Homework 2

Problem 1. Find λ terms representing the logical or and not functions.

Solution. $\mathbf{or} \equiv \lambda xy.x\mathbf{t}y$, for

$$\begin{aligned}
\mathbf{or}\,\mathbf{t}\,\mathbf{t} &\twoheadrightarrow_\beta \mathbf{t}\,\mathbf{t}\,\mathbf{t} \twoheadrightarrow_\beta \mathbf{t},\\
\mathbf{or}\,\mathbf{t}\,\mathbf{f} &\twoheadrightarrow_\beta \mathbf{t}\,\mathbf{t}\,\mathbf{f} \twoheadrightarrow_\beta \mathbf{t},\\
\mathbf{or}\,\mathbf{f}\,\mathbf{t} &\twoheadrightarrow_\beta \mathbf{f}\,\mathbf{t}\,\mathbf{t} \twoheadrightarrow_\beta \mathbf{t},\\
\mathbf{or}\,\mathbf{f}\,\mathbf{f} &\twoheadrightarrow_\beta \mathbf{f}\,\mathbf{t}\,\mathbf{f} \twoheadrightarrow_\beta \mathbf{f}.
\end{aligned}$$

$\mathbf{not} \equiv \lambda x.x\mathbf{f}\mathbf{t}$, for

$$\begin{aligned}
\mathbf{not}\,\mathbf{t} &\twoheadrightarrow_\beta \mathbf{t}\,\mathbf{f}\,\mathbf{t} \twoheadrightarrow_\beta \mathbf{f},\\
\mathbf{not}\,\mathbf{f} &\twoheadrightarrow_\beta \mathbf{f}\,\mathbf{f}\,\mathbf{t} \twoheadrightarrow_\beta \mathbf{t}.
\end{aligned}$$

Problem 2. Prove that

(a) $\mathbf{add}\,\overline{m}\,\overline{n} \twoheadrightarrow_\beta \overline{m+n}$.

(b) $\mathbf{mult}\,\overline{m}\,\overline{n} \twoheadrightarrow_\beta \overline{m \cdot n}$.

Solution. (a) We have

$$\begin{aligned}
\mathbf{add}\,\overline{m}\,\overline{n} &\equiv (\lambda nmfx.nf(mfx))\overline{m}\,\overline{n}\\
&\twoheadrightarrow_\beta \lambda fx.\overline{m}f(\overline{n}fx)\\
&\twoheadrightarrow_\beta \lambda fx.\overline{m}f(f^n x)\\
&\twoheadrightarrow_\beta \lambda fx.f^m(f^n x)\\
&\twoheadrightarrow_\beta \lambda fx.f^{m+n}x\\
&\equiv \overline{m+n}.
\end{aligned}$$

(b) We have

$$\begin{aligned}
\mathbf{mult}\,\overline{m}\,\overline{n} &\equiv (\lambda nmf.n(mf))\overline{m}\,\overline{n}\\
&\twoheadrightarrow_\beta \lambda f.\overline{m}(\overline{n}f)\\
&\twoheadrightarrow_\beta \lambda f.\overline{m}(\lambda y.f^n y)^m\\
&\twoheadrightarrow_\beta \lambda fx.(\lambda y.f^n y)^m x\\
&\twoheadrightarrow_\beta \lambda fx.f^{m\cdot n}x\\
&\equiv \overline{m \cdot n}.
\end{aligned}$$

Problem 3. Compute the β-normal forms of the following terms. Are they strongly normalizable?

(a) $(\lambda xy.yx)((\lambda x.xx)(\lambda x.xx))(\lambda xy.y)$.

(b) $(\lambda xy.yx)(\mathbf{kk})(\lambda x.xx)$.

Solution. (a) We have

$$\begin{aligned}(\lambda xy.yx)((\lambda x.xx)(\lambda x.xx))(\lambda xy.y) &\twoheadrightarrow_\beta (\lambda xy.y)((\lambda x.xx)(\lambda x.xx))\\ &\twoheadrightarrow_\beta \lambda y.y.\end{aligned}$$

But it is not strongly normalizable, since the term $(\lambda x.xx)(\lambda x.xx)$ has no β-normal form.

(b) We have

$$\begin{aligned}(\lambda xy.yx)(\mathbf{kk})(\lambda x.xx) &\twoheadrightarrow_\beta (\lambda x.xx)(\mathbf{kk})\\ &\twoheadrightarrow_\beta (\mathbf{kk})(\mathbf{kk})\\ &\twoheadrightarrow_\beta \mathbf{k}.\end{aligned}$$

And it is strongly normalizable, since any means of reduction will eventually lead to the β-normal form $\mathbf{k}$.

Problem 4. Find a representation of the following functions on integers

(a) $f(n) = \begin{cases} \text{true} & n \text{ is even,} \\ \text{false} & n \text{ is odd.} \end{cases}$

(b) $\exp(n, m) = n^m$.

(c) $\text{pred}(n) = \begin{cases} 0 & \text{if } n = 0, \\ n-1 & \text{otherwise.} \end{cases}$ (Hard)

Solution. (a) Let **xor** be the logical exclusive or function,

$$\mathbf{xor} \equiv \lambda xy.x\,(\mathbf{not}\,y)\,y.$$

Then we have a recursive definition of f as follows,

$$f(n) = \begin{cases} \text{true} & \text{if } n = 0, \\ f(n-1)\,\mathbf{xor}\,\text{true} & \text{otherwise.} \end{cases}$$

So, we have

$$\begin{aligned} f := (\lambda g.\lambda n.\mathbf{ite}\,(\mathbf{iszero}\,n)\,\mathbf{f} \\ (\mathbf{xor}\,\mathbf{t}\,(g(\mathbf{pred}\,n))))\,f \end{aligned}$$

By using the Y combinator, we have

$$\begin{aligned} f := \mathbf{y}(\lambda g.\lambda n.\mathbf{ite}\,(\mathbf{iszero}\,n)\mathbf{t} \\ (\mathbf{xor}\,\mathbf{t}\,(g(\mathbf{pred}\,n)))). \end{aligned}$$

(b) A resucsive definition of the exponential function is as follows,

$$\exp(n, m) = \begin{cases} 1 & \text{if } m = 0, \\ n \cdot \exp(n, m-1) & \text{otherwise.} \end{cases}$$

So, we have

$$\begin{aligned} \mathbf{exp} := (\lambda f.\lambda nm.\mathbf{ite}\,(\mathbf{iszero}\,m)\,\overline{1} \\ (\mathbf{mult}\,n\,(f\,n\,(\mathbf{pred}\,m))))\,\mathbf{exp} \end{aligned}$$

By using the Y combinator, we have

$$\begin{aligned} \mathbf{exp} := \mathbf{y}(\lambda f.\lambda nm.\mathbf{ite}\,(\mathbf{iszero}\,m)\,\overline{1} \\ (\mathbf{mult}\,n\,(f\,n\,(\mathbf{pred}\,m)))) \end{aligned}$$

(c) The predecessor function is defined as follows,

$$\mathbf{pred} := \lambda n.\lambda f.\lambda x.n(\lambda g.\lambda h.h(gf))(\lambda u.x)(\lambda u.u).$$

We can verify that by the following computation,

$$\begin{aligned} \mathbf{pred}\,\overline{m} &\twoheadrightarrow_\beta \lambda f.\lambda x.\overline{m}(\lambda g.\lambda h.h(gf))(\lambda u.x)(\lambda u.u) \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.(\lambda g.\lambda h.h(gf))^m(\lambda u.x)(\lambda u.u) \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.(\lambda g.\lambda h.h(gf))^{m-1}(\lambda h.h((\lambda u.x)f))(\lambda u.u) \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.(\lambda g.\lambda h.h(gf))^{m-1}(\lambda h.hx)(\lambda u.u) \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.(\lambda g.\lambda h.h(gf))^{m-2}(\lambda h.h((\lambda h'.h'x)f))(\lambda u.u) \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.(\lambda g.\lambda h.h(gf))^{m-2}(\lambda h.hfx)(\lambda u.u) \\ &\twoheadrightarrow_\beta \cdots \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.(\lambda h.hf^{m-1}x)(\lambda u.u) \\ &\twoheadrightarrow_\beta \lambda f.\lambda x.f^{m-1}x \\ &\equiv \overline{m-1}. \end{aligned}$$

Problem 5. Suppose two binary relations $\to_1$ and $\to_2$ *commute*, that is, $s \to_1 t_1$ and $s \to_2 t_2$ implies that there exists t such that $t_1 \to_2 t$ and $t_2 \to_1 t$. Let $\to_{12}$ be the union of $\to_1$ and $\to_2$. Prove that if $\to_1$ and $\to_2$ satisfy the diamond property, then so is $\twoheadrightarrow_{12}$.

Solution. Let's define an auxilliary function $\mathbf{edge}(s, t)$ to indicate the particular type of relation that leads from s to t, as follows,

$$\mathbf{edge}(s, t) := \begin{cases} 1 & \text{if } s \to_1 t, \\ 2 & \text{if } s \to_2 t. \end{cases}$$

Consider the fundamental example that some states s, u, v satisfy $s \to_{12} u$ and $s \to_{12} v$.

1. If $\mathbf{edge}(s, u) = 1$ and $\mathbf{edge}(s, v) = 1$, then because of the diamond property of $\to_1$, we can find a state t such that $u \to_1 x$ and $v \to_1 t$.

2. If $\mathbf{edge}(s, u) = 2$ and $\mathbf{edge}(s, v) = 2$, then because of the diamond property of $\to_2$, we can find a state t such that $u \to_2 t$ and $v \to_2 t$.

3. If $\mathbf{edge}(s, u) = 1$ and $\mathbf{edge}(s, v) = 2$, then by the commutativity of $\to_1$ and $\to_2$, we can find a state t such that $u \to_2 t$ and $v \to_1 t$.

4. If $\mathbf{edge}(s, u) = 2$ and $\mathbf{edge}(s, v) = 1$, then by the commutativity of $\to_1$ and $\to_2$, we can find a state t such that $u \to_1 t$ and $v \to_2 t$.

So a state t such that $u \to_{12} t$ and $v \to_{12} t$ always exists, which implies that $\to_{12}$ sastisfies the diamond property. As a result, $\twoheadrightarrow_{12}$ also satisfies the diamond property.

Problem 6. (Optional) Write an algorithm computing the factorial function in Python without using explicit recursion. Sample codes are provided in lambda.py. Note that the use of parenthesis in Python for function application is different from the mathematical way. For example, the term xyz used in classes as an abbreviation for $((xy)z)$ should be written as $x(y)(z)$ in Python in order to be consistent with the Python function call convention.