情報数学 最終課題

72043913/t20391ks 澤田 開杜

概要

情報数学の授業で学んだラムダ計算について下記にまとめて記す。

テーマを選んだ動機

最近のプログラミング言語では、HaskellやScalaのような関数型言語でなくとも、関数が第一級オブジェクトとなっているような言語が多い。私もこの授業を受ける前までは、λ式というと"無名関数"・"関数オブジェクト"といったような認識でしかいなかった。しかし、学んでみると普通のプログラム言語のような表現力を持っており、奥が深く、とても興味を持ったためである。

ラムダ計算

ラムダ記法

ラムダ式とは、いわば名前の無い関数である。例えば$+$の演算子がすでに定義されているとして、引数を1つ取り、それに1を加える関数を定義するとなると、$add1(x) = x + 1$のように記述するだろう。しかし、関数の名前自体は本質的ではない。そこで、この関数を$\lambda x.\, x + 1$と表現することにする。"λ"という記号の直後に引数を記述し、"."の後に関数の本体を記述する。このように仮引数を指定し、式から直接関数を定義する操作を**λ抽象**という。このように定義したλ式を$add1(3)$のように関数を値に適用するには、$(\lambda x.\, x + 1)(3)$のように記述する。

カリー化

先程は$add1$という、引数が一つの関数をもとにラムダ式での形を考えた。ここで新しく$add(x, y) = x + y$という2つの引数を取る関数を考え、それをλ抽象した結果を考えるが、λ式では、一般に$\lambda(x, y).\, x + y$のような複数引数を受け取る形では記述しない。そのため、これをλ抽象するには**カリー化**という手法が大切になる。まずは普通の関数をカリー化するとはどういうことかを考える。関数addは2つの数を受け取って1つの数を返すため、型は$N \times N \to N$である。一言で説明すると、カリー化とはこの$N \times N \to N$のような関数を$N \to N \to N$のような形にすることである。カリー化したaddをadd'と表記すると、add'の例の場合、引数を1つ与えると**"引数を1つ受け取り、1つの値を返す関数(型は$N \to N$)"**を返す。つまり、$add'(1)$とすると先程の$add1$と同じ、引数を1つ受け取り、それに1を足した結果を返す関数を作ることができる。よって、addに対して$add(1)(7)$のように呼び出せば$1 + 7$を計算することができる。これをλ式では$\lambda x.\, (\lambda y.\, x + y)$と記述する。更に、λ式では曖昧性がなければカッコやλを省略することができるため、$\lambda x\, y.\, x + y$と記述するのが一般的である。

λ式の定義とβ変換