3M-3

文脈自由文法の漸次的学習システム *

吉岡 智 † 松本 雅史 £ 中村 克彦 ‡

東京電機大学大学院理工学研究科 § 東京電機大学大学院理工学研究科 § 東京電機大学理工学部 ¶

1 まえがき

われわれは，与えられた正負の文字列例から文脈自由文法を合成するような文法推論の方式の研究を続けており，この方式を実装したSynapseシステムを開発している．現在，Synapseでは生成規則の数が10個程度の文法を合成することができるが，生成規則数が多くなると探索空間が非常に大きくなるため，複雑な文法を合成することができないという大きな問題がある．

本報告では，Synapseシステムの最新の文法合成の結果と漸次的学習方式によって探索時間の問題を解決する方法について述べる．

2 変形 Chomsky 標準形

任意の文脈自由文法 (CFG) $G = (N, T, P, S)$ をこれと等価な，Chomsky 標準形の文法に変換できることが知られている．規則集合探索プロセスの効率化のため，われわれは次の形式の規則のみからなる変形 Chomsky 標準形の文法を用いる．

$$A \to \beta\gamma \quad \beta, \gamma \in N \cup T.$$

この形式の文法は長さが1の文字列を含まないことを除いて一般の文脈自由言語を表すことができる．したがって，変形 Chomsky 標準形の文法は十分に一般的なものであるとみなすことができる．このように1つだけの形式の規則に制限することによって，終端記号の個数が少ない言語に対して一般に規則集合を少なくでき，また文法の合成を簡単化できる．さらに，記号列中の各記号 a を2つの記号 aa に置換えることによって，言語を $A \to a$ なる規則の代わりに変形 Chomsky 標準形の規則 $A \to aa$ を用いた文法で表すことができる．

Synapse:Automatic Synthesis of Context Free Grammars *
Satoru Yoshioka †
Masashi Matsumoto £
Katsuhiko Nakamura ‡
Graduate School of Science and Engineering, Tokyo Denki University §
Faculty of Science and Engineering, Tokyo Denki University ¶

3 帰納的 CYK アルゴリズム

Synapseシステムでは帰納的 CYK アルゴリズムによって正例の文字列と初期規則集合に対して，これを導出するための規則集合が合成される．与えられた文字列が規則集合から導出されるとき，帰納的 CYK アルゴリズムは通常の CYK アルゴリズムと同様に働く．文字列が規則集合から導出できないとき，このアルゴリズムは与えられた記号列を導出するために必要な規則を生成し，規則集合に追加する機能をもっている．以下に帰納的 CYK アルゴリズムの概要を示す．

1. 入力文字列 w に対し，CYK アルゴリズムを適用する．w の構文解析に成功したならば終了．
2. 構文解析に失敗した場合，CYK アルゴリズムの実行において規則を適用した記号の対の集合（テスト集合と呼ぶ）を変数 TS に保存する．
3. TS から記号の対を1つ選択し，これを右辺とした生成規則を合成して規則集合に加える.（この規則の合成には対の選択に加えて，規則の左辺の非終端記号の選択があるが，これらの選択はこの後の処理からバックトラックされるたびにやり直される.）Step 1に戻る．

Step 3において，新たな非終端記号 A を用いて規則 $A \to BC$ を合成した場合，次に合成される規則は右辺に必ず A を含むという制限が可能となる．これによって規則集合の合成の 2 〜 20 倍の効率化が可能となる．

4 Synapse システム

われわれは Prolog と C 言語を用いて Synapse システムを開発している．Prolog 版では，主にアルゴリズムや探索法の実験・改良を行い，C 言語版ではシステムの高速化および機能の充実を図っている．

Synapseは与えられた正負の例に対して，正例の文字列を導出し，負例の文字列を導出しない文脈自由文法を自動合成する．また，指定によりあいまい・非あいまいの両方の文法を合成することが可能である．システムの動作の概要は次の通りである．

表 1: Synapse で合成された文法の例

Language		Set of rules	R	Time	GR	r
(a) balanced parentheses	A	$S \to (\ ) \mid C \ ) \mid SS,\ C \to (\ S$	4	< 1	245	2
	U	$S \to (\ ) \mid C \ ),$ $C \to S(\mid (\ S \mid DS,\ D \to S($	6	4	18023	2
(b) $\#_a(w) = \#_b(w)$	A	$S \to ab \mid ba \mid bC \mid Cb \mid SS,$ $C \to aS \mid Sa$	7	2	6534	2
(c) $\{w \mid w \in \{a,b\}^+,$ $\#_a(w) < \#_b(w)\}$	A	$S \to Sb \mid SS_1 \mid bb \mid bS_1 \mid S_1b,$ $S_1 \to ab \mid ba \mid bC \mid Cb \mid S_1S_1,$ $C \to aS_1 \mid S_1a$	12	35	4.3×10^5	2
(d) $\{ww \mid w \in \{a,b\}^+\}$	A	$S \to CD \mid DC \mid FE \mid GE,$ $C \to aa \mid FE, D \to bb \mid GE,$ $E \to aa \mid bb, F \to EC,$ $G \to ED$	12	10^4	6.7×10^7	3

A: あいまい　　U: 非あいまい

R: 生成規則数　　GR: 合成した総規則数

r: １つの正例に対する最大合成規則数

1. 反復深化のための規則数の上限値を決める．この範囲内で求める規則集合が得られない場合，この制限を緩和する．
2. 正例の１つに対して，帰納的 CYK アルゴリズムを実行する．与えられた制限の中で正例の導出ができない場合，Step １に戻る．
3. 正例の導出に成功したら，すべての負例に対して CYK アルゴリズムによって導出を判定する．負例を導出した場合は Step ２にバックトラックして，帰納的 CYK アルゴリズムによる規則の合成ををやり直す．
4. その他の場合，次の正例を取り出して手順２から操作を繰り返す．すべての正例についての処理が終了したとき，求める規則集合が得られる．

長さの順に正例を与えることによって，漸次的学習が実現される．

5 実行結果

いくつかの文法の合成結果を表１に示す．

5.1 Chomsky 標準形の文法の合成

文法によっては変形 Chomsky 標準形では通常の Chomsky 標準形より規則数が多くなるものがある．この一例が「ww の形でない文字列の集合」の文法 (表１の (d)) である．この文法は，以下に示すような一般の Chomsky 標準形を実質的に変形 Chomsky 標準形として扱う方式により合成された．

- 規則 $A \to aa$，$B \to bb$，$C \to aa \mid bb$ を初期規則集合として与える．
- 文字列の記号を，a は aa，b は bb と置き換え，２つづつ重ねたものとする．
- 規則の右辺は両方とも非終端記号というチョムスキー標準形の規則のみを合成する．

5.2 漸次的学習

言語「a より b を多く含む記号列の集合」(表１の (c)) は，言語「a と b の数が同じである記号列の集合」(表１の (b)) に類似した言語である．Synapse システムにあらかじめ (b) の文法を学習させ，事前知識としてこの文法を初期規則集合として与えることによって，言語 (C) の文法が時間では 1/30 以下，合成した総規則数では 1/170 以下に高速化された（合成時間 = 1(s), GR = 2524）．

6 むすび

一般的なチョムスキー標準形への応用，漸次的な学習によって，複雑な文法の合成が可能となった．

参考文献

[1] Nakamura, K. and Matsumoto, M. ，文脈自由文法の漸次的学習方式，ICGI 2002，2002.