms, 54:40-44, 2005.