1D5-OS-11b-4in

大規模SAT問題の求解のための緩和解法の検討と提案

A relaxation method for solving very large SAT instance

三神 直彬*1　　鍋島 英知*2
Naoaki Mikami　Hidetomo Nabeshima

*1山梨大学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 SAT solving method for large scale instances by a relaxation method. A relaxation method is often used to solve hard combinatorial instances. We use the relaxation technique to solve a large scale instance within limited memory resources. Given an instance, our approach constructs a relaxed instance which consists of some constraints in the instance. If a feasible solution is found from the relaxed instance, the algorithm outputs it. Otherwise, some falsified constraints in the original instance are added to the relaxed one, and it is solved again until a feasible solution is found. In this study, we introduce some extraction methods to construct a relaxed instance. The experimental results show that our approach can solve some large scale instance within limited memory resources although the solving time becomes slower.

1. はじめに

SAT 問題 (satisfiability testing) 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，Cook[1] により初めて NP 完全性が示された問題である．近年，SAT 問題を解くソルバーの飛躍的な性能向上に伴い，原問題を SAT 問題に符号化してから高速な SAT ソルバーを用いて解く問題解決手法が，ソフトウェア・ハードウェア検証，プランニング，スケジューリング，制約充足問題の解法として幅広く研究され，利用されている．しかし SAT は簡易な制約記述言語であるため，実応用問題を SAT 符号化した場合に，しばしば計算機のメモリに乗り切らないような巨大な SAT 問題が生成されることがある．本稿では，そのような巨大な SAT 問題を限られた資源上で求解するための緩和解法を提案する．緩和解法は求解が困難な問題から一部の制約を緩和した緩和問題を生成し，原問題の許容解を得る解法であるが，本稿ではメモリ使用量を減少させるためにこれを利用する．また，メモリ使用量を減らすための効果的な緩和問題の生成手法を提案し，緩和解法を用いることでメモリ容量を超えるような大規模 SAT 問題の求解が可能となることを示す．

本稿の構成は以下の通りである．まず 2 章で SAT 問題と大規模 SAT 問題について説明する．3 章では大規模 SAT 問題を求解するための緩和解法について提案し，4 章で効果的な緩和問題の生成手法について提案する．5 章で提案手法の性能評価を行い，6 章で緩和解法が実際に大規模 SAT 問題を求解可能か調査する．7 章でまとめと今後の課題について述べる．

2. SAT 問題

SAT 問題は与えられた命題論理式が充足可能であるかどうかを決定する問題であり，通常**連言標準形** (Conjunctive Normal Form; CNF) で与えられる．命題変数及びその否定を**リテラル**といい，前者を**正リテラル**，後者を**負リテラル**と呼ぶ．これらのリテラルの選言を**節**と呼び，節を構成するリテラルの個数を，その節の**節長**という．以下に，CNF 式の例を示す．

$$(a \vee b) \wedge (\neg a \vee c \vee \neg d) \wedge (c \vee d) \wedge .....$$

ここで a, b, c, d は命題変数であり，a や $\neg a$ がリテラル，$a \vee b$ が節である．

2.1 大規模 SAT 問題

近年では SAT ソルバーの性能向上を受けて様々な分野で SAT への符号化による問題解決手法が利用されているが，SAT は簡易な制約記述言語であるため符号化によって生成される SAT 問題はしばしば巨大なものとなる．そのため SAT ソルバーでも巨大な SAT 問題を取り扱う場合が増えてきている．例として，制約充足問題の SAT 符号化手法として代表的な順序符号化 [2] では，ドメインサイズ d の整数変数からなる n 項制約は $O(d^{n-1})$ 個の節に符号化され，ドメインサイズが大きくなる程巨大な SAT 問題が生成される．

大規模 SAT 問題を解くためのメモリに関しての問題点は 2 つ考えられる．1 つ目は上記のような与えられる SAT 問題のサイズが非常に巨大で，計算機のメモリ容量を超えてしまうような場合である．2 つ目は求解中に獲得する学習節数についてである．学習節は解の探索に失敗したときに，今後同じ空間を探索してしまうことを防ぐ役割を持つ節である．SAT ソルバーは求解の過程でこのような学習節を動的に追加していくので，与えられた SAT 問題がメモリ容量内に収まるサイズだったとしても，この学習節が増え続けていくことによりメモリ不足となり求解不能となってしまう場合がある．

本稿では 1 つ目の，計算機のメモリ容量を超えるような SAT 問題が与えられた場合を対象とし，そのような問題を求解する手法を提案する．後者に関しては，将来使用される可能性の高い節を特定する学習節評価尺度である LBD[3] を利用することで学習節の大半を破棄可能であることが知られている．

連絡先: 三神 直彬，山梨大学大学院医学工学総合研究部コンピュータ・メディア工学専攻，〒 400 - 8511 山梨県甲府市武田 4 - 3 - 11，E-mail : mikami@nabelab.org

3. 大規模 SAT 問題のための緩和解法

緩和解法ではまず，原問題から一部の制約を緩和した緩和問題を生成する．緩和問題から得られる解のことを緩和解と呼ぶ．緩和解が原問題のすべての制約を充足可能だった場合，このときの緩和解を原問題の実行可能解と呼び，原問題の解が得られたことになる．実行可能解とならなかった場合は，緩和問題を再構成し，再び緩和解を求める．このように，緩和解が原問題の実行可能解となるまで再構成を繰り返していくことで，原問題の解を求めていく．本稿では，原問題から最初に生成される緩和問題のことを初期緩和問題，実行可能解が得られたときの緩和問題を最終緩和問題と呼ぶ．

緩和解法を用いた SAT 問題の求解方法を以下で述べる．SAT 問題においては，節が制約充足問題における制約に該当する．また，SAT 問題中から一部の節を抽出した部分節集合を作ることで，これを緩和問題とする．SAT ソルバーはまず，生成した初期緩和問題について緩和解を求める．緩和解が充足可能となった場合，このときの緩和問題での変数の真偽値割当てで，緩和問題に含まれなかった残りの節集合が充足するかチェックする．すべての節が充足できた場合，この緩和解は原問題の実行可能解であることを示しており，この SAT 問題は充足可能であると判定できる．残りの節集合の一部の節が充足不能であった場合は，緩和問題を再構成し再び SAT ソルバーで求解していく．SAT ソルバーは何度も緩和問題を求解することになるが，各緩和問題で獲得した学習節は再構成した後の緩和問題でも再利用可能なため，探索に失敗した空間を再度求解してしまうということは起こらない．緩和解が充足不能となった場合は，元の SAT 問題も充足不能であると判定できる．これは緩和解が原問題の部分問題であるためである．また，このときの緩和問題は充足不能コアであると考えることができる．

充足不能コアとは，充足不能な SAT 問題中に存在する充足不能な節の部分集合であり，充足不能の原因となった複数の節とそれ以外の節からなる．極小充足不能コアは，充足不能コアから充足不能の原因である節以外の節を取り除いた節の極小部分集合であり，その集合から節をどれか 1 つでも取り除くと充足可能になるような節集合である．緩和解法を用いて充足不能な問題を求解することは，問題中から充足不能コアを見つけることと等しく，また充足不能となった緩和問題中には，極小充足不能コアが含まれていると考えることができる．

ここで緩和解法による SAT 問題求解の利点と欠点について述べる．利点として，緩和問題に含まれなかった節集合はメモリ上に保持しておく必要がなくなり，その分のメモリを節約することができる．欠点としては，求解時間が大幅に増加してしまうことが挙げられる．充足可能な問題の場合，原問題の実行可能解を得られない緩和問題の求解は求解時間の増加に繋がってしまう．充足不能な問題の場合，緩和問題中に極小充足不能コアが含まれていれば緩和解は充足不能となりこの時点で原問題が充足不能であることがわかるが，含まれていない場合はそのときの緩和問題では解を見つけることは不可能であり，その分の探索は無駄となってしまう．本研究では，メモリ不足により求解することが不可能な問題を求解できるようにすることが目的であるため，求解時間の増加については考慮しない．そのため，通常の SAT ソルバーと比較すると求解数が大幅に減少することが予想される．

緩和解法で大規模 SAT 問題を求解する場合，限られたメモリ量の中で求解するために緩和問題は可能な限り小さくなるように生成する必要があり，そのために検討すべき事項は 2 点考えられる．1 つ目は初期緩和問題となる節集合の抽出方法である．初期緩和問題の緩和解が実行可能解となることが最も望ましく，そうでない場合も実行可能解に近い緩和解が得られるよう，原問題の特徴を表すような節を抽出することが必要となる．また，緩和問題のサイズを抑えるためには，実行可能解を得るために必要な節のみから構成されることが望ましい．2 つ目は緩和解が原問題の実行可能解とならなかった場合の，緩和問題の再構成の方法である．緩和問題の再構成として最も単純な方法は，緩和問題に新たに節を加えて制約を強めることである．この場合，実行可能解を得るための最低限の節のみを追加し緩和問題のサイズが不必要に増加しないよう考慮しなければならない．

本研究では前者の初期緩和問題の生成方法について検討し，サイズが最も小さくなるような節の抽出手法を調査する．具体的な緩和問題の生成方法については 4 章で述べる．後者については，単純に緩和解では充足できなかった原問題の制約を追加する手法を用いる．これは SAT 問題の場合，緩和問題に含まれなかった節集合のうち，緩和解で充足不能だった節を緩和問題に追加していく手法となる．

4. 緩和問題の生成

緩和問題となる節集合を抽出する場合，その節集合中には元の SAT 問題の特徴を表すような強い制約を含める必要がある．また，充足不能な問題に対しては 3 章で述べた極小充足不能コアが緩和問題中に含まれていることが望ましい．これは極小充足不能コアを含んだ節集合は必ず充足不能となるため，その時点で SAT 問題全体を充足不能であると判定できるためである．これらを考慮し，以下に示す節抽出手法を提案する．

4.1 短い制約のみを抽出

SAT 問題では，一般に，節長の短い節ほど強い制約として働く．よって節長の短い節を抽出することで単純に問題中の強い制約を緩和問題に含めることができる．また，緩和問題は低コストで生成可能である．

実際の生成方法としては，原問題の全節を節長の昇順でソートし，上位 $k\%$ の節を抽出することで緩和問題を生成する．ソートのためには問題を読み込む必要があるが，外部ソートのアルゴリズムを利用すれば良いので問題は無い．

4.2 節内の変数出現頻度による抽出

各変数が SAT 問題中に何回出現するのかを算出し，節内に存在する変数の出現頻度を元に節を評価，評価値の高い節を抽出する．問題中に多く出現する変数は，真偽値の割り当てによって多くの節に影響を及ぼす．そのため制約違反に関わる場合が多く，その変数を含む節は問題中で強い影響力を持っていると考えられる．

評価値としては，節内変数の出現頻度の最大値，最小値，平均値を用いる．最大値を取った場合，これは出現頻度の高い変数を 1 つでも含んでいる節はすべて強い制約とする楽観的な評価となる．最小値を取った場合，節内のすべての変数の出現頻度が高い場合のみ強い制約と見なす悲観的な評価となる．平均値はこれらの中間的な評価を取ることを狙いとする．各節をこれらの尺度で評価し，全節を評価値の昇順でソート，上位 $k\%$ の節を抽出することで緩和問題を生成する．

4.3 機械学習した分類器による抽出

緩和問題中に極小充足不能コアが含まれていると，緩和解は充足不能となり原問題も充足不能であると判定できる．また初期緩和問題を生成する際，その中に極小充足不能コアを含めることができれば，緩和問題の再構成を行うこと無く充足不

表 1: 各生成手法の性能比較

抽出手法	求解数 (SAT+UNSAT)	緩和問題サイズ		
		Total	SAT	UNSAT
GlueMiniSat	198 (90+108)	100.0%	100.0%	100.0%
Short-clauses	119 (44+75)	44.02%	55.39%	38.60%
Max-var-freq	114 (40+74)	43.51%	61.83%	37.03%
Min-var-freq	113 (41+72)	43.02%	56.39%	36.71%
Avg-var-freq	116 (45+71)	41.97%	54.88%	35.79%
Random Committee	119 (47+72)	46.05%	62.01%	38.38%

能と判定することが可能となる．そこで機械学習により，極小充足不能コアに属する節かどうかを判定する分類器を学習し，この分類器が極小充足不能コアに属すると判定したすべての節を緩和問題として抽出する．以下に分類器の構成方法について述べる．

4.3.1 特徴設計

原問題中の各節が極小充足不能コアに属する節であるかどうかを分類するクラスとして，極小充足不能コアに属する節を正例，属さない節を負例とする[*1]．各節がどちらのクラスに属するかを判定するために，各節から以下の特徴量を抽出する．

SAT 問題中の各節における節長，節内変数の出現頻度，節内リテラルの出現頻度，正リテラル数，負リテラル数の値を利用して，合計 15 個の特徴を設計する．節内変数の出現頻度，節内リテラルの出現頻度では，それぞれ最大値，最小値，平均値を算出し，問題サイズによる差を無くすために正規化を行う．正規化は問題中の総節数，総リテラル数の 2 種類で行う．正リテラル数，負リテラル数は節長で正規化を行う．

4.3.2 学習・検証データ

分類器の作成にはデータマイニングツール Weka[4] を用いる．学習及び検証データの作成には SAT Competition 2011[5] の MUS 部門の問題 300 問を利用した．この問題群に対して極小充足不能コアを抽出することが可能な SAT ソルバーである Haifa-MUC[6] を用いて極小充足不能コアの抽出を行った．Haifa-MUC は SAT Competition 2011 の MUS 部門において，優秀な成績を納めたソルバーである．

各問題中の節に対し，上記の特徴とクラスを合わせた節データを作成する．これらの節データを 1/300 にサンプリングしたものを学習データとして利用し，1/100 にサンプリングしたものを検証データとして利用する．

4.3.3 分類木の評価

上記で設計した特徴を用いて，Weka が提供する分類器の学習アルゴリズム 80 種を評価し，最も良い性能を示した学習アルゴリズムを本稿では採用する．性能の評価には，検証データでの正解率の他に正例再現度を用いる．これは負例を正しく分類することよりも，すべての正例を正しく分類することが望ましいためである．正例再現度が 100%である場合，初期緩和問題中には極小充足不能コアが含まれており，この緩和問題を求解した時点で充足不能であると判定することができる．分類器を作成した結果，Random Committee で作成した分類器が正解率 94.0% ，正例再現度 95.8% で最も高い性能を示した．よって本稿ではこれを採用した．

*1 極小充足不能コアは 1 つとは限らないが，本稿ではある 1 つの極小充足不能コアに属するかどうかでクラス分けを行う．

5. 緩和解法の性能評価

本章では，メモリ容量を超えない問題に対して，緩和解法による求解数を調べ，緩和解法の基本性能を評価する．GlueMiniSat2.2.7[7] に 3 章並びに 4 章で記述した提案手法を実装し，SAT Competition 2011 Application 部門の問題 300 問で実験を行った．実験は 1 問当たり制限時間 1200 秒,CPU Intel(R) Xeon(R) E3-1230 V2 @ 3.30GHz, メモリ 8GB の環境で行った．

実験結果を表 1 に示す．GlueMiniSat は緩和解法を用いない GlueMiniSat2.2.7 での実行結果，Short-clauses は短い節のみの抽出手法，Max-var-freq，Min-var-freq，Avg-var-freq は，それぞれ節内変数の出現頻度の最大値，最小値，平均値による抽出手法，Random Committee は分類器による抽出手法を示している．Random Committee より生成された緩和問題の初期サイズは 12.26%となった．これは SAT 問題中で分類器が極小充足不能コアに属すると判定した節数を表している．それ以外の生成手法では，初期サイズは 10%で固定とした．表 1 中の緩和問題サイズは原問題に対して，求解に成功した最終緩和問題の割合を表しており，値が小さいほど少ない節数で求解できたことを示している．GlueMiniSat ではすべての節を用いて求解するので，緩和問題のサイズはすべて 100%となっている．

表 1 より，GlueMiniSat と比較すると緩和解法を導入した場合での求解数に関しては 40%程度の性能低下が見られたが，原問題のサイズに対して充足可能な問題では 58%，充足不能な問題では 37%の節数で求解可能であることがわかった．生成手法毎の求解数を比較すると，Short-clauses と Random Committee が最も多くの問題の求解に成功している．最終緩和問題のサイズを比較すると，Avg-var-freq が最も少ない節数で求解できていることが分かるが，生成手法による明確な差は見られなかった．

充足可能な問題では，緩和問題で求めた解が元の SAT 問題の解になっている必要があるため，少なくとも原問題に含まれるすべての変数が出現するような節集合が緩和問題に含まれる必要がある．そのため，求解に必要な節数が多く，結果として緩和問題のサイズが大きくなり，求解性能も低下したと考えられる．また充足不能な問題においても，不要な節を緩和問題に含めてしまっていると考えられる．緩和問題に含まれるべきである極小充足不能コアは原問題のサイズに対して非常に小さく，SAT Competition 2011 の MUS 部門では 1%程度である．Application 部門の問題でも充足不能コアは 30%程度であることが知られている．一方で緩和問題のサイズは 10%から最終的に 37%まで増加しているため，求解中に無駄な節を追加していると考えられる．Random Committee は充足不能な問題での性能向上を狙ったが，緩和問題のサイズに他の抽出手法との差は見られなかった．原因としては，極小充足不能コ

表 2: GlueMiniSat2.2.7 でメモリ不足となった問題での求解数

抽出手法	求解数 (SAT+UNSAT)	Time Out
Short-clauses	16 (4+12)	29
Avg-var-freq	14 (4+10)	29
Random Committee	4 (1+3)	13

アの抽出に利用した Haifa-MUC と緩和問題の求解に利用した GlueMiniSat2.2.7 では探索傾向が異なるため，Haifa-MUC が求めた極小充足不能コアに向かって探索が進まなかったことが考えられる．Haifa-MUC は SAT ソルバーでもあるため，この生成手法の課題として，Haifa-MUC を緩和問題の求解に利用した場合についても検証する必要がある．

6. 大規模 SAT 問題に対する性能評価

本章では，緩和解法を用いて実際に大規模 SAT 問題が求解可能かどうか検証する．検証には節内変数出現頻度による抽出手法である Max-var-freq，Min-var-freq，Avg-var-freq のうち最も緩和問題のサイズが小さくなった Avg-var-freq と，Short-clauses，Random Committee の 3 つで行う．

5 章と同様の実験設定から，メモリの使用可能量を 128MB に制限することで疑似的に大規模 SAT 問題を求解するようにした．その結果，GlueMiniSat2.2.7 では 223 問がメモリ不足により求解不能となった．この 223 問のうち，緩和解法を用いることで求解可能となった問題数を表 2 に示す．表中の Time Out は制限時間以内に解けなかった問題数を示している．

表 2 から，GlueMiniSat2.2.7 では求解不能となる大規模 SAT 問題が，緩和解法を導入したことで求解可能となっていることがわかる．また，緩和解法は求解に多くの時間を必要とする解法なので，制限時間を延ばすことで Time Out となった問題も求解できるようになる可能性がある．

しかし GlueMiniSat2.2.7 で求解可能だった問題のうち，緩和解法を導入した場合にメモリ不足となった問題が Short-clauses で 13 問，Avg-var-freq で 12 問，Random Committee で 20 問存在した．これにより，すべての問題で緩和解法が有効ということは無く，一部の問題ではメモリ消費が増えてしまうことがわかった．この原因としては，学習節の増加が考えられる．緩和解法により探索の傾向が変わり，GlueMiniSat2.2.7 よりも多くの学習節を生成する問題が存在する可能性があり，これにより緩和解法を導入した結果求解できなくなる問題が存在していると考えられる．また，Random Committee では他の手法より求解に成功した問題数が少なく，メモリ不足となった問題数が多いことから，メモリ消費がそれ以外の手法より大きくなっていることが分かる．原因として，Random Committee が各節を分類するために 15 個の特徴を保持していることが原因だと考えられる．各特徴はそれぞれが浮動小数点型で，元の SAT 問題の全節分を保持する必要がある．よって SAT 問題中の全節をメモリ上に保持することが不可能な大規模 SAT 問題では，この手法では緩和解法を導入しない GlueMiniSat2.2.7 と同様にメモリ不足となってしまったと考えられる．

7. まとめと今後の課題

本研究では，大規模 SAT 問題における緩和解法を提案した．また，初期緩和問題の生成手法として，節長や変数の出現頻度，機械学習による分類器を利用した手法を示した．評価実験の結果から，充足可能な問題では原問題の 58%程度，充足不能な問題では 37%程度にまで圧縮可能であることを示した．また，緩和解法を用いることでメモリ容量を超える大規模 SAT 問題が求解可能になることを示した．

しかし各提案手法の間に大きな差は見られず，初期緩和問題の生成方法による節の削減には限界があると予想される．緩和問題のサイズを小さくする方法として，初期緩和問題をどのように生成するか，求解の過程での緩和問題の再構成方法，の 2 点を挙げた．さらなる性能向上を目指すには，後者の緩和問題の再構成方法について検討してく必要があると考えられ，今後の課題となっている．また，実際に大規模 SAT 問題を求解する上では，緩和解法を用いたことでメモリ消費が増えてしまう問題も存在していることが分かった．原因として学習節の増加が考えられ，これを削減する方法について検討することで，より多くの大規模 SAT 問題が限られたメモリ量の中でも求解可能となる可能性がある．

参考文献

[1] Cook, S.A.: The complexity of theorem-proving procedures, STOC '71 Proceedings of the third annual ACM symposium on Theory of computing, pp.151-158, (1971).

[2] Tamura, N., Taga, A., Kitagawa, S., and Banbara, M.: Compiling Finite Linear CSP into SAT, Constraints, Vol.14, No. 2, pp.254—272, (2009).

[3] Audemard, G., Simon, L.: Predicting learnt clauses quality in modern SAT solvers, IJCAI'09 Proceedings of the 21st international joint conference on Artificial intelligence, pp.399-404, (2009).

[4] The University of Waikato: Weka 3: Data Mining Software in Java, [http://www.cs.waikato.ac.nz/ml/weka], (2014-03-11).

[5] The international SAT Competitions web page, [http://www.satcompetition.org], (2014-03-11).

[6] Ryvchin, V.: Vadim Ryvchin My Homepage, [http://tx.technion.ac.il/ rvadim/index.html], (2014-03-11).

[7] Nabeshima, H., Iwanuma, K., and Inoue, K.: GlueMiniSat 2.2.7, [http://glueminisat.nabelab.org], (2014-03-11).