HW 9 – Proof Systems

CS 421 – Spring 2011

Revision 1.0

Assigned Thursday, April 21, 2011
Due Thursday, April 28, in class
Extension 48 hours (20% penalty)
Total points 40

1 Change Log

1.0 Initial Release.

2 Overview

After completing this MP, you should have a better understanding of proof systems.

3 Collaboration

Collaboration is NOT allowed on this assignment.

4 Instructions

Fill in the blanks in the following proof trees. Write your answers in the blanks on the page. Submit in hard-copy at the beginning of class.

5 Problems (40 pts)

1. (10 pts) Using the definitions from slide 24 in lecture 23, simplify the expression "`fac 2`". Specifically, you are given these definitions:

```
let W = fun F -> fun f -> F (f f)
let Y = fun F -> (W F)(W F)
let FAC = fun f -> fun x -> if x=0 then 1 else x * f (x-1)
let fac = Y FAC
```

For this problem, use the usual delta and if rules. In addition, you may assume the old "let" rule (i.e. you do not have to transform these lets to function applications). You should type your answer on a separate sheet of paper (copying each line and editing it is the simplest way to do this problem anyway).

This simplification is quite long. Our solution has 33 steps. (In our solution, we always delay delta rules until they are necessary; if you do them earlier, you can save a few steps.) The main problem is just making sure you have copied definitions correctly and that the parentheses are correct. **Solution:** fac 2
$\Longrightarrow$(let) (Y FAC) 2
$\Longrightarrow$(let) ((fun F $\rightarrow$ (W F)(W F)) FAC) 2
$\Longrightarrow$(beta) ((W FAC) (W FAC)) 2
$\Longrightarrow$(let) (((fun F $\rightarrow$ fun f $\rightarrow$ F (f f)) FAC) (W FAC)) 2
$\Longrightarrow$(beta) ((fun f $\rightarrow$ FAC (f f)) (W FAC)) 2
$\Longrightarrow$(beta) (FAC ((W FAC)(W FAC))) 2
$\Longrightarrow$(let) ((fun f $\rightarrow$ fun x $\rightarrow$ if x=0 then 1 else x * f (x-1)) ((W FAC)(W FAC))) 2
$\Longrightarrow$(beta) (fun x $\rightarrow$ if x=0 then 1 else x * ((W FAC) (W FAC) (x-1))) 2
$\Longrightarrow$(beta) if 2=0 then 1 else 2 * ((W FAC) (W FAC) (2-1)))
$\Longrightarrow$(delta) if false then 1 else 2 * ((W FAC) (W FAC) (2-1)))
$\Longrightarrow$(if) 2 * ((W FAC) (W FAC) (2-1)))
$\Longrightarrow$(let) 2 * (((fun F $\rightarrow$ fun f $\rightarrow$ F (f f)) FAC) (W FAC) (2-1))
$\Longrightarrow$(beta) 2 * ((fun f $\rightarrow$ FAC (f f)) (W FAC) (2-1))
$\Longrightarrow$(beta) 2 * (FAC ((W FAC) (W FAC))) (2-1)
$\Longrightarrow$(let) 2 * ((fun f $\rightarrow$ fun x $\rightarrow$ if x=0 then 1 else x * f (x-1)) ((W FAC)(W FAC))) (2-1)
$\Longrightarrow$(beta) 2 * (fun x $\rightarrow$ if x=0 then 1 else x * ((W FAC) (W FAC) (x-1))) (2-1)
$\Longrightarrow$(beta) 2 * (if (2-1)=0 then 1 else (2-1) * ((W FAC) (W FAC) ((2-1)-1)))
$\Longrightarrow$(delta) 2 * (if 1=0 then 1 else (2-1) * ((W FAC) (W FAC) ((2-1)-1)))
$\Longrightarrow$(delta) 2 * (if false then 1 else (2-1) * ((W FAC) (W FAC) ((2-1)-1)))
$\Longrightarrow$(if) 2 * ((2-1) * ((W FAC) (W FAC) ((2-1)-1)))
$\Longrightarrow$(let) 2 * ((2-1) * ((fun F $\rightarrow$ fun f $\rightarrow$ F (f f)) FAC) (W FAC) ((2-1)-1))
$\Longrightarrow$(beta) 2 * ((2-1) * ((fun f $\rightarrow$ FAC (f f))) (W FAC) ((2-1)-1))
$\Longrightarrow$(beta) 2 * ((2-1) * (FAC ((W FAC) (W FAC))) ((2-1)-1))
$\Longrightarrow$(let) 2 * ((2-1) * ((fun f $\rightarrow$ fun x $\rightarrow$ if x=0 then 1 else x * f (x-1)) ((W FAC)(W FAC))) ((2-1)-1))
$\Longrightarrow$(beta) 2 * ((2-1) * (fun x $\rightarrow$ if x=0 then 1 else x * ((W FAC)(W FAC)) (x-1)) ((2-1)-1))
$\Longrightarrow$(beta) 2 * ((2-1) * (if ((2-1)-1)=0 then 1 else ((2-1)-1) * ((W FAC)(W FAC)) (((2-1)-1)-1)))
$\Rightarrow$(delta) 2 * ((2-1) * (if (1-1)=0 then 1 else ((2-1)-1) * ((W FAC)(W FAC)) (((2-1)-1)-1)))
$\Rightarrow$(delta) 2 * ((2-1) * (if 0=0 then 1 else ((2-1)-1) * ((W FAC)(W FAC)) (((2-1)-1)-1)))
$\Rightarrow$(delta) 2 * ((2-1) * (if true then 1 else ((2-1)-1) * ((W FAC)(W FAC)) (((2-1)-1)-1)))
$\Rightarrow$(if) 2 * ((2-1) * 1)
$\Rightarrow$(delta) 2 * (1 * 1)
$\Rightarrow$(delta) 2 * 1
$\Rightarrow$(delta) 2

2. (10 pts) Using the definitions from slide 25, simplify the expression "`(Y Zero) 1`". In this case, however, we make some changes. We want you to use the definitions of true, false, and if from slide 19. That is, you should treat "if" as an ordinary, user-defined function, and assume the built-in function "=" returns one of the expressions true or false. Thus, you'll be working from these definitions:

```
let true = fun x y -> x
let false = fun x y -> y
let if b x y = b x y
let W = fun F -> fun f -> F (f f)
let Y = fun F -> (W F)(W F)
let Zero = fun f -> fun x -> if (x=0) 0 (f (x-1))
```

Again, type these on a separate sheet of paper (or on the same sheet as problem one). For this one, our solution has 23 steps.

Solution: (Y Zero) 1
$\Longrightarrow$(let) ((fun F $\rightarrow$ (W F)(W F)) Zero) 1
$\Longrightarrow$(beta) ((W Zero) (W Zero)) 1
$\Longrightarrow$(let) (((fun F $\rightarrow$ fun f $\rightarrow$ F (f f)) Zero) (W Zero)) 1
$\Longrightarrow$(beta) ((fun f $\rightarrow$ Zero (f f)) (W Zero)) 1
$\Longrightarrow$(beta) (Zero ((W Zero) (W Zero))) 1
$\Longrightarrow$(let) (fun f $\rightarrow$ fun x $\rightarrow$ if (x=0) 0 (f (x-1))) ((W Zero) (W Zero))) 1
$\Longrightarrow$(beta) (fun x $\rightarrow$ if (x=0) 0 (((W Zero) (W Zero)) (x-1))) 1
$\Longrightarrow$(beta) if (1=0) 0 (((W Zero) (W Zero)) (1-1))
$\Longrightarrow$(delta) if true 0 (((W Zero) (W Zero)) (1-1))
$\Longrightarrow$(let) true 0 (((W Zero) (W Zero)) (1-1))
$\Longrightarrow$(let) (fun x y $\rightarrow$ y) 0 (((W Zero) (W Zero)) (1-1))
$\Longrightarrow$(beta) ((W Zero) (W Zero)) (1-1)
$\Longrightarrow$(let) (((fun F $\rightarrow$ fun f $\rightarrow$ F (f f)) Zero) (W Zero)) (1-1)
$\Longrightarrow$(beta) ((fun f $\rightarrow$ Zero (f f)) (W Zero)) (1-1)
$\Longrightarrow$(beta) (Zero ((W Zero) (W Zero))) (1-1)
$\Longrightarrow$(let) (fun f $\rightarrow$ fun x $\rightarrow$ if (x=0) 0 (f (x-1))) ((W Zero) (W Zero))) (1-1)
$\Longrightarrow$(beta) (fun x $\rightarrow$ if (x=0) 0 (((W Zero) (W Zero)) (x-1))) (1-1)
$\Longrightarrow$(beta) if ((1-1)=0) 0 (((W Zero) (W Zero)) ((1-1)-1))
$\Longrightarrow$(delta) if (0=0) 0 (((W Zero) (W Zero)) ((1-1)-1))
$\Longrightarrow$(delta) if true 0 (((W Zero) (W Zero)) ((1-1)-1))
$\Longrightarrow$(let) true 0 (((W Zero) (W Zero)) ((1-1)-1))
$\Longrightarrow$(let) (fun x y $\rightarrow$ x) 0 (((W Zero) (W Zero)) ((1-1)-1))
$\Longrightarrow$(beta) 0

3. (10 pts) Fill in the blanks in the derivation of the type judgment $\emptyset \vdash \texttt{fun x} \to (+\,\texttt{x})\ 1 : \texttt{int} \to \texttt{int}$.

$$\dfrac{\dfrac{}{\Gamma_f \vdash \texttt{f} : (\texttt{int} \to \texttt{int}) \to \texttt{int}} \qquad \dfrac{\dfrac{}{\Gamma_f \vdash + : \texttt{int} \to \texttt{int} \to \texttt{int}} \qquad \dfrac{}{\Gamma_f \vdash 1 : \texttt{int}}}{\Gamma_f \vdash (+\,1) : \texttt{int} \to \texttt{int}}}{\dfrac{\Gamma_f \vdash \texttt{f}\ (+\,1) : \texttt{int}}{\emptyset \vdash \texttt{fun f} \to \texttt{f}\ (+\,1) : ((\texttt{int} \to \texttt{int}) \to \texttt{int}) \to \texttt{int}}}$$

4. (10 pts) Fill in the proof tree for the type judgment $\emptyset \vdash \texttt{fun f} \to \texttt{f}\ (+\,1) : ((\texttt{int} \to \texttt{int}) \to \texttt{int}) \to \texttt{int}$.
   You can use the abbreviation Γ_f for $\emptyset[\texttt{f} : (\texttt{int} \to \texttt{int}) \to \texttt{int}]$.

$$\dfrac{\dfrac{}{\texttt{fun x} \to \texttt{x}\ +\ 1 \Downarrow \texttt{fun x} \to \texttt{x}\ +\ 1} \qquad \dfrac{}{3 \Downarrow 3} \qquad \dfrac{\dfrac{}{3 \Downarrow 3} \qquad \dfrac{}{1 \Downarrow 1}}{3\ +\ 1 \Downarrow 4}}{(\texttt{fun x} \to \texttt{x} + 1)\ 3 \Downarrow 4}$$