The 36th Annual Conference of the Japanese Society for Artificial Intelligence, 2022

類似学習節に基づくCDCLソルバーの高速化

Note on CDCL Inference with Similar Learnt Clauses

趙 振江 *1　　戸田 貴久 *1
Zhenjiang Zhao　　Takahisa Toda

*1電気通信大学大学院情報理工学研究科
Graduate School of Informatics and Engineering, The University of Electro-Communications

The conflict-driven clause learning (CDCL) is a standard algorithmic framework on which almost state-of-the-art SAT solvers are based. During the solving process of the CDCL solver, many learnt clauses are generated, and those turned out to be useless in terms of some criteria are removed. Since the decision commonly relies on heuristics, the same clauses can appear multiple times. Hence, the evaluation of the learnt clause utility has a significant impact on the solver's performance. The recently proposed DL heuristic determines the utility in terms of the number of times clauses are generated. To improve this, we introduce the similarity of learnt clauses and propose a similarity-based clause management method. In experiments we compared our method with the DL, both implemented on top of CaDiCal, and we confirmed that our method outperforms the DL as well as the intact CaDiCal in both PAR-2 scores and the number of solved instances.

1. はじめに

充足可能性問題 (Boolean Satisfiability Problem; SAT) は，与えられた命題論理式を充足する論理変数への真偽値の割当の有無を判定する問題である．SAT はハードウェア・ソフトウェアのモデル検査，テストパターン生成，プランニング問題，制約充足問題，定理証明，暗号技術など，様々な分野に応用されている．

SAT が多くの分野で広く使われている主な理由の一つは，**CDCL ソルバー** (Conflict-Driven Clause Learning Boolean Satisfiability Solver) が実用上非常に有効であるためである [João 09] ．CDCL ソルバーは，SAT 問題の求解過程で矛盾が発生する度に**学習節** (Learnt Clause) を導出する．学習節で同様な矛盾の発生を防ぐことができるため，探索が効率的になる．一方，学習節が多くなるとメモリ不足や単位伝播速度低下を引き起こすため，積極的に不要な学習節を削除するためのヒューリスティックスが活用されている [Audemard 09]．このため，CDCL ソルバーの実行中には，一般に同じ節が複数回学習され，削除される．このような学習節は**重複学習節** (Duplicate Learnt Clause) と呼ばれる．

近年の研究では，重複学習節の削除ポリシーを工夫することで CDCL ソルバーの効率が向上する結果が報告されている [Kochemazov 20]．本論文では，この手法の改善として，類似した学習節を削除しないようにするヒューリスティックスを提案する．

本論文の構成は以下の通りである．まず，2. 節では先行研究について解説する．3. 節において従来の **DL ヒューリスティック** (Duplicate Learnts Heuristic) と提案した **DVDL ヒューリスティック** (Decision Variables Duplicate Learnts Heuristic) について説明する．4. 節において提案手法の実装に関する説明を与える．5. 節では提案ヒューリスティックの評価実験を行う．6. 節でまとめと今後の課題を述べる．

連絡先: 趙 振江：zhenjiang@disc.lab.uec.ac.jp，戸田 貴久：toda@disc.lab.uec.ac.jp，電気通信大学大学院情報理工学研究科，東京都調布市調布ヶ丘 1-5-1

2. 先行研究

本節では，まず学習節データベース管理方法の基本技術の説明からはじめる．その後，重複学習節の削除ポリシーに関する最近の手法（DL ヒューリスティック）を説明する．

CDCL ソルバーの特徴は，SAT 問題の求解過程で学習節を生成して行くことである．学習節は同様な矛盾の発生を防ぐ効果があるため，学習節が多いほど探索領域が小さくなり，不要な探索を防ぐことができる．しかし，多くの学習節があるとメモリ不足や単位伝播速度の低下を引き起こすため，定期的に不要な学習節を削除する必要がある．

不要な学習節を削除する上で最も重要な問題は，ある学習節が他の学習節より優れていることをどのように評価するかということである．良い学習節は削除せずに残すことでより効率的に求解できるようになる．しかし，学習節が将来の探索に役に立つかどうかを事前に知る術はないため，ヒューリスティックスの活用が現実的な手段となる．

2.1 基本技術

最近のソルバーで最もよく使われている学習節の価値を評価する尺度は ***LBD*** (Literal Block Distance) である [Audemard 09]．LBD は現代 CDCL ソルバーのほとんどで利用されている．また，LBD に基づく学習節データベース管理方法としての ***3* 層構造** (Three-tiered Structure) がある．学習節を扱うためのもう一つよく使われる手法は Inprocessing [Järvisalo 12] である．この手法により学習節は短くなり，LBD 値が小さくなる．

学習節の LBD は，その節内の**ブロック**の個数を意味する．ここでブロックは同じ決定レベルで値が割り当てられた変数の集合である．LBD は，学習節内の**決定変数** (Decision Variable) の数と同じである．決定変数は，変数選択ヒューリスティックスによって選択されて値が割り当てられた変数である．

3 層構造は学習節を管理する方法であり，基本的な考え方は学習節を LBD 値に基づいて三層に分類し，層ごとに異なる削除方針を適用することである．この方法は最初に COMINISATPS ソルバー [Oh 15] で導入され，現在ほとんどの CDCL ソルバーで採用されている．特に，SAT Competitions 2016 から 2018 のメイントラックの優勝ソルバーは，すべて COMINISATPS

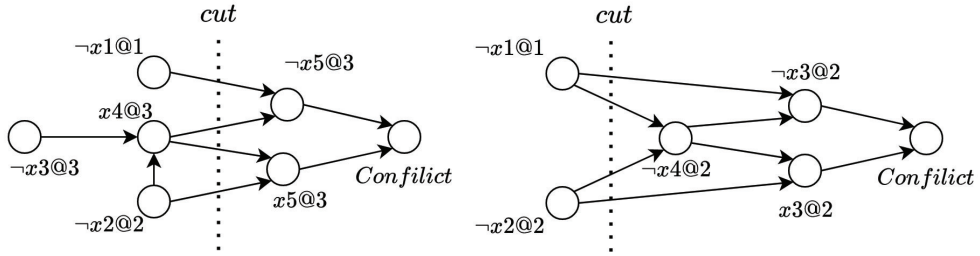
表 1: 学習節とキーの対応関係の例

学習節	キー
$(x1, x2, \neg x4)$	$" - 4\|1\|2\|"$
$(x1, x2)$	$"1\|2"$

(a) DL の場合

学習節	決定変数 (決定レベル)	キー
$(x1, x2, \neg x4)$	$x1@1, x2@2$	$"1\|2"$
$(x1, x2)$	$x1@1, x2@2$	$"1\|2"$

(b) DVDL の場合（決定レベルを@に続く値で表示）



(a) $C1$ が得られるとき　　(b) $C2$ が得られるとき

図 1: 例 1 における含意グラフ

をベースにしている．

3 層構造では，パラメータ $core\_lbd\_cut$ と 2_lbd_cut に基づいて $lbd \leq core\_lbd\_cut$ を満たす学習節を**コア層** (Core Tier) に，$core\_lbd\_cut < lbd \leq 2\_lbd\_cut$ を満たす学習節を**中間層** (Tier2) に，$2\_lbd\_cut \leq lbd$ を満たす学習節を**ローカル層** (Local Tier) に分類する．ローカル層に保持される学習節は価値が低いことを意味し，次の削除操作のときに削除される．ローカル層より価値のある学習節は中間層に保持される．中間層の学習節は次の削除操作での削除は猶予されるが，2 回目の削除操作の適用時点で削除される．中間層の学習節は LBD 値の変化によりコア層に移動する可能性がある．一旦学習節がコア層に移動すれば，以後コア層に保持されつづけ，削除されることはない．

2.2 DL ヒューリスティック

DL ヒューリスティックのアイデアは，学習節の重複回数が多いほど学習節の価値が高いと考え，学習節の重複回数を記録し，その回数に応じてローカル層，中間層，コア層のいずれかに移動することである．従来の 3 層構造では LBD に基づいて学習節を分類していたが，DL ヒューリスティックではこの基準に加えて，重複回数に応じた学習節の分類基準もある．両者の基準による結果に食い違いが生じた場合は，より上位の層を結果として採用している．例えば，LBD に基づく基準で中間層，重複回数に基づく基準でコア層の場合は，コア層に移動させる．

DL ヒューリスティックを実装するために，以下のように，ハッシュテーブルの $Htable$，パラメータとしての lbd_limit，2_dup と $core\_dup$ を準備する．

- $Htable$: 学習節の重複回数を記録するハッシュテーブルである．$Htable$ では学習節がキーで，学習節の重複回数が値である．
- lbd_limit: 学習節の LBD の制限値
- 2_dup: 中間層に移動する判定回数
- $core\_dup$: コア層に移動する判定回数

表 1 の (a) は DL ヒューリスティックで学習節とキーの一対一対応関係の例である．

矛盾から獲得された学習節 C（あるいは学習節に対して in-processing などの単純化を適用して得られた節）に対して，Algorithm 1 のような操作を行う．Algorithm 1 では，節 C の LBD 値を $lbd(C)$ と表し，節 C の文字列表現を $str(C)$ と表す．この文字列を節 C のキーとするとき，ハッシュテーブルで対応する値を $Htable[str(C)]$ と表す．

Algorithm 1 DL ヒューリスティック

DL(C) :
if $lbd(C) \leq lbd\_limit$ **then**
  if $str(C)$ が $Htable$ に登録されている **then**
    $Htable[str(C)] = Htable[str(C)] + 1$
  else
    $Htable[str(C)] = 1$
  end if
end if
if $core\_dup \leq Htable[str(C)]$ **then**
  節 C をコア層に移動させる．
else if $2\_dup \leq Htable[str(C)] < core\_dup$ **then**
  節 C を中間層に移動させる．
else
  節 C をローカル層に移動させる．
end if

3. 提案手法

本節では，DL ヒューリスティックの改善として，学習節の類似性に着目した DVDL ヒューリスティックを提案する．

まずは，DL ヒューリスティックの課題について議論して，DVDL ヒューリスティックの着目点を説明する．

DL ヒューリスティックの活用によりソルバーの性能が向上することが確認されているが，学習節の重複は頻繁に生じるとは限らない．例えば，[Kochemazov 20] において報告されているように，ある命題論理式に対して MINISAT 2.2 ソルバー

を適用したとき，最初の 2000 回のリスタート（約 17 秒）の間に，905,378 個の学習節が生成され，そのうち 6353 個が重複学習節となった．すべての学習節のうち重複学習節の占める割合はたった 0.7% であり，大多数は，2 回の重複（5038 個），3 回の重複（785 個），4 回の重複（234 個）であった．

本論文では，重複をさらに利用するため，節の類似性に着目する．DL ヒューリスティックでは節同士が完全に一致するときに重複と考えていたが，これをもっと緩めて，一定程度の類似性をもつ節同士を重複とみなす．そこで，本論文では以下の類似関係を導入する．

定義 1 学習節 C_1, C_2 が与えられたとき，C_1 と C_2 に含まれる（各学習節の獲得時点における）決定変数が同じであり，決定レベルの順番も同じであるとき，C_1 と C_2 の間に**類似関係**があるという．

例えば，以下の CNF から異なる $C1 = (x1, x2, \neg x4)$ と $C2 = (x1, x2)$ が学習節として得られることを順に説明する．

例 1 $(x2, x3, x4) \wedge (x1, \neg x4, \neg x5) \wedge (x2, \neg x4, x5) \wedge (x1, \neg x3, x4)$

最初に $x1$ を決定変数で選び，偽を割り当てる．単位伝播が生じないので，$x2$ を次に選び，偽を割り当てる．再び単位伝播が生じないので，$x3$ を次に選び，偽を割り当てる．図 1 の (a) の含意グラフのように単位伝播が生じ，矛盾の発生により，学習節 $C1$ が得られる．その後，$x3$ 以降の割り当てをキャンセルした上で，レベル 2 にバックトラックする．今度は，図 1 の (b) の含意グラフのように単位伝播が生じ，学習節 $C2$ が得られる．ここで，$C1$ と $C2$ では，$x1$ と $x2$ だけが決定変数であるので $C1$ と $C2$ の間に類似関係がある．$C2$ を学習したときに $C1$ に対して ”重複” が発生したと考える．

それでは，提案手法の DVDL ヒューリスティックを与える．DVDL ヒューリスティックでは，各学習節 C に対して，決定変数を決定レベルの順に並べて得られる文字列 $sim\_str(C)$ をキーとする．この下で DL ヒューリスティックと同じ Algorithm 1 を適用することで，学習節の重複回数の管理および 3 層への学習節の分類を行う．類似関係にある学習節は同一のキーを持つので，重複回数が増やされる．表 1 の (b) は上例の学習節 $C1, C2$ とキーの対応関係である．

DL ヒューリスティックでは学習節が完全に一致する場合のみ重複回数が増やされるのに対して，DVDL ヒューリスティックは完全一致だけでなく，類似関係にある場合でも重複回数が増やされる．本節の冒頭で議論したように，重複学習節の個数は全学習節に比べて非常に少ない場合があるが，そのような場合でも，DVDL ヒューリスティックでは重複回数がより多くなるので，重複学習節によるソルバーの性能向上の恩恵をより大きく受けられることが期待される．

4. 実装

本節では，提案手法の実装およびパラメータの設定について説明する．

提案手法は，CaDiCaL(SAT competition 2021 release) ソルバーを土台にして DVDL ヒューリスティックの機能を追加することで実装した．CaDiCaL は，かつての minisat がそうであったように，コードの理解や変更が容易であり，最先端のソルバーに比べて性能が大きく劣ることがないことが特徴である．また，sat competition 2021 から CaDiCaL Hacks トラックが追加されるなど，CaDiCaL は標準的なソルバーとして認知されている．これらの点から，本研究での利用に適していると判断した．

予備実験の結果，含まれる決定変数の個数が小さい学習節（特に 0，1 個の場合）を，DVDL ヒューリスティックに従って扱うときにソルバーの性能が悪くなる現象が起きることが分かった．決定変数の個数が小さい学習節はキーが短く，関連性が弱い学習節でも類似と見なし，ソルバーは実際に有効でない学習節もコア層に移動し，性能が悪くなると考えられる．

そのため，実験で使用した提案手法の実装では，学習節に含まれる決定変数の個数が 1 以下である場合，学習節に DVDL を適用しなかった．決定変数の個数が 2 である場合は DVDL のパラメーター $core\_dup$ を 15，2_dup を 5，lbd_limit を 12 にし，個数が 3 以上である場合は DVDL のパラメーター $core\_dup$ を 10，2_dup を 5，lbd_limit を 12 にした．

一方，従来手法の DL ヒューリスティックを実装（cadical-dl）のパラメーターは $core\_dup = 3$，$2\_dup = 2$，$lbd\_limit = 12$ と設定した．

5. 比較実験

提案手法を評価するため比較実験を行う．

5.1 実験の設定

比較ソルバー：以下の 3 つの CDCL ソルバーを比較する．

- *cadical*: CaDiCaL(SAT competition 2021 release) ソルバー [*1]
- *cadical-dl*: 従来手法の DL ヒューリスティックを cadical に実装したソルバー
- *cadical-dvdl*: 提案手法の DVDL ヒューリスティックを cadical に実装したソルバー

ベンチマーク：SAT Competitions 2020[*2] と 2021[*3] で配布されているベンチマーク CNF のうち，メイントラックで利用されたすべてのインスタンス．2020 年の 400 個と 2021 年の 400 個，合計 800 個の CNF からなる．

評価方法：求解数 (SAT+UNSAT) と PAR-2 スコアで各ソルバーの性能を評価する．制限時間は 7000 秒に設定し，その時間内に SAT あるいは UNSAT と報告できた場合に問題を解けたといい，時間内に結果を報告できなかった場合には問題を解けなかったという．PAR-2 スコアは，解けた問題の求解時間の総和に，解けなかった問題について制限時間の 2 倍のペナルティを足し合わせたスコアである．sat competition におけるソルバーの順位付けにおいて利用されている．

計算環境：Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz と 32G メモリを搭載した Linux マシン．

5.2 実験結果と考察

表 2 は各ソルバーの求解数 (SAT+UNSAT) と PAR-2 スコアである．図 2 はそのカクタスプロットである．

提案した cadical-dvdl は cadical と比較すると，求解数が 12 (12+0) 問増加している．ここで，12+0 は出力が SAT となるものが 12 問，UNSAT が 0 問ということを意味する．PAR-2 スコアの観点では約 2.01%の性能向上が見られる．ここで，PAR-2 スコアは解を求めるのに要する時間をはかるための尺

*1 https://github.com/arminbiere/cadical/tree/sc2021
*2 https://satcompetition.github.io/2020/downloads
*3 https://satcompetition.github.io/2021/downloads

表 2: ソルバーごとの求解数 (SAT+UNSAT) と PAR-2 スコア

	SC2020				SC2021			
Solver	SOLVED	SAT	UNSAT	PAR-2	SOLVED	SAT	UNSAT	PAR-2
cadical-sc2021	228	121	107	2696328	267	124	143	2158525
cadical-sc2021-dl	226	119	107	2734950	262	121	141	2236884
cadical-sc2021-dvdl	238	131	107	2616621	269	126	143	2140632

度であり，値が小さいほど，より早く求められることを意味する．

cadical-dvdl は cadical-dl と比較すると，求解数が 19 (17+2) 問増加し，PAR-2 スコアの観点で約 4.31%の性能向上が見られる．

DL ヒューリスティックを実装した cadical-dl は比較ソルバーの中でもっとも性能が悪かった．理由としては，実装方法と使用したベンチマークの違いによる影響の可能性が考えられる．DL ヒューリスティックは土台のソルバーのコードに対して比較的小さい変更だけで実装が可能であり，実装方法だけで性能に大きな差が出ることは考えにくいが，パラメータの設定や実装上の工夫など論文に明確に記述されていないことが影響した可能性が考えられる．

以上をまとめると，提案手法を実装した cadical-dvdl は比較ソルバーの中で一番良い性能を持つことが確認され，特に結果が SAT のインスタンス対して有効であることが分かった．

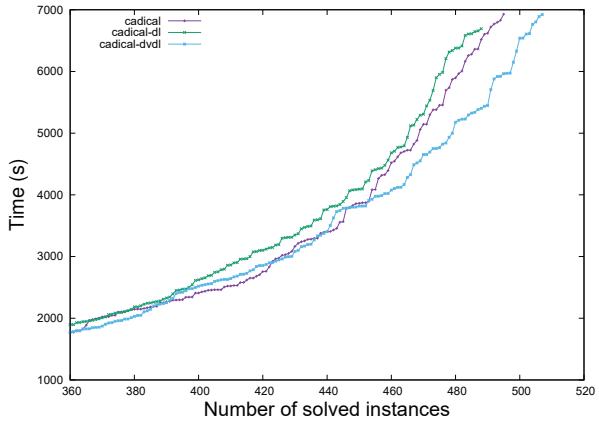


図 2: ソルバーごとの求解時間の比較

6. おわりに

本論文では，CDCL ソルバーの最新の学習節データベース管理手法のひとつである DL ヒューリスティックの改善として，DVDL ヒューリスティックを提案した．DL ヒューリスティックでは学習節の重複を考慮して，3 層構造に基づく学習節の削除ポリシーを決定するのに対して，DVDL ヒューリスティックでは学習節の類似性に着目して，類似の学習節を重複とみなす．これにより，DL ヒューリスティックよりも多くの重複が現れるようになり，重複学習節によるソルバーの性能向上の恩恵をより大きく受けられることが期待される．

SAT Competitions 2020 と 2021 のベンチマーク（合計 800 個）を用いて，オリジナルの cadical，cadical に DL ヒューリスティックを実装したソルバー，cadical に DVDL ヒューリスティックを実装したソルバーの比較実験を行った．その結果，提案手法の DVDL ヒューリスティックは求解数と PAR-2 スコアの両観点でもっとも性能が高いことが確認された．

今後の課題として，より適したパラメータの探索，別の観点から類似関係を与えることの検討などが挙げられる．

参考文献

[Kochemazov 20] Stepan Kochemazov, Oleg Zaikin, Alexander A. Semenov, Victor Kondratiev: Speeding Up CDCL Inference with Duplicate Learnt Clauses, ECAI 339-346, (2020).

[João 09] João P. Marques-Silva, Inês Lynce, and Sharad Malik: Conflict-driven clause learning SAT solvers, in Handbook of Satisfiability, volume 185 of Frontiers in Artificial Intelligence and Applications, 131–153, (2009).

[Audemard 09] Gilles Audemard and Laurent Simon: Predicting learnt clauses quality in modern SAT solvers, in International Joint Conference on Artificial Intelligence (IJCAI), pp. 399–404, (2009).

[Oh 15] Chanseok Oh: Between SAT and UNSAT: the fundamental difference in CDCL SAT, in International Conference on Theory and Applications of Satisfiability Testing (SAT), volume 9340 of Lecture Notes in Computer Science, pp. 307–323, (2015).

[Järvisalo 12] Matti Järvisalo, Marijn Heule, and Armin Biere: Inprocessing rules, in International Joint Conference on Automated Reasoning (IJCAR), vol- ume 7364 of Lecture Notes in Computer Science, pp. 355–370, (2012).