Synapse:文脈自由文法の自動合成システム†

5Q−04

松本　雅史‡　中村　克彦¶

東京電機大学大学院理工学研究科§　東京電機大学理工学部

1　はじめに

本報告では，正の文例と負の文例を入力することによって，帰納推論を用いて自動的に文脈自由文法を合成する文法推論システム Synapse (Synthesis by Analyzing Positive String Examples) について述べる．Synapse システムは，構文解析のための CYK アルゴリズムに規則を合成する機能を加えた帰納的 CYK アルゴリズムを基本としており，次のような特徴をもっている．

- 規則が最小の文法を合成できる．
- あいまいな文法と非あいまいな文法を合成できる．
- すでにある文法の規則に新たな規則を追加して，新しい文法を合成できる．
- 合成に要する時間が少ない．

本報告では，改良された規則生成方式と文法合成の最新の結果について述べる．

2　帰納的 CYK アルゴリズム

2.1　アルゴリズム

合成される文法は，次の形式の変形 Chomsky 標準形の規則からなる．

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

帰納的 CYK アルゴリズムは正例の記号列 w と規則集合 R を入力として，次のように働く．

1. CYK アルゴリズム用いて R から w が導出できるか調べ，導出できれば終了する．このとき同時に下に述べるようなテスト集合を作成する．
2. w が導出できない場合，テスト集合を用いて新たに規則を追加し，1. からのステップを繰りかえす．

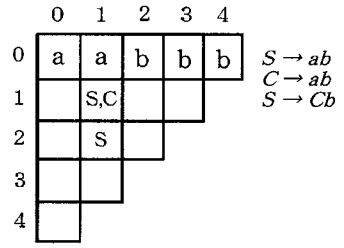

図 1: 入力記号列 aabbb に対して構成される配列

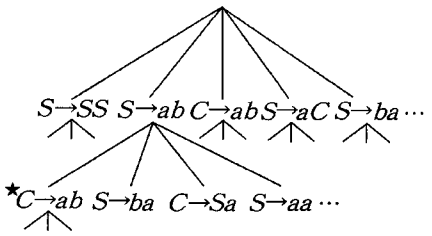

図 2: 探索木

テスト集合とは，CYK アルゴリズムにおいて規則が適用できるか否かを調べる記号の対である．図 1 では $\{aa, ab, bb, aS, \cdots\}$ がテスト集合である．上記のステップ 2. においてこの記号の対を右辺に持つような規則が合成される．

2.2　探索方法

実際には上のアルゴリズムで作られる規則集合のなかで最小のものが図 2 のような探索によって求められる．この探索木は上から順に追加する規則を示している．例えば図の★の位置の場合，規則集合は $\{S \to ab, C \to ab\}$ となる．

解の探索においては，規則数の上限を 1 から順に増やしていく反復深化を用いている．反復深化はそれ以前に行った探索と同じ探索を何度も繰り返し行う可能

†Synapse:Automatic Synthesis of Context Free Grammars
‡Masashi Matsumoto
§Tokyo Denki University Graduate School of Science and Engineering
¶Katuhiko Nakamura

表 1: 文法合成結果と合成時間 (sec)

Language		Rule set	$\#N$	#rules	Time	#G-R	#M-R
$a^n b^n$ $(n \geq 1)$	U	$S \to ab \mid Cb, C \to aS$	2	3	< 1	64	2
$a^m b^n$	A	$S \to ab \mid Sb \mid aC, C \to Sb$	2	4	< 1	84	2
$(1 \leq m \leq n)$	U	$S \to ab \mid aD \mid Cb, C \to ab \mid Cb, D \to Sb$	3	6	< 1	7324	2
Parenthesis	A	$S \to ab \mid Cb \mid SS, C \to aS$	2	4	< 1	376	2
language	U	$S \to ab \mid Cb, C \to Sa \mid aS \mid DS, D \to Sa$	3	6	7	26234	2
$w = w^R$	A	$S \to aa \mid bb \mid bD \mid Ca, C \to aa \mid ab \mid aS,$	3	10	234	1.6×10^6	2
$w \in \{a, b\}$	U	$D \to ab \mid bb \mid Sb$	3	10	131	9.4×10^5	2
$\#_a(w) = \#_b(w)$	A	$S \to ab \mid ba \mid bC \mid Cb \mid SS, C \to aS \mid Sa$	2	7	< 1	24805	2
$2\#_a(w) = \#_b(w)$	A	$S \to bC \mid Cb \mid SS, C \to ab \mid ba \mid bD \mid Db,$					
$w \in \{a, b\}^+$		$D \to aS \mid Sa$	3	9	192	3.3×10^6	2

A: あいまいな文法　U: 非あいまいな文法　w^R: w の逆
$\#N$: 非終端記号の数　#G-R: 生成された全規則数　#rules: 規則数
#M-R: 1つの例から作られた最大規則数　$\#_a(w)$: w の中に含まれる a の文字数

性があるという欠点があるが，深さ優先探索によって実質的に幅優先探索を行うことができる．

3 Synapse システム

Synapse システムは与えられた正例の系列，負例の系列に対して次のように働く．

1. 最初の正例に対して帰納的 CYK アルゴリズムを適用する．
2. この結果作られた規則集合が負例を導出するか否かを調べる．導出してしまう場合は 1. にバックトラックして，同じ正例に対する別の規則集合を生成する．負例を導出しない場合は 1. に戻り次の正例に対して帰納的 CYK アルゴリズムを適用することを繰りかえす．

このプロセスにおいても反復深化によって，最大規則数の制限を順に大きくしながら，探索を繰りかえす．

4 結果とむすび

本システムびよって合成できた文法，および合成時間を表 1 に示す．これは AMD Athlon 1GHz 搭載のマシンでの結果である．正例は記号列の長さが短い順に入力し，負の文例はある程度の長さの記号列までをすべて与えた．本方式では，長さの順に正例を与えたとき，1 ステップで合成される規則数 (#M-R) が小さくなり，合成時間も少なくなる．最初により長い正例を与えた場合の合成時間を表 2 に示す．

表 2: 最初の文字長と合成時間

L	First samples	T	#G-R	#M-R
2	ab	7	26234	2
4	$abab, aabb$	10	38090	2
6	$ababab, aabbab, \cdots$	58	2.0×10^5	3
8	$(ab)^4, aabbaabb, \cdots$	92	9.2×10^5	3

L: 最初に与える正例の長さ　T: 合成時間

本研究では，基本的な文脈自由文法を比較的短時間で合成することに成功した．今後の課題は以下の点である．

- さらに複雑な文法の合成を短時間でできるような改良を行うこと．
- 本方式は文法規則に別の規則を加え，他の文法の推論が可能であるという特徴をもつ．この機能を活かした文法推論について調べること．

参考文献

[1] Katsuhiko Nakamura and Takashi Ishiwata, Synthesizing Context Free Grammars from Sample Strings Based on Inductive CYK Algorithm, LNAI 1891, Fifth International Colloquium on Grammatical Inference, ICGI 2000,(2000)186-195