数理解析研究所講究録
180巻 1973年 20-29

Combinatory Logic について

津田塾大　細井勉

0.

計算機科学において理論的な研究を行なおうとするとき，たいていの場合に，何らかの種類の記号列を扱うための形式的体系が必要となる．形式論理の上にのせられる場合も多いが，処理方法がそこで考えているsemanticsに関係していることから，形式論理にのせることが効果的（あるいは，実質的）でないこともある．しかし，個々の問題（あるいは，研究課題）に応じて，個別に形式的体系を展開する必要があるほど，個々の問題での処理方法が個性的であるとは言えないようである．つまり，ある程度の統一をとって，flexibilityのある1つの形式的体系を用意しておけばよい，と考えられる根拠も多分に観察される．このようなとき，そのような体系を新たに提案する行き方もあるが，既製のものに使えるものがないかと調べることも自然なことであろう．

このような目論見をもってcombinatory logic (CL)を調べ

てみたので，簡単にCLについて紹介したい．CLがこのような目的に適したものであるかどうか，また，仮に適しているとした場合，CLにはどんな有益な結果が用意されているか，という問題点に関しては，残念ながら，まだ結論を得てはいない．CLの紹介を，計算機関係の会合で行なわないで，このような論理関係の会合で行なうわけは，論理の関係者が本来興味を持ってよい話題だと考えるからである．

参考文献

[1] H.B. Curry and R. Feys, Combinatory Logic I, North-Holland, 1958.

[2] R. Feys and F.B. Fitch, Dictionary of Symbols of Mathematical Logic, North-Holland, 1969.

[3] N.D. Goodman, A Simplification of Combinatory Logic, JSL 37 (1972), 225−246.

[4] J.R. Hindley, B. Lercher and J.P. Seldin, Introduction to Combinatory Logic, Cambridge U.P., 1972.

[5] S. Stenlund, Introduction to Combinatory Logic, Uppsala, 1971.

文献[1]とその続篇は，網羅的であり解説もくわしいが，話の本筋はつかみにくい．[2]の第4章は，分量も少なく，CLのoutlineをつかむのに手頃である．[3]は体系の記述において優れているが，CLのふんいきを知っている者でないと理解しにくいようである．[4]は入門用に書かれていてわかり易い本であり，簡単な応用にもふれている．簡潔で要を得ている．[5]も[4]と同様である．初心者用のムダ話的な解説が多いのが特長的．

1．CLの目的

文献[2]によると，CLは関数に関するapplicationとabstractionの2つの過程を抽象的に扱うことを目的とする．applicationとは，関数（あるいは，その名前）とその引数（あるいは，その名前）が与えられたとき，関数にその引数をapplyして，その値（あるいは，その名前）を求める過程であり，abstractionとは，引数に対応する関数値に関する情報が与えられたとき，その関数自身（あるいは，その名前）を見つける過程である．

文献[5]によると，Schönfinkelがはじめて CL を考えだしたときの目論見は，logicにおいて，primitive notionをできるだけ少くしようとすることだったという．とくに，

(bound variable があると複雑になる)代入演算を，modus ponens並に簡単にしたい，ということであったらしい．代入演算を処理するのに，ふつうは，variableが使われているが，この場合，いやらしい例に遭遇する．たとえば，⊃と⊃に関して何かを記述しているはずの⊢ ¬p⊃(p⊃q)を見て，pとqに関する何かが述べられているように錯覚をもつ例がそうである．そこで，variableの消去が問題となる．

CLは，このような目的の下に，かなりきれいに展開されてきていることは確かなようである．しかし，このような目的はさておき，CLにおいて，記号列の処理のための理論がどれだけ展開されているのかということを問題にすることは，それなりの意味があるのではないだろうか．

2. Formulation of the Theory of Combinators

文献[5]によるCLの定義を紹介する．

このTheoryで扱う対象物はtermである．

<u>Def. 2.1.</u> (i) atomic combinators S, K, I は term．

(ii) atomic constant は(もしあれば) term．

(iii) 可算個の variables a, b, c, …… は term．

(iv) X と Y が term なら，(X Y) は term．

(v) Term は，上記により定まるもののみ．

括弧は左から補うこととし，適宜，省略する．X，Y，Z，…などにより，syntactical variable を表わす．

Def. 2.2. atomic combinator だけを含む term を combinator という．（関数の役を果す．）たとえば，S(KS)K．

この Theory での theorem は term の間の equation である．

Def. 2.3. weak equality $=_w$ は，つぎの axiom と rule を満足するとする．

(S) $SXYZ =_w XZ(YZ)$　　(K) $KXY =_w X$

(I) $IX =_w X$　　(r) $X =_w X$

(a) $\dfrac{X =_w Y}{XZ =_w YZ}$　　(b) $\dfrac{X =_w Y}{ZX =_w ZY}$

(c) $\dfrac{X =_w Y}{Y =_w X}$　　(d) $\dfrac{X =_w Y,\ Y =_w Z}{X =_w Z}$

Theorem 2.4. X はたかだか $x_1, \cdots, x_n$ を variable として含む term とする．このとき，combinator F で，関係 $Fx_1 \cdots x_n =_w X$ を満足するものが存在する．

（Combinatory completeness theorem）

この定理の証明としては，つぎに定義する $x_1, \cdots, x_n$ に関する X の abstraction $[x_1, \cdots, x_n].X$ を F とすればよい．（証略）これにより，任意の関数を combinator としてとらえられることがわかる．

<u>Def. 2.5.</u> term $[x_1,\cdots,x_n].X$ をつぎのように定義する．

$n=1$ のとき (i) $[x].X \equiv KX$ if $X \not\ni x$

(ii) $[x].X \equiv I$ if $X \equiv x$

(iii) $[x].X \equiv Y$ if $X \equiv Yx$ and $Y \not\ni x$

(iv) $[x].X \equiv S([x].Y)([x].Z)$ if $X \equiv YZ$

かつ，(i)，(ii)，(iii) の case ではない．

$n>1$ のとき (v) $[x_1,\cdots,x_n].X \equiv [x_1].([x_2,\cdots,x_n].X)$

例 $Fxyz = wx(yz)$ となる F を求める．

$$F \equiv [x,y,z].x(yz) \equiv [x].([y].([z].x(yz)))$$
$$\equiv [x].([y].(S([z].x)([z].yz)))$$
$$\equiv [x].([y].S(Kx)y) \equiv [x].S(Kx)$$
$$\equiv S([x].S)([x].Kx) \equiv S(KS)K$$

weak reduction の定義および Church-Rosser の主要定理があるが，省略する．

3. Combinators

定義により，I は identity operator，K は constant function であることがわかる．2つの関数 f，g が与えられたとき，$Sfgx$ とは，関数

$$Sfgx = f(x, g(x))$$

を定義していると解釈できる．Iは，

$$I \equiv SKK$$

により定義できるので，essential ではない．

CLの理解において重要なことに，つぎのようなことがある．(1) すべての関数あるいは演算子は unary である．(2) 関数はその引数あるいは値に自分と同じレベルの対象物をもつことがある．たとえば，Kxy は値 x をもつわけであるが，この意味は，K は unary function であり，Kx は unary constant function を値とし，したがって $(Kx)y$ は値 x をもつ，ということである．

有名な combinator の例をあげておこう．

(B) $Bfgx \overset{\text{def}}{=} f(gx)$ これは，前に示したように，

$B \equiv S(KS)K$ で定義することができる．（以下同様）

(W) $Wfx = fxx$ $\qquad W \equiv SS(KI)$

(C) $Cfxy = fyx$ $\qquad C \equiv S(BBS)(KK)$

(Φ) $\Phi fghx = f(gx)(hx)$ $\quad \Phi \equiv B(BS)B$

X，Y を combinator としたとき，つぎの Extensionality が成立つかどうかは興味のある問題である．

(Ext) $\forall x,\ Xx = Yx \Rightarrow X = Y$

この (Ext) は，equality として weak equality をとる場合には成立たないことが分っている．この (Ext) が成立つ

ように改造したequalityをstrong equalityというが，その定義は述べない．

primitiveなcombinatorとして，{K, S}をとれば，任意のcombinatorを構成できることは分っているわけであるが，このような例は他にもあり，たとえば，{B, C, W, K}でもよい．

<u>Theorem 3.1.</u> Combinator全体の集合は，strong equalityに関して，Iをidentity，Bを演算子とするsemi-groupである．かつ，$Jfxyz = fx(fzy)$としたとき，{CII, CIJ, B, CI}はその生成元である．

4. λ-conversion

Theory of Combinators (with strong equality)と同等なものに，Theory of λ-conversionがある．ここでは，その定義だけを紹介する．これは，abstractionのためのprimitive operatorをもつことと，variableがessentialであるという点で，基本的態度が，Theory of Combinatorsとは異なる．

基本的には，$x+y$というようなexpressionに名前を与えて，関数として扱えるようにすることである．

primitive symbolsは，λ，(，)，•の4つ．variable

は可算個あり，それらを a, b, c, ‥‥ とする．

このTheoryでは，λ-termを基本的な対象物とする．

<u>Def. 4.1.</u> (i) variable はλ-term.

(ii) atomic constant は，(もしあれば) λ-term.

(iii) X, Y がλ-term なら，(XY) はλ-term.

(iv) X がλ-term で x が variable なら，λx.X はλ-term.

(v) λ-term は，上記により定まるもののみ．

各λ-term は関数を表わすものと解釈される．その関数の引数は関数でもよい．(XY)は，関数XにYをapplyした結果を表わすと解釈される．λの意味は，下の(C3)により，自然に解釈される．

variable がλに関して free であるか bound であるかを，ふつうのように定義する．λ-term Y の中の variable x に，λ-term X を代入することを，ふつうのように定義し，その結果を [X/x].Y で表わす．

Axioms と Rules はつぎのとおり．

Axioms : (C1) $X =_\lambda X$

(C2) $\lambda x.X =_\lambda \lambda y.([y/x].X)$ ただし，y は X の中で free でないとき．

(C3) $(\lambda x.Y)X =_\lambda [X/x].Y$

(C4) $\lambda x.Xx =_\lambda X$ ただし，x は X の中で free でないとき．

Rules : (C5) $\dfrac{X =_\lambda Y}{Y =_\lambda X}$

(C6) $$\frac{X =_\lambda Y, \quad Y =_\lambda Z}{X =_\lambda Z}$$

(C7) $$\frac{X =_\lambda Y}{ZX =_\lambda ZY} \quad \text{および} \quad \frac{X =_\lambda Y}{XZ =_\lambda YZ}$$

(C8) $$\frac{X =_\lambda Y}{\lambda x.X =_\lambda \lambda x.Y}$$

つぎのExtensionalityが成立つことが知られている.

(C9) xがvariableで, XとYに現われていないとき,

$$Xx =_\lambda Yx \quad \text{ならば} \quad X =_\lambda Y$$

5. 応用

応用について, くわしいことは知らないが, つぎのようなものがよく見られる.

(1) wffをcombinatorで表わし, CL流にLogicを展開.

(2) 数をcombinatorで表わし, CL流に, 数論やrecursive function論を展開.

このように, すでに分っていることをCL流に直すことは, 本質的ではないと思いたい. CLの中に見られる"数学"がどんなものであるか, また, どんな応用があるか, などについては, まだよく理解していないので, 何も述べない.