ソフトウェアサイエンス実験 S8 課題 8-1

200911434 青木大祐

平成 24 年 11 月 10 日

8.1.1 テストプログラムの実行

am.ml を実行した結果を以下に示す。

```
# type code = instr list
and instr =
    I_Ldi of int
  | I_Ldb of bool
  | I_Push
  | I_Extend
  | I_Search of int
  | I_Pushenv
  | I_Popenv
  | I_Mkclos of code
  | I_Apply
  | I_Test of code * code
  | I_Add
  | I_Sub
  | I_Mult
  | I_Div
  | I_Greater
  | I_Eq
  | I_Noteq
type am_value =
    AM_IntVal of int
  | AM_BoolVal of bool
  | AM_Closure of code * am_env
  | AM_Env of am_env
and am_stack = am_value list
and am_env = am_value list
val trans : am_value -> am_stack -> am_env -> code -> am_value = <fun>
val am_eval : code -> am_value = <fun>
val code0 : instr list = [I_Ldi 2; I_Push; I_Ldi 1; I_Add]
- : am_value = AM_IntVal 3
val code1 : instr list =
  [I_Pushenv; I_Ldb true; I_Extend; I_Search 0;
   I_Test ([I_Ldi 1], [I_Ldi 2]); I_Popenv]
- : am_value = AM_IntVal 1
val code2 : instr list =
  [I_Pushenv;
   I_Mkclos
    [I_Ldi 0; I_Push; I_Search 0; I_Eq;
     I_Test ([I_Ldi 0],
      [I_Pushenv; I_Ldi (-1); I_Push; I_Search 0; I_Add; I_Push; I_Search 1;
       I_Apply; I_Popenv; I_Push; I_Search 0; I_Add])];
   I_Extend; I_Pushenv; I_Ldi 3; I_Push; I_Search 0; I_Apply; I_Popenv;
   I_Popenv]
- : am_value = AM_IntVal 6
```

正しく計算できていることが分かる。

8.1.2 独自の命令

元々の *am.ml* に加えて、*Sub*, *Mult*, *Div*, *Greater*, *Noteq* の命令を実装した。変更点は以下のとおり。

```
...
  | I_Add                    (* Accumulatorにある値にスタックトップの値
                              * を加えて結果をAccumulator にいれる
                              *)
  | I_Sub                    (* Accumulatorにある値にスタックトップの値
                                 を引いて結果をAccumulatorにいれる *)
  | I_Mult                   (* Accumulatorにある値にスタックトップの値
                                 を掛けて結果をAccumulatorにいれる *)
  | I_Div                    (* Accumulatorにある値にスタックトップの値
                                 で割って結果をAccumulatorにいれる *)
  | I_Greater                (* Accumulatorにある値がスタックトップの値
                                 より大きいかどうかを比較して、結果をAccumulatorにいれる *)
  | I_Eq                     (* Accumulatorにある値とスタックトップの値
                              * が同じ整数であるかどうかをテストして、結
                              * 果をAccumulator にいれる
                              *)
  | I_Noteq                  (* Accumulatorにある値とスタックトップの値
                              * が違う整数であるかどうかをテストして、結
                              * 果をAccumulator にいれる
                              *)


...


(* ASEC machine の状態遷移(計算)*)
let rec trans (a:am_value) (s:am_stack) (e:am_env) (c:code) : am_value =
  let binop (a:am_value) (s:am_stack) (f:instr) : (am_value * am_stack) =
    begin
      match (f,      a,            s               ) with
  |   (I_Add,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_IntVal (n+m), s1)
  |   (I_Sub,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_IntVal (n-m), s1)
  |   (I_Mult,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_IntVal (n*m), s1)
  |   (I_Div,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_IntVal (n/m), s1)
  |   (I_Greater,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_BoolVal (n>m), s1)
  |   (I_Eq ,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_BoolVal(n=m), s1)
  |   (I_Noteq ,  AM_IntVal(n), AM_IntVal(m)::s1) -> (AM_BoolVal(n!=m), s1)
  | _ -> failwith "unexpected type of argument(s) for binary operation"
    end

...

  when List.mem i [I_Add; I_Sub; I_Mult; I_Div; I_Greater; I_Eq; I_Noteq]
  ->
  let (v,s1) = binop a s i
  in trans v s1 e c1
    | _ -> failwith "unknown instruction"

...
```

また、動作を確認するために以下の計算を行った。

```
let code3 =
  [I_Ldi 4; I_Push; I_Ldi 10; I_Sub];;

let _ = am_eval code3;;

let code4 =
  [I_Ldi 2; I_Push; I_Ldi 3; I_Mult];;

let _ = am_eval code4;;

let code5 =
  [I_Ldi 4; I_Push; I_Ldi 8; I_Div];;

let _ = am_eval code5;;

let code6 =
  [I_Ldi 4; I_Push; I_Ldi 3; I_Greater];;

let _ = am_eval code6;;

let code7 =
  [I_Ldi 4; I_Push; I_Ldi 1; I_Noteq];;

let _ = am_eval code7;;
```

実行結果は以下のとおり。

```
val code3 : instr list = [I_Ldi 4; I_Push; I_Ldi 10; I_Sub]
- : am_value = AM_IntVal 6
val code4 : instr list = [I_Ldi 2; I_Push; I_Ldi 3; I_Mult]
```

```
- : am_value = AM_IntVal 6
val code5 : instr list = [I_Ldi 4; I_Push; I_Ldi 8; I_Div]
- : am_value = AM_IntVal 2
val code6 : instr list = [I_Ldi 4; I_Push; I_Ldi 3; I_Greater]
- : am_value = AM_BoolVal false
```

正しく計算できていることが分かる。