3J1-OS7-3

QMaxSAT: Q-dai MaxSAT ソルバー

QMaxSAT: Q-dai MaxSAT Solver

越村 三幸[*1] 安 宣燁[*2] 藤田 博[*1] 長谷川 隆三[*1]
Miyuki Koshimura Xuanye An Hiroshi Fujita Ryuzo Hasegawa

[*1]九州大学 大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University

[*2]九州大学 大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University

We present a partial MaxSAT solver QMaxSATwhich uses CNF encoding of Boolean cardinality constraints. The old version 0.1 was obtained by adapting a CDCL based SAT solver MiniSat to manage cardinality constraints. It was placed first in the industrial subcategory and second in the crafted subcategory of partial MaxSAT category of the 2010 Max-SAT Evaluation. This paper presents new versions 0.2, 0.3, and 0.4. The version 0.2 is obtained by modifying version 0.1 to decrease the number of clauses for the cardinality encoding. The version 0.3 searches a solution by a binary method while both versions 0.1 and 0.2 do by a linear method. The version 0.4 alternates the linear search and the binary search. We compare the four versions by solving MaxSAT instances taken from the 2010 Max-SAT Evaluation.

1. はじめに

SAT（Boolean satisfiability testing）は，ブール式の充足可能性判定であり，最初に NP 完全性が証明されたことで知られる [井上 10]．SAT ソルバーの研究が近年大きく進展したことを受けて，SAT の枠組みを拡張する様々な試みが盛んに行われている．MaxSAT は，SAT を最適化問題に拡張したものである．SAT が制約を満たす「解」を求めるのに対し，MaxSAT は「最適解」を求める．

SAT 問題は，連言標準形（CNF: Conjunctive Normal Form）で表される命題論理式（CNF 式）で与えられるのが一般的である．CNF 式は，節（clause）の連言であり，節は，リテラル（literal）の選言である．リテラルは，変数またはその否定である．変数への値割当のうち，全ての節を充足するような値割当が SAT 解であるのに対し，充足する節の数が最大となる値割当が MaxSAT 解である [Li 09, 平山 10]．したがって，SAT 解があれば，それは MaxSAT 解でもある．一方，SAT 解がない（つまり充足不能な）場合でも，MaxSAT 解はあり，それは，問題がどの程度，充足不能なのかを示している，と考えることができる．

MaxSAT の解法には，近似解法と厳密解法があるが，本論文では，厳密解法のソルバーを扱う．厳密解法のソルバーは大きく，分枝限定法を用いるもの [Pipatsrisawat 07, Heras 08b, Lin 08] と通常の SAT ソルバーを推論エンジンとして用いるもの [Fu 06] に分けられる．後者の特徴は，系統的 SAT ソルバーを繰り返し適用して MaxSAT の解を求めることであり，本論文で述べる QMaxSAT もこれに属し，SAT ソルバー MiniSat 2.0 [Eén 03] を利用している．

SAT ソルバーを利用する MaxSAT ソルバーはさらに，**SAT 解に基づくもの** [Berre 10] と **UNSAT 部分集合に基づくもの** [Ansótegui 09, Manquinho 09] に分類される．QMaxSAT は，前者に属する．SAT 解に基づく（あるいは UNSAT 部分集合に基づく）MaxSAT ソルバーは，MaxSAT 問題から充足可能（あるいは充足不能）な SAT 問題の系列を作り出す．系列中の各 SAT 問題は，直前の SAT 問題を基に生成される．この系列は，充足不能（あるいは充足可能）な SAT 問題で終了する．系列中の最後の充足可能な SAT 問題の SAT 解が，MaxSAT 解と見なせる．

MaxSAT は，重みなし MaxSAT（unweighted MaxSAT），重みなし部分 MaxSAT（unweighted partial MaxSAT），重み付き MaxSAT（weighted MaxSAT），重み付き部分 MaxSAT（weighted partial MaxSAT）に分類される．場合によっては，修飾語である「重みなし」は省略されることがある．QMaxSAT は，重みなし部分 MaxSAT 用のソルバーである．

本論文では，４つの版，第 0.1，0.2，0.3，0.4 版を紹介し，性能比較を行う．第 0.1 版と第 0.2 版は，線形探索を行うのに対し，第 0.3 版は，二分探索を行う．そして，第 0.4 版は，線形探索と二分探索を交互に行う．QMaxSAT は，基数制約（cardinality constraints）を利用して MaxSAT 解を求めるソルバーで，第 0.1 版は，昨年開催された第 5 回 Max-SAT evaluation（Max-SAT 2010）の重みなし部分 MaxSAT 部門（industrial）で優勝，同部門（crafted）で準優勝した．Max-SAT evaluations [Argelich 08, Heras 08a] は 2006 年から毎年開催されている厳密解法の性能を競う競技会である．

以下，2. では，重みなし部分 MaxSAT とその基数制約を用いた解法を概説した後，QMaxSAT の各版について述べる．3. では実験結果を報告し，4. ではまとめと今後の課題を示す．

2. QMaxSAT

CNF 式に対する重みなし部分 MaxSAT（以下，PMS: Partial MaxSAT）問題は，ハード節（hard clause）とソフト節（soft clause）からなる節集合として与えられる．その目的は，全てのハード節を満たし，できるだけ多くのソフト節を満たすような命題変数への値割当を求める，ことである．

さて，C をハード節の集合 H とソフト節の集合 S からな

連絡先: 越村 三幸，九州大学大学院システム情報科学研究院，福岡市西区元岡 744，092-802-3599，092-802-3600，koshi@inf.kyushu-u.ac.jp

る PMS 問題とする．(つまり，$C = H \cup S$ である．) S が n 個のソフト節からなると仮定すると，$S = \{S_1, \ldots, S_n\}$ と書ける．この時，n 個の阻止変数（blocking variable）と呼ばれる新しい論理変数 $b_i (i = 1, \ldots, n)$ を導入して，新しい節集合 $C^b = H \cup S^b$ を考える．ここで，$S^b = \{S_1 \vee b_1, \ldots, S_n \vee b_n\}$ である．こうすると，C の PMS 解を求めることは，C^b を満たすモデルの中で，$\sum_{i=1}^{n} b_i \leq k$ を満たす最小の整数 k を求めることと等価になる．

制約 $\sum_{i=1}^{n} b_i \leq k$ (あるいは，$l \leq \sum_{i=1}^{n} b_i$) は，基数制約と呼ばれ，これを CNF に符号化する手法が幾つも提案されている [Bailleux 03, Sinz 05, Marques-Silvar 07]．QMaxSAT は，Bailleux による符号化 [Bailleux 03] を用いている．この符号化では，n 個の阻止変数 $b_i (1 \leq i \leq n)$ に対して，n 個の新たな命題変数 $v_i (1 \leq i \leq n)$ を用意する．そして，これらを含む 新たな CNF 表現の論理式 $C(b_1, \cdots, b_n, v_1, \cdots, v_n)$ を作る．この論理式は，次のような性質を持つ：

1. m 個の阻止変数に 1 が割り当てられると，始めの m 個の $v_i (1 \leq i \leq m)$ に 1 が割り当てられる．
2. m 個の阻止変数に 0 が割り当てられると，終わりの m 個の $v_{n-i+1} (1 \leq i \leq m)$ に 0 が割り当てられる．

この論理式は，$O(n \cdot log\ n)$ 個の中間変数を含み，$O(n^2)$ 個の節からなる．この論理式を利用すると，基数制約 $l \leq \sum_{i=1}^{n} b_i \leq k$ は，始めの l 個の $v_i (1 \leq i \leq l)$ を 1 にし，終わりの $n - k$ 個の $v_i (k < i \leq n)$ を 0 にすることで得られる．

2.1 第 0.1 版

Algorithm 1 に QMaxSAT の手続きを擬似コードで示す．関数 `solve(A)` は，SAT ソルバー呼出しを表し，SAT 問題 `A` が充足可能なら `true` を充足不能なら `false` を返す．充足可能の場合，配列に `A` のモデル M が格納されるものとする．この配列を走査することにより，M で 1 を割当てられている阻止変数の個数 k を数えることができる．

Algorithm 1 QMaxSAT 第 0.1 版

```
1: A = C^b; {C^b : 阻止変数を付加した問題 }
2: sat = false; first = true;
3: while (solve(A)) do
4:    "M を A のモデルとする";
5:    "M 中で 1 と割当てられた阻止変数の数を k とおく";
6:    sat = true;
7:    if (first) then
8:       first = false;
9:       A = A ∧ C(b_1, ···, b_n, v_1, ···, v_n); { 基数制約を付加 }
10:   end if
11:   for i = k to n do
12:      v_i = 0;
13:   end for{ 制約 Σ_{i=1}^{n} b_i < k の付加 }
14: end while
15: if (sat) then
16:   return 最後のモデル M;
17: else
18:   return 充足不能;
19: end if
```

C^b の最初のモデルが見つかったら，基数制約を課すために論理式 $C(b_1, \cdots, b_n, v_1, \cdots, v_n)$ を作る（9 行目）．`solve(A)` の呼び出しでモデルが得られるたびに，1 を割当てられている阻止変数の数 k を数え，基数制約を更新する（11〜13 行目）．手続きが進むにつれ，k の値は減少していくことに注意されたい．したがって，手続きはいつかは停止し，最後のモデルが C の PMS 解となる（16 行目）．なお，基数制約を課さない C^b が充足不能であれば，元の C も充足不能である（1 8 行目）．

2.2 第 0.2 版

基数制約の CNF 符号化を利用した Max-SAT ソルバーの弱点は，符号化に必要な節の数が膨大になることである．ソフト節の数が数万個あると仮定すると，Bailleux の符号化では，数億個もの節が必要となる．メモリ 4GB の計算機を使った我々の実験では，ソフト節の数が 9 千を超えるような問題では，符号化の途中でメモリ溢れが発生した．第 0.2 版では，Bailleux の符号化を用いつつ，必要な節の数を削減している．

Bailleux の符号化は，次のような節の連言を含む：

$$\bigwedge_{\substack{0 \leq \alpha \leq \lfloor n/2 \rfloor \\ 0 \leq \beta \leq \lceil n/2 \rceil \\ \alpha + \beta = \sigma \\ 0 \leq \sigma \leq n}} (C_1(\alpha, \beta, \sigma) \wedge C_2(\alpha, \beta, \sigma))$$

ここで，n はソフト節の数，$C_1(\alpha, \beta, \sigma)$ と $C_2(\alpha, \beta, \sigma)$ は，それぞれ，$\alpha + \beta \leq \sigma$ と $\sigma \leq \alpha + \beta$ を表す CNF 表現である．さて，QMaxSAT の手続き（Algorithm1）において最初に得られたモデルで，1 を割当てられた阻止変数の数を k_1 としよう．このとき，$k_1 < \sigma \leq n$ に対する $C_1(\alpha, \beta, \sigma)$ と $C_2(\alpha, \beta, \sigma)$ は無意味なので，不要である．第 0.2 版では，不要なこれらの節を作らない．これによって，符号化に必要な節の数は，$O(n^2)$ から $O(n \cdot k_1)$ に削減される．

2.3 第 0.3 版

第 0.1 版と 0.2 版は共に，SAT ソルバーでモデルが得られるとそのモデル中で 1 と割当てられている阻止変数の個数を数える．この個数は，手続きが進むにつれて減少するが，その減少数は，最悪の場合,「一つずつ」である．この点で，両版ともに線形探索を行っている，といえる．第 0.3 版では，モデルで 1 と割当てられている阻止変数の個数の下界と上界を扱うことにより，二分探索を実現する．第 0.1 版と 0.2 版では，上界のみを扱っていたことに注意されたい．

Algorithm 2 に第 0.3 版の手続きを擬似コードで示す．関数 `solve(A,Ass)` は，SAT ソルバーの呼出しを表し，仮定 `Ass` の下で，`A` が充足可能なら `true` を充足不能なら `false` を返す．`LB` は下界，`UB` は上界，`MP` は `LB` と `UB` の中間値を表す．`MP` は，`LB` あるいは `UB` が更新されるたびに更新される（10，19 行目）．SAT ソルバーは，基数制約 $LB \leq \sum_{i=1}^{n} b_i < MP$ の下で呼び出される．最初，下界は 0，上界は n（ソフト節の個数）に初期化される（3 行目）．また，仮定は $\top$（4 行目），つまり何も仮定しない．

`solve(A,Ass)` 呼出しでモデルが得られるたびに，1 を割当てられている阻止変数の数 k を数え，それによって上界 `UB` を更新する（10 行目）．`solve(A,Ass)` が `false` を返した場合，$LB \leq \sum_{i=1}^{n} b_i < MP$ を満たすモデルがないことを意味するので，下界 `LB` をそれまでの `MP` に更新する（19 行目）．下界もしくは上界が更新されたら，それに対応するように制約を更新する（21〜27 行目）．上界と下界の差は，ループを繰り返すたびに，以前の半分以下になる．こうして，二分探索が実現される．

Algorithm 2 QMaxSAT 第 0.3 版

```
1: A = C^b; {C^b : 阻止変数を付加した問題 }
2: sat = false; first = true;
3: LB = 0; UB = n; MP = ⌈UB/2⌉; {n: ソフト節の個数 }
4: Ass = ⊤; { 何も仮定しない }
5: while LB < UB do
6:   if solve(A,Ass) then
7:     "M を A のモデルとする";
8:     "M 中で 1 と割当てられた阻止変数の数を k とおく";
9:     sat = true;
10:    UB = k; MP = LB + ⌈(UB − LB)/2⌉;
11:    if (first) then
12:      first = false;
13:      A = A ∧ C(b_1, ··· , b_n, v_1, ··· , v_n); { 基数制約を付加 }
14:    end if
15:  else
16:    if (!sat) then
17:      return 充足不能;
18:    end if
19:    LB = MP; MP = LB + ⌈(UB − LB)/2⌉;
20:  end if
21:  for i = 1 to LB do
22:    v_i = 1;
23:  end for{LB ≤ Σ_{i=1}^{n} b_i}
24:  for i = UB to n do
25:    v_i = 0;
26:  end for{Σ_{i=1}^{n} b_i < UB}
27:  Ass = ∧_{i=MP}^{UB}(v_i = 0); {Σ_{i=1}^{n} b_i < MP と仮定 }
28: end while
29: return 最後のモデル M;
```

2.4 第 0.4 版

一般には，線形探索よりは二分探索の方が効率がよい，と言われているが，実際には，線形探索の方が速く解ける問題も多い．

k_1 を 2.2 の最終段落と同じ数とし，k_1 が MaxSAT 解であった，と仮定する．この場合，線形探索では，次の SAT ソルバー呼出しで処理が終了するが，二分探索では，$logk_1$ 回の SAT ソルバー呼出しの後，処理が終了する．このように，k_1 が MaxSAT 解に近いような問題では，線形探索の方が，二分探索より優位である．

第 0.4 版は，このような線形探索の優位性を第 0.3 版に取り入れることを狙ったもので，線形探索と二分探索を交互に行う．これは，Algorithm 2 を次のように変更することにより実現される．

- 3 行目と 4 行目の間に，文 "`BIN = 0;`" を挿入．
- 18 行目と 19 行目の間に次の文を挿入．
  "**if** `BIN == 0` **then return** 最後のモデル M **end if**;"
- 26 行目と 27 行目の間に文 "`BIN = 1-BIN;`" を挿入．
- 27 行目を次のようにする："**if** `BIN == 0` **then** `Ass` $= \top$ **else** `Ass` $= \wedge_{i=MP}^{UB}(v_i = 0)$ **end if**"

ここで，変数 `BIN` は，探索モードを表し，`0` は線形探索中，`1` は二分探索中を表す．`BIN` は，`0` と `1` の値を交互にとる．

3. 実験

QMaxSAT を MiniSat 2.0 をベースに実装した．基数制約を扱うために，MiniSAT のトップレベル部分の変更を行った．他の部分は，各種パラメータの初期値を含めてまったく変更してない．

４つの版の性能を比較するために，Max-SAT 2010 の PMS 部門で出題された全 1122 問を利用して実験を行った．PMS 部門は，さらに random，crafted，industrial の三つの副部門に分かれている．表 1 に実験結果を示す．実験は，Core i5-750（4-core 2.66GHz）CPU，4GB メモリのマシン上で行った．制限時間は 30 分とした．表中の数字は，制限時間内に解けた問題の平均 CPU 時間（単位：秒）で，括弧内の数字は，解けた問題数を表す．

第 0.1 版に比べ，第 0.2 版の方が，性能が向上していることが分かる．解けた問題が，crafted では 294 問から 295 問へ，industrial では 379 問から 392 問へ増加している．第 0.1 版は industrial で優勝しているので，この部門での第 0.2 版の性能の高さが分かる．

ソフト節が多すぎて，第 0.1 版では基数制約のための CNF 論理式が 4GB メモリには収まらないものが 13 問あった．それらは全て industrial の問題で，ソフト節の数は，9188 から 101248 であった．第 0.2 版では，13 問全てで CNF 論理式が 4GB メモリに収まった．また，13 問中 8 問については，制限時間内に解くことができた．

第 0.1 版から第 0.2 版への性能向上が明らかなのに比べ，第 0.2 版と第 0.3 版の性能差はほとんどないように見える．Crafted 問題では，0.2 版で解けて 0.3 版で解けない問題が 3 題，逆に 0.2 版で解けなくて 0.3 版で解けた問題が 2 題あり，差し引き，0.3 版で解けた問題が 1 題減った．Industrial 問題では，逆に，0.3 版で解けた問題が 1 題増えているが，この内訳は，0.2 版で解けて 0.3 版で解けない問題が 7 題，逆に 0.2 版で解けなくて 0.3 版で解けた問題が 8 題，であった．

第 0.4 版が，４つの版の中では最も性能が良いと言えるだろう．第 0.3 版と比べ，Random と Industrial では解けた問題数は同じであるが，Crafted では，解けた問題が 4 題増えた．Crafted では，第 0.2 版と第 0.3 版のいずれか一方のみで解けもう一方で解けない問題が 5 題あったが，第 0.4 版は，これら 5 題を全て解くことに成功した．これらの問題に関しては，第 0.4 版は線形探索と二分探索の双方の利点を継承できた，と言っても良いだろう．

4. おわりに

部分 MaxSAT ソルバー QMaxSAT の４つの版（第 0.1〜0.4 版）の性能評価を行った．基数制約の CNF 符号化に要する節数を削減した第 0.2 版によって，第 0.1 版ではメモリ不足で手に負えなかった問題を解くことができた．解の二分探索を行う第 0.3 版と線形探索を行う第 0.2 版では，総合的には性能に大差はなかった．線形探索と二分探索を交互に行う第 0.4 版が，総合的な性能では，４つの版の中で最も優れていた．

QMaxSAT の実装は，基数制約を取り扱えるように SAT ソルバー MiniSat を改良することによりなされている．改良したのは，問題入力手続きと `main` 関数のみである．入力手続きでは，阻止変数をソフト節に付加し，`main` 関数では，Algorithm1 の手続きを行うループを設定した．第 0.1 版と MiniSat のソースレベルの違いは，基数制約を生成する C++コードを含めても百数十行であり，MiniSat 全体の 5%ほどである．この点で，実装は非常に簡単である．

表 1: QMaxSAT 各版の比較

Solver	問題数	第 0.1 版	第 0.2 版	第 0.3 版	第 0.4 版
Random	240	45.04 (31)	30.48 (31)	31.54 (32)	49.93 (32)
Crafted	385	101.69 (294)	94.84 (295)	104.44 (294)	111.61 (298)
Industrial	497	55.28 (379)	67.06 (391)	60.91 (392)	66.77 (392)

平均時間〔秒〕（解けた問題数）

基数制約の CNF 符号化には，Bailleux のものを用いているが，最近，よりメモリ効率の良い符号化が提案された [Asín 09, Codish 10]．これらの符号化に要する節数は，$O(n \cdot log^2 k_1)$ であり，Bailleux のものに比べ少ない．Bailleux の符号化をこれらに置き換えることにより節数を削減し，さらなる性能改善をはかっていきたい．

また今後，重み付き MaxSAT にも適用できるように QMaxSAT を拡張していく予定である．

謝辞

本研究は科研費（20240003）の助成を受けたものである．

参考文献

[Ansótegui 09] Ansótegui, C., Bonet, M. L., and Levy, J.: Solving (Weighted) Partial MaxSAT through Satisfiability Testing, in *Proc. of SAT 2009*, pp. 427–440 (2009)

[Argelich 08] Argelich, J., Li, C.-M., Manyà, F., and Planes, J.: The First and Second Max-SAT Evaluations, *JSAT*, Vol. 4, pp. 251–278 (2008)

[Asín 09] Asín, R., Nieuwenhuis, R., Oliveras, A., and Rodríguez-Carbonell, E.: Cardinality Networks and Their Applications, in *Proc. of SAT 2009*, pp. 167–180 (2009)

[Bailleux 03] Bailleux, O. and Boufkhad, Y.: Efficient CNF Encoding of Boolean Cardinality Constraints, in *Proc. of CP 2003*, pp. 108–122 (2003)

[Berre 10] Berre, D. L. and Parrain, A.: The Sat4j library, release 2.2, *JSAT*, Vol. 7, pp. 59–64 (2010)

[Codish 10] Codish, M. and Zazon-Ivry, M.: Pairwise Cardinality Networks, in *Proc. of LPAR-16*, pp. 154–172 (2010)

[Eén 03] Eén, N. and Sörensson, N.: An Extensible SAT-solver, in *Proc. of SAT 2003*, pp. 502–518 (2003)

[Fu 06] Fu, Z. and Malik, S.: On Solving the Partial MAX-SAT Problem, in *Proc. of SAT 2006*, pp. 252–265 (2006)

[Heras 08a] Heras, F., Larrosa, J., Givry, de S., and Schiex, T.: 2006 and 2007 Max-SAT Evaluations: Contributed Instances, *JSAT*, Vol. 4, pp. 239–250 (2008)

[Heras 08b] Heras, F., Larrosa, J., and Oliveras, A.: MiniMaxSat: An Efficient Weighted Max-SAT Solver, *J. of Artificial Intelligence Research*, Vol. 31, pp. 1–32 (2008)

[Li 09] Li, C. M. and Manyà, F.: *MaxSAT, Hard and Soft Constraints*, Vol. 185 of *Frontiers in Artificial Intelligence and Applications*, chapter 19, pp. 613–631, IOS Press (2009)

[Lin 08] Lin, H., Su, K., and Li, C.-M.: Within-Problem Learning for Efficient Lower Bound Computation in Max-SAT Solving, in *Proc. of AAAI-08*, pp. 351–356 (2008)

[Manquinho 09] Manquinho, V., Marques-Silva, J., and Planes, J.: Algorithms for Weighted Boolean Optimization, in *Proc. of SAT 2009*, pp. 495–508 (2009)

[Marques-Silvar 07] Marques-Silvar, J. and Lynce, I.: Towards Robust CNF Encodings of Cardinality Constraints, in *Proc. of CP 2007*, pp. 483–497 (2007)

[Pipatsrisawat 07] Pipatsrisawat, K. and Darwiche, A.: Clone: Solving Weighted Max-SAT in a Reduced Search Space, in *Proc. of AI 2007*, pp. 223–233 (2007)

[Sinz 05] Sinz, C.: Towards an Optimal CNF Encoding of Boolean Cardinality Constraints, in *Proc. of CP 2005*, pp. 827–831 (2005)

[井上 10] 井上 克巳, 田村 直之：SAT ソルバーの基礎, 人工知能学会誌, Vol. 25, No. 1, pp. 57–67 (2010)

[平山 10] 平山 勝敏, 横尾 真：*-SAT: SAT の拡張, 人工知能学会誌, Vol. 25, No. 1, pp. 105–113 (2010)