1E3-OS-4-4

学習節評価尺度LBDに基づく並列SATソルバーの提案

大橋 弘幸[*1]　　鍋島 英知[*2]
Hiroyuki Ohashi　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 Enginnering, 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 Interdiciplinary Graduate School of Medicine and Engineering, University of Yamanashi

In this paper, we propose a parallel SAT solver based on LBD that is one of evaluation criteria for learnt clauses. CDCL solvers learn a large amount of clauses which are resolvents of the original problem. Learned clauses are useful knowledge to avoid repeating conflicts which have occurred in the past search process. On the other hand, CDCL solvers need to reduce learnt clauses in order to restrain the memory consumption and the speed decline of unit propagations. LBD is an evaluation criteria of learnt clauses, and shows superior performance in the past SAT competitions. However, there is few work on parallel SAT solvers based on LBD criteria. We propose a parallel SAT solver based on LBD which is used for solving a problem by each SAT solver and also used for sharing learnt clauses between solvers.

1. はじめに

本研究では，学習節評価尺度 LBD を用いる逐次 SAT ソルバー GlueMiniSat [2] を並列化し，その性能を評価する．

命題論理式の充足可能性判定問題 (satisfiability testing) である SAT 問題はハードウェア・ソフトウェア検証，プランニングなど様々な分野で用いられている．SAT ソルバーは SAT 問題を解く過程で学習節を生成し保持することで同じ矛盾の繰り返しを回避する．一方で学習節を過剰に保持するとメモリを消費し，また単位伝搬速度が低下するため，学習節の取捨選択が必要となる．その際に用いられる学習節評価尺度の 1 つに LBD (Literal Blocks Distance) [1] がある．この LBD を用いた逐次 SAT ソルバーには，Glucose [1]，GlueMiniSat [2] などがある．GlueMiniSat は Glucose を元に作られた逐次 SAT ソルバーであり，SAT 競技会において優秀な成績を収めている．一方で，マルチコア技術の発達に伴い並列処理により高速に SAT 問題を解く並列 SAT ソルバーの研究も盛んに行われている．しかし LBD を用いた並列 SAT ソルバーはほとんどない．そこで本研究では，GlueMiniSat の並列版を開発しその性能を評価する．

本論文の構成は次のとおりである．まず 2 節において SAT 問題と CDCL ソルバーについて解説する．3 節においては学習節評価尺度 LBD を紹介し，4 節では並列 SAT ソルバーについて概説する．5 節では我々の LBD に基づく並列 SAT ソルバーを提案し，6 節ではその評価実験を示す．7 節で本研究をまとめ，今後の課題について述べる．

2. SAT 問題と SAT ソルバー

SAT とは，与えられた命題論理式を真にできるかを判定することであり，SAT 問題は通常連言標準形で与えられる．命題変数及びその否定を**リテラル**，リテラルの選言を**節**という．連言標準形は節の連言で与えられる．

連絡先: 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 : ohashi@nabelab.org

現在主流の SAT ソルバーは，**DPLL** 手続き [3] に**矛盾からの節学習** (Conflict-Driven Clause Learning) [9] [10] を導入した CDCL ソルバーである．DPLL は現在多くの SAT ソルバーの基礎となっているアルゴリズムであり，命題変数の真偽の組合せを網羅的に探索する．まず，単位節 (未割り当てのリテラルが 1 つで残りのリテラルの値が偽の節) がある限り，それを真にする**単位伝搬**を行う．SAT 問題を解くためにはすべての節を真にする必要があり，単位節のリテラルは必ず真にしなければならない．単位節がなければ，**変数選択ヒューリスティクス**に基づき，変数を 1 つ選び真または偽を割り当てる．この変数を**決定変数**と呼び，決定変数の数を**決定レベル**という．探索の途中で**矛盾** (同じ命題変数に真と偽を同時に割り当てる) が起きた場合，最後に選択した決定変数まで戻り，その決定変数の真偽値を反転して探索する．これらを繰り返して解を探索する．

CDCL ソルバーではこの DPLL 手続きに加えて矛盾からの節学習を行う．**矛盾からの節学習**とは，矛盾発生時にその原因を解析し，以降の探索で同じ矛盾を回避するために，その原因を否定した節を学習する．これを**学習節**とよぶ．

学習節が生成される過程を図 1 を使用して説明する．この図は含意グラフ (implication graph) と呼ばれ，ある決定レベルにおける単位伝搬の様子を示している．図において各ノードは真の値が割り当てられたリテラルを示している．図では f が決定変数であり，その決定レベルを l とする．白色のノードは f と同じ決定レベル l で真が割り当てられたリテラルを表す．一方，黒色のノードは決定レベル l 以前に真が割り当てられたリテラルを表す．あるノードから別のノードへ向かって伸びている矢印は，そのノードのリテラルを真にすることで矢印の先にあるリテラルが真になるという単位伝搬を表している．例えば，リテラル g に向かってリテラル f と e から矢印が伸びているが，これは節 $(\neg f \vee \neg e \vee g)$ においてリテラル f とリテラル e が真になったために，その節が単位節となり，リテラル g を真にしなければならなくなったことを示している．

そして図 1 の単位伝搬では最終的にリテラル m とリテラル $\neg m$ に同時に真を割り当てなければならないという矛盾が発生している．この矛盾の原因を解析する際に，まず決定変数の

リテラルから m または $\neg m$ まで矢印をたどっていくときに必ず通るリテラルを探す．この例では f と g がそれである．このリテラルを UIP(Unique Implication Point) と呼び，最も矛盾に近い UIP を FirstUIP と呼ぶ．ここで，ある UIP よりも矛盾に近く決定レベル l で値の決まったリテラルを conflict side，それ以外のリテラルを reason side という 2 つのグループに分割する．この時 reason side から conflict side に向かって矢印を伸ばしているリテラルが矛盾の原因である．図 1 では FirstUIP のリテラル g とリテラル b，c，a が矛盾の原因である．つまり，これらのリテラルに同時に真を割り当てるという状態が発生すると矛盾が生じる．従ってこの状態を否定する節 $(\neg a \vee \neg b \vee \neg c \vee \neg g)$ が学習節となる．この様にして FirstUIP より得られる学習節が有用であることは Zhang らにより示されている． [12]

3. 学習節評価尺度 LBD

学習節を過剰に保持するとメモリを消費し，単位伝搬速度も低下する．そこで通常 CDCL ソルバーでは，ある尺度に基づき学習節の有用性を評価し，有用でない学習節を削除することを定期的に行う．

最も簡単な評価尺度は学習節の長さである．節の長さとはその節に含まれる変数の数である．短い学習節ほど単位節になりやすく，より多くの真偽値の組合せを取り除けるため有用である．

代表的な CDCL ソルバーである MiniSat [8] では**活性度** (activity) という評価尺度を用いる．活性度は学習節毎に与えられる．活性度は学習節が生成された時とその学習節が矛盾の原因に寄与した時に増加し，ある活性度が閾値を超えるとすべての活性度が定数で割られて減少する．このため最近の矛盾に関与しない学習節は活性度が低くなり，関与する学習節は活性度が高くなる．学習節を削減する際には活性度の低い学習節を削除する．すなわち最近の矛盾に関わった学習節を残すようにする評価尺度である．

一方で Glucose [1]，GlueMiniSat [2] では LBD (Literal Brocks Distance) [1] という評価尺度を利用している．LBD とは学習節に含まれるブロック数を示す．ブロックとは，同じ決定レベルで一斉に値が割り当てられた節内の変数の集合である．ブロックに含まれる変数は，今後の探索においてもある決定レベルで再び同じ塊となって現れることが期待できる．このブロックをあたかも 1 つの命題変数とみなす．例えばある学習節の長さが 100 であっても LBD が 2 であれば，その学習節はあたかも長さが 2 の学習節とみなせ，実際に長さが 2 の学習節程ではないものの単位伝搬を引き起こしやすくより有用であると考えられる．すなわち変数数ではなく，ブロック数の小さい学習節ほど有用とする評価尺度である．

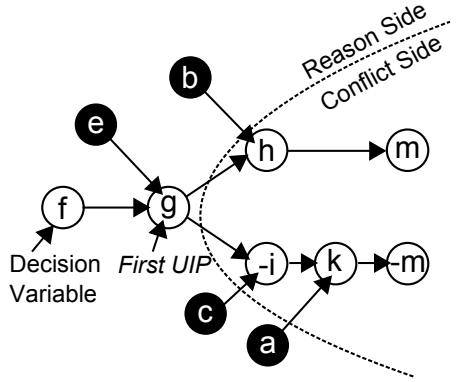

図 1: 含意グラフ

GlueMiniSat では LBD を強化した Pseudo LBD [2] という尺度に基づき，Glucose よりも多くの学習節を保持する戦略をとる．

4. 並列 SAT ソルバー

マルチコア技術の普及に伴い，並列処理により SAT 問題を解く並列 SAT ソルバーの研究が盛んに行われている．代表的な並列 SAT ソルバーとして ManySAT [6] や c-sat [7] などがある．それらの SAT ソルバーにも取り入れられている並列 SAT ソルバーの代表的な技術として学習節共有と多重戦略を紹介する．

並列 SAT ソルバーでは，複数の SAT ソルバーを同時に動かして SAT 問題を解いていく．その際にあるソルバーで生成された学習節を他のソルバーに送り共有することを行う．これを**学習節共有**という．これによりソルバー間で同じ矛盾を回避できるため探索の効率化が期待できる．また学習節が過剰に送信されるのを避けるため，通常は送信する学習節の長さを制限する．

また，すべての並列プロセスにおいて，同じ探索戦略をとる SAT ソルバーで SAT 問題を解いては意味がない．そこで通常は各プロセスで使用する SAT ソルバーを変えるか，あるいは同じ SAT ソルバーを用いるがパラメータは各プロセスで異なるようにする．この様に各プロセスで異なる探索戦略を用いて SAT 問題を解くことを**多重戦略**と呼ぶ．

5. LBD に基づく並列 SAT ソルバー

本研究では，LBD に基づく逐次 SAT ソルバー GlueMiniSat に，学習節共有機構と多重戦略機構を導入し，学習節共有時の評価尺度として LBD を用いた並列 SAT ソルバーを開発した．開発した並列版 GlueMiniSat は，システム全体を管理し学習節の収集分配を行う Master と，実際に SAT 問題を解く複数の Worker から構成される．また Master-Worker 間の通信には MPI を利用する．

5.1 学習節共有機構

学習節の共有には c-sat などで用いられている **Master-Worker モデル**を使用する．ある Worker で得られた学習節は一度 Master に送られ，Master を介して他の Worker に送られる．並列版 GlueMiniSat では，送信する学習節の長さだけでなく LBD も制限する．評価実験では，

- 学習節の長さを 8 以下
- LBD が 3 以下で長さが 100 以下

の 2 通りの制限を使用した．学習節の長さを 8 以下としたのは ManySAT や c-sat などの並列 SAT ソルバーで長さ 8 以下の学習節を共有しているためである．また LBD が 3 以下でかつ長さが 100 以下とした理由として，GlueMiniSat では前述の Pseudo LBD が 3 以下の学習節を保持するようにしていることと，LBD が小さくても巨大な学習節が送信されることを避けるためである．

5.2 多重戦略機構

各 Worker は GlueMiniSat を使用して SAT 問題を解くが，そのパラメータは Worker 間で異なる．GlueMiniSat は Glucose と MiniSat に基づいて開発されており，MiniSat には変

数選択時における変数の選び方や活性度の上昇度などを決定する様々なオプションがある．本研究で開発した並列版 GlueMiniSat では，その中の変数選択に影響を与える *var_decay* と *polarity* の 2 種類のオプションにより指定されるパラメータが Worker 間で異なる．

- *var_decay*: 変数の活性度の増分を制御するパラメータである．値域は $0.0 < var\_decay \leq 1.0$ であり GlueMiniSat では 0.95 を初期値とする．変数の活性度は前述の学習節に与えられる活性度とは異なり変数選択ヒューリスティクスに用いられ，MiniSat や GlueMiniSat では変数選択時に活性度の高い変数を選択する．活性度は対応する変数が生成された学習節に含まれると *var_inc* だけ増化し，定期的に減少する．増分 *var_inc* はリスタートの度に $1/var\_decay$ 倍される．従って *var_decay* が 0 に近い程，最近の矛盾に関与した変数が選択されやすい．

- *polarity*: 変数選択時に変数に割り当てる真偽値を決めるパラメータである．GlueMiniSat には実装されていないため，我々が追加実装した．GlueMiniSat では phase chaching [5] という手法に基づき，選択した変数には以前の探索空間で割り当てられていた真偽値をそのまま割り当てるという戦略をとっている．並列版 GlueMiniSat に追加実装した polarity のオプションは，変数に常に偽を割り当てる False，逆に常に真を割り当てる True，ランダムに真偽値を割り当てる Random の 3 種類である．

各 Worker のパラメータは予備実験により決定した．予備実験では，polarity のオプションを追加実装した GlueMiniSat に表 1 に示すパラメータのいずれかを指定して SAT 2009 競技会 Application 部門の問題 292 問を制限時間 1000 秒で解かせた．そうして得られた結果の中から総求解数が最も高くなるパラメータの 4 つの組合せを選び，各 Worker に指定するパラメータとした．評価実験では 1，2，3，4 番のパラメータを使用する．

6. 評価実験

逐次版の GlueMiniSat と本研究で開発した並列版 GlueMiniSat とで性能を比較した．実験環境は Mac mini，Core 2 Duo 1.83GHz，2GB RAM である．並列版 GlueMiniSat ではこのマシンを Master に 1 台，各 Worker に 1 台ずつの計 5 台を使用している．制限時間は 1 問あたり 1000 秒とした．評価実験に使用した問題は SAT Competition 2009 の問題の中から選んだ 55 問である．逐次版 GlueMiniSat が 1000 秒から 3000 秒かけて解いた問題 21 問と 1000 秒未満の間に解いた問題 34 問を使用した．

また，並列版 GlueMiniSat は 3 パターンの設定で実験を行っており，それぞれ学習節共有をしない場合，長さを 8 以下の学習節を共有する場合，LBD3 以下でかつ長さ 100 以下の学習節を共有する場合である．

表 1: 予備実験で使用したパラメータ

	1	2	3	4	5	6
var_decay	0.95	0.96	0.94	0.95	0.96	0.94
polarity	phase saving	phase saving	phase saving	Flase	False	False

表 2: 逐次版と並列版 GlueMiniSat の結果

	問題数	逐次版	並列版 共有なし	並列版 共有 (1)	並列版 共有 (2)
SAT < 1000	11	10	11	10	9
SAT ≥ 1000	10	0	5	8	8
UNSAT < 1000	23	23	23	23	23
UNSAT ≥ 1000	11	0	3	11	11
all	55	34	42	52	51

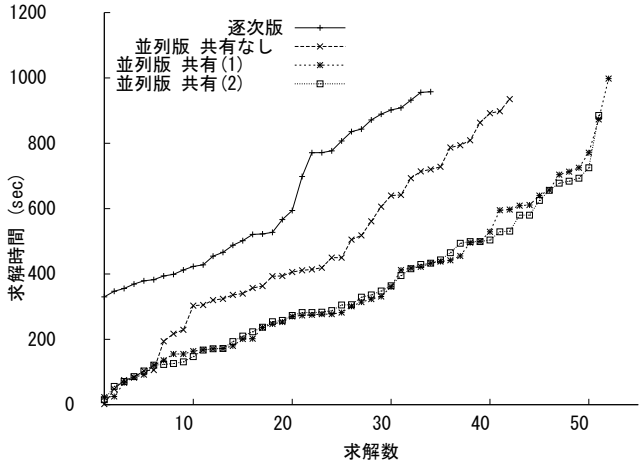

図 2: 実験結果

実験結果を表 2 に示す．各表の SAT， UNSAT は解いた問題の種類を示し，それぞれ SAT 問題， UNSAT 問題を示す．また < 1000，≥ 1000 はそれぞれ逐次版 GlueMiniSat を用いて解いた時の求解時間が 1000 秒未満，1000 以上 3000 秒以下であったことを示す．また逐次版とは逐次版 GlueMiniSat の結果を示している．同様に並列版は並列版 GlueMiniSat の結果を示しており，共有なしは学習節共有なしの場合，共有 (1) は長さ 8 以下の学習節を共有する場合，共有 (2) は LBD 3 以下かつ長さ 100 以下の学習節を共有する場合を示している．表中の数値はその項目での求解数を示している．例えば，学習節共有を行わない場合の並列版 GlueMiniSat は逐次版が 1000 秒以上かけて解いた SAT 問題を 5 問解いている．

表 2 より，多重戦略機構の導入によって逐次版 GlueMiniSat が 1000 秒以上かけなければ解けない問題を 1000 秒以内で解けるようになり，特に SAT の求解数が大きく向上した．すなわち多重戦略機構は SAT 問題に対し有効にはたらいた．そして，学習節共有を行うことでさらに求解数が向上した．特に今回用意したすべての UNSAT 問題を解くことに成功しており，UNSAT 問題の求解数向上に学習節共有機構が大きく寄与したと言える．しかし一方で，逐次版 GlueMiniSat が 1000 秒以内に解いていた SAT 問題をそれぞれ 1 問，2 問ずつ解けなくなった．また求解数は 1 問少ないが，本実験において LBD3 以下かつ長さ 100 以下の学習節を共有する戦略は，長さ 8 以下の学習節を共有する戦略とほぼ同等の結果となった．今後 UNSAT 問題を増やしてさらに評価する必要がある．

また，逐次版 GlueMiniSat と各並列版 GlueMiniSat の求解時間を図 2 に示す．縦軸は求解時間を秒単位で示しており，横軸は求解数を示している．縦軸と交わる部分のプロットは 1 問目の問題の求解時間を示す．

図 2 より逐次版 GlueMiniSat と比較すると，学習節共有を行わない場合の並列版 GlueMiniSat は求解数・求解時間共に性能が向上しており，多重戦略機構の効果が顕著に現れている．学習節共有を行った場合の並列版 GlueMiniSat は行わない場合よりもさらに性能が向上しているが，共有 (1) と共有 (2) ではそれほど性能に差は見られない．

以上のことから，共有する学習節の長さがある程度長くても LBD が十分小さければ，短い共有節と同じくらい求解数の向上に寄与できると言える．

7. まとめと今後の課題

本研究では，学習節評価尺度 LBD に基づく並列 SAT ソルバーを開発しその性能を評価した．LBD に基づく並列 SAT ソルバーの開発にあたり，LBD に基づく逐次 SAT ソルバー GlueMiniSat を並列化し，多重戦略機構と Master-Worker モデルによる学習節共有機構を取り入れることを導入した．

その結果，多重戦略機構と学習節共有により性能を大幅に向上させることに成功した．また，共有する学習節の長さを 8 以下に制限した場合と LBD3 以下かつ長さ 100 以下に制限した場合では，両者はほぼ同等の性能を示した．このことから，共有する学習節の長さがある程度長くても LBD が十分小さければ求解数の向上に寄与できると言える．

今後の課題として，今回の実験では問題数が 55 問だけだったのでさらに問題数を増やして実験・評価を行う必要がある．またより多くの共有制限や多重戦略機構のパラメータの組合せを試す必要がある．

参考文献

[1] Andemard, G. and Simon, L.： Predicting Learnt Clauses Quality in Modern SAT Solvers, Proc. IJCAI-09，pp.399-404，(2009).

[2] 鍋島英知, 岩沼宏治, 井上克巳：GlueMiniSat 2.2.5：単位伝搬を促す学習節の積極的獲得戦略に基づく高速 SAT ソルバー，コンピュータソフトウェア，(2012)，(再録予定).

[3] Davis, M.，Logemann G. and Loveland, D. W.：A machine program for theorem-proving. Communications of the ACM，5(7)，pp.394-397，(1962).

[4] 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，pp.68-76，(2010).

[5] Pipatsrisawat，K. and Darwiche，A.：A Lightweight Component Caching Scheme for Satisfiability Solvers, Proc. SAT-07，pp.294-299，(2007).

[6] Hamadi，Y.，Jabbour，S.，and Sais，L.：ManySAT：a Parallel SAT Solver，Journal on Satisfiability，Boolean Modeling and Computation 6，pp.245-262，(2009).

[7] Ohmura，Kei.，Udea，Kazunori.： c-sat: A Parallel SAT Solver for Clusters，Theory and Applications of Satisfiability Testing-SAT 2009，pp.524-537，(2009).

[8] Eén，N. and Sörensson，N.；An Extensible SATsolver, SAT，Giunchiglia，E. and Tacchella，A. (eds.)，Lecture Notes in Computer Science，Vol.2919，Springer，pp.502-518，(2003).

[9] Bayardo，R. J. and Schrag，R. : Using CSP Look-Back Techniques to Solve Real-World SAT Instances, AAAI/IAAI, Kuipers，B. and Webber，B. L. (eds.)，AAAI Press / The MIT Press，pp.203-208，(1997).

[10] Silva，J. P. M. and Sakallah，K. A. : GRASP: A Search Algorithm for Propositional Satisfiability, IEEE Trans. Computers，Vol.48，No.5，pp.506-521，(1999).

[11] Gomes，C. P.，Selman，B.，and Crato，N. : Heavy-Tailed Distributions in Combinatorial Search，CP，Smolka，G. (ed.)，Lecture Notes in Computer Science，Vol.1330，Springer，pp.121-135，(1997).

[12] Zhang，L. , madigan，C. F. , Moskewicz，M. W. , and Malik，S. : Efficient Conflictt Driven Learning in Boolean Satisfiability Solver，ICCAD，pp.279-285，(2001).