分類型問題に対する最適な仮説検証木の効率的な生成法：MINISTAR 法†

赤埴 淳一††

分類型問題解決における大きな問題の一つに，与えられた情報が解を求めるのに不十分な場合に対処しなければならないという問題がある．本稿では，情報はテストを行うことによって収集されると仮定し，行うべきテストの順序を表した仮説検証木を生成することにより，この問題の解決をはかる．我々の目的は，最適な仮説検証木，すなわち行うべきテストのコストの期待値が最小となる仮説検証木を効率的に生成することである．最適な仮説検証木を生成する方法として，ガンマ最小平均法が提案されている．ガンマ最小平均法はすべての仮説検証木を含む潜在的な木を，コストの期待値を計算しながら探索する方法である．しかし，探索の方法が深さ優先であるため，計算量が膨大になるという問題がある．本稿では，最適な仮説検証木の効率的な生成法として ministar 法を提案する．本法はガンマ最小平均法の改良であり，以下の2点の特徴を持つ探索法である：・コストの期待値の下界値をトップダウンに計算する．・この下界値を用いて最良優先探索を行う．次に，ministar 法の正当性について述べ，本法の探索空間が常にガンマ最小平均法の探索空間の部分空間になることから本法の優位性を示す．さらに計算機による実験を行い，テスト数が10個のときに探索空間がガンマ最小平均法の約10分の1で済むことを示す．

1. はじめに

分類型問題[3]とは，可能な解の集合が与えられたときに，与えられた情報から解を求める問題である．例えば，診断問題は考える原因（すなわち解）の集合が与えられたときに，診断対象で観測された症状（すなわち情報）をもとに原因を求める問題であり，分類型問題の典型例である．分類型問題解決における大きな問題の一つに，与えられた情報が解を求めるのに不十分な場合に対処しなければならないという問題がある．診断問題でいえば，観測された症状が原因を求めるのに不十分な場合がこれに当る．

このような問題に対処する方式として，仮説推論[4),8)]を応用することが提案されている．仮説推論とは，解の候補を仮説として生成し，仮説を検証するために収集すべき情報を決定・収集することを繰り返して問題を解決する推論方式であり，大きく分けて次の二つの技術を用いて実現される．

1. 仮説生成法：与えられた情報から，解の候補を仮説として生成する．
2. 仮説検証法：生成された仮説を検証するために収集すべき情報を決定する．

仮説推論に関する従来の研究は，仮説生成法に関するものがほとんどであった．しかし，分類型問題に仮説推論を適用する場合には，効率的な仮説検証法の実現が重要になる．そこで本稿では，効率的な仮説検証法の実現法を対象とする．

本稿では，情報はテストを行うことによって収集するものと仮定する．この仮定のもとで仮説検証は，行うべきテストを決定することに帰着する．あるテストを行い，その結果にしたがってさらに次のテストを行うというように，行うべきテストは木で表現できる．このようなテストの木を仮説検証木と呼ぶ．診断問題について考えると，仮説検証木は診断木（fault tree）に対応する．さらに，次のものが与えられていることを仮定する．

- 考えうる解の集合および各解の生起確率
- 実行可能なテストの集合と各テストの取りうる結果の集合および各テストの実行に要するコスト

診断問題について考えると解の生起確率とは故障の発生確率であり，このような仮定・前提条件は診断問題に応用する場合に適切なものである．テストのコストおよび解の生起確率を用いて，仮説検証木に対するコストの期待値，すなわち仮説検証木に対応するテストを行うのに要するコストの期待値を定義することができる．コストの期待値が最小となる仮説検証木を最適な仮説検証木と呼ぶ．我々の目的は最適な仮説検証木を効率的に求めることである．

仮説検証木を生成する方法として，最小エントロピー法[1)]，ガンマ最小平均法（gamma miniaverage

† MINISTAR: An Efficient Method for Generating the Optimal Test Tree in Classification Problem Solving by JUN-ICHI AKAHANI (NTT Communications and Information Processing Laboratories).

†† NTT 情報通信処理研究所

method)[9] が提案されている．最小エントロピー法とは，解の生起確率から計算されるテスト実行後のエントロピーの期待値が最小となるテストを，実行すべきテストとして決定する方法である．しかし，エントロピーはテスト回数の推定値を与えるものであり，テストのコストが一定でない場合には，適用することができない．テストのコストを考慮した拡張が提案されている[2),7)]が，最適な仮説検証木が求められるとは限らないという問題がある．ガンマ最小平均法は，すべての仮説検証木を含む潜在的な木を，コストの期待値を計算しながら深さ優先で探索することにより，最適な仮説検証木を求める方法である．この方法は，探索の際に，アルファ・ベータ枝刈り[6]に類似したガンマ枝刈りを行い，探索の効率化をはかっているが，探索方法が深さ優先のため計算量が問題となる．

そこで本稿では，最適な仮説検証木の効率的な生成法として ministar 法を提案する．ministar 法は，ガンマ最小平均法の改良であり，以下の2点の特徴を持つ．

1. コストの期待値の下界値をトップダウンに計算する．
2. この下界値を用いて最良優先探索を行う．

また，ガンマ最小平均法との対比でいえば，ministar 法はアルファ・ベータ探索法の代わりに SSS* 探索法[10]を用いているということができる．

本稿の構成を以下に示す：第2章で分類型問題に対する仮説推論の応用について述べ，最適な仮説検証木を定義する．次に，第3章で ministar 法の定義を与え，その正当性について述べる．さらに，第4章で計算機による実験結果について述べ，他の研究との比較・評価を行う．最後に，第5章でまとめを行う．

2. 仮説推論を用いた分類型問題解決

本稿では，分類型問題をテストを行って得られた情報から解を求めることと考える．実行可能なテストの集合とその取りうる結果の集合が与えられているとする．これらをそれぞれ次のように表す．

- TESTS：実行可能なテストの集合
- Outcomes (*Test*)：テスト *Test* の取りうる結果の集合

テストを行って得られた情報を観測と呼ぶ．観測 OBS をテストとその結果の組の集合で次のように表現する．

$$\mathrm{OBS}=\{<Test, Outcome> \mid Test\in\mathrm{TESTS}, Outcome\in\mathrm{Outcomes}(Test)\}$$

また，考えうる解の集合を SOLUTIONS で表す．

以上の定義のもとで，分類型問題は実行可能なテストの集合 TESTS，各テストの取りうる結果の集合 Outcomes (*Test*) (*Test*∈TESTS)，考えうる解の集合 SOLUTIONS が与えられたときに，観測 OBS から解 S∈SOLUTIONS を求める問題と定義することができる．

分類型問題を解くためには，観測から解への写像が必要となる．知識処理では，このような写像を経験的なルールや対象の構造・動作記述で表現している．ここでは，観測が解を求めるのに不十分な場合に対処するため，観測から解の集合への写像で表現する．この解の集合を仮説と呼び，写像を仮説生成写像と呼ぶ．仮説生成写像は，形式的には観測 OBS から考えうる解の集合 SOLUTIONS の部分集合への写像であり，*HYP*(OBS) で表す．

さて，いくつかのテストを行ってある観測が得られたとしよう．仮説生成写像により，この観測に対する仮説が生成されるが，生成された仮説が複数の解を含むとき，さらにテストを行って仮説を絞り込むことが必要になる．このとき，どのようなテストをどのような順序で行うかが問題となる．あるテストを行い，その結果にしたがってさらに次のテストを行うというように，行うべきテストは木で表現できる．このようなテストの木を仮説検証木と呼ぶ．仮説検証木は次のように定義できる．

定義 2.1 仮説検証木とは以下の性質を持つ木である．

1. テストに対応する TEST 節点と，解に対応する SOLUTION 節点との2種類の節点から構成される．
2. すべての末端節点は SOLUTION 節点であり，それ以外の節点はすべて TEST 節点である．
3. 各 TEST 節点から出ている各リンクは，その節点に対応するテストの取りうる値に対応する．

例 2.1 次のような分類型問題を考える．実行可能なテストは $Test_1$, $Test_2$, $Test_3$ の3個であり，

$$\mathrm{TESTS}=\{Test_1, Test_2, Test_3\},$$

それぞれのテストの取りうる結果は $O_{11}, O_{12}, \cdots$ であるとする．

$$\mathrm{Outcomes}(Test_1)=\{O_{11}, O_{12}\},$$

$$\mathrm{Outcomes}(Test_2)=\{O_{21}, O_{22}, O_{23}\},$$

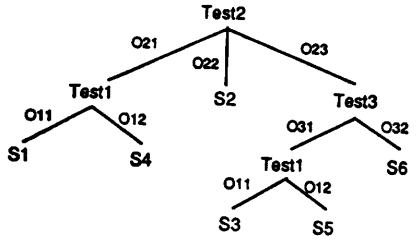

図 1 例 2.1 の分類型問題に対する仮説検証木の一例
Fig. 1 A test tree for the classification problem in example 2.1.

Outcomes ($Test_3$)= $\{O_{31}, O_{32}\}$.

また，考えうる解は $S_1, S_2, \cdots, S_6$ の 6 個であり，

SOLUTIONS= $\{S_1, S_2, S_3, S_4, S_5, S_6\}$,

次のような仮説生成写像が与えられているとする．

$HYP(\{<Test_1, O_{11}>\})=\{S_1, S_2, S_3\}$,
$HYP(\{<Test_1, O_{12}>\})=\{S_4, S_5, S_6\}$,
$HYP(\{<Test_2, O_{21}>\})=\{S_1, S_4\}$,
$HYP(\{<Test_2, O_{22}>\})=\{S_2\}$,
$HYP(\{<Test_2, O_{23}>\})=\{S_3, S_5, S_6\}$,
$HYP(\{<Test_3, O_{31}>\})=\{S_1, S_3, S_5\}$,
$HYP(\{<Test_3, O_{32}>\})=\{S_2, S_4, S_6\}$,
$HYP(\mathrm{OBS})=\bigcap_{obs\in \mathrm{OBS}} HYP(obs)$.

これらは，テスト $Test_1$ の結果が O_{11} であったときに $\{S_1, S_2, S_3\}$ という仮説が生成されること等を表しており，最後の式は複数のテストとその結果との組に対しては各テスト・結果に対する仮説の積集合が仮説として生成されることを表している．

この分類問題に対する仮説検証木の一例を**図 1** に示す．図の仮説検証木は，まずテスト $Test_2$ を行い，その結果が O_{22} ならば解 S_2 が得られ，結果が O_{21} ならばさらに $Test_1$ を行い，その結果が O_{11} ならば解 S_1 が，O_{12} ならば解 S_4 が得られること等を表している．

次に，テストに要するコストおよび解の生起確率から，仮説検証木に対するコストの期待値を定義する．テスト $Test$ のコスト，解 $Solution$ の生起確率をそれぞれ $Cost(Test)$, $Pr(Solution)$ で表す．

定義 2.2 観測 OBS のもとで仮説検証木 T に対するコストの期待値 $E(T, \mathrm{OBS})$ は次の式で与えられる．

$$E(T, \mathrm{OBS})=E_T(n_0, \mathrm{OBS}),$$

ここで，n_0 は仮説検証木 T の根節点であり，$E_T(n, \mathrm{OBS})$ は以下のように定義される．

1. 末端節点でない TEST 節点 n に対して，

$$E_T(n, \mathrm{OBS})=Cost(Test_n)+\sum_{O\in \mathrm{Outcomes}(Test_n)} Prob(<Test_n, O>|\mathrm{OBS})*E_T(n_0, \mathrm{OBS}\cup\{<Test_n, O>\}),$$

ここで，$Test_n$ は節点 n に対応するテスト，n_0 はテスト $Test_n$ の結果 O に対応するリンクの後継者節点を表すものとする．また，$Prob(<Test, Outcome>|\mathrm{OBS})$ は観測 OBS のもとでテスト $Test$ の結果が $Outcome$ となる確率を表しており，次式で定義される．

$$Prob(<Test, Outcome>|\mathrm{OBS})=\frac{Prob(\mathrm{OBS}\cup\{<Test, Outcome>\})}{\sum_{X\in \mathrm{Outcomes}(Test)} Prob(\mathrm{OBS}\cup\{<Test, X>\})},$$

ただし，

$$Prob(\mathrm{OBS})=\sum_{s\in HYP(\mathrm{OBS})} Pr(s).$$

2. 末端節点 n に対して

$$E_T(n, \mathrm{OBS})=Risk(Solution_n).$$

ここで，$Solution_n$ は節点 n に対応する解を表す．また，$Risk(Solution)$ は唯一の解が見つかる前にテストを打ち切る危険度を表すものであり，唯一の解が見つかった場合 0 が代入される．

例 2.2 例 2.1 の分類問題で，次のようなテストのコストと解の生起確率が与えられているとする．

$Cost(Test_1)=10$, $Cost(Test_2)=20$,
$Cost(Test_3)=30$,
$Pr(S_1)=0.01$, $Pr(S_2)=0.02$, $Pr(S_3)=0.03$,
$Pr(S_4)=0.04$, $Pr(S_5)=0.05$, $Pr(S_6)=0.06$.

このとき，例 2.1 の仮説検証木に対するコストの期待値は，次のようにして計算される．

$$\begin{aligned}
&E(T, \{\}) \\
&=E_T(Test_2, \{\}) \\
&=Cost(Test_2) \\
&\quad +Prob(<Test_2, O_{21}>|\{\}) \\
&\quad *E_T(Test_1, \{<Test_2, O_{21}>\}) \\
&\quad +Prob(<Test_2, O_{22}>|\{\}) \\
&\quad *E_T(S_2, \{<Test_2, O_{22}>\}) \\
&\quad +Prob(<Test_2, O_{23}>|\{\}) \\
&\quad *E_T(Test_3, \{<Test_2, O_{23}>\}),
\end{aligned}$$

ここで，テストの結果に対する確率は，

$$Prob(<Test_2, O_{21}>|\{\})=\frac{Prob(\{<Test_2, O_{21}>\})}{\sum_{X\in\{O_{21}, O_{22}, O_{23}\}} Prob(\{<Test_2, X>\})}$$

$$=\frac{Pr(S_1)+Pr(S_4)}{(Pr(S_1)+Pr(S_4))+Pr(S_2)+(Pr(S_3)+Pr(S_5)+Pr(S_6))}$$
$$=0.05/0.21=0.238.$$

のように計算できる．結局，コストの期待値は

$$(T,\{\})=20+0.238*10+0.095*0$$
$$+0.667*(30+0.571*10+0.429*0)$$
$$=46.2$$

となる．

我々の目的は，最適な仮説検証木—すなわちコストの期待値が最小となるような仮説検証木—を効率的に求めることである．次章で，最適な仮説検証木の効率的な生成法を与える．

3. Ministar 法

本章では，前章で定義した最適な仮説検証木を効率的に求める方法 (ministar 法と呼ぶ) について述べる．最適な仮説検証木を求めるためには，すべての仮説検証木を生成し，その中でコストの期待値が最小となるものを選択すればよい．すべての仮説検証木を含む木を ministar 木と呼ぶ．

3.1 Ministar 木

まず，ministar 木と ministar 値を定義する．これらはそれぞれ AND/OR 木と minimax 値[6] に類似するものである．

定義 3.1 ministar 木とは以下の性質を持つ木である．

1. MIN と STAR の2種類の節点を持つ．
2. 根節点とすべての末端節点は MIN 節点である．
3. MIN 節点のすべての後継者節点は STAR 節点であり，STAR 節点のすべての後継者節点は MIN 節点である．
4. 各 MIN 節点 m に対して，以下を満たす実数値 $p(m)$ が付随している．
   - $0\leq p(m)\leq 1$
   - 根節点 m_0 に対して $p(m_0)=1$
   - 各 STAR 節点 s に対して
   $$\sum_{m\in son(s)} p(m)=1,$$
   ただし，$son(n)$ は節点 n のすべての後継者節点の集合を表す．
5. 各 STAR 節点 s に対して，実数値 $c(s)$ (>0) が付随している．
6. 各末端節点 m に対して，実数値 $r(m)$ (≥ 0) が付随している．

ministar 木の例を図2に示す．

定義 3.2 節点 n の ministar 値は $g(n)$ で表される実数値関数であり，以下のように定義される．

1. 末端節点でない MIN 節点 m に対して
   $$g(m)=\min_{s\in son(m)} g(s).$$
2. STAR 節点 s に対して
   $$g(s)=c(s)+\sum_{m\in son(s)} p(m)*g(m).$$
3. 末端節点 m に対して
   $$g(m)=r(m).$$

ministar 木の MIN 節点と STAR 節点はそれぞれ観測とテストに対応するものである．分類型問題が与えられたとき，以下のようにして ministar 木を構成することができる．

1. 初期観測に対応して根節点 m_0 を生成し，1を $p(m_0)$ として付随する．
2. 観測 OBS に対応する MIN 節点があるとき，その時点で実行可能なテスト *Test* と観測 OBS の組 (OBS, *Test*) に対応する STAR 節点 s をその後継者節点として生成し，テスト *Test* のコスト *Cost*(*Test*) を $c(s)$ として付随する．
3. 観測とテストの組 (OBS, *Test*) に対応する STAR 節点があるとき，テスト *Test* の取りうる結果 *Outcome*∈Outcomes (*Test*) に対して，観測 OBS∪<*Test*, *Outcome*> に対応する MIN 節点 m をその後継者節点として生成し，確率 *Prob*(<*Test*, *Outcome*>|OBS) を $p(m)$ として付随する．

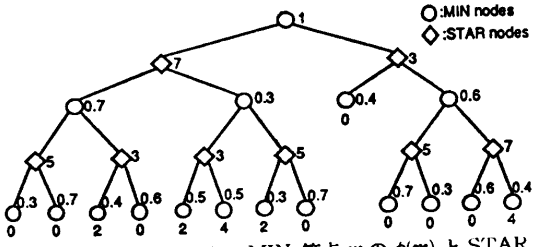

図 2 ministar 木の例．MIN 節点 m の $p(m)$ と STAR 節点 s の $c(s)$ の値を各節点の横に示す．末端節点 m の $r(m)$ 値を各節点の下に示す

Fig. 2 A ministar tree. $p(m)$ of the MIN node m and $c(m)$ of the STAR node s are shown next to the nodes. $r(m)$ of the terminal nodes m are shown below the nodes.

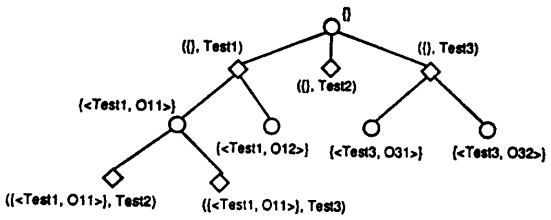

図 3 例2.1の分類型問題に対する ministar 木の一部，観測を MIN 節点の横，観測とテストの組を STAR 節点の横に示す

Fig. 3 The part of the ministar tree corresponding to the classification problem in ex. 2.1. Observations and a pair of observations and test are shown next to the corresponding MIN and STAR node, respectively.

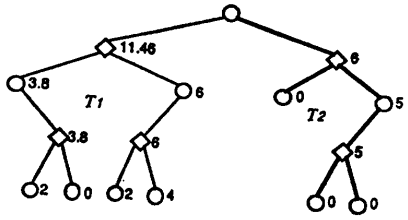

図 4 図2の ministar 木に対する二つの異なる決定木 T_1 と T_2．決定値 $f_{T_1}(n)$ と $f_{T_2}(n)$ を各節点の横に示す

Fig. 4 Two different decision trees T_1 and T_2 of the ministar tree of Fig. 2. The decision evaluation $f_{T_1}(n)$ and $f_{T_2}(n)$ are shown next to the nodes.

4. 以下の条件を満たす MIN 節点を末端節点とする．
   - 観測に対応する仮説が1個である．
   - 実行可能なテストがない．

   さらに，前者の場合には0を，後者の場合にはある正数を $r(m)$ として付随する．

例 3.1 例2.1の分類問題に対応する ministar 木の一部を**図3**に示す．根節点は初期観測 $\{\}$ に対応する MIN 節点であり，実行可能テストに対応して3個の後継者節点を持っている．観測とテストの組 $(\{\}, Test_1)$ に対応する STAR 節点は $Test_1$ の取りうる結果に対応して2個の後継者節点を持っている．

次に，AND/OR 木の解木 (Solution Tree) に類似した ministar 木の決定木を定義する．決定木は仮説検証木に対応するものである．

定義 3.3 ministar 木 G の決定木 T は次の性質を持つ木である．

1. 決定木 T の根節点は ministar 木 G の根節点である．
2. G の末端節点でない MIN 節点が T に含まれるとき，その後継者節点のうち唯一つの節点が T に含まれる．
3. G の STAR 節点が T に含まれるとき，そのすべての後継者節点が T に含まれる．

決定木 T の節点 n に対する決定値は，$f_T(n)$ で表される実数値関数であり，以下のように定義される．

1. 末端節点でない MIN 節点 m に対して
   $f_T(m)=f_T(s)$, where $s\in son(m)$.
2. STAR 節点 s に対して
   $$f_T(s)=c(s)+\sum_{m\in son(s)} p(m)*f_T(m).$$
3. 末端節点 m に対して
   $f_T(m)=r(m)$.

図2の ministar 木に対する二つの異なる決定木 T_1 と T_2 を**図4**に示す．上の定義から明らかのように，ministar 木の任意の節点の ministar 値は，任意の決定木の対応する節点の決定値よりも大きくなることはない．

定理 3.1 G を ministar 木，T を G の一つの決定木とする．このとき，T の各節点 n に対して次式が成立する．

$g(n)\leq f_T(n)$.

さらに，T_0 のすべての節点 n に対して $g(n)=f_{T_0}(n)$ となるような決定木 T_0 が存在する．

このような決定木 T_0 を G の最適決定木と呼ぶ．例えば，図4の決定木 T_2 は図2の ministar 木の最適決定木である．

決定木と決定値はそれぞれ仮説検証木とそのコストの期待値に対応するものである．したがって，与えられた潜在的な ministar 木から最適決定木を求めれば，最適な仮説検証木を得ることができる．次節でそのアルゴリズムを与える．

3.2 Ministar アルゴリズム

本節では，与えられた潜在的な ministar 木から最適決定木を求めるアルゴリズム (ministar アルゴリズムと呼ぶ) について述べる．このアルゴリズムは，ministar 木に SSS*アルゴリズム[10] を応用したものと考えることができる．

ministar アルゴリズムは四つ組

$$(node, state, merit, weight)$$

のリスト (OPEN リストと呼ぶ) に対する操作として定義される．ここで，*node* は節点であり，*state* は節点の状態を LIVE あるいは SOLVED で表したものである．さらに，*merit* は節点の付加情報であ

表 1 操作 $\Theta(t, \mathrm{OPEN})$
Table 1 The operation $\Theta(t, \mathrm{OPEN})$.

場合	$t=(n,s,m,w)$ に対する条件			操作
	nのタイプ	s	条件	
1	MIN	SOLVED	$next(n)\neq nil$	$(next(n), \mathrm{LIVE}, m, w)$ を OPEN に積む
2	MIN	SOLVED	$next(n)=nil$	$(parent(n), \mathrm{SOLVED}, m, w)$ を OPEN に積む
3	MIN	LIVE	nは末端節点	$(n, \mathrm{SOLVED}, m+w*p(n)*r(n), w)$ を OPEN の，より大きいか等しい merit を持つすべての四つ組の後に置く
4	MIN	LIVE		すべての $s\in son(n)$ に対して，$(s, \mathrm{LIVE}, m+w*p(n)*c(s), w*p(n))$ を OPEN の，より大きいか等しい merit を持つすべての四つ組の後に置く
5	STAR	SOLVED		$p=parent(n)$ とし，$(p, \mathrm{LIVE}, m, w/p(p))$ を OPEN に積み，pのすべての子孫を OPEN から削除する
6	STAR	LIVE		$(first(n), \mathrm{LIVE}, m, w)$ を OPEN に積む

表 2 図5の計算に対する OPEN リスト
Table 2 The complete OPEN list for the computation in Fig. 5.

ステップ	Θの場合	OPEN
1	—	$(m_0,$ LIVE, 0, 1)
2	4	$(s_1,$ LIVE, 4, 1) $(s_2,$ LIVE, 5, 1) $(s_3,$ LIVE, 8, 1)
3	6	$(m_1,$ LIVE, 4, 1) $(s_2,$ LIVE, 5, 1) $(s_3,$ LIVE, 8, 1)
4	3	$(s_3,$ LIVE, 5, 1) $(m_1,$ SOLVED, 6.1, 1) $(s_3,$ LIVE, 8, 1)
5	6	$(m_3,$ LIVE, 5, 1) $(m_1,$ SOLVED, 6.1, 1) $(s_3,$ LIVE, 8, 1)
6	3	$(m_1,$ SOLVED, 6.1, 1) $(m_3,$ SOLVED, 6.2, 1) $(s_3,$ LIVE, 8, 1)
7	1	$(m_3,$ LIVE, 6.1, 1) $(m_3,$ SOLVED, 6.2, 1) $(s_3,$ LIVE, 8, 1)
8	3	$(m_3,$ SOLVED, 6.2, 1) $(s_2,$ LIVE, 8, 1) $(m_2,$ SOLVED, 8.2, 1)
9	1	$(m_4,$ LIVE, 6.2, 1) $(s_2,$ LIVE, 8, 1) $(m_2,$ SOLVED, 8.2, 1)
10	3	$(m_4,$ SOLVED, 7.8, 1) $(s_2,$ LIVE, 8, 1) $(m_2,$ SOLVED, 8.2, 1)
11	2	$(s_5,$ SOLVED, 7.8, 1) $(s_2,$ LIVE, 8, 1) $(m_2,$ SOLVED, 8.2, 1)
12	5	$(m_5,$ SOLVED, 7.8, 1)

り，*weight* は節点の重み付けを表したものである．

Ministar アルゴリズム

1. OPEN に $\{(m_0, \mathrm{LIVE}, 0, 1)\}$ を代入する．
2. OPEN から先頭の要素 $t=(n,s,m,w)$ を削除する．
3. $n=m_0$ かつ $s=$SOLVED ならば停止する．
4. OPEN に $\Theta(t, \mathrm{OPEN})$ を代入する．
5. 2へ行く．

ここで，四つ組 t とリスト OPEN に対する操作 $\Theta(t, \mathrm{OPEN})$ は**表1**で与えられる．表中，*parent*(*n*), *first*(*n*), *next*(*n*) はそれぞれ節点 *n* の親節点，最も左にある (leftmost) 後継者節点，右にある兄弟節点を表すものとする．

図5にこのアルゴリズムの実行例を示す．ここで節点の番号は状態 LIVE のときに探索される順序を示している．**表2**に対応する OPEN リストを示す．表2のステップは状態が LIVE または SOLVED のときに探索される順序を示すものであり，図5の節点の番号と表2のステップが対応しないことに注意されたい．計算手順は以下の通りである．まず，状態 LIVE を持つ根節点 m_0 が OPEN リストに代入される（表2のステップ1参照）．次に，根節点が削除され，操作 Θ_4 により根節点のすべての後継者節点が OPEN リストに積まれる．このとき，各節点 *s* の $c(s)$ の値が *merit* に代入され，この *merit* にしたがって昇順にソートされる（ステップ2）．以下同様に繰り返す．

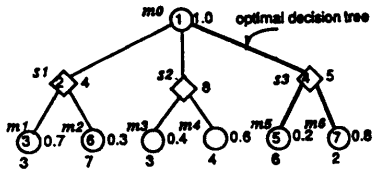

図 5 ministar アルゴリズムの計算例．節点の番号は探索される順序を示す
Fig. 5 Computation of the ministar algorithm. The node number indicates the exploring order.

3.3 Ministar アルゴリズムの正当性

前節で述べたアルゴリズムの正当性は，このアルゴリズムによって導かれる根節点の *merit* の値とその ministar 値が等しくなることから示すことができる．次の命題は ministar アルゴリズムの定義から成立する．

命題 3.2 状態 SOLVED を持つ MIN 節点 m の *merit* 値は次の性質を持つ．

1. 節点 m から到達可能な任意の節点の *merit* 値よりも小さくない．
2. 状態 LIVE のときの MIN 節点 m の *merit* 値を $m_i(m)$ とするとき，次のように定義される $m_s(m)$ と等しい．

$$m_s(m)=m_i(m)+w(m)*g(m).$$

ここで，$w(m)$ は次のように定義される．

（a） 根節点 m_0 に対して，

$$w(m_0)=1.$$

（b） MIN 節点 m 対して，

$$w(m)=p(m_o)*w(m_o),$$

ただし， $m_o=parent(parent(m))$.

MIN 節点 m に対する $m_s(m)$ と $w(m)$ の値は状態 SOLVED のときの *merit* と *weight* の値にそれぞれ一致する．すなわち，*merit* はボトムアップに定義される ministar 値の下界値をトップダウンに計算しているということができる．また，この命題は表1の操作 Θ_5 における枝刈りを保証するものである．次の定理はこの命題から導かれるもので，ministar アルゴリズムの正当性を保証する．

定理 3.3 状態 SOLVED を持つ根節点の *merit* の値は，根節点における ministar 値と等しい．

4. 評　　価

本章では，ministar 法とガンマ最小平均法[9]との理論的な比較を行い，計算機を用いた実験結果について述べる．理論的比較では，ministar 法の探索空間が常にガンマ最小平均法の探索空間の部分集合になることから ministar 法の優位性を示す．また，実験による比較では，探索される節点数を用いて性能比較を行う．

ガンマ最小平均法は，ministar 木に対してアルファ・ベータ探索法[6]を応用したものと考えることができる．すなわち，miniaverage backed-up と呼ばれる値（ministar 法では決定値に対応する）を計算しながら ministar 木を深さ優先で探索し，最小の miniaverage backed-up 値を求めるものである．探索の際に，もし miniaverage backed-up 値がそれ以前に探索した節点の値よりも大きくなれば，枝刈りが行われる．図5の ministar 木に対してガンマ最小平均法を適用した例を**図6**に示す．ここで，節点の番号は探索される順序を表している．

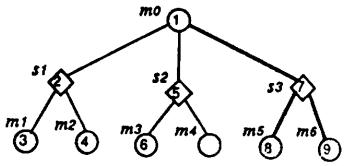

図 6 図5の ministar 木に対するガンマ最小平均法の計算例．節点の番号は探索される順序を示す
Fig. 6 Computation of the gamma miniaverage method for the ministar tree of Fig. 5. The node number indicates the exploring order.

ministar 法は MIN 節点が探索される度に OPEN リストをソートするため，ガンマ最小平均法に比べて計算が複雑になる．しかし，MIN 節点を探索する度に計算される仮説および条件付き確率の計算量の方が，ソーティングよりも多いと考えられる．したがって，探索される MIN 節点の個数をアルゴリズムの効率性の評価基準として採用することにする．

miniaverage backed-up 値が ministar 法の決定値と等しくなることから，ministar 法が探索する節点はガンマ最小平均法では必ず探索されることがいえる．このことは，ministar 法の探索空間が常にガンマ最小平均法の探索空間の部分集合になることを示している．図2の ministar 木にこの二つの方法を適用した結果を**図7**に示す．この例は ministar 法にとって最も有利な例であり，探索される MIN 節点がガンマ最小平均法では 13 個であるのに対し，ministar 法では5個で済んでいる．

いくつかの分類型問題に対して，ministar 法，ガン

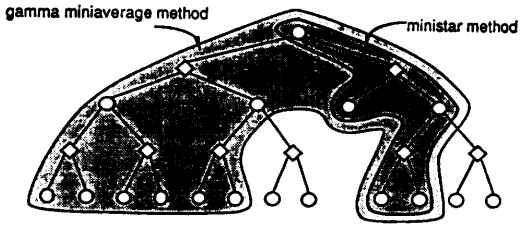

図 7 図2の ministar 木に対する ministar 法とガンマ最小平均法との探索空間の比較
Fig. 7 Comparison of the explored nodes for the ministar tree in Fig. 2.

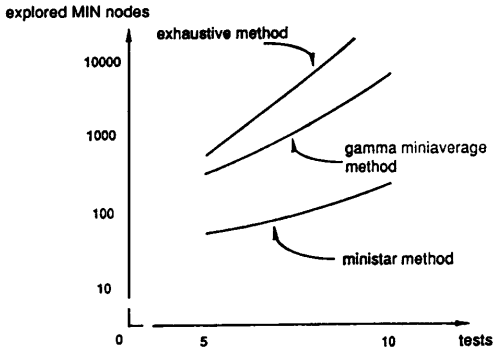

図 8 いくつかの分類型問題に対する実験結果
Fig. 8 Experimental results for several classification problems.

マ最小平均法および全数探索法を適用した実験結果を図8に示す．実験は，仮説生成写像，テストのコスト，解の生起確率を乱数を用いて生成し，テストの個数に対して探索された MIN 節点の平均個数がどのように変化するかを調べたものである．この実験結果より，ministar 法はガンマ最小平均法に比べて約 10 分の 1 の計算量で済むことがわかる．

5. おわりに

分類型問題における最適な仮説検証木の効率的な生成方法について述べた．本稿で提案した方式は従来の方式と比べて約 10 分の 1 の計算量で，最適な仮説検証木を生成できることを示した．

しかし，テスト数に対して計算量が指数的に増加する問題は依然として残っている．実用のためには，探索の段数を制限し，最下段の節点に対しては何らかの評価関数（第2章で導入した *Risk*）を適用することが考えられる．評価関数として，エントロピー，仮説数等が考えられるが，どのような評価関数を用いればよいかが今後の課題である．

謝辞 本研究の機会を与えて頂いた NTT 情報通信処理研究所の中野良平主幹研究員に感謝します．また，熱心に御討論頂いた同研究所の石田亨主幹研究員に感謝します．

参考文献

1) Ben-Bassat, M.: Myopic Policies in Sequential Classifications, *IEEE Trans. Comput.*, Vol. 27, pp. 170-178. (1978).
2) Cantone, R.R., Lander, W.B., Marrone, M.P. and Gaynor, M.W.: IN-ATE: Fault Diagnosis as Expert System Guided Search, in Bolc, L. and Coombs, M.J. (eds.), *Computer Expert Systems*, Spinger, New York (1986).
3) Clancey, W.J.: Heuristic Classification, *Artif. Intell.*, Vol. 27, pp. 289-350 (1985).
4) de Kleer, J.: An Assumption-based TMS, *Artif. Intell.*, Vol. 28, pp. 127-162 (1986).
5) de Kleer, J. and Williams, B.C.: Diagnosing Multiple Faults, *Artif. Intell.*, Vol. 32, pp. 97-130 (1987).
6) Nilsson, N.: *Principles of Artificial Intelligence*, Tioga, Palo Alto, CA (1980).
7) Pipitone, F.: The FIS Electronics Troubleshooting System, *IEEE Comput.*, Vol. 19, No. 7, pp. 68-75 (1986).
8) Poole, D.: Default Reasoning and Diagnosis as Theory Formation, Tech. Rept. CS-86-08, Department of Computer Science, University of Waterloo (1986).
9) Slagle, J.R. and Richard, C.T.L.: Application of Game Tree Searching Techniques to Sequential Pattern Recognition, *Comm. ACM*, Vol. 14, pp. 103-110 (1971).
10) Stockman, G.C.: A Minimax Algorithm Better than Alphabeta?, *Artif. Intell.*, Vol. 21, pp. 179-198 (1983).

（平成元年2月20日受付）
（平成元年6月13日採録）

赤埴 淳一（正会員）

1960年生．1983年京都大学工学部数理工学科卒業．1985年同大学院修士課程修了．同年日本電信電話(株)入社．以来，知識処理の研究に従事．推論方式，知識獲得に興味を持つ．現在，NTT 情報通信処理研究所知識処理研究部研究主任．ソフトウェア科学会会員．