Homework 7

SNU 4190.310, 2015 가을

Kwangkeun Yi

Due: 12/09(Wed), 24:00

이번 숙제의 목적은:

- M언어로 짜여진 프로그램이 무난히 실행될 수 있는지를 미리 확인해 주는 안전장치를 갖추어 본다.
- 우선 단순 타입 시스템(simple type system)을 장착해보고, 다음으로는 안전하면서 좀더 정교한 다형 타입 시스템(polymorphic type system)을 장착해서 편리하면서도 안전한 프로그래밍 환경을 완성해 본다.

Exercise 1 (40pts) "저지방 M"

수업시간에 구축해간 프로그래밍 언어에 약간의 간을 한 언어가 M 이다. TA가 제공하는 M 실행기의 틀 위에, 수업시간에 확인한 안전한 단순 타입 시스템(simple type system)을 설치하라.

즉, 첨부되는 M의 정의에서 타입 시스템의 미흡한 부분을 완성하고, 그것의 충실한 타입 유추기를 구현해서 M 실행기 앞단에 장착하도록 한다. 그러한 M 실행기는 항상 잘 도는 건강한 프로그램만 실행하게 된다. 그래서 건강해진 밥상, M 실행기를 즐길 수 있기를.

Bon appétit! Help yourself! 请吃好! □

Exercise 2 (60pts) "저지방 고단백 M"

M 실행기 위에, 위 숙제에서 구현한 단순 타입 시스템(simple type system)을 대신해서, 보다 정교한 let-다형 타입 시스템(let-polymorphic type system)을 장착하자. 그래서, 단순 타입 시스템이 받아들이는 프로그램은 물론이고, 다형 타

입의 함수(polymorphic function, 다양한 타입의 인자에 적용될 수 있는 함수)를 가지는 프로그램까지 받아들여 지도록.

예를들어, 아래와 같은 잘 도는 프로그램들이 단순 타입 시스템에서는 받아들여지지 않았으나, let-다형 타입 시스템에서는 받아들여지게 된다.

TA가 제공하는 M 실행기의 틀 위에 let-다형 타입 시스템을 장착하라.

```
(* example 1: polymorphic toys *)

   let val I = fn x => x
       val add = fn x => x.1 + x.1
       val const = fn n => 10
   in
     I I;
     add(1, true) + add(2, "snu 310 fall 2011");
     const 1 + const true + const "kwangkeun yi"
   end

(* example 2: polymorphism with imperatives *)

   let val f = fn x => malloc x
   in
      let val a = f 10
          val b = f "pl"
          val c = f true
      in
        a := !a + 1;
        b := "hw7";
        c := !c or false
      end
   end

(* example 3: polymorphic swap *)

   let val swap =
           fn order_pair =>
              if (order_pair.1) (order_pair.2)
               then (order_pair.2)
               else (order_pair.2.2, order_pair.2.1)
   in
     swap(fn pair => pair.1 + 1 = pair.2, (1,2));
     swap(fn pair => pair.1 or pair.2, (true, false))
   end

(* S K I combinators *)
```

```
let val I = fn x => x
    val K = fn x => fn y => x
    val S = fn x => fn y => fn z => (x z) (y z)
in
 S (K (S I)) (S (K K) I) 1 (fn x => x+1)
end
```

□