2G1-OS3-6

証明木作成プログラムを用いたCCG意味合成の実装

Implementation of Semantic Composition in CCG by a Proof Tree Generation Program

尾崎有梨 Yuri Ozaki　櫻井加奈子 Kanako Sakurai　浅井健一 Kenichi Asai　戸次大介 Daisuke Bekki

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
Advanced Sciences, Graduate School of Humanities and Sciences, Ochanomizu University

The correspondence between Combinatory Categorial Grammar (CCG) and logical systems/type theories in the sense of Curry=Howard=Lambek isomorphism has been recently paid attention in the field of mathematical linguistics. However, it is not clear in this context how to interpret some 'fine-grained' natures of CCG, such as the type variables and their unifications. This study investigates such relation between CCG and logical systems/type theories through the process of implementing the semantic composition of CCG utilizing a proof tree generating program.

1. はじめに

文法理論と論理や型システムとの対応をとることは、自然言語処理に有用なものであり、どのように自然言語が数学的に表現できるかを考察するうえでも重要である。また文法理論の一種である CCG(Combinatory Categorial Grammar:[1][2][3]) は等位接続現象に優れているなど頑健なパーザのための文法理論として知られており、CCG の文法理論をもとに型システムとの対応を考えることにより、自然言語の数理的理解につながると考える。

そこで [14] ではそのための第一段階として CCG 統語導出を証明木作成プログラムを用いて実装することで、CCG とシーケント計算の対応を考察した。本研究では続いて意味表示の実装を行う。しかし、意味表示には統語構造と密接に関わっており、さらに、もともとそれ自身が論理的表現であるうえに、[14] であがった型変数のような問題点にも CCG の型システムとしての解釈を目的するうえで、数学的な考察が必要であると考えられる。

本論文では、4 節で CCG の意味構造の概要や、統語構造との関係を述べ、5 節ではその関係性を導くような CCG の数学的概念の対応を提案し、6 節も含め、実装を行う上での仮定をたてる。

2. CCG の概要

CCG の概要については [1][2][3] を参照のこと。以下に CCG の統語導出規則をあげる。

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

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

$$>_{S_\times}\frac{\Gamma \Rightarrow f : (X/Y)\backslash Z \quad \Delta \Rightarrow g : Y\backslash Z}{\Gamma, \Delta \Rightarrow \lambda x.fx(gx) : X\backslash Z} \qquad <_{S_\times}\frac{\Gamma \Rightarrow g : Y/Z \quad \Delta \Rightarrow f : (X\backslash Y)/Z}{\Gamma, \Delta \Rightarrow \lambda x.fx(gx) : X/Z}$$

$$>_T\frac{\Gamma \Rightarrow a : X}{\Gamma \Rightarrow \lambda f.fa : \boldsymbol{T}/(\boldsymbol{T}\backslash X)} \qquad <_T\frac{\Gamma \Rightarrow a : X}{\Gamma \Rightarrow \lambda f.fa : \boldsymbol{T}\backslash(\boldsymbol{T}/X)}$$

$$<_\Phi>\frac{\Gamma_1 \Rightarrow f_1 : X \quad \cdots \quad \Gamma_{n-1} \Rightarrow f_{n-1} : X \quad \Delta \Rightarrow \circ : CONJ \quad \Gamma_n \Rightarrow f_n : X}{\Gamma_1, \cdots, \Gamma_{n-1}, \Delta, \Gamma_n \Rightarrow \lambda\vec{x}.(f_1\vec{x}) \circ \cdots \circ (f_n\vec{x}) : X}$$

連絡先: 〒 112-8610 東京都文京区大塚 2-1-1
ozaki.yuri@is.ocha.ac.jp

上から順に、関数適用規則 (>)、逆関数適用規則 (<)、順関数合成規則 (>B)、逆関数合成規則 (<B)、順型繰り上げ規則 (>$\boldsymbol{T}$)、逆型繰り上げ規則 (<$\boldsymbol{T}$)、順関数交差置換規則 (>$S_\times$)、逆関数交差置換規則 (<$S_\times$)、等位接続規則 (<Φ>) という。X や Y、$\boldsymbol{T}$ は任意の統語範疇である。

3. 型システムとの関連

CCG における構成素は、記号列、型付きラムダ計算で表された意味表示、統語範疇とで構成されている。一方型システムは、環境・項・型で構成されており、構造的に CCG と型システムには対応関係がある。これにより、組み合わせ規則と型付け規則の間にも本来対応があると考えられる。

そこで [14] では、CCG における統語導出過程の実装を行った。実装例は以下の通りである。

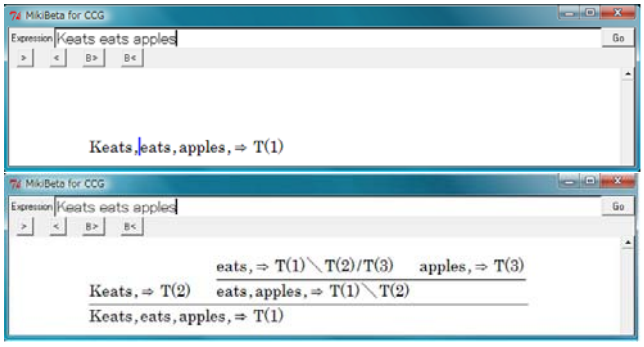

図 4: 証明木作成プログラムを用いた統語導出の実装

この実装の目的とは以下の問題を調べるためであった。

1. CCG の記号列・意味表示・統語範疇の全ての情報がプログラミング言語の環境・値 (式)・型として記述されうるか。
2. 1 が可能である場合、CCG の統語範疇の推論を型推論の問題とみなして計算できるか。

型繰り上規則で用いられる範疇変数には型システムにはな

$$\dfrac{keats \Rightarrow NP : schools \quad <\dfrac{eats \Rightarrow (S\backslash NP)/NP : \lambda y.\lambda x.\lambda e.eat(e,x,y) \quad apples \Rightarrow NP : apples}{eat, apples \Rightarrow S\backslash NP : \lambda x.\lambda e.eat(e,x,apples)}}{Keats, eats, apples \Rightarrow S : \lambda e.eat(e, keats, apples)}$$

図 1: CCG 導出例

$$\dfrac{cooks \Rightarrow (S\backslash NP)/NP : \lambda y.\lambda x.\lambda e.cook(e,x,y) \quad >\dfrac{eats \Rightarrow (S\backslash NP)/NP : \lambda y.\lambda x.\lambda e.eat(e,x,y) \quad apples \Rightarrow NP : apples}{eats, apples \Rightarrow S\backslash NP : \lambda x.\lambda e.eat(e,x,apples)}}{}$$

図 2: 統語論的エラーの例

$$\dfrac{Schools \Rightarrow NP : schools \quad <\dfrac{eats \Rightarrow (S\backslash NP)/NP : \lambda y.\lambda x.\lambda e.eat(e,x,y) \quad apples \Rightarrow NP : apples}{eat, apples \Rightarrow S\backslash NP : \lambda x.\lambda e.eat(e,x,apples)}}{Schools, eat, apples \Rightarrow S : \lambda e.eat(e, schools, apples)}$$

図 3: 意味論的エラーの例

い特徴が見受けられ、型推論における型変数と似ているのではないかと考察をした。(以下は型繰り上げ規則実行例)

$$\dfrac{>^T\dfrac{Keats \Rightarrow NP}{Keats \Rightarrow \boldsymbol{T}/(\boldsymbol{T}\backslash NP)} \quad eats \Rightarrow (S\backslash NP)/NP}{Keats, eats \Rightarrow S}$$

本研究では統語構造に続いて意味構造の実装も目指すが、CCG の意味構造は次節に述べるように数学的関わりが深い。そのため型繰上規則のさらなる考察を含め、CCG が型システムや論理とどのような関係であるかを考察する必要がある。

4. CCG の意味合成

4.1 CCG 意味構造の概要

意味合成も含めた CCG の導出の例は図 1 の通りである。本研究では [12] に倣い、CCG の意味表示には型付きラムダ計算を使用するものとする。
例として、CCG での *eats* の意味表示は次のようである。

$$\lambda y.\lambda x.\lambda e.eat(e,x,y)$$

これは、x eats y を表し、eat には、「誰が」の x と「何を」の y という引数が 2 つ必要であることを示す。
一方型システムにおいての演算子は

$$+(3,5)$$

のように表され、+ の演算子は第 1 引数と第 2 引数を加算することを意味し、CCG の意味表示は型システムにおける演算子のようなものといえる。

4.2 意味構造と統語構造の関係

さらに、意味構造と統語構造の関係について考察する。

CCG で導出に失敗した場合は、統合的な不整合によるものである。一方、統語的に整合であり導出に成功する場合も、意味的にも整合とは限らない (意味表示の真偽は状況から判断されるものなので、文法的には well-formed とされる。)

図 2 のように,“cooks eats apples” という文法的に間違っている文は最後まで導出できない (意味を生成することができない) が、図 3 の “schools eat apples” のように意味表示が間違っているものは、それが間違いでも最後まで正しく統語導出することができる。

このように、CCG では統語構造によって、その文が意味構造としても正しい文に生成されるかが決まる。一方型システムでは、以下のように項が計算できるかどうかは、型に依存している。

int 型	+	int 型	→	整合
string 型	+	int 型	→	不整合

このように、CCG では統語構造が、型システムでは型が、整合性があるかかを決定づけている。意味表示や項は、統語構造、型に依存して挙動が決定する。このことから、CCG と型システムにおいて、上にも述べた通り、統語構造と型、意味構造と項が対応が取れていると自然に考えられる。

5. CCG と数学的概念の対応への 1 つの仮定

Lambek 計算 [4] は [10] らの古典的 Categorial Grammar をシーケント計算の形式で定式化したもので、「統語導出＝証明」という考え方を提示した。その後、線形論理、ファジー論理などの部分構造論理の発展により Lambek 計算もその一種と考えられるようになった。古典論理の構造規則は以下の 3 つである。

$$\dfrac{\Gamma \Rightarrow \Sigma}{\Gamma, A \Rightarrow \Sigma} \qquad \dfrac{\Gamma, A, A \Rightarrow \Sigma}{\Gamma, A \Rightarrow \Sigma} \qquad \dfrac{\Gamma_1, A, \Gamma_2, B, \Gamma_3 \Rightarrow \Sigma}{\Gamma_1, B, \Gamma_2, A, \Gamma_3 \Rightarrow \Sigma}$$

左から Weakning,Contraction,Exchange という規則で、Lambek 計算ではこれらの構造規則が成り立たない。しかしこれはまさに、CCG の環境部分の規則に等しくなっていて、CCG が Lambek 計算の一種といえる。

部分構造論理のカリー・ハワード対応を考えると、それに対応するラムダ計算があると考えられる。一方、[11] は、統語論に Categorial Grammar を、意味論にラムダ計算を使用した。よって、Lambek 計算と部分構造論理、ラムダ計算の関係によって Lambek-λ 計算が提案された。

また、CCG は言語学的要請から生まれたが、組合せ論理との対応も示唆されている。

組合せ論理はヒルベルトシステムとカリー・ハワード対応が成り立っている。そこで、CCG と論理にも対応関係が期待されるが、CCG は部分構造論理でもあるので、CCG は部分構造的ヒルベルトシステムであり、型システムの観点からみれば、部分構造的組合せ論理といえる。これは、CCG での意味構造はある種の Lambek -λ 計算と言えるのではないかという仮定が立てられる。しかし、CCG における型変数、統語範疇が Lambek 計算とのどのような関係であるかといった詳細は明らかにはなっていない。[9] では CCG の一種と Associative Lambek 計算の等価性が示されている。しかし、ここで用いられた CCG は [1] の CCG で主張されている Principle of Directional Consistency 及び、Principle of Directional Inheritance を考慮していない。従って [9] の CCG は自然言語の文法理論では過剰生成であると考えられ、[1] にあるような CCG において Lambek 計算との関係性を考察する必要がある。

6. まとめ：CCG と型システムの相違点

4.2 節より、統語範疇を型として組み合わせる CCG と、型があえば計算できる型システムは対応しているといえる。

一方で、CCG と型システムの相違点をあげると、CCG では環境から型と項を同時に推論するのに対し、型システムは、環境と項から型を推論するという点があげられる。このことが、より一般的な部分から生じるものであるときを考え、5. 節のカリー・ハワードの関係を述べた状況から substructural なラムダ計算としての CCG の位置づけを考察したい。ところが、CCG には現実の文法記述に必要とされる拡張的な概念がいくつか用意されており、その中には型繰上にともなう型変数と、それらの間の単一化 (unification) 演算のようなものがあるが、それはカリー・ハワード同型の枠組みの中では対応物をみつけられないものである。

それに対して、[14] において、ヒンドレー・ミルナー流の型推論アルゴリズムのなかに現れる型変数と、CCG の型変数が対応しているのではないかと考えた。

我々が CCG と型システムの対応を調べるために証明木作成プログラムによる実装を手段としている理由はこの点にある。つまり、記号体系としての型システムには現れず、型システムの実装のみに現れるような概念に対応物が見出される場合があるということである。

ところが、部分構造ラムダ計算における型推論の仕組みはそれほど明らかではないが、先行研究ではそこまで考察をしておらず CCG において、統語構造では実装として対応がみつかったが、意味構造では逆に、実装する点では型推論の初期状態として、項が不明な状態で推論を開始するのが自然であるが、型 system の型推論では考えにくい状況である。

そのために我々は意味合成の実装においては、不明な項を表すメタ変数が必要になるであろう。

従って、型システムとして実装することによって、逆に CCG と型システムの相違点が浮かび上がるということが推測され、CCG 導出 GUI を実装中である。

参考文献

[1] Steedman, Mark. *Surface Structure and Interpretation*, The MIT Press, 1996.

[2] Steedman, Mark. *The Syntactic Process*, The MIT Press, 2001.

[3] Steedman, Mark, and Baldridge, Jason. *Combinatory Categorial Grammar*, In Borsley, R. and Borjars, K. (eds.), *Non-Transformational Syntax*. Blackwell, 2007.

[4] Lambek, J. *The Mathematics of Sentence Structure*, American Mathematical Monthly, 65, pp.154-170, 1958.

[5] Buszkowski, W. *The logic of types*, 1987.

[6] Wansing, H. *The Logic of Information Structures*, 1993.

[7] Polakow, J. and F. Pfenning. *Natural deduction for intuitionistic non-commutative linear logic*, 1999.

[8] Buszkowski, W. *Mathematical linguistics and proof theory*, 1997.

[9] Jäger, Gerhard. *Lambek Grammars as Combinatory Categorial Grammars*, L.J. of the IPGL, Vol 9 No6, pp.781-792, 2001.

[10] Ajudkiewicz, K. *Die Syntakische Konnexitat*, Studia Philosophica 1, pp.1-27, 1935. Transl. In: McCall, S.(Ed.): Polish Logic in 1929-1939. Oxford: Clearendon, 1967.

[11] Montague, R. *The Proper Treatment of Quanification in Ordinary English*, In: J. Hintikka, J.Moravcsic, and P.Suppes (eds.); Approaches to Natural Language. Dordrecht, Reidel, pp.221-242, 1973

[12] 戸次 大介, 「日本語文法の形式理論—活用体系・統語導出・意味合成—」, くろしお出版, 2010.

[13] 櫻井 加奈子, 浅井 健一, 「汎用的に証明木を作成する『Mikiβ』」, 第 12 回プログラミングおよびプログラミング言語ワークショップ PPL2010, 2010.

[14] 尾崎 有梨, 櫻井 加奈子, 浅井 健一, 戸次 大介, 「証明木作成プログラムを用いた CCG 統語導出の実装」, 言語処理学会第 16 回年次大会発表論文集, pp.334-336, 東京大学, 2010.