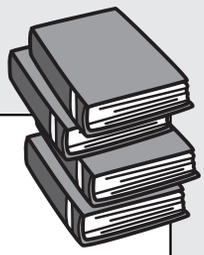

Daniel P. Friedman and David S. Wise : CONS Should not Evaluate Its Arguments

Automata, Languages and Programming: Third International Colloquium, edited by S. Michaelson and R. Milner, pp.257-284（1976）

本論文は遅延評価（Lazy Evaluation）の概念を提示した論文として，Henderson と Morris の論文（P. Henderson and J. Morris: A Lazy Evaluator, Proc. 3rd ACM Symposium on Principles of Programming Languages, pp.95-103, 1976）と並び，よく引用される．遅延評価を意味論のベースにする関数型言語研究のよりどころとなった論文の 1 つである．LISP を知っている人ならば，「cons はその実引数を評価すべきでない」という主張は直ちに理解できる．同意をするかどうかはともかくとして．

論文の着想は明快である．現代風に説明すると以下のようになる．まず，関数を基本表記とする小さな言語を考える．そこで使う語は，関数記号と変数である．数学では関数の定義や呼び出しを f（t1, …, tn）のように書くが，ここでは話を LISP につなげていくので，（f t1 … tn）のように書く．f が関数記号である．プログラムは，変数あるいは（f t1 … tn）である．ここで，t1,…,tn はプログラムである（帰納的に定義していることに注意）．関数記号には 2 つの種類がある．1 つは定義関数記号で，何らかの計算をする関数に名前を付けるのに用いる．つまり，定義関数記号はユーザが定義する関数，あるいはシステムに組み込まれた関数の名前である．それ以外の関数記号は構成子記号と呼ばれる．c を構成子記号とし，（c t1 … tn）の評価を考える．c が関数を定義する記号でないので，このプログラムは評価されても，（c t1'… tn'）のようになるだけで，c が先頭にくるプログラム表現であることは変わらない．ここで，t1,…,tn は評価されて，各々 t1',…,tn' になるとする．このようにプログラムの枠が変わらないので，構成子記号は，データを格納するコンテナを作るのに用いられる．

純粋 LISP では，cons という構成子記号が 1 つ備わっている．cons によって作られるコンテナを LISP では cons セルと呼ぶ．コンテナに入れるデータは，コンテナを作ってそのデータを入れるときには，わざわざ評価する必要がない．実際にデータが使われるのは，コンテナからデータが取り出されて計算に利用されるときである．そのときに，評価すればよい．これが，本論文の具体的な主張である．

この方法の利点は次の点にある．まず，余分な計算をしなくてすむ可能性がある．たとえば，（cons A B）で B は使われないかもしれない．もし使われないのならば，B の評価は無駄である．それどころか，もし，B の評価が停止しない計算を引き起こすならば，答えが出なくなってしまう．評価中に一部の計算で止まらなくても，意味のあるプログラムは数多くある．たとえば，論文で示される次の関数 terms は，列 1/1, 1/4, 1/9, … , $1/n^2$, … を計算する．

```
(terms n) ≡ (cons (reciprocal (square n))
                  (terms (add1 n)))
```

cons セルを作るときに，両方の実引数を評価すると停止しない．しかし，実引数を評価せずに，必要となったときに cons セルに格納されたデータを評価するならば，上の列のどの数も得ることができる．たとえば，上の列の 3 番目の数 1/9 は

```
(car (cdr (cdr (terms 1))))
```

を評価することで得られる．

本論文では，さらに，純粋 LISP のインタプリタを再定義することによって，これまでに述べた機能が容易に実現できること，再定義した純粋 LISP インタプリタはもとのインタプリタよりも強力であることを，理論的に示している．

本論文は構成子の実引数の遅延評価を論じている．実際，有意義な例の多くが，構成子の実引数の遅延評価である．しかしながら，この着想は構成子に限る必要はなく，すべての関数記号について当てはまる．後に開発された Haskell のような関数型言語は，一般的な枠組みで遅延評価を実現している．

最後に，Henderson らの論文との関連にふれる．両論文は LISP を例にとり遅延評価を相異なる枠組みを用いて理論的に解明した点で，同等に意義深い．なお，本論文は Henderson らの論文を引用し，独立な仕事であると述べている．

（平成 15 年 2 月 25 日受付）

井田哲雄／筑波大学電子・情報工学系
Tetsuo.Ida@acm.org