CS 221/321	**Programming Languages**	**Homework Solution 4**
Fall 2009		**Due Nov 12, 2009**

1. [75 points] We can augment the MinML language by adding pairs (binary Cartesian products). Concretely, this amounts to adding three new expression forms to the abstract syntax, as shown here:

$$e ::= \ldots \mid (e,\, e) \mid \texttt{fst}(e) \mid \texttt{snd}(e)$$

The basic pair expression has the form $(e_1,\, e_2)$, where e_1 and e_2 are arbitrary expressions. Its value is a pair made up of the values of e_1 and e_2. The expression $\texttt{fst}(e)$ projects out the first component of the pair denoted by e, while $\texttt{snd}(e)$ yields the second component. Thus if $v = (2, \texttt{true})$, then $\texttt{fst}(v) = 2$ and $\texttt{snd}(v) = \texttt{true}$. Note that the first and second components of a pair can have different types, and also that those types can be arbitrary; a pair can have primitive values, functions, or pairs as components.

The definition of a value is also extended to include pair values:

$$v ::= \ldots \mid (v,\, v)$$

i.e., a pair of values is a value.

The type expressions are correspondingly extended with a product form:

$$\tau ::= \ldots \mid \tau * \tau$$

As with the function arror operator, the product operator for types is writen using infix notation.

(a) [10 points]. Add new typing rules for the three new expression forms (note that intuitively, a value like $(2, \texttt{true})$ has the product type $\texttt{int} * \texttt{bool}$).

Solution:

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (e_1, e_2) : \tau_1 * \tau_2} \quad \text{(P1)}$$

$$\frac{\Gamma \vdash e : \tau_1 * \tau_2}{\Gamma \vdash \texttt{fst}(e) : \tau_1} \quad \text{(P2)} \qquad\qquad \frac{\Gamma \vdash e : \tau_1 * \tau_2}{\Gamma \vdash \texttt{snd}(e) : \tau_2} \quad \text{(P3)}$$

(b) [15 points]. Add new small-step evaluation rules for the transition relation $\mapsto$ to cover the new expression forms. Evaluation of a pair expression should be *left-to-right*, as it is for the arguments of `plus` and `apply`. [Hint: there will be only 6 new rules, two of which will be instructions.]

Solution: Note that in rules (PE1) and (PE2) the pair must be fully evaluated before the projections can be evaluated. One can imaging "call-by-name" versions of these rules that did not evaluate the discarded pair element, but this would conflict with the search rules, which specify left-to-right evaluation of pair expressions. Note also that there is no instruction rule for pair expressions since (v_1, v_2) is a value, and hence a final state in the transition system.

$$\frac{}{\texttt{fst}((v_1, v_2)) \mapsto v_1} \quad \text{(PE1)} \qquad \frac{}{\texttt{snd}((v_1, v_2)) \mapsto v_2} \quad \text{(PE2)}$$

$$\frac{e \mapsto e'}{\texttt{fst}(e) \mapsto \texttt{fst}(e')} \quad \text{(PE3)} \qquad \frac{e \mapsto e'}{\texttt{snd}(e) \mapsto \texttt{snd}(e')} \quad \text{(PE4)}$$

$$\frac{e_1 \mapsto e_1'}{(e_1, e_2) \mapsto (e_1', e_2)} \quad \text{(PE5)} \qquad \frac{e_1 \mapsto e_1'}{(e_1, e_2) \mapsto (e_1', e_2)} \quad \text{(PE6)}$$

(c) [10 points]. State the new clauses in the Inversion Theorem (Theorem 9.1, p. 53) and the Canonical Forms Lemma (Lemma 10.2, p. 61) needed to deal with pairs.

Solution:
Inversion Theorem:
(1) $\Gamma \vdash (e_1, e_2) : \tau_1 * \tau_2 \implies \Gamma \vdash e_1 : \tau_1 \text{ and } \Gamma \vdash e_2 : \tau_2$
(2) $\Gamma \vdash \texttt{fst}(e) : \tau \implies \exists \tau_2 . \Gamma \vdash e : \tau * \tau_2$
(3) $\Gamma \vdash \texttt{snd}(e) : \tau \implies \exists \tau_1 . \Gamma \vdash e : \tau_1 * \tau$

(d) [20 points]. Give the new case of the proof of the Progress Theorem relating to pair expressions of the form $(e_1,\ e_2)$.

Solution: The assumption of the theorem is that $\vdash e : \tau$, where $e = (e_1, e_2)$, and the proof is by induction on the derivation of the typing judgment. By the Inversion Theorem, we have $\tau = \tau_1 * \tau_2$ and

$$(1) \quad \vdash e_1 : \tau_1$$

$$(2) \quad \vdash e_2 : \tau_2$$

The induction hypotheses are

$$(IH1) \quad e_1 \text{ avalueor } e_1 \mapsto e_1'$$

$$(IH2) \quad e_2 \text{ avalueor } e_2 \mapsto e_2'$$

If e_1 and e_2 are both values, then $e = (e_1, e_2)$ is also a value, and we are done. If $e_1 \mapsto e_1'$ then $\mathit{stepe}(e_1', e_2)$ by (PE5). Finally, if e_1 is a value and $e_2 \mapsto e_2'$ then $\mathit{stepe}(e_1, e_2')$ by (PE6).

(e) [20 points]. Give the new cases of the Preservation Theorem relating to expressions of the form $\texttt{fst}(e)$.

Solution: The proof assumptions are that $\vdash e : \tau$ and $e \mapsto e'$, and the proof is by induction on the derivation of the transition judgement.

Case $e = \texttt{fst}((v_1, v_2))$ and $e \mapsto v_1$, with $e' = v_1$ (rule (Pe1)). Then $e = (e_1, e_2)$ and by inversion we have $\vdash v_1 : \tau$, hence $\vdash e' : \tau$, as required.

Case $e = \texttt{fst}(e_1)$ and $e' = \texttt{fst}(e_1')$ where $e_1 \mapsto e_1'$ (rule (PE3)). By inversion, there exists a type τ_2 such that $\vdash e_1 : \tau * \tau_2$. The induction hypothesis is $\vdash e_1' : \tau * \tau_2$. Thus by the rule (P2) from part (a) we have $\vdash \texttt{fst}(e_1') : \tau$, i.e. $\vdash e' : \tau$.