

関数型言語 Standard ML の プログラミング教育

山口 文彦 榎本 進
(東京理科大学)



✓ 関数型言語教育

東京理科大学工学部情報科学科では、学部2年生約130名を対象に、後期の約半年間をかけて、関数型言語 Standard ML のプログラミング教育を行っている。関数型言語は、関数呼び出しの組合せによって計算手順を表現することを基本的な考え方とするプログラミング言語である。この授業は、情報科学を学ぶ学生が身につけるべき教養としてのプログラミングを習得させることを目的としており、特に、手続き型以外の計算モデルに基づいた言語に触れることで、プログラミングに対して柔軟な考え方を身につけてもらうことを意図している。

本学科のプログラミング言語の実習は、1年次にC言語の基礎を行い、2年次以降から本格的な実習を始めている。2年次前期には、15週×1.5コマ×週2回でJava言語の実習を行う。ここまでの、手続き型言語の理解と習得を目標とする。2年次後期には、15週×1.5コマ×週2回でML言語の実習を行っている。また、実習と並行して、15週×1コマのML言語についての講義が選択科目として設定されている。選択科目ではあるが、対象となる学生のほぼ全員がこちらの講義も履修している。実習では、ML言語処理系としてSML/NJ^{☆1}システムを採用している。

関数型言語におけるプログラミングでは、定義しようとする関数ごとに、その働きを考える必要があるため、問題を機能ごとに分割するように思考が自然に誘導される¹⁾。さらに、それぞれの関数は、基本的には返戻値以外の出力を持たない。そのため、

単一の機能を持った部品(としての関数)を考えればよい、というように設計過程を単純化できる。このように考える訓練をすることは、問題を分析する能力の向上につながるものと考えられる。また、SML/NJでは、実際には実行時にコンパイルを行っているが、ユーザはコンパイルしていることをほとんど意識せず、インタプリタのように対話的に実行することができる。そのため、個々の関数に具体的な値を入れた場合の返戻値を観察することが比較的容易であり、学生が自発的にコードのテストをする際の敷居が低くなっていると考えられる。

✓ 実習内容

実習は以下のように進められる²⁾。

- 最初は、対話モードでプログラムを入力しながら、基礎となる変数や関数の定義・MLの基本的データである整数、実数、ブール、文字、文字列、組およびリスト・さらに、再帰関数の定義・適用を学ぶ。また、パターンを用いた関数呼び出しを学び、関数の再帰的定義を基底段階と帰納段階とに分け、パラメータの順序に基づいて定義することを学ぶ。
- エディタを用いたプログラミングを学び、関数等の定義をテキストファイルとして作成・保存し、MLシステムに読み込んで実行可能にする。また、高階関数^{☆2}とカリー化を用いたプログラミングを学び、関数の部分的具体化や高階関数上の演算子等を通して高階関数の理解を深める。
- リストの **reverse** 関数程度の関数を書かせる問題を50問ほど用意し、それらから各自に6~10

^{☆1} Standard ML of New Jersey, <http://www.smlnj.org>

^{☆2} 関数を引数や返戻値とするような関数を高階関数と呼ぶ。

●パターン

パターンを用いると if 文を用いなくても、リストやデータ構成子の構造によって処理を分けることができる。たとえば、リストの長さを求める関数は、入力が nil のときとコンスセルの場合に分けられるので、次のように書くことができる。

```
fun length [] = 0
| length (x::xs) = 1 + length xs
```

●カリー化

n 要素の組を引数とする関数を、 n 個の引数を 1 つずつ受け取る関数にする操作をカリー化と呼ぶ。たとえば、次の関数 plus は整数 2 つの組を受け取る関数だが、add は引数を 1 つずつ受け取る関数である。

```
fun plus(x,y) = x+y;
fun add x y = x+y;
```

それぞれの型は、plus が $(\text{int} * \text{int}) \rightarrow \text{int}$ 、add が $\text{int} \rightarrow (\text{int} \rightarrow \text{int})$ である（省略可能な括弧を明示している）。この add は 2 つの整数値を受け取るとそれらの和を返す関数だが、1 つだけ具体的な引数を与えると 1 引数の関数が得られる。たとえば、add 1 は (x) だけが具体化されて、 y を受け取って $1+y$ を返す関数となる。このように関数に引数を部分的に与えて関数を得ることを、部分的具体化と呼ぶ。部分的具体化によって、既存の関数から派生する関数を簡単に定義することができる。なお、次のように定義される関数 curry を用いると、2 要素の組を受け取る関数をカリー化することができる。

```
fun curry f x y = f(x,y);
```

たとえば、curry plus によって、add と同じ動作をする関数が得られる。

●reverse 関数

リストの要素を逆順に並べ替える関数。たとえば、リストを連結する演算子 @ を用いて、次のように定義できる。

```
fun reverse [] = []
| reverse (x::xs) = (reverse xs)@[x]
```

間ほどを割り当てて、実習で定義させる。また、データ型や文字列処理について学ぶ。特に、再帰的構造をもつデータ型では、そのデータを処理する再帰関数がデータ型の再帰的構造に基づいて定義できることを学ぶ。さらに、その応用としてソーティングのプログラムを学ぶ。

- 簡単な数値解析・ソート・文字処理・データ型の定義等の課題を実習する。
- スタックや二分木、キューなどを題材として、抽象データ型について学ぶ。さらに、いろいろなデータ型の要素を持つこれらのデータ型をファンクタと呼ばれる総称データ型で定義できることを学ぶ。
- 応用的な課題として、簡単なコンパイラまたはゲーム木探索のプログラミングを行う。これらの課題は年によって異なるものを課している。

基本的な使い方の説明に続いて、より進んだ概念

の説明に入る前に、簡単な課題を用いて反復的に訓練を行っている。また、応用的な課題を進めるうちに、学生が自発的に試行錯誤することによって、理解を深めることを意図している。

✓ 授業の特徴と評価

本授業の大きな特徴の 1 つは、通常の授業の 1.5 倍の長さの実習を水曜と金曜の週 2 回行っている点である。これは、学生に考える時間を与えられる点と、忘れないうちに作業の続きができる点で有効な時間割になっていると考えられる。学習意欲を向上させるために、簡単なものから段階的に課題を難しくしていき、最後には学生が自由に工夫できるような課題を設定している。

2002 年以降 3 年次後期に、任意の言語で階乗計算程度の簡単なプログラムを解答するペーパーテストを実施している。直前の演習で用いた言語は、年に



図-1 実習を行う教室の様子。160台のWindows端末が並ぶ。端末2台につき1つ、教員用端末や書画カメラの画面を映すモニタがある。

よって異なるが、JavaまたはC言語である。それにもかかわらず、毎年1～2割の学生がMLを選択している。学生がこのテストに解答する際には、どの言語であれば書きやすいかを考えて選択しているというよりも、自分の好んだ言語、得意な言語を選んでいるように思われる。MLが好まれるのは、いわゆるおまじないが少なく、プログラムを短く書くことが理由ではないだろうか。また、このテストでは再帰を用いると簡単に解答できるような出題をしており、CやJavaでも再帰を教えるはいるものの、学生にとって再帰的なプログラムを書くには、MLの方が慣れているという事情があるかもしれない。

2006年度からの4年間は、最後の応用的な課題として、毎年異なるゲームを題材に、ゲーム木探索のプログラミングを課した³⁾。学生同士が自由に対戦することができ、随時、Ratingによる順位を見ることができ、教員やTAもプログラムを作成して、この対戦に参加することで、学生のモチベーション向上に努めた。課題の成績はレポートによって採点され、対戦結果とは無関係である。しかし、強いプログラムを書く学生は、よいレポートを提出する傾向にある。なお、Rating上位10名がその後所属した研究室を調べてみたところ、過半数が計算機科学

をテーマとする研究室に配属されていたことが分かった。その中でもレポートの点数が良かった者は、ほとんどが計算機系研究室に進んでいる。また、レポート提出後にもRatingが更新されている形跡があった。指示待ちの受身人間が増加しているといわれる中、一部ではあっても、課題に興味を持って自発的に取り組んでいたことが伺えたのは、意味のあることではないだろうか。

参考文献

- 1) Abelson, H. and Sussman, G. J. : Structure and Interpretation of Computer Programs MIT Press (1996).
- 2) 児玉泰司 : ML演習—The Meta Language—, 東京理科大学, 学内資料 (1999).
- 3) 山口文彦, 榎本 進 : SMLによるゲーム木探索の演習のためのソフトウェア, 2010年夏のプログラミングシンポジウム報告集 (Sep. 1-3, 2010).

(平成22年7月31日受付)

山口文彦 (正会員) yamagu@is.noda.tus.ac.jp

昭和46年生。平成13年慶應義塾大学大学院理工学研究科計算機科学専攻後期博士課程単位取得退学。同年より平成19年まで東京理科大学理工学部情報科学科助手および助教。現在は東京理科大学大学院非常勤講師。博士(工学)。

榎本 進 (正会員) nomoto@is.noda.tus.ac.jp

昭和24年生。昭和53年東京工業大学大学院工学研究科電子物理工学専攻博士課程修了。工学博士。同年より東京理科大学理工学部情報科学科。現在は東京理科大学専任講師。非常勤講師。博士(工学)。

訂 正

本誌 51 卷 11 号 (2010 年 11 月号) の「プログラミング, 何をどう教えているか: 関数型言語 Standard ML のプログラミング教育」に一部誤りがありました. お詫びして訂正いたします.

p.1491 著者 榎本 進様のご略歴 (誤) 最終行: 非常勤講師. 博士 (工学). (正) 削除する

訂 正

本誌 51 卷 11 号 (2010 年 11 月号) の「プログラミング, 何をどう教えているか: 関数型言語 Standard ML のプログラミング教育」に一部誤りがありました. お詫びして訂正いたします.

p.1491 著者 榎本 進様のご略歴 (誤) nomoto@is.noda.tus.ac.jp (正) enomoto@is.noda.tus.ac.jp