## CIS 705 — Programming Languages — Spring 2009

# Assignment 6

### Due by 2:30 p.m. on Thursday, May 7

The context for this assignment is Chapters 11–12 of *TAPL*.

### Reformulating the Simply Typed Lambda Calculus with Unit and Products

If $f$ is a function and $x, y$ are elements of our universe, we define the function $f[x \mapsto y]$ from $\mathsf{dom}(f) \cup \{x\}$ to $\mathsf{ran}(f) \cup \{y\}$ by, for all $z \in \mathsf{dom}(f) \cup \{x\}$,

$$f[x \mapsto y](z) = \begin{cases} y, & \text{if } z = x, \\ f(z), & \text{if } z \neq x. \end{cases}$$

If $f$ is a function and $X$ is a set, then $f/X$ is the function from $\mathsf{dom}(f) \setminus X$ to $\mathsf{ran}(f)$ such that, for all $z \in \mathsf{dom}(f) \setminus X$, $(f/X)(z) = f(z)$.

We reformulate the syntax of the simply typed lambda calculus with unit and products as follows, where variables $x$ are as usual. Our *types* are defined by:

| | |
|---|---|
| $T ::=$ | types: |
| $\mathsf{Unit}$ | unit type |
| $T \times T$ | product type |
| $T \to T$ | function type |

As usual, $\to$ associates to the right, and $\times$ has higher precedence than $\to$. Our *terms* are defined by:

| | |
|---|---|
| $t ::=$ | terms: |
| $\mathsf{unit}$ | unit constant |
| $(t, t)$ | pair |
| $\mathsf{fst}\, t$ | first projection |
| $\mathsf{snd}\, t$ | second projection |
| $x$ | variable |
| $\lambda x : T.\, t$ | abstraction |
| $t\, t$ | application |

And our *values* are defined by:

| | |
|---|---|
| $v ::=$ | values: |

| | |
|---|---|
| $\mathsf{unit}$ | unit constant |
| $(v, v)$ | pair value |
| $\lambda x : T.\, t$ | abstraction value |

As usual, application associates to the left and abstractions extend as far as possible. And $\mathsf{fst}$ and $\mathsf{snd}$ have higher precedence than application.

In contrast to TAPL's approach, we do *not* identify abstractions up to the renaming of bound variables, so that, e.g., $\lambda x : \mathsf{Unit}.\, x = \lambda y : \mathsf{Unit}.\, y$ iff $x = y$. The *free variables* of a term $t$ ($\mathsf{FV}(t)$) is defined recursively as follows:

$$\begin{aligned}
\mathsf{FV}(\mathsf{unit}) &= \emptyset, \\
\mathsf{FV}((t_1, t_2)) &= \mathsf{FV}(t_1) \cup \mathsf{FV}(t_2), \\
\mathsf{FV}(\mathsf{fst}\, t) &= \mathsf{FV}(t), \\
\mathsf{FV}(\mathsf{snd}\, t) &= \mathsf{FV}(t), \\
\mathsf{FV}(x) &= \{x\}, \\
\mathsf{FV}(\lambda x : T.\, t) &= \mathsf{FV}(t) \setminus \{x\}, \\
\mathsf{FV}(t_1\, t_2) &= \mathsf{FV}(t_1) \cup \mathsf{FV}(t_2).
\end{aligned}$$

A term is *closed* iff it has no free variables; otherwise it is *open*.

A *simultaneous substitution* (or just *substitution*) $\sigma$ is a function such that $\mathsf{dom}(\sigma)$ is a finite subset of the variables, and $\mathsf{ran}(\sigma)$ is a subset of the closed values. The result of *applying* a substitution $\sigma$ *to* a term $t$ ($t\,\sigma$) is defined recursively by:

$$\begin{aligned}
\mathsf{unit}\,\sigma &= \mathsf{unit}, \\
(t_1, t_2)\,\sigma &= (t_1\,\sigma, t_2\,\sigma), \\
(\mathsf{fst}\, t)\,\sigma &= \mathsf{fst}(t\,\sigma), \\
(\mathsf{snd}\, t)\,\sigma &= \mathsf{snd}(t\,\sigma), \\
y\,\sigma &= \begin{cases} \sigma(y), & \text{if } y \in \mathsf{dom}(\sigma), \\ y, & \text{if } y \notin \mathsf{dom}(\sigma), \end{cases} \\
(\lambda x : T.\, t)\,\sigma &= \lambda x : T.\, t\,(\sigma / \{x\}), \\
(t_1\, t_2)\,\sigma &= (t_1\,\sigma)(t_2\,\sigma).
\end{aligned}$$

The *evaluation relation* $\boxed{t \rightarrow t'}$ between *closed* terms is defined inductively by:

$$\frac{t_1 \rightarrow t_1'}{(t_1, t_2) \rightarrow (t_1', t_2)} \qquad \text{(E-Pair1)}$$

$$\frac{t_2 \rightarrow t_2'}{(v_1, t_2) \rightarrow (v_1, t_2')} \qquad \text{(E-Pair2)}$$

$$\frac{t \to t'}{\mathsf{fst}\, t \to \mathsf{fst}\, t'} \qquad \text{(E-Fst)}$$

$$\mathsf{fst}(v_1, v_2) \to v_1 \qquad \text{(E-FstVal)}$$

$$\frac{t \to t'}{\mathsf{snd}\, t \to \mathsf{snd}\, t'} \qquad \text{(E-Snd)}$$

$$\mathsf{snd}(v_1, v_2) \to v_2 \qquad \text{(E-SndVal)}$$

$$\frac{t_1 \to t_1'}{t_1\, t_2 \to t_1'\, t_2} \qquad \text{(E-App1)}$$

$$\frac{t_2 \to t_2'}{v_1\, t_2 \to v_1\, t_2'} \qquad \text{(E-App2)}$$

$$(\lambda x : T.\, t)v \to t\,\{(x, v)\} \qquad \text{(E-AppAbs)}$$

So, in all but the last rule, $t$, $t'$, $t_1$, $t_1'$, $t_2$, $t_2'$, $v$, $v_1$ and $v_2$ are *closed*, whereas, in (E-AppAbs), $v$ is closed but $\mathsf{FV}(t) \subseteq \{x\}$. A closed term $t$ is a *normal form* iff there is no closed term $t'$ such that $t \to t'$. A closed term $t$ is *stuck* iff $t$ is a normal form but $t$ is not a value. A closed term $t$ *converges* iff there is a closed value $v$ such that $t \to^* v$; otherwise, it *diverges*.

An easy induction on the evaluation relation suffices to show that **evaluation is deterministic**: for all closed terms $t$, $t'$ and $t''$, if $t \to t'$ and $t \to t''$, then $t' = t''$.

A *typing context* (or just *context*) $\Gamma$ is a function such that $\mathsf{dom}(\Gamma)$ is a finite subset of the variables, and $\mathsf{ran}(\Gamma)$ is a subset of the types. The *typing relation* $\boxed{\Gamma \vdash t : T}$ between typing contexts, terms and types is defined inductively by:

$$\Gamma \vdash \mathsf{unit} : \mathsf{Unit} \qquad \text{(T-Unit)}$$

$$\frac{\Gamma \vdash t_1 : T_1 \qquad \Gamma \vdash t_2 : T_2}{\Gamma \vdash (t_1, t_2) : T_1 \times T_2} \qquad \text{(T-Pair)}$$

$$\frac{\Gamma \vdash t : T_1 \times T_2}{\Gamma \vdash \mathsf{fst}\, t : T_1} \qquad \text{(T-Fst)}$$

$$\frac{\Gamma \vdash t : T_1 \times T_2}{\Gamma \vdash \mathsf{snd}\, t : T_2} \qquad \text{(T-Snd)}$$

$$\frac{(x, T) \in \Gamma}{\Gamma \vdash x : T} \qquad \text{(T-Var)}$$

$$\frac{\Gamma[x \mapsto T_1] \vdash t : T_2}{\Gamma \vdash \lambda x : T_1.\, t : T_1 \to T_2} \qquad \text{(T-Abs)}$$

$$\frac{\Gamma \vdash t_1 : T_1 \to T_2 \qquad \Gamma \vdash t_2 : T_1}{\Gamma \vdash t_1\, t_2 : T_2} \qquad \text{(T-App)}$$

We say that a closed term $t$ is *well-typed* iff $\emptyset \vdash t : T$ for some type $T$.

An easy induction on the typing relation suffices to show that, for all contexts $\Gamma$, terms $t$ and types $T$, if $\Gamma \vdash t : T$, then $\mathsf{FV}(t) \subseteq \mathsf{dom}(\Gamma)$. Another induction on the typing relation shows the **weakening lemma**: for all contexts $\Gamma$ and $\Gamma'$, terms $t$ and types $T$, if $\Gamma \vdash t : T$ and $\Gamma \subseteq \Gamma'$, then $\Gamma' \vdash t : T$. And another induction on the typing relation suffices to show the **uniqueness of typing**: for all contexts $\Gamma$, terms $t$ and types $T$ and $T'$, if $\Gamma \vdash t : T$ and $\Gamma \vdash t : T'$, then $T = T'$.

The **canonical forms lemma** holds:

- If $v$ is a value and $\emptyset \vdash v : \mathsf{Unit}$, then $v = \mathsf{unit}$.
- For all types $T_1$ and $T_2$, if $v$ is a value and $\emptyset \vdash v : T_1 \times T_2$, then $v = (v_1, v_2)$, where $\emptyset \vdash v_1 : T_1$ and $\emptyset \vdash v_2 : T_2$.
- For all types $T_1$ and $T_2$, if $v$ is a value and $\emptyset \vdash v : T_1 \to T_2$, then $v = \lambda x : T_1.\, t$ for a variable $x$ and term $t$ such that $\{(x, T_1)\} \vdash t : T_2$.

The **progress theorem** holds: for all closed terms $t$, if $t$ is well-typed, then $t$ is not stuck. And the **preservation theorem** holds: for all closed terms $t$ and $t'$ and types $T$, if $\emptyset \vdash t : T$ and $t \to t'$, then $\emptyset \vdash t' : T$.

Define a function $R$ from types to sets of terms, by recursion (where we write $R(T)$ as $R_T$):

- $R_{\mathsf{Unit}} = \{\, t \mid \emptyset \vdash t : \mathsf{Unit}$ and $t$ converges $\}$;
- $R_{T_1 \times T_2} = \{\, t \mid \emptyset \vdash t : T_1 \times T_2$ and $t$ converges and $\mathsf{fst}\, t \in R_{T_1}$ and $\mathsf{snd}\, t \in R_{T_2} \,\}$;
- $R_{T_1 \to T_2} = \{\, t \mid \emptyset \vdash t : T_1 \to T_2$ and $t$ converges and, for all terms $s$, if $s \in R_{T_1}$, then $t\, s \in R_{T_2} \,\}$.

Note that, for all types $T$, $R_T \subseteq \{\, t \mid \emptyset \vdash t : T$ and $t$ converges$\}$. We often write $R_T(t)$ instead of $t \in R_T$.

**Exercise 1 (10 Points)**

Prove that, for all terms $t$ and substitutions $\sigma$, if $\mathsf{FV}(t) \cap \mathsf{dom}(\sigma) = \emptyset$, then $t\,\sigma = t$.

**Exercise 2 (15 Points)**

Prove that, for all terms $t$ and substitutions $\sigma$, $\mathsf{FV}(t\,\sigma) = \mathsf{FV}(t) \setminus \mathsf{dom}(\sigma)$. (This result is needed to see that the rule (E-AppAbs) of the definition of $\to$ is valid.)

**Exercise 3 (15 Points)**

Prove that, for all contexts $\Gamma$, terms $t$, types $T$ and substitutions $\sigma$, if $\Gamma \vdash t : T$ and, for all $x \in \mathsf{dom}(\Gamma) \cap \mathsf{dom}(\sigma)$, $\emptyset \vdash \sigma(x) : \Gamma(x)$, then $\Gamma/\mathsf{dom}(\sigma) \vdash t\,\sigma : T$.

**Exercise 4 (25 Points)**

Prove that, for all types $T$, for all closed terms $t$ and $t'$, if $\emptyset \vdash t : T$ and $t \rightarrow t'$, then $R_T(t)$ iff $R_T(t')$.

**Exercise 5 (30 Points)**

Prove that, for all contexts $\Gamma$, terms $t$, types $T$ and substitutions $\sigma$, if $\Gamma \vdash t : T$, $\mathsf{dom}(\Gamma) \subseteq \mathsf{dom}(\sigma)$, and for all $x \in \mathsf{dom}(\Gamma)$, $R_{\Gamma(x)}(\sigma(x))$, then $R_T(t\,\sigma)$.

**Exercise 6 (5 Points)**

Prove that all closed, well-typed terms converge.