正規化制約を用いたCCG構文解析とその実装

尾崎 博子　　戸次 大介
お茶の水女子大学大学院 人間文化創成科学研究科理学専攻情報科学コース
{ozaki.hiroko, bekki}@is.ocha.ac.jp

1　はじめに

現在、社会の情報化が進み大量の電子テキストが流通するようになっている。そこでそれらのデータを有効活用するための構文解析技術の重要性が高まってきている。構文解析を行うにあたり様々な文法理論が存在するが、そのうちの一つであるCCG（Combinatory Categorial Grammar：組み合わせ範疇文法 [1]）は多くの語彙項目といくつかの組み合わせ規則から成る。CCGは自然言語処理において有用性が示されているが、その一方で簡単な文に対してさえ同じ意味を持つ複数の構文木が導出されることがある。

これは解析処理効率の観点からは問題であり、1つの意味に対しては構文木を一意に決定したい。そこで必要となってくるのが正規化解析技術 [2] [3] である。これは構文解析のための効率的で完全性のある手法である。範疇文法は論理の証明論と対応しているため、このような論理学の手法を応用できることが利点の1つである。

本研究では、CCGの型繰り上げ規則を規則として設定せず、辞書内限定の適用とした場合の正規化制約を設定し、その妥当性を証明した上で、構文解析器を実装する。

2　構文解析

形式文法には様々な種類があるが、最も基本的なものの1つにCFG（Context Free Grammar：文脈自由文法）がある。しかし、CFGでの解析には、完全な文法を書くにはルール数が増大してしまう、という問題がある。これを解決するために、単語自体に情報を詰め込んでルール数を少なくする、という語彙化文法の方法がある。

たとえば、“John”という単語は名詞であり、三人称単数という素性を持つ。この情報を型付き素性構造を用いて以下のように表すことができる。

$$\boxed{\text{John}} \quad \begin{bmatrix} np \\ \text{POS: 名詞} \\ \text{PER: 三人称} \\ \text{NUM: 単数} \end{bmatrix}$$

npは型の名前、POS, PER, NUMは型npが持つ素性である。型付き素性構造をデータ構造とするのがLiLFeSというプログラミング言語 [4] であり、本研究の実装はLiLFeSを用いている。

3　CCG

CCGは辞書といくつかの規則で構成されている。辞書は多数の語彙項目を含んでおり、語彙項目は記号列、統語範疇、意味表示から成る。例えば、

$$\text{John} \vdash \ S/(S\backslash NP) \ : \ \lambda P.Pj$$
$$\text{sees} \vdash \ S\backslash NP/NP \ : \ \lambda y.\lambda x.\lambda e.see(e,x,y)$$

のように各語について語彙項目が定められている。

さらに、語からより大きな構成素を作るために組み合わせ規則がある。今回実装した組み合わせ規則は以下の4つである。

組み合わせ規則

$$(>)\frac{\Gamma \vdash X/Y : f \quad \Delta \vdash Y : a}{\Gamma,\Delta \vdash X : fa} \qquad (<)\frac{\Gamma \vdash Y : a \quad \Delta \vdash X\backslash Y : f}{\Gamma,\Delta \vdash X : fa}$$

$$(>B)\frac{\Gamma \vdash X/Y : f \quad \Delta \vdash Y/Z : g}{\Gamma,\Delta \vdash X/Z : \lambda x.fgx} \qquad (<B)\frac{\Gamma \vdash Y\backslash Z : g \quad \Delta \vdash X\backslash Y : f}{\Gamma,\Delta \vdash X\backslash Z : \lambda x.fgx}$$

4　語彙項目などの定義

統語範疇、意味表示は以下のように定義する。

統語範疇

$s \mid np \mid X/Y \mid X\backslash Y$

意味表示

$x \mid \lambda x.M \mid MN \mid pred(x_1, \cdots, x_n)\ (1 \leq n \leq 3)$

ただし、統語範疇の np については格・人称・単複の素性構造を付加している。

5 正規化解析

第1節で述べたように、CCG のような関数合成規則を持つ範疇文法では1つの意味に対して複数の異なる構文解析結果が指数的に導出される。これを spurious ambiguity という。たとえば、"John sees vincent." という文に対しては以下のような2つの異なる構文解析結果が得られる。

$$\heartsuit\quad \dfrac{\dfrac{\text{John}}{S/(S\backslash NP)}\ (>)\quad \dfrac{\dfrac{\text{sees}}{S\backslash NP/NP}\quad \dfrac{\text{vincent.}}{T\backslash(T/NP)}}{S\backslash NP}\ (<)}{S}\ (>)$$

$$\spadesuit\quad \dfrac{\dfrac{\dfrac{\text{John}}{S/(S\backslash NP)}\quad \dfrac{\text{sees}}{S\backslash NP/NP}}{S/NP}\ (>B)\quad \dfrac{\text{vincent.}}{T\backslash(T/NP)}}{S}\ (<)$$

(図 1)"John sees vincent." の導出

しかし、意味が同じである導出結果を全て計算することは非効率的である。そのため、1つの意味に対しては唯一の解析結果（正規形）を見つけたい。正規形が見つかれば、処理中に余分な結果を随時枝刈りすることができ、処理効率の向上につながる。これが正規化解析である。

5.1 関連研究

[2] では、以下のような正規化制約を定めている。

a. No constituent produced by $> B_n$, any $n \geq 1$, ever serves as the primary (left) argument to $> B_{n'}$, any $n^{'} \geq 0$.

b. No constituent produced by $< B_n$, any $n \geq 1$, ever serves as the primary (right) argument to $< B_{n'}$, any $n^{'} \geq 0$.

上の制約は安全性と完全性が証明されている。また、型繰り上げ規則の適用は辞書内に限定されている。

[3] では以下のような正規化制約を定めている。

1. The output of $> B^{n \geq 1}$ $(resp. < B^{n \geq 1})$ cannot be primary functor for $> B^{n \leq 1}$ $(resp. < B^{n \leq 1})$

2. The output of $> B^{1}$ $(resp. < B^{1})$ cannot be primary functor for $> B^{n \geq 1}$ $(resp. < B^{n \geq 1})$

3. The output of $> B^{m}$ $(resp. < B^{m})$ cannot be secondary functor for $> B^{n>m}$ $(resp. < B^{n>m}$

4. The output of $> T$ cannot be primary input to $> B^{n>0}$ if the secondary input is the output of $< B^{m>n}$. The output of $< T$ cannot be primary input to $< B^{n>0}$ if the secondary input is the output of $> B^{m>n}$

5. The output of forward (or backward) type-raising $> T$ $(resp. < T)$ cannot be the functor in application $>$ $(resp. <)$

[2] では示されていなかった一般合成規則と型繰り上げ規則についても対処できるように正規化制約が拡張された。また、この制約を基にした構文解析への効果の検証も行われている。

型繰り上げ規則

$$(>T)\dfrac{\Gamma \vdash X : a}{\Gamma \vdash T/(T\backslash X) : \lambda P.Pa} \qquad (<T)\dfrac{\Gamma \vdash X : a}{\Gamma \vdash T\backslash(T/X) : \lambda P.Pa}$$

5.2 提案

本研究では CCG の型繰り上げ規則を規則としては設定せず、固有名詞等は辞書において繰り上げられた型を持つものとする。

これは、型繰り上げ規則の採用には以下のような問題があるからである。

- 同一の構成素に対して繰り返し適用可能な規則のため、ナイーブな実装では構文解析が停止しない。
- 型変数を使用するためトップダウンの解析が難しい。
- Extraction のような言語現象では型繰り上げ規則があると問題が発生する。[5]

よって辞書内での固有名詞の定義は以下のようになっている。

$$\begin{array}{lcll} \text{John} & \vdash & S/(S\backslash NP_{nom,3,sg}) & : \lambda P.Pj \quad \text{(主格)} \\ & \vdash & T\backslash(T/NP_{acc}) & : \lambda P.Pj \quad \text{(対格)} \end{array}$$

しかし、[2], [3] の制約では図1における正規形を決定することはできないという問題がある。

そこで、[3] の **4.**,**5.** を以下のように意味表示に言及する形式の制約に変更する。

⋆ The category whose semantic representation of the form $\lambda P.Px$ cannot be functor in $> B^{n \geq 0}$ $(resp. < B^{n \geq 0})$ if the argument is the output of $< B^{m>n}$ $(resp. > B^{m>n})$

$$>B^{n\geq 0}\frac{T/(T\backslash Z):\lambda P.Px \qquad <B^{m>n}\dfrac{Y\ |_{m-1}\ Z_{m-1}\cdots|_{n-1}\ Z_{n-1}\backslash Z\ |_n\ Z_n\cdots|_1\ Z_1 : g \quad X\backslash Y : f}{X\ |_{m-1}\ Z_{m-1}\cdots|_{n-1}\ Z_{n-1}\backslash Z\ |_n\ Z_n\cdots|_1\ Z_1 : \lambda x_1.\cdots\lambda x_m.fgx_1.\cdots x_m}}{X\ |_{m-1}\ Z_{m-1}\cdots|_{n-1}\ Z_{n-1}\ |_n\ Z_n\cdots|_1\ Z_1 : \lambda y_1.\cdots\lambda y_{m-1}.fgy_1\cdots y_nxy_{n+1}\cdots y_{m-1}}$$

(図 2) 制約 $\star$ における非正規形

$$<B^{m-1}\frac{>B^{n}\dfrac{T/(T\backslash Z):\lambda P.Px \quad Y\ |_{m-1}\ Z_{m-1}\cdots|_{n-1}\ Z_{n-1}\backslash Z\ |_n\ Z_n\cdots|_1\ Z_1 : g}{Y\ |_{m-1}\ Z_{m-1}\cdots|_{n-1}\ Z_{n-1}\ |_n\ Z_n\cdots|_1\ Z_1 : \lambda x_1.\cdots\lambda x_n.gx_1\cdots x_nx} \qquad X\backslash Y : f}{X\ |_{m-1}\ Z_{m-1}\cdots|_{n-1}\ Z_{n-1}\ |_n\ Z_n\cdots|_1\ Z_1 : \lambda y_1.\cdots\lambda y_{m-1}.fgy_1\cdots y_nxy_{n+1}\cdots y_{m-1}}$$

(図 3) 制約 $\star$ における正規形

6 検証

第 6 節で述べた制約が構文解析において 1 つの意味に対し唯一の導出を見つけられているかを検証する。そのために、[3] の制約 **1.**, **2.**, **3.** と制約 $\star$ 下での安全性と完全性の証明を行った。

安全性とは以下のような性質である。

安全性

> 全ての構文木 α について、α と意味的に同値な正規形構文木 $NF(\alpha)$ が存在する

非正規形の構文木が与えられた場合、それと意味的に同値な正規形構文木が存在することを示すことで安全性の証明とする。

[証明]

1.2.3. に関しては [3] で証明済みとする。そのため、$\star$ についてのみ安全性を証明すればよい。

制約 $\star$ に従い、図 2 のような非正規形構文木が与えられた場合、図 3 のように意味的に同値な正規形構文木が存在する。よって制約 $\star$ に関する安全性が証明された。以上より、本研究の制約下での安全性が証明された。

完全性とは以下のような性質である。

完全性

> 葉が同じである正規形構文木 $\alpha \neq \alpha'$ が与えられたとき、α と α' は意味的に同値ではない

完全性の証明については [3] を参考にし、構成素の並びが同じである正規形構文木は意味的に異なることを示すこととする。

本研究では、制約の条件より長さ 3 の構成素における場合においてこれを示せばよい。証明の詳細については省略するが、場合分けは以下の通りである。

i) Y, $(X\backslash Y)/Z$, Z から X が導出される場合
ii) X/X, $(X \mid \alpha) \mid \beta$, $(X \mid \alpha)\backslash(X \mid \alpha)$ から $(X \mid \alpha) \mid \beta$ が導出される場合
iii) Y, $((X \mid \alpha)\backslash Y) \mid \beta$, $X\backslash X$ から $(X \mid \alpha) \mid \beta$ が導出される場合
iv) Y, $(((X\backslash Y) \mid \alpha)/Z) \mid \beta$, Z から $(X \mid \alpha) \mid \beta$ が導出される場合
v) Y_A, Y_B, $(((X\backslash Y_1) \mid \alpha)\backslash Y_2) \mid \beta$ から $(X \mid \alpha) \mid \beta$ が導出される場合
vi) Y_A, $(((X\backslash Y_1) \mid \alpha)/Z$, $((Z \mid \beta)\backslash Y_2) \mid \gamma$ から $(((X \mid \alpha) \mid \beta)\backslash Y_2) \mid \gamma$ か $(((X\backslash Y_1) \mid \alpha) \mid \beta) \mid \gamma$ が導出される場合

以上より本研究の制約下での完全性が証明された。

7 実装

以上の分析を基に、正規化制約を組み込んだ構文解析器の CYK 法による実装を行った。また、意味表示については β 簡約された結果が出るようになっている。

なお、本研究では CCG の組み合わせ規則として第 3 節に挙げた 4 つのみを用いているため、実装においてもこれらに関する規則と正規化制約のみを実装している。

以下は “John sees vincent.” の構文解析の実行例である。R は解析に用いた規則、C は統語範疇、S は β 簡約された意味表示、である。

```
> ?- test("John sees vincent",C,R,S).
   |~r_mp          ~|
   | C1:lex_        |
R: |    |~l_mp   ~| |
   | C2:| C1:lex_ | |
   |_   |_C2:lex__|_|
C: s
S: "\lambda e.(see(e,j,v))"
Enter ';' for more choices, otherwise press ENTER --> ;
no
```

(図 4) 実行例

8　まとめと今後の課題

本研究では統語範疇、意味を同時に計算する CCG 構文解析器の実装を行った。また、型繰り上げ規則のない CCG での正規化制約を設定し、完全性と安全性を証明し、構文解析器へ組み込んだ。今後の課題として、辞書を拡張することで解析対象を広げること、対象言語を日本語にも拡張することなどがある。

参考文献

[1] Steedman, M. J. (2000) The Syntactic Process (Language, Speech, and Communication). The MIT Press.

[2] Eisner, J. (1996) “Efficient Normal-Form Parsing for Combiantory Categorial Grammar”, In the Proceedings of ACL ’96 Proceedings of the 34th annual meeting on Association for Computational Linguistics. pp.79-86.

[3] Hockenmaier, J. and B. Yonatan. (2010) “Normal-form parsing for Combinatory Categorial Grammars with generalized composition and typeraising”, In the Proceedings of the Proceeding of COLING 2010. pp.465-473.

[4] LiLFeS ホームページ, http://www.nactem.ac.uk/lilfes/index.html

[5] Ozaki, H. and Bekki D. (2012) “Extractability as the Deduction Theorem in Subdirectional Combinatory Logic”, In the Proceedings of LACL 2012, pp.186-200.

[6] 尾崎博子, 戸次大介. (2012) “部分方向性組み合わせ論理における日本語のかき混ぜ文の正規化制約”, 第 26 回人工知能学会 JSAI 2012.