ICOT Technical Memorandum: TM-0473

TM-0473

学習の形式モデルとしての文法推論の意義

高田裕志

March, 1988

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

学習の形式モデルとしての文法推論の意義

(Implications of Grammatical Inference as a formal model for learning)

高田裕志

yuji@iias.fujitsu.junet

富士通(株)国際情報社会科学研究所

要旨

文法推論を学習の形式モデルとしたときに、モデルから得られる論理的帰結の意義を認知科学の側面から考察する。まず、制御集合にもとづく線型言語の文法推論の方法と、その方法から得られる帰結を述べる。そして、モデルから得られる教師と計算量に関する帰結が学習においてもつ意義を考察する。教師の能力は学習の達成の認知を、計算量は学習の難易度の認知を決定する。これより、線型言語をモデルとする対象の学習における理想的な問題設定が示される。

1 導入

学習の研究は認知科学の中心的かつ重要な研究課題の1つである。従来の研究の関心は主に認識モデルにあり、形式モデルによる考察はほとんど行なわれていない。モデルは主に内観法で構築され、モデル検証の方法は観察・実験や計算機によるシミュレーションが中心である。その場合、モデル自体にあいまい性があり、モデルから得られる帰結が無意味である危険性がある。

一方、計算機科学では帰納的推論に関する興味深い結果が数多く得られている[5]。モデルは形式的に定義され、推論問題に関するさまざまな帰結が論理的に得られている。しかし、アルゴリズムの停止性や計算の効率といった理論的側面に関心が多く、認識モデルに対する関心はあまりない。また、心理学や教育学などの経験科学への貢献に対する関心もほとんどない。

一般に、学習と帰納的推論は区別される。それゆえ、計算機科学における帰納的推論の理論を学習全体の形式モデルとすることはできない。しかし、学習において帰納的推論が用いられていることは明らかであろう。したがって、帰納的推論の機能に関しては、計算機科学の理論を形式モデルとすることで、さまざまな帰結を論理的に得ることができる。

本稿では、文法推論を学習の形式モデルとしたときに、モデルから得られる論理的帰結の意義を認知科学の側面から考察する。特にここでは、線型言語の文法推論をモデルとする。線型文法は記号列変換機能の形式モデルとなることが可能である。記号列変換機能は人間の認知過程で重要な役割を演ずる。

まず、2節で、制御集合にもとづく線型言語の文法推論の方法とその方法の性質を述べる。次に、3節で、モデルから得られる教師と計算量に関する帰結が学

習においてもつ意義を考察する。教師の能力は学習の達成の認知を、計算量は学習の難易度の認知を決定する。これより、線型言語をモデルとする対象の学習における理想的な問題設定が示される。

2 制御集合にもとづく線型言語の文法推論

記号列の集合から言語を生成する文法を見つける問題を「文法推論 (grammatical inference)」という。文法推論アルゴリズムは、言語を生成する文法を見つけたとき、言語を同定したといわれる。

実用的な文法推論アルゴリズムの研究は、主に正則集合のクラスに対して行なわれてきた。正則集合よりも大きいクラスに対しては、あまり良い結果が得られていない。文脈自由言語を推論するアルゴリズムの多くは枚挙法にもとづいており、一般には計算量が大きくなりすぎるため、実用的とは言えない。

線型言語のクラスは正則集合のクラスを真に含み、文脈自由言語のクラスに真に含まれる。線型言語のクラスに対する推論アルゴリズムの研究として、Biermann [6]、Radhakrishnan and Nagaraja [14]、Tanatsugu [20]などの研究がある。これらのアルゴリズムは文法の自己埋め込み性にもとづいて言語を同定するが、記号列の中の自己埋め込み性を発見する手続きは効率が悪い。

そこで、我々は制御集合にもとづく文法推論の方法を提案する。まず、線型言語に関する表現定理を示す。この表現定理により、ある固定された線型文法の生成規則をアルファベットとする正則集合を同定することが、線型言語を同定することに対応する。

なお、ここで用いる形式言語と文法推論の概念及び用語に関しては付録を、補題と定理の証明は文献 [17] を参照。

2.1 表現定理

始めに、アルファベット Σ 上の任意の線型言語 L はある固定された線型文法 G と正則な制御集合 C によって生成されることを示す。

定義　線型文法 $G = (N, \Sigma, \Pi, S)$ とは

$$A \to uBv,\ A \to u$$

の形をした生成規則だけからなる文脈自由文法である。ここで、$A, B \in N$、$u, v \in \Sigma^*$。

線型文法 G によって生成される言語 $L = L(G)$ を線型言語という。

定義　$\Sigma = \{a_1, a_2, \ldots, a_n\}$ をアルファベットとする。このとき、生成規則の集合 Π^0 が

$$\Pi^0 = \left\{ \begin{array}{llll} S^0 \to a_1 S^0, & S^0 \to a_2 S^0, & \ldots, & S^0 \to a_n S^0, \\ S^0 \to S^0 a_1, & S^0 \to S^0 a_2, & \ldots, & S^0 \to S^0 a_n, \\ S^0 \to a_1, & S^0 \to a_2, & \ldots, & S^0 \to a_n, \\ S^0 \to \lambda & & & \end{array} \right\},$$

からなる線型文法 $G^0 = (\{S^0\}, \Sigma, \Pi^0, S^0)$ を万能 (universal) という。

アルファベット Σ に対して、万能線型文法 G^0 はただ 1 つ存在し、$\Sigma^* = L(G^0)$ である。

定義 ([8]) $G = (N, \Sigma, \Pi, S)$を線型文法とする。このとき、Π^* の部分集合 C を G に対する制御集合 (control set) という。また、言語

$$L_C(G) = \{w \in \Sigma^* \mid S \underset{G}{\overset{\alpha}{\Rightarrow}} w,\ \alpha \in C\}$$

を制御集合 C とともに文法 G によって生成された言語という。

$G = (N, \Sigma, \Pi, S)$ を線型文法、$G^0 = (\{S^0\}, \Sigma, \Pi^0, S^0)$ を万能線型文法とする。Π^*からΠ^{0*} への準同型写像 h を次のように定義する:

$$h(\pi) = \begin{cases} \pi^0 & \text{ただし } \pi^0 : S^0 \to \lambda,\ \pi : S \to \lambda \text{ のとき、} \\ \pi_i^0 & \text{ただし } \pi_i^0 : S^0 \to a,\ \pi : A \to a \text{ のとき、} \\ \pi_j^0 & \text{ただし } \pi_j^0 : S^0 \to aS^0,\ \pi : A \to aB \text{ のとき、} \\ \pi_k^0 & \text{ただし } \pi_k^0 : S^0 \to S^0a,\ \pi : A \to Ba \text{ のとき。} \end{cases}$$

G に対応する非決定性有限オートマトン $M = (K, \Pi, \delta, S, F)$ を次のように定義する:

- $K = N \cup \{q_F\}$ ただし $q_F \notin N$、

$$\delta(S, \pi^0) = \begin{cases} \{q_F\} & \pi^0 : S^0 \to \lambda,\ \pi \in h^{-1}(\pi^0) \text{ かつ } \pi : S \to \lambda \text{ のとき} \\ \emptyset & \pi^0 : S^0 \to \lambda \text{ かつ } h^{-1}(\pi^0) = \emptyset \text{ のとき} \end{cases}$$

- $$\delta(A, \pi_i^0) = \begin{cases} \{q_F\} & \pi_i^0 : S^0 \to a,\ \pi \in h^{-1}(\pi_i^0) \text{ かつ } A \to a \text{ のとき} \\ \emptyset & \pi_i^0 : S^0 \to a \text{ かつ } h^{-1}(\pi_i^0) = \emptyset \text{ のとき} \end{cases}$$

$$\delta(A, \pi_j^0) = \begin{cases} \{B \mid \pi : A \to aB \in \Pi,\ \pi \in h^{-1}(\pi_j^0)\} & \pi_j^0 : S^0 \to aS^0 \text{ のとき} \\ \{B \mid \pi : A \to Ba \in \Pi,\ \pi \in h^{-1}(\pi_j^0)\} & \pi_j^0 : S^0 \to S^0a \text{ のとき、} \end{cases}$$

- $F = \{q_F\}$。

線型文法 G、万能線型文法 G^0 と、G に対応する非決定性有限オートマトン M に関して次の補題が成立する。

補題 2.1 任意の $w \in \Sigma^*$、$A \in N$、$\alpha \in \Pi^*$ に対して、$A \underset{G}{\overset{\alpha}{\Rightarrow}} w$ である必要十分条件は $S^0 \underset{G^0}{\overset{h(\alpha)}{\Rightarrow}} w$ かつ $\delta(A, h(\alpha)) \ni q_F$ である。

補題2.1 から線型言語に関する次の 2 つの表現定理を得る。

定理 2.2 $G^0 = (\{S^0\}, \Sigma, \Pi^0, S^0)$ を万能線型文法とする。このとき、アルファベット Σ 上の任意の線型言語 L に対して $L = L_C(G^0)$ となる正則制御集合 C が存在する。

定理 2.3 $G^0 = (\{S^0\}, \Sigma, \Pi^0, S^0)$ を万能線型文法、C を G^0 に対する正則制御集合とする。このとき、$L = L_C(G^0)$ は線型言語である。

表現定理2.2、2.3 から、万能線型文法に対する正則制御集合を推論することによって、任意の線型言語を推論することができる。

2.2 線型言語の文法推論

一般に、線型言語の文法推論問題を正則集合の推論問題に還元することはできない。なぜならば、任意の線型言語に対して、ただ1つの正則制御集合を構成的に求めることはできないからである。線型言語 L を生成する線型文法は無限にあるので、制御集合

$$C = \{\alpha \mid S^0 \overset{\alpha^0}{\underset{G^0}{\Longrightarrow}} w,\ w \in L\} = \bigcup_{L(G)=L} h(\{\alpha \mid S \overset{\alpha}{\underset{G}{\Longrightarrow}} w,\ w \in L\})$$

は正則制御集合の無限和であり、C が正則であるかどうかは一般には判らない。しかし、線型文法に関する何らかの補助情報を用いることで、任意の線型言語に対してただ1つの正則制御集合を定義することができる。この補助情報には、文法の変数の個数の上限や文法の代表標本 (representative sample) などが考えられる。

文法の変数の個数の上限を設定すると、任意の線型言語 L に対して、L を生成する簡約された線型文法の個数は有限になる。したがって、上記の制御集合 C は正則制御集合の有限和となり、任意の線型言語に対してただ1つ存在する。

線型文法 G の代表標本は次のように定義される。

定義 ([17]) $G = (N, \Sigma, \Pi, S)$ を簡約された線型文法とする。$L(G)$ の要素 w は $S \overset{\alpha\pi\alpha'}{\underset{G}{\Longrightarrow}} w$ のとき、生成規則 π を働かす(exercise) という。G の代表標本とは、G の任意の生成規則 π に対して、π を働かす記号列 w を少なくとも1つ含む $L(G)$ の有限な部分集合である。

RS を L のある有限部分集合とする。L を生成しかつ、RS を代表標本とする変数の数が最小の線型文法が存在するならば、その個数は有限である。したがって、任意の線型言語 L に対して、L を生成する線型文法の代表標本を補助情報としても、ただ1つの正則制御集合を定義することができる。

このように、文法の変数の個数の上限や文法の代表標本を補助情報として利用可能な場合、線型言語の文法推論問題を正則集合の推論問題に還元することが可能になる。ただし、推論アルゴリズムは与えられた補助情報が示す条件を満たすような正則制御集合を推論しなければならない。また、その正則制御集合は構成的には定義されない。したがって、自然な問題設定においては、線型言語の推論アルゴリズムは極限においてのみ言語を同定するにすぎないことが予想される。

さらに、推論に要する時間計算量は指数関数のオーダになることが予想される。任意の記号列 w に対して、万能線型文法 G^0 において開始記号 S^0 から w を得る導出は $2^{|w|}$ 個ある。したがって、素朴なアルゴリズムでは、推論に要する時間計算量は $|w|$ の指数関数のオーダになる。

推論の対象を線型言語のサブクラスに限ると、そのクラスの言語の文法推論問題を正則集合の推論問題に還元することが可能になる。次に、還元可能な2つのサブクラスを示す。

1つは文法の構造記述をもつ線型言語のクラスである。ここで言う文法の構造とは導出木のことである。かっこ文法(parenthesis grammar) [12] を用いることによって、文法の構造を言語の各記号列中に記述することができる。

定義　$G=(N,\Sigma,\Pi,S)$ を線型文法とする。G のかっこ文法 $[G]$は$[G]=(N,\Sigma,\Pi',S)$ で表され、Π' は Π の各要素 $A\to x$ を $A\to[x]$ で置き換えることによって得られる。ここで、'[' と ']' は Σ の要素でない特別の記号である。

線型文法 G のかっこ文法 $[G]$ によって生成される言語 $L_s=L([G])$ を構造記述をもつ線型言語(linear language with structural description) という。

構造記述をもつ線型言語に関しても、線型言語の場合と同様の表現定理が成立する。そして、任意の構造記述をもつ線型言語に対して、ただ1つの正則制御集合を構成的に定義することができる。この場合、補助情報を必要としない。したがって、構造記述をもつ線型言語の文法推論問題は万能線型文法に対する正則制御集合の推論問題に還元可能である [19]。

第2のサブクラスはeven線型言語のクラス [1] である。

定義　$A\to uBv$ の形をした任意の生成規則が $|u|=|v|$ の条件を満たす線型文法 G_e をeven線型文法という。

even線型文法によって生成される言語 $L_e=L(G_e)$ をeven線型言語という。

even線型言語に関しても、線型言語の場合と同様の表現定理が成立し、任意のeven線型言語に対して、補助情報なしでただ1つの正則制御集合を構成的に定義することができる。それゆえ、even線型言語の文法推論問題も万能線型文法に対する正則制御集合の推論問題に還元可能である [18]。

したがって、構造記述をもつ線型言語 L_s とeven線型言語 L_e の文法推論問題に関して次の定理が成立する。

定理 2.4 ([19]、[18])　構造記述をもつ線型言語 L_s と *even* 線型言語 L_e の文法推論問題は正則集合の推論問題に還元される。

構造記述をもつ線型言語 L_s とeven線型言語 L_e を推論するアルゴリズムを構築するには、フロントエンド処理アルゴリズムを用意して、それを正則集合推論アルゴリズムに付け加えればよい。一般に、言語を推論するアルゴリズムは記号列を入力、出力とし、仮説である文法を出力とする。このとき、正則集合を推論するアルゴリズムは万能線型文法の付随語を入力、出力とし、正則制御集合の仮説を出力とする。そこで、フロントエンド処理アルゴリズムは次の3つの処理をすればよい:

- 入力記号列を万能線型文法 G^0 において構文解析し、G^0 の付随語を求め、それを正則集合推論アルゴリズムに渡す。
- 正則集合推論アルゴリズムの出力である付随語から記号列を生成し、出力する。
- 正則制御集合の仮説を文法に変換し、出力する。

この模様を図1に示す。

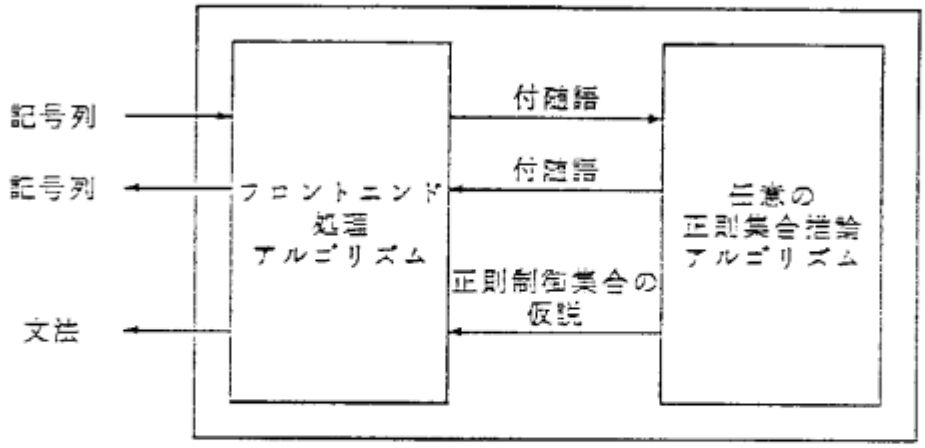

図 1: 構造記述をもつ線型言語 L_s や even 線型言語 L_e を推論するアルゴリズム

正則集合の文法推論問題はよく研究されており、多くの実用的なアルゴリズムが提案されている [2,3,7,11]。なかでも、Angluin [2] のアルゴリズムは時間計算量の点で現在最も優れたアルゴリズムである。このアルゴリズムが推論に要する時間計算量は、推論すべき正則集合を受理する最小状態決定性有限オートマトンの状態数を m、反例の最大長を n とすると、m と n の多項式のオーダである。したがって、このアルゴリズムを用いて L_s や L_e を推論するアルゴリズムを構築した場合、推論に要する時間計算量は推論すべき言語を生成する文法が必要とする変数の最小数を s、反例の最大長を t とすると、s と t の多項式のオーダになる。

構造記述をもつ線型言語 L_s、even 線型言語 L_e はどちらも次の 2 つの性質をもつ:

1. 各言語に対する万能線型文法は無あいまいである。さらに、任意の言語に対して、それを生成する文法から無あいまいな文法を構成的に求めることが可能である。

2. 言語の同値性判定問題が可解である。

正則制御集合を受理する決定性有限オートマトンが対応する文法 $[G]$ や G_e は無あいまいである。また、任意の言語 L_s や L_e に対して、ただ 1 つの正則制御集合が構成的に定義されるので、言語の同値性判定問題が正則制御集合の同値性判定問題に還元される。したがって、表現定理から L_s と L_e の性質 1、2 が導かれる。

線型言語、構造記述をもつ線型言語、even 線型言語の 3 つのクラスに関する性質と制御集合にもとづく文法推論に関する性質を、表 1 にまとめる。かっこをつけて示したものは、予想される性質である。有限同定が可能というのは、そのクラスの言語を有限の計算で同定する推論アルゴリズムが存在することを意味す

	線型言語 L	構造記述をもつ線型言語 L_s	even線型言語 L_e
正則集合推論問題への還元可能性	補助情報が必要	還元可能	還元可能
時間計算量	(指数関数)	多項式	多項式
同定の基準	(極限同定のみ)	有限同定が可	有限同定が可
無あいまいな文法	求められない	求められる	求められる
言語の同値性判定	決定不能	決定可能	決定可能

表1: 推論アルゴリズムと言語の性質

る。ただし、有限の計算で同定するには、補助情報を与えるか、言語の同値性に関する教師が必要である。表1が示すように、言語の性質と推論アルゴリズムの性質には密接な関係がある。

3 学習の形式モデルとしての意義

ある実体がある対象を学習するとする。実体は現在の計算機をモデルとすることが可能とする。学習すべき対象は形式言語をモデルとし、生成文法によって記述されているとする。このとき、学習の問題は言語の文法推論問題をモデルとし、推論アルゴリズムは学習アルゴリズムのモデルとなる。

ここでは、学習すべき対象は線型言語をモデルとし、線型文法によって記述されているとする。そして、制御集合にもとづく線型言語の文法推論のアルゴリズムを学習アルゴリズムの形式モデルとする。このとき、形式モデルから導かれる帰結の意義を、教師、計算量の2つの観点から考察する。

3.1 教師の意義

文法推論アルゴリズムは計算過程で入出力を行う。入出力は『質問応答』の形で行われる。推論アルゴリズムの質問に答えるのが「教師」である。教師は現在の計算機をモデルとすることが可能であり、質問には常に正しく答えると仮定する。ここでは、次の3種類の質問を考える:

- 記号列の言語への所属に関する質問
- 補助情報に関する質問 (補助情報の要求)
- 言語の同値性に関する質問

推論アルゴリズムからある記号列 w が推論の対象である言語 L の要素であるかどうかを質問された場合、もし $w \in L$ ならば、教師は「はい」と答え、そうでないならば、「いいえ」と答える。補助情報が推論アルゴリズムから求められた場合には、教師は自分自身が想定する文法の変数の個数、または代表標本をアルゴリズムに与える。言語の同値性に関する質問は、仮説である文法 G_H が推論の対

象である言語 L を生成するかどうかという形で行われる。もし $L = L(G_H)$ であるならば、教師は「はい」と答え、そうでないならば、反例として L と $L(G_H)$ の対称差の要素を1つ推論アルゴリズムに与える。

答えうる質問の種類によって、教師の能力は異なる。ここでは、所属に関する質問のみに答えることができる教師を T_M、所属に関する質問と補助情報の要求に答えることが可能な教師を T_A、所属に関する質問と言語の同値性に関する質問の両方に答えることが可能な教師を T_E で表す。

定理2.4より、構造記述をもつ線型言語 L_s、even 線型言語 L_e を推論するアルゴリズムが存在するかどうかは、正則集合を推論するアルゴリズムが存在するかどうかによって決まる。教師 T_M のもとで任意の正則集合を極限において同定する推論アルゴリズムが存在する [11]。したがって、教師 T_M のもとで L_s、L_e を極限において同定する制御集合にもとづく推論アルゴリズムが存在する。さらに、正則集合を有限の計算の後に同定するアルゴリズムは、補助情報が与えられたときのみ存在する [13]。したがって、T_M のもとで L_s、L_e を有限の計算の後に同定する制御集合にもとづく推論アルゴリズムは存在しない。また、教師 T_A のもとで正則集合を有限の計算で同定する推論アルゴリズムが存在する [3]。したがって、L_s、L_e を T_A のもとで有限の計算で同定するアルゴリズムが存在する。T_E のもとで正則集合を有限の計算で同定する推論アルゴリズムも存在する [2]。それゆえ、T_E のもとで L_s、L_e を有限の計算で同定するアルゴリズムも存在する。

教師 T_M のもとでは、任意の線型言語 L に対してただ1つの正則制御集合が構成的には定義されない。しかし、文法に関する補助情報をアルゴリズム自身が発見することにより、L を極限において同定するアルゴリズムが存在するかもしれない。構成的にではないが、補助情報を用いるとただ1つの正則制御集合を定義できるので、T_A のもとでは少なくとも L を極限において同定する推論アルゴリズムが存在するであろう。教師 T_M、T_A のもとで L を有限の計算で同定する推論アルゴリズムは存在しない。そのようなアルゴリズムが存在すれば、L の同値性判定問題が決定可能になってしまい、矛盾が生じる[1]。教師 T_E のもとでは、線型言語を有限の計算で同定するアルゴリズムが存在する。その際、アルゴリズムの停止は教師への質問によって決定される。ただし、線型言語に対して言語の同値性を判定する能力を持つ教師を仮定することは不自然である。なぜならば、任意の2つの線型言語の同値性を判定するアルゴリズムは存在しないからである。それゆえ、2.2節の表1には、線型言語を極限でのみ同定可能であるという予測を記した。

推論アルゴリズムに関する教師と言語のクラスの関係を表2にまとめる。ただし、かっこをつけて示したものは、予想される性質である。

極限同定と有限同定の違いは、推論アルゴリズムの停止性にある。極限において言語を同定するアルゴリズムは、停止することはない。これに対して、有限の計算の後に言語を同定するアルゴリズムは、言語を同定した後、停止する。

線型言語をモデルとする対象をある実体が学習するとする。この場合、極限同定と有限同定の違いは、実体が学習の達成、つまり対象を学習したという事実を

[1]鎌原研究員の示唆による。同研究員に特に感謝する。

	線型言語 L	構造記述をもつ線型言語 L_s	even線型言語 L_e
T_M	(極限同定)	極限同定	極限同定
T_A	(極限同定)	有限同定	有限同定
T_E	有限同定	有限同定	有限同定

表2: 推論アルゴリズムに関する教師と言語のクラスの関係

認知できるかどうかにかかわる。有限同定可能な推論アルゴリズムが存在する場合、実体は対象を学習したことを認知でき、認知した後は対象を「知っている」と確信できる。一方、極限同定可能な推論アルゴリズムしか存在しえない場合、実体は学習したことを決して認知できず、対象を「知っている」と確信できない。

制御集合にもとづく文法推論をモデルとする学習では、実体の学習の達成に関する認知は言語のクラスと教師の能力によって決定される。まず、言語のクラスから教師の能力と認知の関係を示す。対象が線型言語 L のみをモデルとすることが可能であるならば、実体に学習したと確信させるためには、教師は T_E の能力をもっていなければならない。もし教師が T_M か T_A の能力しか持たない場合、実体はその対象を知っていると確信することはないであろう。対象が構造記述をもつ線型言語 L_s や、even 線型言語 L_e をモデルとすることが可能であるならば、実体に学習したと確信させるためには、教師は T_A または T_E の能力を持っていればよい。教師が T_M の能力しか持たないとき、実体は学習の達成を認知することはできない。

次に、教師の能力から言語のクラスと認知の関係を示す。T_M の能力しか持たない教師は、実体に L、L_s、または L_e をモデルとする対象を学習したことを認知させることができない。逆に、T_E の能力を持つ教師は、どの対象に対しても実体に学習したことを認知させることが可能である。T_A の能力を持つ教師は L_s や L_e をモデルとする対象に対して実体に学習したことを認知させることは可能であるが、L のみをモデルとすることができる対象を実体に学習したことを認知させることは不可能である。

また、実体の学習の達成に関する認知から、教師の能力と、実体が対象のモデルと想定している言語のクラスが推定される。実体が対象を学習したと確信しているとする。この場合、実体の学習アルゴリズムのモデルとしては有限同定可能な推論アルゴリズムが対応する。対象が線型言語 L のみをモデルとすることが可能なことが自明であるならば、教師は T_E の能力をもつと推定される。さらに、対象が L_s や L_e をモデルとするこが可能であるならば、教師は T_A の能力でもかまわないことがわかる。逆に、教師が T_A の能力しか持たないならば、実体が対象のモデルを L_s や L_e としていることが推定される。教師が T_E の能力を持つならば、対象のモデルを L とすることも可能である。教師が T_M の能力しか持たないならば、実体がモデルとして想定しうる言語のクラスはなく、制御集合にもとづく文法推論アルゴリズムを学習のアルゴリズムのモデルとすることが妥当でないことがわかる。

3.2 計算量の意義

ここでいう計算量とは時間計算量である。領域計算量は考慮しないが、時間計算量と領域計算量の間には密接な関係がある。

多項式のオーダの時間計算量のアルゴリズムとは、現在の計算機で効率よく実行可能なアルゴリズムである。指数関数のオーダの時間計算量のアルゴリズムは、組み合わせ爆発などを起こしてしまい、現在の計算機では効率よく実行することができない。したがって、多項式のオーダの時間計算量のアルゴリズムを学習アルゴリズムのモデルとする実体は、与えられた学習の課題を「やさしい」と認知し、指数関数のオーダの時間計算量のアルゴリズムをモデルとする実体は「難しい」と認知する。

2.2節で述べたように、構造記述をもつ線型言語 L_1 や even 線型言語 L_e を多項式の時間計算量で推論するアルゴリズムが存在する。これに対して、線型言語 L を推論するアルゴリズムの時間計算量は指数関数のオーダであることが予想される。したがって、L を実体に多項式のオーダの時間計算量で学習させるには、対象の構造に関する情報を与えてモデルを L_1 とするか、モデルが L_e のサブクラスになるように対象を制限してやらなければならない。ここで重要なのは、推論アルゴリズムの時間計算量の評価が教師の能力と独立であることである。

また、時間計算量から実体が対象に対して想定するモデルのクラスが推定される。制御集合にもとづく線型言語の文法推論をモデルとする学習の問題をある実体が多項式時間の計算量で解くことができるとする。もし実体が L_1 をモデルとすることができない、つまり構造に関する情報を発見できない対象を学習しているならば、実体が対象のモデルを L_e としていることが推定される。逆に、L_e を対象のモデルとすることはできないことがわかっているならば、対象のモデルを L_1 としていることが推定される。L_1、L_e のどちらも対象のモデルとすることができないならば、制御集合にもとづく文法推論アルゴリズムがこの実体の学習アルゴリズムのモデルとして妥当でないことがわかる。このように、時間計算量は、学習における構造記述の情報の必要性、対象のモデルとなる言語のクラス、さらには、文法推論アルゴリズムのモデルとしての妥当性を推定する。

3.3 記号列変換機能の学習

ここでは、教師と計算量の意義を具体的に述べるために、あるアルファベットの記号列を別のアルファベットの記号列に変換する機能を学習する問題について考察する。

実体を人間(以後、学習者と呼ぶ)、学習の対象を記号列のペア (w_i, w_o) からなる集合とする。ここで、w_i は入力記号列、w_o は出力記号列であり、学習者は入力記号列を出力記号列に変換する機能を学習する。このような記号列変換は人間にとって基礎的かつ重要な機能である。

形式的には、学習者の変換機能は6項組 $T = (K, \Sigma, \Delta, \mu, q_0, F)$ で定義される。ここで、K は状態の空でない有限集合、Σ は入力アルファベット、Δ は出力アルファベット、μ は $K \times \Sigma^* \times \Delta^* \times K$ の有限な部分集合、q_0 は初期状態、F は最終状態の集合である。$K \times \Sigma^* \times \Delta^*$ 上の関係 $\longmapsto$ を、Σ^* の各要素 w に対して、$(p, x, y, q) \in \mu$ かつ

$z_2 = z_1 y$ のとき、

$$(p, xw, z_1) \longmapsto (q, w, z_2)$$

と定義する。関係 $\overset{*}{\longmapsto}$ は、$\longmapsto$ の反射的推移的閉包である。変換機能 T が変換可能な入力記号列と、それに対応する出力記号列のペアの集合 $T(T)$ は、

$$T(T) = \{(w_i, w_o) | (q_0, w_i, \lambda) \overset{*}{\longmapsto} (q, \lambda, w_o),\ q \in F\}$$

である。

w^T で記号列 w の鏡像を表わす。たとえば、記号列 $w = abcd$ に対して、$w^T = dcba$ である。$T = (K, \Sigma, \Delta, \mu, q_0, F)$ を任意の変換機能とする。T に対して、線型文法 $G = (K, \Sigma \cup \Delta \cup \{\#\}, \Pi, q_0)$ を定義する。ここで、Π は次のように定義される:

- $(q, w_i, w_o, p) \in \mu$ に対して、$q \rightarrow w_i p w_o^T \in \Pi$、
- $q \in F$ に対して、$q \rightarrow \# \in \Pi$。

変換機能 $T = (K, \Sigma, \Delta, \mu, q_0, F)$ と線型文法 $G = (K, \Sigma \cup \Delta \cup \{\#\}, \Pi, q_0)$ に関して、次の定理が成立する。

定理 3.1　任意の入出力記号列のペア $(w_i, w_o) \in \Sigma^* \times \Delta^*$ に対して、$(w_i, w_o) \in T(T)$ である必要十分条件は $w_i \# w_o^T \in L(G)$ である。

定理3.1 より、学習の対象のモデルを線型言語 L、実体の変換機能のモデルを線型文法とする。

$T = (K, \Sigma, \Delta, \mu, q_0, F)$ を変換機能、$G = (K, \Sigma \cup \Delta \cup \{\#\}, \Pi, q_0)$を T に対する線型文法とする。このとき、G のかっこ文法 $[G]$ によって構造記述をもつ線型言語 L_s を得る。また、μ を $K \times \Sigma \times \Delta \times K$ の部分集合に制限することで、G は even 線型文法になる。したがって、対象のモデルは even 線型言語 L_e になる。

例題　　我々人間は、馴染みのない物の数を認知するのに、具体的な馴染みやすい物に対応させると認知しやすくなる。たとえば、ローマ数字の表わす数は認知しにくい。そこで、各ローマ数字の文字をその文字が表わす算用数字と算術記号 '+' と '−' に変換する。変換した記号列を算術式と見て、演算を実行すれば、容易にローマ数字を認知できる。

ここでは、記号 I、V、X だけからなるローマ数字を対象とする。ローマ数字では、I が 1 に、V が 5 に、X が 10 にそれぞれ対応する。数を構成する記号は、対応する算用数字の値が大きい順に左から右へ並べられ、数の値は各記号の値の加算によって得られる。もし値の小さい記号が大きい記号の左に現われたならば、負の値をもつと見なされる。したがって、XVI は 16 で、IXX は 19 である。そこで、ローマ数字から算術式への変換機能 $T_R = (K, \Sigma, \Delta, \mu, q_p, F)$ を次のように定義する：

- $K = \{q_p, q_{n_1}, q_{n_2}\}$、
- $\Sigma = \{\mathrm{I}, \mathrm{V}, \mathrm{X}\}$、

- $\Delta = \{0, 1, 5, +, -\}$、
- $\mu = \left\{ \begin{array}{lll} (q_p, \mathrm{I}, +1, q_p), & (q_p, \mathrm{V}, +5, q_{n_1}), & (q_p, \mathrm{X}, +10, q_{n_2}), \\ (q_{n_1}, \mathrm{I}, -1, q_{n_1}), & (q_{n_1}, \mathrm{V}, +5, q_{n_1}), & (q_{n_1}, \mathrm{X}, +10, q_{n_2}), \\ (q_{n_2}, \mathrm{I}, -1, q_{n_2}), & (q_{n_2}, \mathrm{V}, -5, q_{n_2}), & (q_{n_2}, \mathrm{X}, +10, q_{n_2}) \end{array} \right\}$、
- $F = K$。

ただし、右から左へと走査したほうが変換しやすいので、ローマ数字の鏡像が入力記号列として与えられるものとする。変換機能 T_R によって、ローマ数字 XVI は算術式 $+1+5+10$ に、IXX は $+10+10-1$ に変換される。これらの算術式の値は簡単に求められる。

変換機能 T_R に対応する線型文法 G_R は、前述の方法により求められる。この文法 G_R によって生成される言語 $L_R = L(G_R)$ は線型言語であり、構造記述をもつ線型言語のクラスや、even 線型言語のクラスには属さない。

このように、記号列変換機能は人間の認知において重要な役割を演ずる。

さて、教師が学習者に記号列の変換機能を教育することを考える。この場合、教師も人間と仮定する。対象のモデルが構造記述をもつ線型言語 L_s や even 線型言語 L_e でよいならば、定理2.4、3.1 より、変換機能の学習の問題は正則集合の同定の問題に還元される。したがって、教師は学習者にまず変換機能と線型文法の関係を教え、そして正則集合の同定方法とフロントエンド処理の機能を教えればよい。教えられた学習アルゴリズムにより、学習者は任意の変換機能を多項式時間で学習することが可能である。また、入出力ペアから帰納的に学習させる場合、教師が T_E、T_A の能力を持っていれば、学習者に学習したことを認知させることが可能である。しかし、T_M の能力しか持たない場合には、認知させることは不可能である。

対象のモデルが線型言語 L でなければならないならば、教師は表現定理にもとづいて正則集合の同定方法との類推から変換機能の学習方法を教えることになる。この際、教えられた学習アルゴリズムにより学習者が任意の変換機能を学習するのに、指数関数の時間計算量を必要とするであろう。また、教師が T_M、T_A の能力しか持たない場合、学習者は学習したことを認知することはできない。教師自身も言語の同値性を判定することは一般にはできないから、T_E の能力を持つことはない。

したがって、学習者が一般的な変換機能を効率良く学習し、学習の達成を認知するには、教師が対象の構造に関する情報を学習者に与える能力と、T_E か T_A の能力を持っていればよい。特に、対象の構造に関する情報は学習者にとって非常に重要な要因である。先のローマ数字の変換機能の例では、比較的、ローマ数字の構造を発見しやすく、学習者が効率良く学習することが期待できる。

我々は観察や実験などによって、学習者のアルゴリズムの時間計算量を求め、学習の達成の認知を知ることが可能である。もし L をモデルとする対象を学習する時間計算量が多項式時間であるならば、構造記述の情報を学習者が発見し、利用していることが推定される。あるいは、構造記述の情報を発見することができ

ないでいるならば、その学習者は対象のモデルをL_eとして学習し、L_eの要素でない入出力記号列のペアは学習できないと推定される。どちらの場合でもなければ、制御集合にもとづく推論アルゴリズムよりも効率の良いアルゴリズムを用いていることが予想される。逆に、もし対象がL_rやL_eをモデルとするにもかかわらず、指数関数の時間計算量を必要としたならば、学習者は制御集合にもとづく推論アルゴリズムを知らないことが推定される。また、教師がT_Mの能力しか持たない場合や、教師はT_Aの能力を持つがLのみが対象のモデルとなりうる場合に、学習者が学習の達成を認知したと確信した場合、教師はその認知が誤りであることを指摘できる。

このように、文法推論を記号列変換機能の学習の形式モデルとすることにより得られる教師や計算量に関する帰結は、教育、観察・実験における重要な示唆ならびに仮説である。

4 結論

本稿では、文法推論を学習の形式モデルとしたときに、モデルから得られる帰結の意義を認知科学の側面から示した。教師の能力は実体の学習の達成の認知を、計算量は学習の難易度の認知を決定する。線型言語をモデルとする対象の学習においては、実体が対象の構造に関する情報を発見でき、教師が補助情報を与える能力または対象の同値性を判定する能力を持つとき、理想的な問題設定となる。

ここで述べた以外にも、文法推論を学習の形式モデルとすることにより、さまざまな意義を得ることができる。[4] では、教師の能力と推論問題の関係が考察されている。[21] では、前置詞(prefix) の所属に関する質問に答える能力を持つ教師とsimple言語の推論問題の関係が考察されている。また、[15] では構造記述を持つ文脈自由言語の効率的な推論アルゴリズムが述べられている。さらに、認知科学の観点からは、事例の提示の順序と推論アルゴリズムの能力の関係を研究することも重要である。

このように、文法推論問題を形式モデルとすることにより得られる帰結は、認知科学においてより多くの意義を持つ。また、心理学や教育学などの経験科学にとっても重要な意義を持つ。

謝辞

日頃、有益な助言をいただく国際研・学習システム研究グループの横森、榊原、石坂各研究員と西田研究員に感謝します。なお、本研究は第五世代コンピュータプロジェクトの一環として行なったものである。

参考文献

[1] V. Amar and G. Putzolu. On a family of linear grammars. *Information and Control*, 7:283–291, 1964.

[2] D. Angluin. Learning regular sets from queries and counter-examples. *Information and Computation*, 75:87–106, 1987.

[3] D. Angluin. A note on the number of queries needed to identify regular languages. *Information and Control*, 51:76–87, 1981.

[4] D. Angluin. *Types of queries for concept learning.* TR 479, YALEU/DCS, 1986.

[5] D. Angluin and C. H. Smith. Inductive inference : theory and methods. *ACM Computing Surveys*, 15(3):237–269, 1983.

[6] A. W. Biermann. A grammatical inference program for linear languages. In *Proceedings of Forth Hawaii International Conference on System Sciences*, pages 121–123, 1971.

[7] A. W. Biermann. An interactive finite-state language learner. In *Proceedings of First USA-JAPAN Computer Conference*, pages 13–20, 1972.

[8] S. Ginsburg and E. H. Spanier. Control sets on grammars. *Mathematical Systems Theory*, 2(2):159–177, 1968.

[9] M. A. Harrison. *Introduction to Formal Language Theory.* Addison-Wesley, Reading, Massachusetts, 1978.

[10] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, Massachusetts, 1979.

[11] H. Ishizaka. Inductive inference of regular languages based on model inference. 1987. To appear in Lecture Notes in Computer Science, Logic Programming '87.

[12] R. McNaughton. Parenthesis grammars. *Journal of the ACM*, 14(3):490–500, 1967.

[13] E. F. Moore. Gedanken-experiments on sequential machines. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, pages pp.129–153, Princeton Univ. Press, Princeton, N.J., 1956.

[14] V. Radhakrishnan and G. Nagaraja. Inference of even linear grammars and its application to picture description languages. *Pattern Recognition*, 21(1):55–62, 1988.

[15] Y. Sakakibara. *Inferring parsers of context-free languages from structural examples.* Research Report 81, IIAS-SIS, FUJITSU LIMITED, 1987.

[16] A. Salomaa. *Formal Languages.* Academic Press, Inc., New York, 1973.

[17] Y. Takada. *A constructive method for grammatical inference of linear languages based on control sets.* Research Report 78, IIAS-SIS, FUJITSU LIMITED, 1987.

[18] Y. Takada. Grammatical inference for even linear languages based on control sets. Submitted.

[19] Y. Takada. Inductive inference for linear grammars based on control sets. Submitted.

[20] K. Tanatsugu. A grammatical inference for context-free languages based on self-embedding. *Bulletin of Informatics and Cybernetics*, 22:149–163, 1987.

[21] T. Yokomori. Learning simple languages in polynomial time. 1988. Private memo.

付録

ここでは、本稿で用いる形式言語の概念と用語を説明する。さらに詳しい説明は、[9]、[10]、[16] を参照せよ。

Σ をアルファベット、Σ^* を Σ 上のすべての記号列の集合とする。ただし、Σ^* は空記号列 λ を含む。$|u|$ を記号列 u の長さとする。

決定性有限オートマトン M は5項組 $(K, \Sigma, \delta, q_0, F)$ である。ここで、K は状態の空でない有限集合、Σ は有限集合である入力アルファベット、δ は $K \times \Sigma$ から K への遷移関数、q_0 は初期状態で K の要素、F は最終状態の集合で K の部分集合である。次のように、遷移関数 δ を $K \times \Sigma^*$ から K への関数に拡張する: 任意の状態 q に対して、

$$\delta(q, \lambda) = q$$

任意の記号列 $u, v \in \Sigma^*$ に対して、$\delta(q, uv) = \delta(\delta(q, u), v)$。

M によって受理されるすべての記号列の集合は $T(M)$ で表され、

$$T(M) = \{u \in \Sigma^* \mid \delta(q, u) \in F\}$$

である。

非決定性有限オートマトン M' は5項組 $(K, \Sigma, \delta', q_0, F)$ である。K、Σ、q_0、F は決定性有限オートマトンの場合と同じであるが、δ' は $K \times \Sigma$ から 2^K への関数である。遷移関数 δ' も $2^K \times \Sigma^*$ から 2^K への関数に拡張される: 任意の状態 $q \in K$ に対して、

$$\delta'(q, \lambda) = \{q\}$$

任意の記号列 $u, v \in \Sigma^*$ に対して、

$$\delta'(q, uv) = \{p \mid \text{ある状態 } r \in \delta'(q, u) \text{ に対して、} p \in \delta'(r, v)\},$$

$Q \subseteq K$ に対して、

$$\delta'(Q, u) = \bigcup_{q \in Q} \delta'(q, u)。$$

M' によって受理される記号列の集合 $T(M')$ は

$$T(M') = \{u \in \Sigma^* \mid \delta'(q_0, u) \cap F \neq \emptyset\}$$

である。定義により決定性有限オートマトンは非決定性有限オートマトンである。

集合 R が非決定性有限オートマトンによって受理されるならば、R を受理する決定性有限オートマトンが存在する。決定性有限オートマトンによって受理される Σ^* の部分集合 R を正則という。

文脈自由文法 G は 4 項組 (N,Σ,Π,S) である。ここで、N は変数の空でない有限集合である。N と Σ は共通要素を持たないと仮定する。$N\cup\Sigma$ を V で表す。Π は生成規則の空でない有限集合である: 各生成規則は $A\to x$ の形で表される。ここで、A は変数、x は V 上の記号列である。ここでは、ラベル π_i をつけて生成規則を区別する。S は特別な変数で、開始記号と呼ばれる。

$G=(N,\Sigma,\Pi,S)$ を文脈自由文法とする。V^* の記号列の間に次の関係 $\overset{\pi}{\underset{G}{\Rightarrow}}$ を定義する。任意の V^* の記号列 x と y に対して、$x=uAv$、$y=uxv$、$\pi_i: A\to x$ が G の生成規則ならば、$x\overset{\pi_i}{\underset{G}{\Rightarrow}}y$ である。このとき、生成規則 $\pi_i: A\to x$ が記号列 x に適用され、y が得られた、と言われる。

$x_0,x_1,\ldots,x_k$ を V^* の記号列とする。

$$x_0\overset{\pi_1}{\underset{G}{\Rightarrow}}x_1,\ x_1\overset{\pi_2}{\underset{G}{\Rightarrow}}x_2,\ldots,x_{k-1}\overset{\pi_k}{\underset{G}{\Rightarrow}}x_k$$

であるならば、$x_0\overset{\alpha}{\underset{G}{\Rightarrow}}x_k$ と表し、付随語 α をもつ G における x_0 から x_k への導出と言う。ここで、$\alpha=\pi_1\pi_2\cdots\pi_k$ である。

文脈自由文法 $G=(N,\Sigma,\Pi,S)$ によって生成される言語 $L(G)$ は、

$$L(G)=\{w\in\Sigma^*\mid S\overset{\alpha}{\underset{G}{\Rightarrow}}w,\ \alpha\in\Pi^*\}$$

である。文脈自由文法 G_1、G_2 は、$L(G_1)=L(G_2)$ のとき、同値であるといわれる。

次の条件を満たす文脈自由文法 $G=(N,\Sigma,\Pi,S)$ を簡約された文脈自由文法という:

- Π は $A\to A$ の形をした生成規則を含まない、
- $A\to\lambda$ の形をした生成規則が Π に含まれるならば、A は開始記号 S である、
- 任意の変数 A に対して、$S\overset{\alpha}{\underset{G}{\Rightarrow}}uAv\overset{\alpha'}{\underset{G}{\Rightarrow}}w$ である $w\in L(G)$ が存在する。

任意の文脈自由文法 G に対して、$L(G)=L(G')$ である簡約された文脈自由文法 G' が存在する。

$G=(N,\Sigma,\Pi,S)$ を文脈自由文法とする。導出

$$u_1A_1v_1\overset{\pi_1}{\underset{G}{\Rightarrow}}u_2A_2v_2\overset{\pi_2}{\underset{G}{\Rightarrow}}\cdots\overset{\pi_{n-1}}{\underset{G}{\Rightarrow}}u_nA_nv_n$$

において、$u_i\in\Sigma^*$ $(1\le i\le n)$ のとき、この導出を最左導出という。$L(G)$ の中に 2 つ以上の相異なった最左導出を有する語が存在しないとき、G を無あいまいであるという。

$\mathcal{M}$ がある未知の言語 L を推論しようとする文法推論アルゴリズムとする。$\mathcal{M}$ が L のしだいに増加する事例の集まりに対して繰り返し適用され、$\mathcal{M}$ の仮説である文法の無

限列が $G_1, G_2, \ldots$ のように生成されるものとする。もしある数 m が存在して、G_m が L を生成する文法であり、

$$G_m = G_{m+1} = G_{m+2} = \cdots$$

とすると、$\mathcal{M}$ はこの事例の系列の上で極限において L を正しく同定するといわれる。