ソフトウェア構成特論 第6回

大学院理工学研究科 電気電子情報工学専攻　篠埜 功

2015年5月21日

1　はじめに

今回は、型無しラムダ計算に関する操作的意味論について紹介する。

2　型無しラムダ計算 (untyped lambda calculus)

ラムダ式は1930年代にChurchにより考案されたものであり、β変換と呼ばれる変換によりラムダ式を変換していくことによって行われるのが計算であるという考えである。その計算体系をラムダ計算と呼んでいる。後にラムダ式に型をつけた体系が考えられ、それを型付きラムダ計算、型のついてないのを型無しラムダ計算と呼ぶ。

ラムダ式（の集合）は以下のように定義する。

$$\begin{array}{rcl} \mathtt{t} & := & \mathtt{x} \\ & | & \lambda \mathtt{x}.\mathtt{t} \\ & | & \mathtt{t}\ \mathtt{t} \end{array}$$

値は以下のように定義する。

$$\mathtt{v} := \lambda \mathtt{x}.\mathtt{t}$$

ラムダ式の評価規則は以下のように定義する。

$$\frac{\mathtt{t}_1 \to \mathtt{t}_1'}{\mathtt{t}_1\ \mathtt{t}_2 \to \mathtt{t}_1'\ \mathtt{t}_2}\ \text{(E-APP1)} \quad \frac{\mathtt{t}_2 \to \mathtt{t}_2'}{\mathtt{v}_1\ \mathtt{t}_2 \to \mathtt{v}_1\ \mathtt{t}_2'}\ \text{(E-APP2)} \quad \frac{}{(\lambda \mathtt{x}.\mathtt{t}_{12})\ \mathtt{v}_2 \to [\mathtt{x} \mapsto \mathtt{v}_2]\mathtt{t}_{12}}\ \text{(E-APPABS)}$$

これはsmall-step semanticsである。ラムダ式のbig-step semanticsによる評価規則はこの講義では扱わない。

E-APPABS規則中の$[\mathtt{x} \mapsto \mathtt{s}]\ \mathtt{t}$は、ラムダ式$\mathtt{t}$中で自由に現れる変数$\mathtt{x}$をラムダ式$\mathtt{s}$で置き換えたラムダ式を表し、以下のように定義される。

$$\begin{array}{ll} [\mathtt{x} \mapsto \mathtt{s}]\mathtt{x} = \mathtt{s} & \\ [\mathtt{x} \mapsto \mathtt{s}]\mathtt{y} = \mathtt{y} & \text{if } \mathtt{y} \neq \mathtt{x} \\ [\mathtt{x} \mapsto \mathtt{s}](\lambda \mathtt{y}.\ \mathtt{t}_1) = \lambda \mathtt{y}.\ [\mathtt{x} \mapsto \mathtt{s}]\mathtt{t}_1 & \text{if } \mathtt{y} \neq \mathtt{x} \text{ and } \mathtt{y} \notin FV(\mathtt{s}) \\ [\mathtt{x} \mapsto \mathtt{s}](\mathtt{t}_1\ \mathtt{t}_2) = ([\mathtt{x} \mapsto \mathtt{s}]\mathtt{t}_1)([\mathtt{x} \mapsto \mathtt{s}]\mathtt{t}_2) & \end{array}$$

ただし、ラムダ抽象に対する置換適用時には、必要に応じて置換対象のラムダ抽象式の束縛変数の付け替えを行う。例えば、$[\mathtt{x} \mapsto \mathtt{y}\ \mathtt{z}](\lambda \mathtt{y}.\ \mathtt{x}\ \mathtt{y})$は、まず$\lambda \mathtt{y}.\ \mathtt{x}\ \mathtt{y}$を$\lambda \mathtt{w}.\ \mathtt{x}\ \mathtt{w}$に置き

換える（$\mathtt{w}$ は他の名前、例えば $\mathtt{v}$ などでも良い）。すると $[\mathtt{x} \mapsto \mathtt{y}\ \mathtt{z}](\lambda \mathtt{w}.\ \mathtt{x}\ \mathtt{w})$ となり、定義に従い、$\lambda \mathtt{w}.\ \mathtt{y}\ \mathtt{z}\ \mathtt{w}$ が得られる。また、$[\mathtt{x} \mapsto \mathtt{y}](\lambda \mathtt{x}.\ \mathtt{x})$ だと、まず $\lambda \mathtt{x}.\ \mathtt{x}$ を $\lambda \mathtt{w}.\ \mathtt{w}$ に置き換える（$\mathtt{w}$ は他の名前、例えば $\mathtt{v}$ などでも良い）。すると $[\mathtt{x} \mapsto \mathtt{y}](\lambda \mathtt{w}.\ \mathtt{w})$ となり、定義に従い、$\lambda \mathtt{w}.\ \mathtt{w}$ が得られる。

ラムダ抽象の束縛変数（とそれに対応する変数）の名前の付け替えを α 変換と呼ぶ。
（注意）ラムダ式 $\lambda \mathtt{x}.\ \mathtt{x}\ \mathtt{y}$ の束縛変数 $\mathtt{x}$ を $\mathtt{y}$ にして $\lambda \mathtt{y}.\ \mathtt{y}\ \mathtt{y}$ とするのは α 変換ではない（もともと自由変数だった $\mathtt{y}$ が束縛されてしまうので）。また、ラムダ式 $\lambda \mathtt{x}.\ \lambda \mathtt{y}.\ \mathtt{x}$ の束縛変数 $\mathtt{x}$ を $\mathtt{y}$ にして $\lambda \mathtt{y}.\ \lambda \mathtt{y}.\ \mathtt{y}$ とするのも α 変換ではない（内側のラムダ抽象 $\lambda \mathtt{y}.\ \mathtt{x}$ では $\mathtt{x}$ は自由変数だったのに $\lambda \mathtt{y}.\ \mathtt{y}$ では自由変数ではなくなっているので）。このように、自由変数が自由変数でなくなる場合は名前の付け替えをしてはいけない。

ラムダ式 $\mathtt{t}$ の自由変数 (free variable) の集合 $FV(\mathtt{t})$ は以下のように定義される。

$$
\begin{aligned}
&FV(\mathtt{x}) = \{\mathtt{x}\} \\
&FV(\lambda \mathtt{x}.\ \mathtt{t}_1) = FV(\mathtt{t}_1) \setminus \{\mathtt{x}\} \\
&FV(\mathtt{t}_1\ \mathtt{t}_2) = FV(\mathtt{t}_1) \cup FV(\mathtt{t}_2)
\end{aligned}
$$

（情報工学科出身者への注意 1）プログラミング言語論ではラムダ計算の β 変換（あるいは β 簡約, β reduction） $\xrightarrow[\beta]{}$ は、ラムダ式中に 2 カ所以上 redex がある場合に変換先が 2 つ以上あり得る形で定義したが、上記の意味論では計算の順序が一通りに限定されている。
（情報工学科出身者への注意 2）プログラミング言語論ではラムダ抽象に対する置換は 3 通りに場合分けして行ったが、ここでは、置換の前に置換対象のラムダ抽象の束縛変数の名前の付け替えを必要に応じて行うことにしたので、場合分けがなくなっている。

例 1. ラムダ式 $\lambda \mathtt{x}.\mathtt{x}\ \mathtt{y}$ の自由変数の集合は $\{\mathtt{y}\}$ である。計算過程は黒板で示す。

練習問題 1. ラムダ式 $\lambda \mathtt{x}.\mathtt{z}\ (\lambda \mathtt{y}.\mathtt{x}\ \mathtt{y})$ の自由変数の集合を求めよ。

例 2. ラムダ式 $(\lambda \mathtt{x}.\mathtt{x}\ \mathtt{y})\ (\lambda \mathtt{y}.\mathtt{y})$ を上記評価規則により正規形になるまで評価すると $(\lambda \mathtt{y}.\mathtt{y})\ \mathtt{y}$ になる。評価の過程は黒板で示す。

（注意）ラムダ式 $(\lambda \mathtt{y}.\mathtt{y})\ \mathtt{y}$ は正規形である。ラムダ式 $\mathtt{y}$ は値ではないので、E-APPABS 規則は適用できない。

練習問題 2. ラムダ式 $(\lambda \mathtt{x}.\mathtt{x}\ (\lambda \mathtt{y}.\mathtt{y}))\ (\lambda \mathtt{y}.\mathtt{y})$ を上記評価規則により正規形になるまで評価せよ。

上記では α 変換の定義は直感的な説明のみしかしていないが、α 変換（α 同値関係）は置換を用いて定義できる。

$$
\lambda \mathtt{x}.\mathtt{t} =_\alpha \lambda \mathtt{y}.[\mathtt{x} \mapsto \mathtt{y}]\mathtt{t} \qquad (\mathtt{y} \notin FV(\mathtt{t}))
$$

形式的には、まず置換を α 変換（あるいは α 同値関係）を用いずに（ただしラムダ抽象の場合は場合分けをして）定義し、その後 α 変換を定義すればよい。
（補足 1）言葉使いとして、変換（あるいは簡約）と同値関係と 2 つ使ったが、これらは方向があるものを変換（簡約）、ないものを同値関係と言っている。例えば、ラムダ式 $(\lambda \mathtt{x}.\mathtt{x})\ \mathtt{y}$ に β 変換を施すとラムダ式 $\mathtt{y}$ になるという言い方をし、また、ラムダ式 $(\lambda \mathtt{x}.\mathtt{x})\ \mathtt{y}$ とラムダ式 $\mathtt{y}$ は β 同値であるという言い方もする。

（補足 2）上記の評価規則によるラムダ式の評価の順番は一意であるが、置換を行う際に名前の付け替えが行われる場合、名前の選び方が自由であるので、束縛変数名が異なる、様々な（α 同値関係にある）ラムダ式が評価結果として得られうる。