

Programming Languages and Types

Homework 12

Yi Dai

January 30, 2013

1 Simply-Typed λ -Calculus

1.1 Typing Derivation

Tell whether each of the following terms in the simply-typed λ -calculus with all the extensions introduced in the lecture is well-typed. If it is, give a typing derivation for it; if not, give the reason. For very large terms, you can name their sub-terms and type them individually.

1. **pred (succ false)**
2. $\lambda f : \mathbf{Nat} \rightarrow \mathbf{Nat} . \lambda n : \mathbf{Nat} . f \ (f \ (\mathbf{succ} \ n))$
3. **if (iszero (succ 0)) then true else 0**
4. $\{tru = \mathbf{succ} \ 0, tru = \mathbf{true}\} \text{ as } \{tru : \mathbf{Bool}, one : \mathbf{Nat}\}$
5. **let $b = \mathbf{false}$ in (iszero b)**
6. **let $p = (0, \mathbf{succ} \ 0)$ in (snd p , fst p)**
7. **case (inl 0) of inl $x \Rightarrow \mathbf{false}$ | inr $x \Rightarrow \mathbf{true}$**

8.

```
fix ( $\lambda f_{ise} : (\mathbf{Nat} \rightarrow \mathbf{Bool}) \rightarrow (\mathbf{Nat} \rightarrow \mathbf{Bool})$  .  
   $\lambda n : \mathbf{Nat}$  .  
    if (iszero  $n$ )  
      then true  
      else if (iszero (pred  $n$ ))  
        then false  
        else  $f_{ise}$  (pred (pred  $n$ )) )
```

1.2 Programming with Extensions

1. Complete the addition function *add* in the simply-typed λ -calculus extended with Peano numbers (**0** and **succ**) and fixed point operator **fix**.¹

```
fix ( $\lambda f_{add} : (\mathbf{Nat} \rightarrow \mathbf{Nat} \rightarrow \mathbf{Nat}) \rightarrow (\mathbf{Nat} \rightarrow \mathbf{Nat} \rightarrow \mathbf{Nat})$  . ?)
```

2 System- \mathcal{F}

2.1 Parametric Polymorphism

2.2 Typing Church-Encodings

¹During the exercise session, I gave the wrong type $(\mathbf{Nat} \rightarrow \mathbf{Nat} \rightarrow \mathbf{Nat})$ to the variable that is to be bound to the fixed point. Please refer to the exercise sheet **ex12.pdf**. In this homework exercise, I have given the type for *fadd*, to remind of the mistake I made.