論　文

Technical Papers

木パターン上の決定リストの学習とその推論制御への応用

Learning Decision Lists over Tree Patterns and Its Application to Inference Control

小林　聡*1　堀　浩一*2　大須賀　節雄*2
Satoshi Kobayashi　Kohichi Hori　Setsuo Ohsuga

*1 電気通信大学情報工学科
Dept. of Computer Science, The University of Electro-Communications, Tokyo 182, Japan.
*2 東京大学工学部
Faculty of Engineering, The University of Tokyo, Tokyo 153, Japan.

1992年11月5日　受理

Keywords: decision lists, decision trees, tree patterns, PAC learning, explanation based learning.

Summary

This paper introduces a new concept, *a decision list over tree patterns* (*DLTP*), which is a natural extension of a decision list, for dealing with tree structured objects.

First, we show a theoretical result of the learnability of this class. We define the class, k-node-DLTP, which is a subclass of decision lists over tree patterns whose number of tuples is bounded by k+1. A hardness result on the learnability of this class is shown. Then we propose a practical learning algorithm based on an information theoretical evaluation function. This algorithm is an extension of Quinlan's ID3 algorithm.

One of the most interesting application areas of this work is inference control. We can use decision lists over tree patterns for controlling the application of rules in SLD resolution process. We apply the proposed algorithm to learning strategies for applying rules and experimental results in several domains are presented. These experiments show the effectiveness of the proposed heuristic evaluation function for finding small size hypotheses. From the view point of Explanation Based Learning, our method can be regarded as a method for refining domain rules so as not to backtrack while solving training examples.

1. はじめに

学習研究は，知識獲得ボトルネックを解決する一つのアプローチとして工学的に注目を浴びているが，その理論と実際のギャップが大きいことも事実である．その最も重要な要因の一つは，学習問題が組合せ論的爆発の問題をかかえており，効率の良いアルゴリズムの開発が困難であることである．この解決には，人間が行っているように，学習に利用可能なさまざまな情報を利用することが考えられる．その一つのアプローチとして，領域知識を積極的に利用した学習の枠組みである，説明に基づく学習(Explanation Based Learning，以下EBLという)[Mitchell 86]が提案され，多くの研究が行われている．EBLでは，学習された知識が常にその領域知識から演繹的に導かれることが保証されるという好ましい性質を持つが，多くの未解決問題も残されている[Minton 89, Mitchell 86, Rajamoney 87]．

本研究では，Utility問題[Minton 89]を取り扱う．

Utility問題は，EBLによって学習された結果が必ずしもシステムの計算効率を改善するとは限らないという問題であるが，その解決のために操作性規範を改善する必要性が指摘されてきた[Minton 89, 山田 89,

山村 89]．本研究では，与えられた例題集合に対し，バックトラックしない知識であることを操作性規範とする．もちろん，常にどのようなプログラムもバックトラックをしない記述に変更できるとは限らないが，バックトラックは計算コストのなかで大きな割合を占める要因の一つであり，本研究は，Utility 問題を解決するための一つのアプローチであると考えられる．

2章で，ルールの選択を制御するための制御構造として，木パターン上の決定リストを導入する．そして，**3章**においてその学習可能性に関する議論を行い，**4章**で実際的なアルゴリズムが示される．**5章**では，そのルール適用条件学習への応用を示す．

2. 木パターン上の決定木および決定リスト

木パターンは，項書換えシステムや，論理プログラムなどに幅広く応用されている言語である．その学習に関する研究もさかんに行われている[Arimura 91, Plotkin 70]．この章では，決定リスト[Rivest 87]を拡張して木パターンをデータとして取り扱うことができるようにする．まず，木パターンを次のように定義する．

Σ をファンクタを表す有限のアルファベットとし，V を Σ と共通の要素を持たない変数を表すアルファベットの集合とすると，$\Sigma \cup V$ 上の木パターンは次のように帰納的に定義される[Arimura 91]．

(1) 引数の数が 0 であるようなファンクタ a は木パターンである．

(2) 変数は木パターンである．

(3) 順序木 $f(t_1, \cdots, t_n)$ は，木パターンである．ここで，f は n 引数のファンクタであり，各 t_i $(1 \leq i \leq n)$ は木パターンである．

また，変数がすべて異なるような木パターンを正則であるという．

ここで，定義された木パターンを本稿では，木構造とみなすことがある．例えば，$\{a, b, \cdots\} \cup \{X, Y, \cdots\}$ 上の木パターン $a(b(c(X, e(f)), g(h, i(Y))), j)$ は，図 1 のような木構造とみなされる．

また，$\Sigma \cup V$ 上の木パターンすべてからなる集合を $TP(\Sigma)$ と表す．

代入とは，引数の数が 0 の木をそれ自身に移すような木パターンから木パターンへの準同形写像 θ である．ここで，木パターン p, q について，$p=\theta(q)$ ならば，$p \leq' q$ と表す．例えば，$a(b(c, e), e, f) \leq' a(b(c, X), X, Y)$ である．木パターン p が定義する言語とは，次のように定義される木パターンの集合 $L(p)$ である．

$$L(p)=\{t \in TP(\Sigma) : t \leq' p\}$$

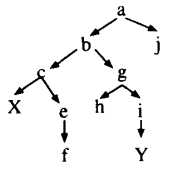


図 1 木パターン

【定義 1】 木パターン上の決定リストとは，

$$(tp_1, c_1), (tp_2, c_2), \cdots, (tp_r, c_r), (X, c_{r+1})$$

なる 2 項組の列である．ここで，各 tp_i は $TP(\Sigma)$ の要素であり，各 c_i はクラスの集合 C の要素である．X は変数ただ一つからなる木パターンを表す．

木パターン上の決定リストは，$TP(\Sigma)$ から C への写像を次のように定義する．ある木パターン t が与えられると，決定リストの最初の要素 (tp_1, c_1) から順に，$t \leq' tp_i$ かどうかがテストされる．もし，$t \leq' tp_i$ なら，c_i を関数値として返し，そうでなければ，次のリストの要素に対して同様の操作を繰り返す．

例えば，

$$(a(X, X), c_1), (a(X, Y), c_2), (b(a(X), Y), c_3), (X, c_4)$$

なる決定リストでは，$a(b, c)$ は c_2 に分類され，$b(a(c), e(Z))$ は c_3 に分類される．

ここで，この決定リストを SLD 導出手続きにおける知識の選択のための制御メカニズムとして利用できることに注目してほしい．すなわち，入力の木パターンは現在導出手続きにおいて選択されているサブゴールであり，出力のクラスは適用すべきルールの id である．もしそのような決定リスト

$$(tp_1, c_1), (tp_2, c_2), \cdots, (tp_r, c_r), (X, c_{r+1})$$

が獲得できれば，それを用いて次のようにルールを特殊化できる．$i=1$ から順に，tp_i とルール $c_i : A \leftarrow B_1, \cdots, B_n$ のヘッド A の最汎単一化子を θ とすると，c_i を $\theta(A \leftarrow !, B_1, \cdots, B_n)$ に特殊化して知識集合（最初は空集合）の最後に加えていく．このような特殊化は得られた知識選択の決定リストの制御情報を忠実に実現する．ただし，対象とするルールはホーン節で記述され，しかも各ホーン節の引数は，入力引数と出力引数に分けられ，入力引数はサブゴールとして選択されたときには常に定数になっており，出力引数は互いに異なる変数になっているものとする．

例えば，次のようなルールを考える．

r 1 $del([X|Y], X, Z) : -del(Y, X, Z).$

r 2 $del([X|Y], Z, [X|W]) : -del(Y, Z, W).$

r 3 $del([], X, []).$

このルールは，$del(X, Y, Z)$ において，リスト X に現れる Y という要素をすべて削除したリスト Z を返すことを意図して書かれている．しかし，このままでは，この論理プログラムは正しく動かない．ここで，

$(del([], X, Y), r3), (del([X|Y], X, Z), r1), (X, r2)$

なる決定リストでルールの適用を制御すれば，目的の動作が得られる．

本稿で取り扱う問題は，人間から与えられる具体的な正しいトレースから，このような制御構造を学習する問題である．例えば，上記の例で，

$del([1, 2, 3], 2, X_1)$ → r2 を適用 → $del([2, 3], 2, X_2)$ → r1 を適用 → $del([3], 2, X_3)$ → r2 を適用 → $del([], 2, X_4)$ → r3 を適用 → 空節

というトレースが与えられると，木パターンをそれぞれ，$del([1, 2, 3], 2, X_1)$ → r2, $del([2, 3], 2, X_2)$ → r1, $del([3], 2, X_3)$ → r2, $del([], 2, X_4)$ → r3 に分類するような，決定リストを学習する問題に帰着される．

次章では，木パターン上の決定リストの学習問題を理論的に解析する．

3. 理論的背景

木パターン t とそのクラス c の 2 項組を例題と呼ぶ．また，例題の有限集合を**例題集合**と呼ぶ．木パターン上の決定リスト L が定義する関数が，例題集合 S に矛盾しないとき，**L は S に対して無矛盾である**という．

リストの 2 項組の個数がたかだか $k+1$ で抑えられる決定リストで表現し得る関数のクラスを **k-node-DLTP** で表す．

最初に，木パターン上の決定リストについて，次のような問題を考える．

〈**k-node-DLTP 問題**〉

問題例：例題集合 $S=\{s_1, s_2, \cdots, s_n\}$

質問：S に無矛盾な $dl \in$ k-node-DLTP が存在するか．

まず，k-node-DLTP 問題の NP-完全性を示す．還元の方法は，基本的に，[Pitt 88]の定理 3.2 と同様の方針であり，k 彩色可能性問題を還元することによって示される．k 彩色可能性問題は次のように定義される．

問題例：彩色される頂点の集合 $V=\{v_1, \cdots, v_n\}$，および制約(辺)集合 $S=\{s_1, \cdots, s_m\}$．ただし，各制約は，V の 2 要素からなる部分集合である．

問題：各制約内の 2 頂点を同色に塗らないような k 彩色は可能か．

まず，以下の補題を示しておく．

［**補題 1**］ 辺の数が m のグラフ G において，G が k 彩色可能かどうかを判定する問題は，$m \leq k$ ならば多項式時間で判定可能である．

《**証明**》 一般に連結なグラフのみを考えれば十分である．

$m+1 \leq k$ ならば，明らかに k 彩色可能である．

$m=k$ の場合を考察する．

- グラフが木の場合：根を一つ定め，根つき木とする．そして，深さが偶数の頂点を 1 の色で彩色し，深さが奇数の頂点を 2 の色で彩色すれば，制約を違反することはないので，2 彩色可能である．したがって，$2 \leq k=m$ ならば YES，そうでなければ NO と答えればよい．
- グラフが閉路を含む場合：k 彩色可能である．なぜなら，一般に，連結なグラフの頂点の数を n とし，辺の数を m とすると，$m=1$ のときは $n=2$ であり，辺が一つ増えるに従って，頂点はたかだか一つしか増えないから，$n \leq m+1$ が成り立つ．ところで，グラフが閉路を含むなら，頂点の数は少なくとも一つは減るから，$n \leq m$ が成り立つ．頂点の数 n のグラフは n 彩色可能であり，$n \leq m=k$ だから，グラフは k 彩色可能である． □

次に，以下の四つの条件を満たす例題集合を考える．また，クラスは，P(正例)と N(負例)の 2 種類があるものとする．

(a) すべての例題の木パターンは，n 個の子を持つ根が r でラベルされた木である．しかも，根 r の i 番目の子は，必ず，t_i または f_i でラベルされている．

(b) 正例は，r の子のうち一つだけ，f で始まるラベルを持ち，それ以外はすべて t で始まるラベルを持つ．

(c) 負例は，r の子のうち二つだけ，f で始まるラベルを持ち，それ以外はすべて t で始まるラベルを持つ．

(d) 例願集合の木パターンはそれぞれお互いに異なる．

ここで，簡便のため，(T_i, P) なる決定リストの要素を**正の要素**，(T_i, N) なる要素を**負の要素**と呼ぶことにする．

このとき，次の補題が成り立つ．

［**補題 2**］ 条件(a), (b), (c), (d)を満たす例題集合を S とし，その負例の数を m とするとき，S に無矛盾な $dl \in$ k-node-DLTP が存在し，$k<m$ ならば，S に無矛

盾で最後の要素のみを負の要素とするような $dl' \in$ k-node-DLTP が存在する．

《証明》 証明は**付録**に示す． □

【定理 1】 k-node-DLTP 問題は NP-完全である．

《証明》 明らかに k-node-DLTP 問題は NP に属する．

k 彩色可能性問題の問題例から，次のように k-node-DLTP 問題の問題例を構成する．まず，クラス集合は $\{P, N\}$ とする．ここで，P, N はそれぞれ，正例，負例を意味する．正例として，根が r でラベルされ，n 個の子を持つ木を n 個構成する．ただし，i 番目の木の左から j 番目の子のラベルは，$i \neq j$ なら t_j とし，$i=j$ のときは f_j とする．つまり，子のラベルが一つだけ f で始まるラベルで，他のラベルはすべて t で始まるラベルとなっているものを n 個構成する(図 2 (b))．負例としては，各制約 s_i に対して根が r でラベルされ，n 個の子を持つ木を次のように生成する (図 2 (c))．$s_i = \{v_{i1}, v_{i2}\}$ のとき，i_1 番目と i_2 番目の子のラベルをそれぞれ f_{i1}, f_{i2} とし，それ以外の子のラベルは j 番目の子なら t_j とする．

このような構成が多項式時間でできることは明らかである．よって，k 彩色可能性問題が解を持つことと，このように構成された k-node-DLTP 問題が解を持つことが同値であることを示せばよい．

まず，k 彩色可能であるとする．ただし，少なくとも 1 回は各色が用いられているものとしてよい．このとき，各色 i について，根が r でラベルされ，n 個の子を持つ木パターンを次のように構成する．色 i に彩色されていない頂点の番号を $j_1, j_2, \cdots, j_p$ とすると，j_q $(1 \leq q \leq p)$ 番目の子のラベルをそれぞれ t_{j_q} として，それ以外の子のラベルを互いに異なる変数とする．このようにしてできた木パターンを T_i とすると，$(T_1, P), \cdots, (T_k, P), (X, N)$ なる k-node-DLTP は正しく正例，負例を分類する(図 2 (d))．なぜなら，少なくとも 1 回は各色が使われているので，正例はすべて満足される．負例についても，f で始まるラベルが付けられている子のラベルを f_{i1}, f_{i2} とすると，各色 i について，頂点 v_{i1}, v_{i2} のうち，どちらか一方は色 i で塗られていない．すなわち，T_i に含まれることはない．

逆に，例題集合に矛盾しない $dl \in$ k-node-DLTP が存在するなら，k 彩色可能であることを示す．

まず，補題 1 より，$m \leq k$ なら k 彩色可能かどうかは多項式時間で判定できるので，$m > k$ としてよい．また，構成された例題集合は条件(a), (b), (c), (d)を満たすので，補題 2 の条件が成り立つ．よって，補題 2 より，得られた dl を最後の要素以外を正の要素にした dl' に変更することができる．

また変形された dl' はさらに，その各木パターンを次のようにして，f で始まる頂点を含まないような木パターンに変形することができる．もし，f で始まる頂点が二つ以上あれば，それは正例を一つも含まないので消去することができる．f で始まる頂点がただ一つ f_i であれば，そのような木パターンに含まれる可能性のある正例はただ一つであるから，代わりに，根が r でその i 番目の子どもは X とし，それ以外の j 番目の子どもは t_j とした木パターンに変更することができる．なぜなら，このような木パターンは負例を含むことはないからである．

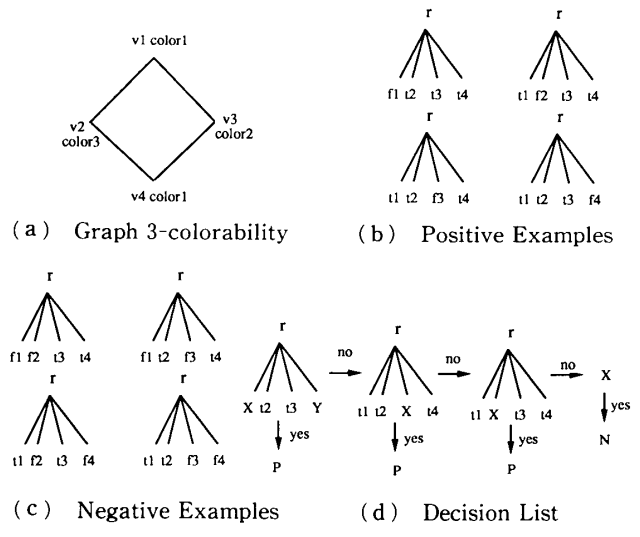
このように変形して得られた $dl'' = (T_1'', P), \cdots, (T_k'', P), (X, N)$ から，次のように頂点 v_j の彩色 c を決める．

$$c = \min\{i : T_i'' \text{ で } j \text{ 番目の子が変数である}\}$$

このような彩色により，制約はすべて満足される．なぜなら，もし制約 s_j が破られて s_j の要素が同じ色 i で塗られたとすると，j 番目の負例が i 番目の木パターン T_i'' に含まれてしまうからである．

以上により，k-node-DLTP 問題が NP-完全であることが示された． □

定理 1 は，例題集合に矛盾しないできるだけリストの要素数の少ない（サイズの小さい）決定リストを求める問題が，NP-困難であることを示している．また，この事実は，木パターン上の決定リストの学習問題が計算量の観点から見て困難であることを示している．実際，確率的近似学習の枠組みとして近年注目を集めている PAC 学習の枠組みでは，NP≠RP ならば k-node-DLTP という関数のクラスは多項式時間で PAC 学習可能でないことが示せる[小林 93]．



(a) Graph 3-colorability　(b) Positive Examples

(c) Negative Examples　(d) Decision List

図 2 還元方法

4. 木パターン上の決定リストの学習

定理1より，k-node-DLTPを効率良く学習するアルゴリズムを見つけるのは困難であると予想される．そこでこの章では，情報論的評価関数を用いて，できるだけリストの要素の数が少ない決定リストを求める手法を与える．決定木の学習は広く研究が行われているが，与えられた例題集合に矛盾しない最小コストの決定木を求める問題もやはり，NP-困難であることが知られている[Hyafil 76]．そこで，Quinlanはできるだけ小さいサイズの決定木を求めるのに，情報論的評価関数を利用する手法[Quinlan 86]や，最小記述長原理そのものを利用する手法[Quinlan 89]を提案している．本研究では，前者の手法を改良して木パターン上の決定リストに適用することを試みた．

Quinlanらの手法との大きな相違は，決定木のテスト集合が明示的に与えられていないということである．テストとなる木パターンは与えられた例題集合からボトムアップに求められる．その際には，最小汎化[Plotkin 70]を利用する．

まず，与えられた例題集合Sをクラス別に分類して，$S_1, S_2, \cdots, S_n$とする．nはクラスの総数である．

ここで，$\mathrm{pat}(S)=\{t : (t, c)\in S\}$, $\mathrm{lgg}'(S)=\mathrm{lgg}(\mathrm{pat}(S))$と定義する．ここで，$\mathrm{lgg}(S)$は，$S$が木パターンの集合であるとき，$S$の要素すべての最小汎化を表す．また，$m_{i,j}=|\{t : t\leq' \mathrm{lgg}'(S_i)\wedge(t, c)\in S_j\}|$, $k_i=\sum_{j=1}^{n} m_{i,j}$と置く．ここで，例題集合$S_1,\cdots, S_n$の複雑さを表す関数$f$を次のように定める．

$$f(\{S_1,\cdots, S_n\})=\sum_{i=1}^{n}\left(|S_i|\cdot\sum_{j=1}^{n}-\frac{m_{i,j}}{k_i}\cdot\log\frac{m_{i,j}}{k_i}\right)$$

直観的に説明すると，

$$\sum_{j=1}^{n}-\frac{m_{i,j}}{k_i}\cdot\log\frac{m_{i,j}}{k_i}$$

は，S_iと他のS_j $(i\neq j)$が干渉しあっている度合いを表していると考えることができる．fはそれら各S_iの複雑さを$|S_i|$で重み付けして加え合わせたものである．

このような設定のもとで，次のようなHeuristicsに従って決定頂点のできるだけ少ないDLTPを求める．

H1 各$i=1,\cdots, n$について，S_iをそのまますべて取り除くことができるなら，取り除く．

H2 **H1**が適用できない場合，できるだけ複雑度fが小さくなるように，木パターンの集合を取り除く．

具体的には，学習アルゴリズムは次のように与えられる．

```
DLTPLEARN
  begin
    L= null list;
    while(∃i(S_i≠φ)) do
      flag=NO;
      for each 1≤i≤n do
        if ∀j (i≠j⇒∀s∈pat(S_j)(s≰'
          lgg'(S_i))) then do
            L の最後に (lgg'(S_i), c_i) を加える；
            S_i=φ ；
            flag=YES;
            break;
        end
      end
      if flag=YES then continue;
      for each 1≤i≤n do e_i=∞;
      for each 1≤i≤n do
        次の条件を満たす S_i の部分集合 S_i' で
          f({S_1,…, S_{i-1}, S_i-S_i', S_{i+1},…. S_n})
          を極小化するようなものを求める；…(*)
            (条件)  ∀j(i≠j⇒∀s∈pat(S_j)(s≰'
            lgg'(S_i')))};
        得られた S_i' に対し，e_i=f({S_1,…, S_{i-1}, S_i
        -S_i', S_{i+1},…, S_n}) と置く；
      end
      {e_1,…, e_n} において最小値を与える e_p を求める；
      L の最後に (lgg'(S_p'), c_p) を加える；
      S_p=S_p-S_p';
    end
    L を出力する；
  end
```

ただし，(*)においては，S_i'の初期候補として空集合を選び，S_i'に順次fを最小化するようなS_iの要素を一つずつ加えていく山登り法を用いる．

次章では，この学習アルゴリズムをルールの適用制御へ応用し，いくつかの実験例を示す．

5. ルールの適用条件学習への応用

この章では，ホーン節に限定したルールの適用条件の学習への応用を示す．適用条件といっても，ホーン節のヘッドのリテラルの特殊化のみをここでは取り扱

う．この場合，問題のトレースにおいて各ルールが適用される直前の選択されたサブゴールが分類対象であり，適用したルールの *id* がクラスとなる．例題集合は，目標概念の導出木から，各導出において選択されたサブゴールと適用されたルールの2項組を抽出することによって得られる．

5・1 ロボット計画立案への応用

ロボット計画立案への応用例として，図3のような連続する4部屋に箱が一つ置いてあり，それを部屋4のゴールgに移動する問題を実験に使用した．ロボットr，箱bの存在場所としては，部屋の中央c，右側の戸のそばrd，左側の戸のそばldの3か所を考え，戸は開いた状態と閉じた状態の2通りを考えた．

また，オペレータとしては，右側の戸へ動くgtrd，左側の戸へ動くgtld，戸をくぐるgtr，戸を開けるopen，箱を持つpick_up，箱を置くput_down，箱へ動くgtb，ゴールへ動くgtgの八つである．この問題の場合，問題例の総数は1 152となる．対象とするルールの一部を図4（a）に示す．

st(b(X1), r(X2), at(X3), have(X4), on(X5), open(X6, X7, X8))で，b(X1), r(X2)のX1, X2はそれぞれ箱，ロボットの部屋(rm1, rm2, rm3, rm4)を示し，at(X3), on(X5)のX3, X5はそれぞれ箱，ロボットの位置（c, rd, ld, g）を表す．have(X4)のX4はロボットが持っているものを表し，何も持っていないときはnilとなる．また，open(X6, X7, X8)でX6, X7, X8は，戸d1，d2，d3が開いているかいないかを表し，開いているときはそれぞれ，d1，d2，d3となり，閉まっているときはnilとなる．

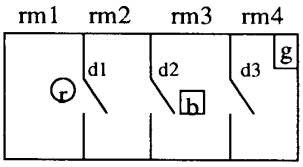


図3 ロボット計画立案

```
gtg:
op(st(b(X1),r(rm4),at(X3),have(X4),on(X5),open(X6,X7,X8)),[gtg|OP]):-
      not(X3=g),
    op(st(b(X1),r(rm4),at(g),have(X4),on(X5),open(X6,X7,X8)),OP).

put_down:
op(st(b(X1),r(X2),at(X3),have(b),on(X5),open(X6,X7,X8)),[put_down|OP]):-
      op(st(b(X1),r(X2),at(g),have(nil),on(X3),open(X6,X7,X8)),OP).
```

（a） robot planning domain knowledge

```
equiv_func_main(X,X):-
  specification(X),
 !.
equiv_func_main(X,Y):-
  equiv_func(X,Z),
   equiv_func_main(Z,Y).

equiv_func(X,X).

equiv_func(neg(neg(X)),X).

equiv_func(and(X,Y),nor(neg(X),neg(Y))).

equiv_func(and(X,Y),and(Z,W)):-
  equiv_func(X,Z),
  equiv_func(Y,W).

equiv_func(neg(X),neg(Y)):-
  equiv_func(X,Y).
```

（b） circuit design domain knowledge

図4 領域知識

この領域知識には，ロボットと箱の位置関係を述べるようなバイアスがなく，バックトラックを行わないようなプログラムに変換するのはかなり複雑になる．本手法の有効性を示すためにあえてこのような領域を選んだ．

70題の問題例を10題ずつランダムに生成し，各問題例に対して人間から解を与えてもらい，そのトレースを獲得する．得られたトレースの集合から，前章で提案した学習アルゴリズムへの例題集合として，〈サブゴール，適用すべきルール〉という2項組の情報を抽出する．そして，学習アルゴリズムを適用し，得られた制御構造が，未知の問題例をどの程度解くことができるかを実験した．具体的には，各学習結果に対して50題の未知の問題例をランダムに生成してその正当率を求めた．実験結果を図5に示す．これらの学習されたルールは当然学習に用いられた例題もすべて解くことができた．70題の例題をもとに学習されたルールの数は35であり，未知の問題例に対する正当率は96%であった．

また，この手法をブロックの積換え問題に対して適用した．三つのブロックをスタックAからスタックBへ積み換える例題を教えたところ，任意個のブロックの積換えを完全に解くことが可能になった．これは，提案した手法がうまく数の一般化を行っていることを示している．

5・2 論理回路設計への応用

論理回路設計への応用として，論理積標準形で書かれたブール関数をNORゲートを利用して設計する問題を考えた．与えた領域知識はド・モルガン則などのブール関数の公式に関する知識と，変形されたブール

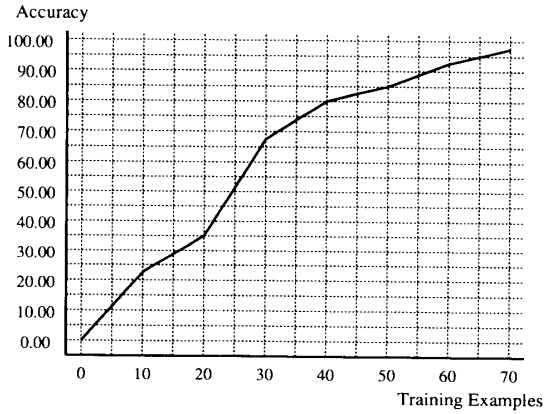


図5 ロボット計画立案におけるCoverage

関数がNORとNOTから構成されなければならないという仕様である．具体的には最初の知識の一部は図4(b)のように与えられる．specificationは仕様を記述する述語である．

この応用例の場合，このままのプログラムでは，公式の適用を一つずつ実行しては仕様との一致を確かめる非常に効率の悪いプログラムである（実際無限ループに入ってしまう）．そこで，equiv_funcなる述語をヘッドに持つルールの適用条件を学習させた．

学習に用いた例題は，まず，問題例を20題生成し，そのなかから5題をランダムに選び，それらを人間に解いてもらったトレースを前節同様に利用して，学習アルゴリズムを適用した．その結果，残りの15題すべての問題を解くことができた．

学習した結果，例えば，次のような関数の変形をバックトラックなしに実現することができるようになった．

and[x, y]⇒nor[neg(x), neg(y)]
and[neg(x), neg(y)]⇒nor[x, y]
and[or[neg(x), a], or[y, b], or[neg(e), neg(f)]]⇒
　　nor[nor[neg(x), a], nor[y, b], nor[neg(e), neg(f)]]
and[x, neg(y), neg(z), w, s, t]⇒
　　nor[neg(x), y, z, neg(w), neg(s), neg(t)]

ただし，上記の記法は簡便化したものであり，and[x, y, z]などは，実際にはand(x, and(y, z))を表している．

5・3 関連研究

ルールの適用条件の学習を取り扱った研究としては，バージョン空間法を利用したLEXがある[Mitchell 83]．LEXの手法と本手法の大きな違いは，LEXが個々のルールが適用可能な概念を学習するのに対し，サブゴールから適用可能なルールを決定する関数として，ルールの制御情報を学習している点にある．また，特に決定リストの要素の順番に意味があることが特徴的である．本稿で提案した手法は，バージョン空間法で定義されるバイアスを，項の構造のなかに組み込むことにより，数式処理のような領域にも適用可能である．

説明に基づく学習の枠組みを，ルールの適用条件学習に適用した例として，PRODIGY[Minton 89]，Meta-LEX[Keller 87]，AXA-EBL[Cohen 90]など，多数の研究がある．これらの研究との違いは，本手法がマクロの生成ではなく，変数の特殊化に焦点を当てており，その理論的な解析，特に学習可能性についての議論を与えている点である．

また，本研究で提案した木パターン上の決定リストは，非常に一般的な概念であり，木構造を持った対象の分類問題などに適用の可能性を持っているものと思われる．

6. おわりに

本稿では，木パターン上の決定リストという概念を提案し，その学習可能性の困難性を理論的に示した．そして，4章では，Quinlanらが決定木の学習に対して行ったのと同様に，評価関数を用いてデータからできるだけ小さい木パターン上の決定リストを求める手法を示した．木パターン上の決定リストの学習問題と，通常の決定木の学習の大きな相違点は，木パターンの場合はテスト集合が明示的に与えられていないということである．したがって，データからボトムアップにテストとなる木パターンを生成しなければならないが，本稿では，最小汎化を利用した．応用例としては，ルールの適用条件の学習への応用を示し，非常に本手法が未知の問題に対するcoverageが優れていることがわかった．

しかし，提案した手法は，ヒューリスティックなものであり，その目標とする関数への収束性などは保証されていない．今後の課題としては，サブゴールの情報だけでなく，トレースにおけるオペレータ列の情報も利用した枠組みを理論的に展開することが可能かどうかを探っていきたいと考えている．

謝辞

工業技術院機械技術研究所の阿久津達也氏には，本論文の内容に関して貴重な議論をしていただきました．ここに感謝致します．

◇参考文献◇

[Arimura 91] Arimura, H., Shinohara, T. and Otsuki, S.: Polynomial Time Inference of Unions of Tree Pattern Languages, *Proc. 2nd Workshop on Algorithmic Learning Theory*, pp. 105-114 (1991).

[Cohen 88] Cohen, W. W.: Generalizing Number and Learning from Multiple Examples in Explanation Based Learning, *Proc. Int. Conf. on Machine Learning '88*, pp. 256-269 (1988).

[Hyafil 76] Hyafil, L. and Rivest, R. L.: Constructing Optimal Binary Decision Trees is NP-Complete, *Inf. Processing Let.*, Vol. 5, pp. 15-17 (1976).
[Keller 87] Keller, R. M.: Defining Operationality for Explanation Based Learning, *Proc. AAAI'87*, pp. 482-487 (1987).
[小林 93] 小林 聡：説明に基づく学習—複数例題からの構造の抽出—, 東京大学博士論文 (1993).
[Minton 89] Minton, S.: Quantitative Results Concerning the Utility of Explanation-Based Learning, *Proc. AAAI '89*, pp. 564-569 (1989).
[Mitchell 83] Mitchell, T., Utgoff, P. and Banerji, R.: Learning by Experimentation: Acquiring and Refining Problem Solving Heuristics, Carbonell, J. and Michalski, R. and Mitchell, T. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Tioga Publishing Co. (1983).
[Mitchell 86] Mitchell, T. M., Keller, R. M. and Kedar Cabelli, S. T.: Explanation-Based Generalization: A Unifying View, *Machine Learning*, Vol. 1, pp. 47-80 (1986).
[Pitt 88] Pitt, L. and Valiant, L. G.: Computational Limitations on Learning from Examples, *J. ACM*, Vol. 35, pp. 965-984 (1988).
[Plotkin 70] Plotkin, G. D.: A Note on Inductive Generalization, *Machine Intell.*, Vol. 5, pp. 153 163 (1970).
[Quinlan 86] Quinlan, J. R.: Induction of Decision Trees, *Machine Learning*, Vol. 1, pp. 81 106 (1986).
[Quinlan 89] Quinlan, J. R. and Rivest, R. L.: Inferring Decision Trees Using Minimum Description Length Principle, *Inf. and Comput.*, Vol. 80, pp. 227 248 (1989).
[Rivest 87] Rivest, R. L.: Learning Decision Lists, *Machine Learning*, Vol. 2, pp. 229 246 (1987).
[Rajamoney 87] Rajamoney, S. and Dejong, G.: The Classification, Detection and Handling of Imperfect Theory Problem, *Proc. IJCAI'87*, pp. 242 248 (1987).
[山田 89] 山田誠二, 辻 三郎：完全因果性によるマクロオペレータの選択的学習, 人工知能学会誌, Vol. 4, No. 3, pp. 321 329 (1989).
[山村 89] 山村雅幸, 小林重信：EBL の複数例題下への拡張, 人工知能学会誌, Vol. 4, No. 4, pp. 389 397 (1989).

〔担当編集委員・査読者：寺野隆雄〕

◇ 付　　録 ◇

補題 2 の証明

S に無矛盾な $dl \in$ k-node-DLTP が与えられたとき，リストの最後の要素以外はすべて正の要素となるように変形できることを示せば十分である．

まず，dl の最後の要素以外は，各木パターンの根はすべて r と仮定してよい．なぜなら，もし変数ならすべての例題を含むので取り除くことができるし，r 以外の定数ならば一つの例題も含むことがないので取り除くことができるからである．

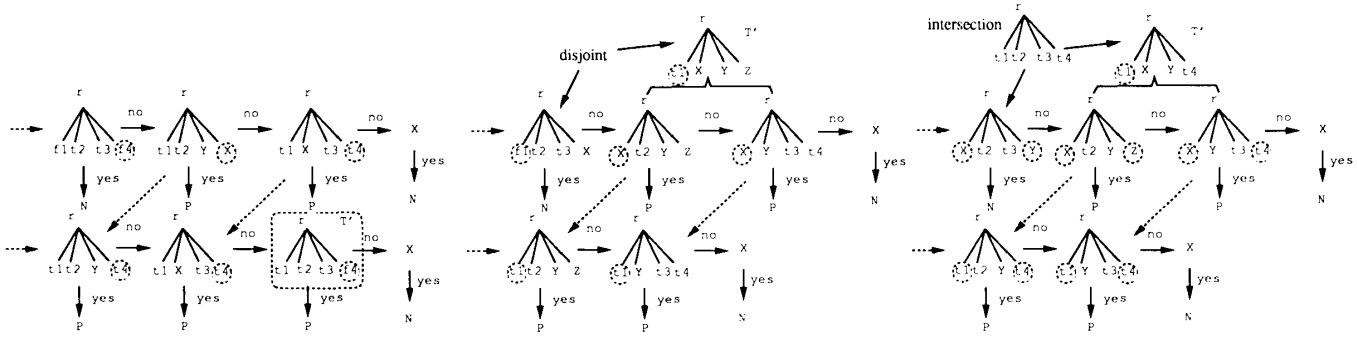
また，dl の最後の要素以外は，根 r の子を n 個持つ木パターンであり，i 番目の子で定数のものは，t_i または f_i に限られると考えてよい．もし，そうでなければ，それは例題を一つも含まない木パターンであり，取り除くことができるからである．また，dl 中の木パターンは正則であると仮定してよい．正則でない木パターンは，S の要素を一つも含まないからである．

dl の最後の要素が負の要素である場合とそうでない場合に大きく場合分けして証明する．

(1) **dl の最後が負の要素のとき**：負の要素のうち一番最後から 2 番目のものを (T_i, N) とする．このとき，以下の(1)〜(3)の操作を繰り返して適用することにより，決定リストの要素の数を増やすことなく，最後の要素以外を正の要素にすることができる．

(1) もし T_i において，根のラベルが r でその n 個の子のなかで，最初が f で始まるラベルが二つ以上ある場合は，その一つを f_s とする．このとき，(T_i, N) より後の正の要素の木パターンを，s 番目の子のラベルを t_s に付け換えたものに変更する．そして，(T_i, N) を dl から取り除く．こうすることにより，負例が誤って P に分類されることはない．また，根が r でラベルされ，その子が s 番目だけ f_s でラベルされ，それ以外は各 j 番目が t_j でラベルされた木を T' とし，これを用いて (T', P) を dl の最後から 2 番目に付け加える．これにより正例も誤って分類されることがなくなり，リストの長さを変えることなく，負の要素 (T_i, N) を取り除くことができる(図 6 (a))．

(2) 木パターン T_i が一つだけ f で始まるラベルの子を持つ場合を考える．T_i は，根が r で j 番目の子が f_j でラベルされているものとする．このとき，(T_i, N) 以降で P とクラス分けされる正例は次のような木パターン T' に必ず含まれる．すなわち，根が r でラベルされ，j 番目の子が t_j にラベルされ，それ以外の子は異なる変数でラベルされている木パターンである．なぜなら，j 番目の子が f_j の正例は，T_i に含まれてしまうからである．したがって，(T_i, N) 以降の正の要素の木パターンを，T' との交わりをとったものに変更しても矛盾は



(a) Decision List Transformation(1)　(b) Decision List Transformation(2)　(c) Decision List Transformation(3)

図 6 決定リストの変更方法

生じない(ただし，交わりが空ならその要素を取り除く)．そして，(T_i, N)を取り除けばよい(図6(b))．ここで，T_iとT'は言語として交わりを持たないので，負例も誤って分類されることはない．このように構成し直すことにより，リストの長さが一つ減少することに注意する．

(3) 木パターンT_iが一つもfで始まるラベルの子を持たないときを考える．T_iは，根がrで$i_1, \cdots, i_p$番目の子がそれぞれ，異なる変数でラベルされているものとする．このとき，(T_i, N)以降でPとクラス分けされる正例は次のような木パターンT'に必ず含まれる．すなわち，根がrでラベルされ，そのn個の子のうち，i_q $(1 \leq q \leq p)$番目の子はt_{iq}でラベルされ，それ以外の子はそれぞれ異なる変数でラベルされている木パターンである．なぜなら，i_q番目の子がf_{iq}である正例(ただし，$1 \leq q \leq p$)は，T_iに含まれてしまうからである．したがって，(T_i, N)以降の正の要素の木パターンを，T'との交わりをとったものに変更しても矛盾は生じない(ただし，交わりが空ならその要素を取り除く)．そして，(T_i, N)を取り除けばよい(図6(c))．ここで，T_iとT'の言語としての交わり(子がすべてtで始まるラベルを持つ木パターン)には，負例が含まれないので，負例は誤って分類されることはない．この場合もリストの長さが一つ減少する．

(2) ***dl*の最後が正の要素のとき**：変更の仕方は上述と全く同様に行えばよい．ただし，1回目の変更で(X, N)なる2項組をリストの最後に付け加えなければならないので，リストの長さが一つ増えてしまう可能性がある．しかし，$m > k$より，負の要素のうち少なくとも一つの木パターンは，二つ以上の負例を含まなければならない．したがって，負の要素の木パターンのなかで，fで始まるラベルをせいぜい一つしか持たないものが存在する．よって，この要素を取り除くときにリストの長さが一つ減少する．よって，リストの長さを増やすことなく変形が行える．

著者紹介

小林　聡(正会員)

1988年東京大学工学部航空学科卒業．1990年同大学院工学系研究科修士課程修了．1993年同大学院博士課程修了．博士(工学)．同年，電気通信大学情報工学科助手．人工知能，学習理論の研究に従事．特に，説明に基づく学習，計算論的学習理論，ゲノム情報処理などに興味を持つ．情報処理学会会員．

堀　浩一(正会員)，**大須賀　節雄**(名誉会員)は，前掲(Vol. 8, No. 2, p. 200)参照．