

# Errata, CS\_336/CS\_M36 (part 2): Interactive Theorem Proving, Lent Term 2005

Anton Setzer\*

May 4, 2005

## Coursework 1

- Question 3: Replace “ $(\lambda x.(\lambda y.y y)) \tilde{\Omega}$ ” by “ $(\lambda x.(\lambda y.y y)) (\tilde{\Omega} \tilde{\Omega})$ ”

## Coursework 2

- Question 3:  
The following is not a mistake in the coursework, but it looks neither, if one

– replaces

```
One :: Set
    = data ast
```

```
ast :: One
    = ast@_
```

by

```
Nil2 :: Set
    = data nil
```

```
nil :: Nil2
    = nil@_
```

– replaces

```
Tuple (A :: Set)
      (n :: N)
      :: Set
    = case n of
        (Z) -> One
        (S n') -> Cons A (Tuple A n')
```

by

---

\*Dept. of Computer Science, University of Wales Swansea, Singleton Park, Swansea SA2 8PP, UK. Email: [a.g.setzer@swan.ac.uk](mailto:a.g.setzer@swan.ac.uk)

```

Tuple (A :: Set)
      (n :: N)
      :: Set
      = case n of
          (Z) -> Nil2
          (S n') -> Cons A (Tuple A n')

```

## 2: The simply typed $\lambda$ -calculus and term rewriting

- Slide 2-28, last line: Replace “ $x \longrightarrow' y$  implies  $x \longrightarrow^* y$ ” by “ $x \longrightarrow^* y$  implies  $x \longrightarrow' y$ ”.
- Slide 2-31, last line: Replace “ $x \longleftrightarrow' y$  implies  $x \longleftrightarrow^* y$ ” by “ $x \longleftrightarrow^* y$  implies  $x \longleftrightarrow' y$ ”.
- Slide 2-51: The statement of the lemma can be strengthened. It holds already for any weakly normalising and confluent reduction system.
- Slide 2-51, line after **Proof**:  
Replace “by simply reducing  $a$ ” by “by simply reducing  $x$ ”.
- Slide 2-105: Replace in the second item twice “ $r \longrightarrow_\beta t$ ” and “ $s \longrightarrow_\beta t$ ” by “ $r \longrightarrow_\beta^* t$ ” and “ $s \longrightarrow_\beta^* t$ ”
- Slide 2-106: Replace everywhere on this slide  $\tilde{\Omega}$  by  $\Omega$ .
- Slide 2-130: Replace twice  $\lambda x : A.r$  by  $\lambda(x : A).r$ .
- Slide 2-140: Replace “use the first way” by “use the second way”.
- Slide 2-153: Replace

$$\begin{aligned}
 a2 &:: A \rightarrow A \\
 &= a2'
 \end{aligned}$$

by

$$\begin{aligned}
 a2' &:: A \rightarrow A \\
 &= a2
 \end{aligned}$$

- Slide 2-193 (distributed as 2h-1): Replace **twice**  $\lambda x^\sigma.t s$  by  $(\lambda x^\sigma.t) s$ .
- Slide 2-226: Replace  $\pi_i^3$  by  $\pi_i^2$ .
- Slide 2-227: Replace  $\pi_i^3$  by  $\pi_i^2$ ,  $\pi_0^3$  by  $\pi_0^2$  (twice),  $\pi_1^3$  by  $\pi_1^2$  (twice).
- Slide 2-228: Replace  $\pi_i^3$  by  $\pi_i^2$ ,  $\pi_2^3$  by  $\pi_2^2$  (twice).
- Slide 2-237: Replace

$$\begin{aligned}
 ab &:: AB \\
 &= \text{struct}\{a = a'; b = b'\} :: AB
 \end{aligned}$$

by

$$\begin{aligned}
 ab &:: AB \\
 &= \text{struct}\{a = a'; b = b'\}
 \end{aligned}$$

(i.e. omit second occurrence of  $:: AB$ ).

- Slide 2-264: Replace in the right diagram  $a.b$  by  $a.c$  and  $c.d$  by  $b.d$ .
- Slide 2-272: The notation

`data Person = person (g :: Gender)(n :: Name)`

hasn't been introduced yet. It is essentially equivalent to

`Person :: Set = data person (g :: Gender)(n :: Name)`

(more details will be discussed later). So replace it by that definition.

- Slide 2-303: Replace “is the curried form of  $f$ ” by “is the uncurried form of  $f$ ”  
(applies to  $\lambda x^{\text{int} \times \text{int}}.f \pi_0(x) \pi_1(x) : (\text{int} \times \text{int}) \rightarrow \text{int}$ ).

### 3: The logical framework

- Slide 3-15: Replace

$$x_1 : A_1, \dots, x_n : A_n \Rightarrow A = B$$

by

$$x_1 : A_1, \dots, x_n : A_n \Rightarrow A = B : \text{Set}$$

- Slide 3-29: Replace “without the logical framework” by “without the full logical framework”.
- Slide 3-40: Replace “`nil : N`” by “`nil : NatList`”.