2I1-3

ポートフォリオ型戦略の導入による結論発見システム SOLAR の効率改善

A Portfolio Strategy for the Consequence Finding System SOALR

村松 匠[*1] 鍋島 英知[*2]
Takumi Muramatsu Hidetomo Nabeshima

[*1]山梨大学大学院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 portfolio strategy for the consequence finding system SOLAR to improvement the performance. SOLAR has multiple search strategies and it is difficult to choose the best strategy to a given consequence finding problem by a user. For this problem, we introduce a portfolio search strategy which extracts a feature vector from a given problem, and selects an apposite strategy and the parameters for solving the problem efficiently. We construct portfolio search strategies by using various kinds of machine learning algorithms and evaluate the strategies. The experimental results show the usefulness of the portfolio search strategies.

1. はじめに

本研究では，一階述語論理の結論発見器 SOLAR [1] に対し，その実用性改善のためポートフォリオ型戦略を導入する．SOLAR とは一階述語論理で記述された結論発見問題を解くシステムである．**結論発見** とは公理集合から自明ではない興味深い結論を導出することをという．結論発見は演繹・帰納・発想推論を実現する手段として，多くの応用分野に適用できる有用な枠組みである．結論発見を行う手続きとして, **SOL 導出** [2] がある．SOLAR は SOL 導出を**タブロー計算法** [3] で再定式化した SOL タブロー計算法に基づく効率のよい実装である．公理集合から導出可能な結論は一般に無限にあるため, SOLAR ではその中でもユーザーが指定する特定の条件を満たす結論を発見する．また SOALR は結論発見システムであるが, 結論発見は反駁発見 (定理証明) の一般化であるため, SOLAR は定理証明器としても利用可能である．

現在, SOLAR には多数の探索戦略, パラメータが存在する．しかし, ユーザーが全ての問題に対して適切にパラメータを選択するの困難である．そこで, 本研究では SOLAR の実用性改善のため，問題の特徴量から適切な探索戦略を自動決定する分類器を種々の学習アルゴリズムにより構築することで，より効果的に結論を枚挙できることを示す．

2. 諸定義

2.1 SOL 導出 [2]

まず諸定義を行う．述語とその否定を**リテラル**という．そしてリテラルを選言 ($\vee$) で繋げたものを**節**という．生成領域 $\mathcal{P}$ とは求めたい結論 (**特徴節**) の条件であり, リテラルの集合 L と特定条件 $Cond$ によって定義される．ある節 C に含まれる全てのリテラルが L に含まれ, かつ C が特定条件 $Cond$ を満たしている時, C は $\mathcal{P}$ に**属する**という．節集合 Σ に対して, Σ の論理的帰結で生成領域 $\mathcal{P}$ に属する集合を $Th_{\mathcal{P}}(\Sigma)$ とし, $\mu\Sigma$ を Σ 中の任意の節により，真に包摂されない節集合と定義する．**節タブロー** T とはラベル付き順序木であり, T の根以外のノードがリテラルでラベル付けをされている．基本的にリテラルとノードは同一視する．あるノードの直下のリテラルが $L_1, L_2, \cdots, L_n$ である時，節 $L_1 \vee L_2 \vee \cdots \vee L_n$ を**タブロー節**と呼ぶ．特に根の直下のタブロー節を**先頭節**と呼ぶ．また, 根以外の全ての非葉ノード L が直下の子として $\neg L$ を持つタブローを**連結タブロー**と呼ぶ．**マーク付きタブロー**とはいくつかの *closed* または *skipped* で印づけたされた節タブローである．マーク付けをされていないタブローは**サブゴール**と呼ばれ，後に出てくる推論規則を用いることで印づけすることができる．タブロー節 T が**解けた**とは，T の全てのノードに対して *closed* または *skipped* で印づけされたタブローの状態である．*skip(T)* を *skipped* でマーク付けされたリテラルの集合とする．**選択関数** ϕ とはタブローからサブゴールへの写像である．本稿では，選択関数は後述されるサブゴール選択関数で決定したサブゴールを返す写像とする．

2.2 SOL タブロー計算法 [3]

本節では **SOL タブロー** を定義する．Σ を**節集合**，C を**節**，$\mathcal{P}$ を**生成領域**，ϕ を**選択関数**とする．$\Sigma + C$ と $\mathcal{P}$ からの ϕ による節 S の **SOL 演繹**は次の条件を満たす, タブロー列 $T_0, T_1, \cdots, T_n$ である．

1. T_0 は，先頭節 C のみからなるタブローである．
2. T_n は解けたタブローである．
3. 各 T_i において，$skip(T_i)$ は $\mathcal{P}$ に属する．
4. T_{i+1} は以下の手順で導出する．まず，ϕ によりサブゴール K を選択する．次に，以下の推論規則のいずれかを T_i に適用し T_{i+1} を得る．なお, サブゴールの選択方法と, 推論規則の選択方法については, 次章で詳細を示す．

   a Skip : もし，$skip(T_i) \cup K$ が，$\mathcal{P}$ に属するならば K に *skipped* とラベル付けする．

連絡先: 山梨大学大学院医学工学総合教育部 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 : muramatsu@nabelab.org

b Factoring：もし $skip(T_i)$ がリテラル L を含み K と L が θ によって単一化可能ならば， K に *skipped* とラベル付けし，T_i に θ を適用する．

c Extension ：$\Sigma \cup C$ から 節 B を選択し変数の名前替えをした亜種を $B' = L_1 \vee L_2 \vee \cdots \vee L_m$ とする．もし，$\neg K$ とリテラル L_j が θ によって単一化可能ならば，サブゴール K に新しい子 $L_1, L_2, \cdots, L_m$ を付加し，L_j には *closed* とラベル付けする．得られたタブローに θ を適用する．

d Reduction：もし, K が祖先ノードに L を持ち，$\neg K$ と L が θ によって単一化可能ならば， K に *closed* をラベル付けするし，T_i に θ を適用する．

本稿では SOL タブロー計算法における 4 種類の推論規則を**オペレータ**と呼ぶ．また, SOL タブロー法における探索空間を表現するためにタブロー探索木 [2] を定義する．**タブロー探索木** $\mathcal{T}$ とは各ノードがタブローでラベル付けされた木であり，以下の手順で構成される．

1. $\mathcal{T}$ の根は先頭節 C のみからなる タブローである．
2. $\mathcal{T}$ の葉ではない各ノード T は T のサブゴール $\phi(\ T\ )$ にオペレータを適用して得られるタブローを子に持つ．

3. SOLAR の実行パラメータ

SOLAR は探索戦略などを指定する 3 種類の実行パラメータを持っており, ユーザーは結論発見問題を解く際に, それらのパラメータを指定することができる．

3.1 探索戦略

探索戦略はタブロー，またはタブロー探索木をどのような順序で探索するのかを決定する．以下に SOLAR の扱える探索戦略戦略の一部を示す．

- DFID :タブローに対して深さ優先反復深化探索を行う．
- DFIDO [4]: タブロー探索木の深さに対する DFID 探索を行う．
- RED [4]: 与えられた制限時間内で全ての可能性 (タブロー探索木における兄弟ノード) に対し均等に探索を行う．

3.2 サブゴール選択関数

サブゴール選択関数は, タブローから，オペレータを適用すべきノードを選択する．SOLAR で利用可能なサブゴール選択関数の一部を以下に示す．

- sym： リテラルに含まれる定数項, 関数項の数が少ない一般的なノードを優先的に選択する．
- rsym： リテラルに含まれる定数項, 関数項の数が多い具体的なノードを優先的に選択する

3.3 オペレータ選択関数

オペレータ選択関数はサブゴール選択関数により選択されたノードに対し，どのオペレータを適用するかを決定する．以下は, SOLAR で指定可能なオペレータ選択関数の例である．

- ext： 探索木における分岐数が少なくなるオペレータを選択する．
- rext： 探索木における分岐数が多くなるオペレータを選択する．

4. ポートフォリオ型戦略の導入

SOLAR では，探索戦略，サブゴール選択関数，オペレータ選択関数のそれぞれをユーザが指定可能である．本稿では，これらの組合せを**実行パラメータ**と呼ぶ．実行パラメータの選択によって，しばしば求解時間が変化する．また最適な実行パラメータは問題に依存し，それを人手により事前に選択することは困難である．さらに結論発見問題には，一般に無限の解が存在するため，与えられた制限時間内に効率よく解を列挙するためにも適切な実行パラメータの選択は重要となる．そこで SOALR にポートフォリオ型戦略を導入し, 問題の特徴に応じて自動で実行形式を指定する手法を提案する．本研究では, 以下の 4 つの分類器の構築を行った．

1 単一型分類器：1 つの分類器を用いて, 3 つのパラメータを一意に決定する．

2 複合型分類器：3 つの分類器を用いて, 探索戦略, サブゴール選択関数, オペレータ選択関数のそれぞれを決定する．

3 多段型分類器：2 値分類を行う分類器を多段接続する．各 2 値分類器 c_i はある実行パラメータの組合せ P_{ci} に対応する．各分類器 c_i は，与えられた問題を解くのに実行パラメータ P_{ci} が適しているかどうかを判定し，適さなければ後続の分類器に判定をゆだねる．

4 多段アンサンブル型分類器：多段型と同様に 2 値分類器を多段接続して, 実行パラメータが適しているかを判定する．多段型との違いとして, 各分類器 ac_i は複数の 2 値分類器の集合である．最終的な ac_i の判定は内部の分類器集合による多数決によって決定する．

4.1 特徴量の選定

ポートフォリオ型戦略導入のために，問題から特徴量を抽出した．静的な特徴量として，高速な定理証明器のひとつである E[5] に用いられているものを参考にし, 問題中の節数やリテラル数, 項の深さの平均など 53 種類を抽出した．また動的な特徴量として, SOLAR をデフォルトのパラメータで 10 秒間実行し, 推論速度や各オペレータの適用回数など 84 種類を抽出した．動的な特徴量を抽出する際の SOLAR の実行パラメータは探索戦略が DFID, サブゴール選択関数が SYM/ext (サブゴールとして，具体的かつ適用可能なオペレータ数が少ないものを選択)，オペレータ選択関数が ext (サブゴールの子供ノード数が少なくなるオペレータを選択) である．

5. 分類器の構築と評価実験

分類器構築のため, 訓練事例として定理証明のためのベンチマーク問題集 TPTP-v3.5.0 に含まれる CNF 形式の問題 6346 問を用いた．実験環境は Core2 Duo 3.00Ghz，4GB RAM である．実行パラメータは，探索戦略から 3 種類，サブゴール選択関数から 26 種類， オペレータ選択関数から 11 種類との計 858 種類である．また，一問あたりの制限時間は 60 秒とした．

実験結果を表 1 に示す．解けた総問題数は 858 種類の実行パラメータのうち少なくとも 1 つ以上の実行パラメータで解けた問題数を示す．共通で解けた問題数は，どの実行パラメータでも解けた問題数を示す．単体パラメータで解けた最大問題数は 858 種類の実行パラメータの中で最も解けた数を示す．

実験結果より， 858 種類のパラメータをもし適切に選択することが可能であるならば, 6346 問中 3221 問解けることが分

表 1: 訓練事例収集実験

総問題数	6346 問
解けた総問題数	3221 問
共通に解けた問題数	1336 問
単体実行パラメータで解けた最大問題数	2386 問

表 2: 予備実験

静的特徴量のみ		動的特徴量も含む	
分類器名	正分類数	分類器名	正分類数
RSS	2158	DecisionStump	2354
Logit Boost	2144	RSS	2340
RILB	2142	J48	2228

かる．また, ある単体実行パラメータを用いれば最大 2386 問解くことが可能であるため, これが提案手法のベースラインとなる．なお，単体で最も解けた実行パラメータの内訳は，探索戦略が RED，サブゴール選択関数が SYM/ext ，オペレータ選択関数が sym (代入に含まれる項数が少ないオペレータを選択) であった．

単一型，複合型, の分類構築には, データマイニングツールボックス Weka[6] に含まれているアルゴリズム 21 種類を用いた．多段アンサンブル型分類器の学習は Weka に含まれる AdaBoost [7] を使用した．多段型分類器の構築には, 2 値分類器 SVM の実装のひとつである LibSVM[8] を使用した．

まず, 予備実験として訓練事例や各事例の与える分類クラスの妥当性を検証した．分類器構築にあたり, 各事例に対する分類クラスとして, その事例を最も速く解くことができた実行パラメータ (これを**最速分類クラス**と呼ぶ) を与えた．総分類クラス数は 526 個である．分類器を作成するための訓練事例として, 少なくともひとつの実行パラメータで解けた事例 3221 問を用いた．

分類器構築に静的特徴量のみを使用した場合と，静的＋動的特徴量を使用した場合の結果を，表 2 に示す．以降, 全ての実験結果は 10 点交差検定の結果である．表には, 最も優れた性能を示した学習アルゴリズム上位 3 つまでを示している．分類器名は分類器を作成したアルゴリズム名を示す．正分類数は正しく分類クラスを出力した総数を示す．

表 2 より，いずれの結果もベースラインを下回っていることが分かる．精度が上がらない原因として，与える分類クラス数が 500 以上だったため学習が困難であったと考えられる．また，分類クラスが最速分類クラスであったため，問題を多く解くという観点では精度が上がらなかった可能性がある．さらに, どの実行パラメータでも共通に解ける問題に関して考慮をしていなかったことも学習が困難になった一因と考えられる．

そこで，ある事例に対する分類クラスとして，その事例を解くことができた実行パラメータのうち，求解数が最大のもの (これを**最大求解数分類クラス**と呼ぶ) を与え，再度分類器を構築した．この場合，分類クラス数は 100 個となった．訓練事例として全 3221 問から最大求解分類クラス 100 個で共通に解ける事例数 1445 問を除いた 1776 問を用いた．共通に解ける問題を除く理由は，訓練事例に学習に不必要な情報 (ノイズ) を含めないようにするためである．

最大求解数分類クラスを用い，分類器構築に静的特徴量のみを使用した場合の単一型, 複合型, 多段型, 多段アンサンブル型の実験結果を 表 3 に示す．同様に静的＋動的特徴量を使用した場合のそれぞれの実験結果を，表 4 に示す．正分類数は分類器が正しく分類クラスを分類した数を示す．求解数は解ける問題数を示す．総求解数は求解数に共通に解ける問題数 1445 問を加えた数である．

表 3 の考察として, 静的特徴量のみの実験結果は, 予備実験で目標となったベースラインの 2386 問を超える結果となった．総求解数は 2606 問となり, 全 3221 問中の 80 % をカバーできる結果となった．表 4 でも同様に動的特徴量を含めた場合の総求解数は 2613 問で, 全 3221 問中 81% をカバーできる結果となった．また予備実験で設定されたベースラインを超える結果となり, 新たに設定した分類クラス, 訓練事例は分類に有用であったと考えられる．しかし, 静的特徴量のみと比べると総求解数にあまり差異はない．その原因として, 動的特徴量と与えたデータがあまり分類に有用ではなかったため, 分類が複雑化してしまった可能性が考えられる．よって今後の課題として, より有用な動的特徴量を模索していくことが挙げられる．

6. マルチクラスラベリング

前述の実験では, 各事例に与えた分類クラスは 1 つであった．しかし, それぞれの事例に対して解ける実行パラメータはひとつでなく複数存在する場合が多い．そこで, 各事例に 1 つ以上の分類クラスを与える**マルチクラスラベリング**を評価した．分類器構築には, Weka のツールキットである **Mulan** [9] に実装されている分類アルゴリズムを用いた．各事例には, その事例を解くことができる複数の実行パラメータを分類クラスとして与えた．分類器作成のための訓練事例数は, 全問題から最大求解数分類クラスで共通して解ける問題数 1445 問を除いた 1776 問である．

静的特徴量のみの結果を表 5 に動的特徴量を含めた結果を表 6 に示す．考察として, ベースラインである 2386 問を超える結果となり, 静的特徴量のみ場合では前述の実験結果よりも最大求解数 30 問ほど向上する結果となった．また, 3221 問中 82 % をカバーできる結果となった．しかし, 静的特徴量のみと動的特徴量も含めた場合では, 静的特徴量のみの方が最大求解数が多いという結果になってしまった．この結果からも, 動的特徴量の選定が今回の分類に適していなかった可能性がある．

7. まとめと今後の課題

本研究では, 結論発見システム SOLAR にポートフォリオ型戦略を導入することで効率改善を図った．そのため 5 種類の分類器を，様々な機械学習アルゴリズムを用いて構築し評価実験を行った．

評価実験により, 単一パラメータによる最大求解数の 2386 問よりも約 230 問多く問題解決できることを示した．今後の課題として, 動的な特徴量を含めた結果の伸びがあまり芳しくなかったので, より有用な動的特徴量の選定を行うことが挙げられる．

謝辞

本研究の一部は科学研究費補助金 基盤研究 (A) の助成を受けたものである．

表 3: 静的な特徴量のみの単一型, 複合型, 多段型, 多段アンサンブル型

単一型				複合型			
分類器名	正分類数	求解数	総求解数	分類器名	正分類数	求解数	総求解数
RotationForest	1026	1161	2606	KStar	897	1136	2581
IBk	907	1157	2602	IBk	838	1123	2568
KStar	946	1156	2601	IB1	804	1116	2561
多段型				アンサンブル型			
分類器名	正分類数	求解数	総求解数	分類器名	正分類数	求解数	総求解数
LibSVM	1028	1084	2529	AdaBoost	941	960	2405

表 4: 動的特徴量を含める単一型, 複合型, 多段型, 多段アンサンブル型

単一型				複合型			
分類器名	正分類数	求解数	総求解数	分類器名	正分類数	求解数	総求解数
RotationForest	1033	1168	2613	RotationForest	948	1137	2582
PART	915	1148	2593	J48graft	876	1124	2569
END	1050	1143	2588	J48	804	1121	2566
多段型				アンサンブル型			
分類器名	正分類数	求解数	総求解数	分類器名	正分類数	求解数	総求解数
LibSVM	941	960	2405	AdaBoost	1009	1051	2496

表 5: 静的特徴量のみのマルチクラスラベリング

分類器名	求解数	総求解数
MLKNN	1197	2642
BRKNN	1194	2639
IBIL_ML	1161	2606

表 6: 動的特徴量を含めるマルチクラスラベリング

分類器名	求解数	総求解数
MLKNN	1155	2600
IBIL_ML	1140	2585
BRKNN	1121	2566

参考文献

[1] Hidetomo Nabeshima, Koji Iwanuma, Katsumi Inoue, and Oliver Ray. Solar : An automated deduction sysytem for consequence finding. Journal of AI Communications, Vol. 23, No. 2-3, pp183 - 203, 2010

[2] Katumi Inoue. Linear resolution for consequence finding. Artificial Interlligence, Vol. 56, pp. 301-353, 1992.

[3] Koji Iwanuma, Katsumi Inoue , and Ken Satoh. Completeness of pruning methods for consequence finding procedure SOL. In Procedings of FTP - 2000, pp89-100, 2000

[4] Kenshirou Suzuki, Hidetomo Nabeshima, Koji Iwanuma. A multiple search strategy for consequenace finding procedure SOL tableau calculus. SIG-FPAI-B003,pp 19 - 24, 2010.

[5] Stephan Schulz. E - A brainiac theorem prover. Journal of AI Communications, Vol .15, No 2-3, pp 111-126,2002.

[6] Weka 3: Data Mining Software in Java, http://www.cs.waikato.ac.nz/ml/weka/, 2011/4/30

[7] ブースティング, http://www.eb.waseda.ac.jp/ murata/ryotaro.nishino/openhouse/boosting3.php/ , 2011/4/30

[8] LIBSVM – A Library for Support Vector Machines, http://www.csie.ntu.edu.tw/ cjlin/libsvm/, 2011/4/30

[9] Mulan: A Java Library for Multi-Label Learning, http://mulan.sourceforge.net/, 2011/4/30