『プログラム言語論』 第2回 演習 の略解

亀山

課題 2-1.

MiniC 処理系で，フィボナッチ関数 (ex11.c) に引数 n を与えて実行したときのスタックの深さの最大値を n の式で答えなさい。

答. fib(n) の計算に必要なスタックの深さの最大値を S(n) と置く。fib(n) は fib(n-1) を呼び、その計算が終わったら fib(n-2) を呼んでいる。よって、S(n) は S(n-1)+1 と S(n-2)+1 の最大値に等しい。

$$S(n) = max\{S(n-1)+1, S(n-2)+1\}$$

また、$S(1) = S(2) = 1$ である。(引数が 1,2 のとき、再帰呼び出しをせずに終了する。) これらから、$S(n) = n-1$ ($n > 1$ のとき) となる。

課題 2-2.

MiniC 処理系の「実行時のモード」は現時点で 0〜7 の 8 つである．ただし，0 と 3 は前回と同じ (動的束縛) であるので，今回の対象外であり，それ以外の 6 つのモード (1,2,4,5,6,7) は，すべて静的束縛である．各モードは，以下のバリエーションのいずれかである．

- 複数の引数を持つ関数において、引数を左から右に順番に計算 (評価) するか、右から左に計算するか。
- call-by-value か call-by-name か call-by-need か.

合計で 6 通りの処理方式があり、それが、モードの 1,2,4,5,6,7 のどれかに対応している。

そこで、各モードが、上記のどの組み合わせであるかを答えなさい。

答. 参考ファイル ex13.c,ex14.c 等を実行することにより、以下のことがわかる。

- 実行モード 0: 動的束縛、値呼び、左から右
- 実行モード 1: 静的束縛、値呼び、左から右
- 実行モード 2: 静的束縛、値呼び、右から左
- 実行モード 3: 動的束縛、値呼び、右から左
- 実行モード 4: 静的束縛、名前呼び
- 実行モード 5: 静的束縛、名前呼び
- 実行モード 6: 静的束縛、必要呼び
- 実行モード 7: 静的束縛、必要呼び

実行モード 4,5 と、実行モード 6,7 は、それぞれ引数を「左から右」に計算するか、その逆かが異なっているはずだが、どのようなプログラムでも見分けがつかない。なぜなら、名前呼びや必要呼びでは、実引数の評価は、関数適用の瞬間には発生せず、それらに対応する仮引数が関数本体で実際に評価されるときに、発生するからである。

課題 2-3. (発展課題; 興味のある人のみ)

いろいろなプログラム言語 (C 言語以外) が，(1) 動的束縛か静的束縛か，(2) 引数の評価順序は左から右か，その逆か，(3) call-by-value/call-by-name/call-by-need のどれか，について，調査せよ．ただし，この課題をやる

場合には,「文献にこう書いてあった」では駄目で、「実際このプログラムを動かして、こうなったからこうだ」と書きなさい (プログラムのソースコードをレポートに付けなさい)。

具体的な例は、ex13.c 等と同じ directory にある various.txt ファイルにいくつか書かれているので参考にするとよい．なお,「いろいろなプログラム言語 (C 言語以外)」の使い方や起動方法は自分で調べるものとする．(教員・TA は，すべてのプログラム言語の使用方法を知っているわけでない．皆さんしか知らないプログラム言語での実行例を提出するのは大歓迎である。)

答. これは、いろいろ試してみてほしい。OCaml は、(言語仕様としては不定の部分もあるが、coins マシンにある処理系では) 静的束縛、値呼び、右から左 という戦略である。Haskell は、静的束縛、必要呼びである。emacs-lisp は、動的束縛、値呼び、左から右である。