Name:____

CS 578 Programming Language Semantics – Mid-term Exam May 2, 2024

This exam has 4 questions; most have several sub-parts. The worth of each question and sub-part is indicated in square brackets. There are 75 points in total, and you have 75 minutes for the exam. Please write your answers on the exam paper in the spaces provided. The exam is closed book.

All the questions concern the simply typed λ-calculus extended with Naturals and Pairs, which will be denoted $\lambda_{\rightarrow,\mathbb{N},\times}$. For your reference, syntax and semantic rules for $\lambda_{\rightarrow,\mathbb{N},\times}$ are provided at the end of the exam.

1. [20 pts.] Derivations

Consider the $\lambda_{\rightarrow,\mathbb{N},\times}$ term

$$\mathtt{t} = (\lambda \mathtt{f}{:}\mathtt{Nat} \rightarrow (\mathtt{Nat} \times \mathtt{Nat}).\mathtt{pred}\ ((\mathtt{f}\ \ \mathtt{0}).\mathtt{1}))\,(\lambda \mathtt{x}{:}\mathtt{Nat}.\{\mathtt{succ}\ \ \mathtt{x},\mathtt{0}\})$$

When answering the following questions, you may abbreviate `Nat` by `N`, `succ` by `S` , and `pred` by `P` to save writing time.

(a) [5 pts.] Show the sequence of one-step evaluation transitions (in the small-step or contextual semantics) that lead from `t` to the normal form `0`. It is *not* necessary to give the full derivation for each transition. (Hint: 4 steps are needed.)

Answer:

$$\begin{array}{ll}
 & (\lambda \mathtt{f}{:}\mathtt{N} \rightarrow (\mathtt{N} \times \mathtt{N}).\mathtt{P}\ \ ((\mathtt{f}\ \ \mathtt{0}).\mathtt{1}))\,(\lambda \mathtt{x}{:}\mathtt{N}.\{\mathtt{S}\ \ \mathtt{x},\mathtt{0}\}) \\
\rightarrow & \mathtt{P}\ \ (((\lambda \mathtt{x}{:}\mathtt{N}.\{\mathtt{S}\ \ \mathtt{x},\mathtt{0}\})\mathtt{0}).\mathtt{1}) \\
\rightarrow & \mathtt{P}\ \ (\{\mathtt{S}\ \ \mathtt{0},\mathtt{0}\}.\mathtt{1}) \\
\rightarrow & \mathtt{P}\ \ (\mathtt{S}\ \ \mathtt{0}) \\
\rightarrow & \mathtt{0}
\end{array}$$

(b) [5 pts.] Draw a complete derivation tree using the big-step rules to show that $\mathtt{t} \Downarrow \mathtt{0}$. To save writing, you can drop the type annotations on lambda-bound variables. Your tree should have as few nodes as possible. (Hint: 9 nodes are sufficient.)

Answer:

$$\dfrac{
\dfrac{}{\lambda \mathtt{f}.\mathtt{P}((\mathtt{f}\ \ \mathtt{0}).\mathtt{1}) \Downarrow \lambda \mathtt{f}.\mathtt{P}((\mathtt{f}\ \ \mathtt{0}).\mathtt{1})}\ \text{B-VALUE}
\qquad
\dfrac{}{\lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\} \Downarrow \lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\}}\ \text{B-VALUE}
\qquad
\dfrac{\dfrac{\dfrac{
\dfrac{}{\lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\} \Downarrow \lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\}}\ \text{B-VALUE}
\quad
\dfrac{}{\mathtt{0} \Downarrow \mathtt{0}}\ \text{B-VALUE}
\quad
\dfrac{}{\{\mathtt{S0},\mathtt{0}\} \Downarrow \{\mathtt{S0},\mathtt{0}\}}\ \text{B-VALUE}
}{(\lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\})\,\mathtt{0} \Downarrow \{\mathtt{S0},\mathtt{0}\}}\ \text{B-APP}
}{((\lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\})\ \ \mathtt{0}).\mathtt{1} \Downarrow \mathtt{S0}}\ \text{B-PROJ1}
}{\mathtt{P}(((\lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\})\ \ \mathtt{0}).\mathtt{1}) \Downarrow \mathtt{0}}\ \text{B-PREDSUCC}
}{(\lambda \mathtt{f}.\mathtt{P}((\mathtt{f}\ \ \mathtt{0}).\mathtt{1}))\,(\lambda \mathtt{x}.\{\mathtt{Sx},\mathtt{0}\}) \Downarrow \mathtt{0}}\ \text{B-APP}$$

(c) [10 pts.] Draw a derivation tree using the typing rules to show that $\emptyset \vdash \mathtt{t : Nat}$. Use of auxiliary definitions for contexts Γ that arise along the way is recommended! (Hint: Your tree should have 12 nodes.)

Answer:

$$
\dfrac{
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{
          \dfrac{\mathtt{f} : \mathtt{N} \to (\mathtt{N} \times \mathtt{N}) \in \Gamma_f}{\Gamma_f \vdash \mathtt{f} : \mathtt{N} \to (\mathtt{N} \times \mathtt{N})}\ \text{T-VAR}
          \qquad
          \dfrac{}{\Gamma_f \vdash \mathtt{0} : \mathtt{N}}\ \text{T-ZERO}
        }{\Gamma_f \vdash \mathtt{f\ 0} : \mathtt{N} \times \mathtt{N}}\ \text{T-APP}
      }{\Gamma_f \vdash \mathtt{(f\ \ 0).1} : \mathtt{N}}\ \text{T-PROJ1}
    }{\Gamma_f \vdash \mathtt{P\ \ ((f\ \ 0).1)} : \mathtt{N}}\ \text{T-PRED}
  }{\emptyset \vdash \lambda \mathtt{f{:}N} \to (\mathtt{N} \times \mathtt{N})\mathtt{.P\ \ ((f\ \ 0).1)} : (\mathtt{N} \to (\mathtt{N} \times \mathtt{N})) \to \mathtt{N}}\ \text{T-ABS}
  \qquad
  \dfrac{
    \dfrac{
      \dfrac{
        \dfrac{\mathtt{x} : \mathtt{N} \in \Gamma_x}{\Gamma_x \vdash \mathtt{x} : \mathtt{N}}\ \text{T-VAR}
      }{\Gamma_x \vdash \mathtt{S\ \ x} : \mathtt{N}}\ \text{T-SUCC}
      \qquad
      \dfrac{}{\Gamma_x \vdash \mathtt{0} : \mathtt{N}}\ \text{T-ZERO}
    }{\Gamma_x \vdash \{\mathtt{S\ \ x,0}\} : \mathtt{N} \times \mathtt{N}}\ \text{T-PAIR}
  }{\emptyset \vdash \lambda \mathtt{x{:}N.}\{\mathtt{S\ \ x,0}\} : \mathtt{N} \to (\mathtt{N} \times \mathtt{N})}\ \text{T-ABS}
}{\emptyset \vdash (\lambda \mathtt{f{:}N} \to (\mathtt{N} \times \mathtt{N})\mathtt{.P\ \ ((f\ \ 0).1)})\ (\lambda \mathtt{x{:}N.}\{\mathtt{S\ \ x,0}\}) : \mathtt{N}}\ \text{T-APP}
$$

where $\Gamma_f = \emptyset, \mathtt{f} : \mathtt{N} \to (\mathtt{N} \times \mathtt{N})$ and $\Gamma_x = \emptyset, \mathtt{x} : \mathtt{N}$.

2. [20 pts.] Properties

The following meta-properties hold for language $\lambda_{\rightarrow,\mathbb{N},\times}$ under small-step or contextual semantics.

- **Determinacy** (of one-step evaluation): If $\mathtt{t} \rightarrow \mathtt{t}'$ and $\mathtt{t} \rightarrow \mathtt{t}''$ then $\mathtt{t}' = \mathtt{t}''$.
- **Uniqueness** (of normal forms): If $\mathtt{t} \rightarrow^* \mathtt{u}$ and $\mathtt{t} \rightarrow^* \mathtt{u}'$, where $\mathtt{u}$ and $\mathtt{u}'$ are both normal forms, then $\mathtt{u} = \mathtt{u}'$.
- **Progress**: If $\vdash \mathtt{t} : \mathtt{T}$ then either $\mathtt{t}$ is a value or else $\exists \mathtt{t}'$ such that $\mathtt{t} \rightarrow \mathtt{t}'$.
- **Preservation**: If $\vdash \mathtt{t} : \mathtt{T}$ and $\mathtt{t} \rightarrow \mathtt{t}'$ then $\vdash \mathtt{t}' : \mathtt{T}$.

For each of the following alternative languages, state which, if any, of the properties are **false**, and, for each such property, give a brief counter-example demonstrating that the property does not hold.

(a) [5 pts.] Language $\lambda_{\rightarrow,\mathbb{N},\times}$ with the addition of a small-step rule

$$\{\mathtt{v}_1, \mathtt{v}_2\}.1 \rightarrow \mathtt{v}_2 \qquad \text{(E-FUNNY1)}$$

Answer:

- Determinacy fails: e.g. $\{\lambda\mathtt{x}\mathtt{:Nat.x}, \mathtt{0}\}.\mathtt{1}$ steps to either $\lambda\mathtt{x}\mathtt{:Nat.x}$ or $\mathtt{0}$.
- Uniqueness fails with the same counterexample.
- Preservation fails: the same example has type $\mathtt{Nat} \rightarrow \mathtt{Nat}$ but can step to term $\mathtt{0}$ of type $\mathtt{Nat}$.
- Progress still holds. (Adding an additional stepping rule can never make progress fail!)

(b) [5 pts.] Language $\lambda_{\rightarrow,\mathbb{N},\times}$ with the removal of small-step rule E-PROJ1.

Answer:

- Progress fails: for example $\{\mathtt{pred\ succ\ 0}, \mathtt{0}\}.\mathtt{1}$ has type $\mathtt{Nat}$, but is not a value and does not step.
- Other properties still hold.

(c) [5 pts.] Language $\lambda_{\rightarrow,\mathbb{N},\times}$ under contextual semantics with a change of the context grammar to:

$$\texttt{C} ::= [\ ] \mid \texttt{C t} \mid \texttt{v C} \mid \texttt{pred C} \mid \texttt{succ C} \mid \texttt{C.1} \mid \texttt{C.2} \mid \{\texttt{C,t}\} \mid \{\texttt{t,C}\}$$

Answer:

- Determinacy fails: {pred 0,pred 0} now reduces to either {0,pred 0} or to {pred 0,0}.
- However, Uniqueness still holds, since ultimately the same reductions are made on the way to a normal form, regardless of order.
- Progress and Preservation also still hold.

(d) [5 pts.] Language $\lambda_{\rightarrow,\mathbb{N},\times}$ with the addition of the typing rule

$$\Gamma \vdash \texttt{pred 0} : \texttt{Nat} \rightarrow \texttt{Nat} \qquad \text{(T-FUNNY2)}$$

Answer:

- Preservation fails: pred 0 has type Nat $\rightarrow$ Nat but reduces to 0, which does not have type Nat $\rightarrow$ Nat.
- Progress does not fail: although we can construct an example like (pred 0) 0 which violates type safety (since it is well-typed but can never reduce to a value), it can still take a single step (to 0 0).
- Since the stepping rules do not change, Determinacy (and hence Uniqueness) still hold.

3. [20 pts.] Progress

The following Progress theorem holds for the small-step semantics of $\lambda_{\to,\mathbb{N},\times}$:

Theorem. If $\vdash \mathtt{t} : \mathtt{T}$ then either $\mathtt{t} \to \mathtt{t}'$ for some $\mathtt{t}'$, or $\mathtt{t}$ is a value.

An incomplete proof of this theorem is given below. Complete the proof by filling in the four missing cases (marked by a ?). You may assume the following lemma without proof:

- **Canonical Forms Lemma**:
  1. If $\mathtt{v}$ is a value of type $\mathtt{Nat}$, then $\mathtt{v}$ is a numeric value $\mathtt{nv}$.
  2. If $\mathtt{v}$ is a value of type $\mathtt{T}_1 \to \mathtt{T}_2$, then $\mathtt{v} = \lambda\mathtt{x}\mathbin{:}\mathtt{T}_1\mathtt{.t}_2$.
  3. If $\mathtt{v}$ is a value of type $\mathtt{T}_1 \times \mathtt{T}_2$, then $\mathtt{v} = \{\mathtt{v}_1, \mathtt{v}_2\}$.

Proof. By induction on the structure of the typing derivation $\vdash \mathtt{t} : \mathtt{T}$. We proceed by case analysis on the root ("final") rule in the derivation.

- Case T-VAR: ?

  Answer:
  Impossible, since $\mathtt{t}$ is typable in the empty context.

- Case T-ABS: ?

  Answer:
  $\mathtt{t} = \lambda\mathtt{x}\mathbin{:}\mathtt{T.t}_1$ is already a value.

- Case T-APP: $\mathtt{t} = \mathtt{t}_1\ \mathtt{t}_2$

  By inversion on the derivation, we have

  $\vdash \mathtt{t}_1 : \mathtt{T}_{11} \to \mathtt{T}$
  $\vdash \mathtt{t}_2 : \mathtt{T}_{11}$

  By induction on the sub-derivations, we have that each of $\mathtt{t}_1$ and $\mathtt{t}_2$ can either take a step or is a value. There are three cases:

  - If $\mathtt{t}_1$ can take a step, then $\mathtt{t}$ can take a step by E-APP1.
  - If $\mathtt{t}_1$ is a value but $\mathtt{t}_2$ can take a step, then $\mathtt{t}$ can take a step by E-APP2.
  - If both $\mathtt{t}_1$ and $\mathtt{t}_2$ are values, then since by Canonical Forms $\mathtt{t}_1 = \lambda\mathtt{x}\mathbin{:}\mathtt{T}_{11}\mathtt{.t}_{11}$, we know $\mathtt{t}$ can take a step by E-APPABS.

- Case T-ZERO: $\mathtt{t} = \mathtt{0}$ is already a value.

- Case T-SUCC: $\mathtt{t} = \mathtt{succ}\ \mathtt{t}_1$

  By inversion on the derivation, we have

  $\vdash \mathtt{t}_1 : \mathtt{Nat}$

  By induction on the sub-derivation, $\mathtt{t}_1$ can either take a step or is a value.

- In the former case, t can take a step by E-SUCC.
- In the latter case, Canonical Forms says that t_1 is a numeric value nv_1. So $t = succ\ t_1$ is already also a (numeric) value.

• Case T-PRED: Similar to T-SUCC.

• Case T-PAIR: ?

Answer:

$t = \{t_1, t_2\}$

By inversion on the derivation, we have

$\vdash t_1 : T_1$
$\vdash t_2 : T_2$

By induction on the two sub-derivations, we have that each of t_1 and t_2 can take a step or is a value. There are three cases:

- If t_1 can take a step, t can take a step by E-PAIR1.
- If t_1 is a value but t_2 can take a step, t can take a step by E-PAIR2.
- If both t_1 and t_2 are values, then t is itself already a value.

• Case T-PROJ1: ?

Answer:

$t = t_1.1$

By inversion on the derivation, we have

$\vdash t_1 : T \times T_2$

By induction on the sub-derivation, either t_1 can take a step or it is a value.

- In the former case, t an take a step by E-PROJ1.
- In the latter case, Canonical Forms tells us that t_1 has the form $\{v_1, v_2\}$, so t can take a step by E-PAIRBETA1.

• Case T-PROJ2: Similar to T-PROJ1.

4. [15 pts.] True or False

Say whether each of the following assertions about $\lambda_{\rightarrow,\mathbb{N},\times}$ is true or false. If true, give a brief informal justification. If false, give a **concrete** counterexample.

(a) [5 pts.] If term $\texttt{t}$ is not well-typed, then $\texttt{t}$ is stuck under small-step semantics.

Answer:
False. For example,the term $(\lambda\texttt{x:Nat.0})\ \{\texttt{0,0}\}$ is ill-typed, but it steps to $\texttt{0}$.

(b) [5 pts.] If $\texttt{t} \rightarrow^* \texttt{t}'$ then $\mathit{size}(\texttt{t}') \leq \mathit{size}(\texttt{t})$, where as usual *size* is the number of nodes in the abstract syntax tree representation of the term.

Answer:
False. If we apply a lambda whose argument is used more than once in its body, the term's size can grow. For example,

$(\lambda\texttt{x:Nat.}\{\texttt{x,x}\})\,(\texttt{succ succ succ succ 0})$

has size 10, but steps to

$\{\texttt{succ succ succ succ 0,succ succ succ succ 0}\}$,

which has size 11.

(c) [5 pts.] If $\texttt{t}$ is a closed term (i.e., it has no free variables) and $\texttt{t} \rightarrow^* \texttt{t}'$, then $\texttt{t}'$ is a closed term.

Answer:
True. Informally, the only rule that removes a variable binding (hence possibly changing a variable from bound to free) is E-AppAbs, but this rule replaces every use of that variable with the (closed) argument value. (We can prove this formally using an inductive argument very similar to that for Preservation.)

Syntax and Rules for Simply Typed λ-calculus with Naturals and Pairs ($\lambda_{\rightarrow,\mathbb{N},\times}$)

Syntactic forms:

`t`	::=		*terms:*
		`x`	*variable*
		λ`x:T.t`	*abstraction*
		`t t`	*application*
		`0`	*constant zero*
		`succ t`	*successor*
		`pred t`	*predecessor*
		`{t,t}`	*pair*
		`t.1`	*first projection*
		`t.2`	*second projection*
`v`	::=		*values:*
		λ`x:T.t`	*abstraction value*
		`nv`	*numeric value*
		`{v,v}`	*pair value*
`nv`	::=		*numeric values:*
		`0`	*zero value*
		`succ nv`	*successor value*
`T`	::=		*types:*
		$\mathtt{T} \rightarrow \mathtt{T}$	*type of functions*
		`Nat`	*type of naturals*
		$\mathtt{T}_1 \times \mathtt{T}_2$	*type of pairs*
Γ	::=		*contexts:*
		$\emptyset$	*empty context*
		$\Gamma, \mathtt{x} : \mathtt{T}$	*term variable binding*

Typing rules:

$$\frac{\mathtt{x} : \mathtt{T} \in \Gamma}{\Gamma \vdash \mathtt{x} : \mathtt{T}} \tag{T-Var}$$

$$\frac{\Gamma, \mathtt{x} : \mathtt{T}_1 \vdash \mathtt{t}_2 : \mathtt{T}_2}{\Gamma \vdash \lambda \mathtt{x}{:}\mathtt{T}_1\mathtt{.t}_2 : \mathtt{T}_1 \to \mathtt{T}_2} \tag{T-Abs}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathtt{T}_{11} \to \mathtt{T}_{12} \qquad \Gamma \vdash \mathtt{t}_2 : \mathtt{T}_{11}}{\Gamma \vdash \mathtt{t}_1\ \mathtt{t}_2 : \mathtt{T}_{12}} \tag{T-App}$$

$$\Gamma \vdash \mathtt{0} : \mathtt{Nat} \tag{T-Zero}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathtt{Nat}}{\Gamma \vdash \mathtt{succ}\ \mathtt{t}_1 : \mathtt{Nat}} \tag{T-Succ}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathtt{Nat}}{\Gamma \vdash \mathtt{pred}\ \mathtt{t}_1 : \mathtt{Nat}} \tag{T-Pred}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathtt{T}_1 \qquad \Gamma \vdash \mathtt{t}_2 : \mathtt{T}_2}{\Gamma \vdash \{\mathtt{t}_1\mathtt{,t}_2\} : \mathtt{T}_1 \times \mathtt{T}_2} \tag{T-Pair}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathtt{T}_{11} \times \mathtt{T}_{12}}{\Gamma \vdash \mathtt{t}_1\mathtt{.1} : \mathtt{T}_{11}} \tag{T-Proj1}$$

$$\frac{\Gamma \vdash \mathtt{t}_1 : \mathtt{T}_{11} \times \mathtt{T}_{12}}{\Gamma \vdash \mathtt{t}_1\mathtt{.2} : \mathtt{T}_{12}} \tag{T-Proj2}$$

Small-step evaluation rules:

$$\frac{\mathtt{t_1} \to \mathtt{t_1'}}{\mathtt{t_1\ t_2} \to \mathtt{t_1'\ t_2}} \qquad \text{(E-APP1)}$$

$$\frac{\mathtt{t_2} \to \mathtt{t_2'}}{\mathtt{v_1\ t_2} \to \mathtt{v_1\ t_2'}} \qquad \text{(E-APP2)}$$

$$(\lambda \mathtt{x}\mathtt{:}\mathtt{T_{11}}\mathtt{.}\mathtt{t_{12}})\ \mathtt{v_2} \to [\mathtt{x} \mapsto \mathtt{v_2}]\mathtt{t_{12}} \qquad \text{(E-APPABS)}$$

$$\frac{\mathtt{t_1} \to \mathtt{t_1'}}{\mathtt{succ\ t_1} \to \mathtt{succ\ t_1'}} \qquad \text{(E-SUCC)}$$

$$\mathtt{pred\ 0} \to \mathtt{0} \qquad \text{(E-PREDZERO)}$$

$$\mathtt{pred\ succ\ nv_1} \to \mathtt{nv_1} \qquad \text{(E-PREDSUCC)}$$

$$\frac{\mathtt{t_1} \to \mathtt{t_1'}}{\mathtt{pred\ t_1} \to \mathtt{pred\ t_1'}} \qquad \text{(E-PRED)}$$

$$\{\mathtt{v_1},\mathtt{v_2}\}\mathtt{.1} \to \mathtt{v_1} \qquad \text{(E-PAIRBETA1)}$$

$$\{\mathtt{v_1},\mathtt{v_2}\}\mathtt{.2} \to \mathtt{v_2} \qquad \text{(E-PAIRBETA2)}$$

$$\frac{\mathtt{t_1} \to \mathtt{t_1'}}{\mathtt{t_1.1} \to \mathtt{t_1'.1}} \qquad \text{(E-PROJ1)}$$

$$\frac{\mathtt{t_1} \to \mathtt{t_1'}}{\mathtt{t_1.2} \to \mathtt{t_1'.2}} \qquad \text{(E-PROJ2)}$$

$$\frac{\mathtt{t_1} \to \mathtt{t_1'}}{\{\mathtt{t_1},\mathtt{t_2}\} \to \{\mathtt{t_1'},\mathtt{t_2}\}} \qquad \text{(E-PAIR1)}$$

$$\frac{\mathtt{t_2} \to \mathtt{t_2'}}{\{\mathtt{v_1},\mathtt{t_2}\} \to \{\mathtt{v_1},\mathtt{t_2'}\}} \qquad \text{(E-PAIR2)}$$

Contextual Semantic Rules:

$$\frac{\texttt{t} \rightarrow_{cmp} \texttt{t}'}{\texttt{C}[\texttt{t}] \rightarrow \texttt{C}[\texttt{t}']} \qquad \text{(E-STEP)}$$

$$\texttt{C} ::= [\ ] \mid \texttt{C t} \mid \texttt{v C} \mid \texttt{pred C} \mid \texttt{succ C} \mid \texttt{C.1} \mid \texttt{C.2} \mid \{\texttt{C,t}\} \mid \{\texttt{v,C}\}$$

$$(\lambda\texttt{x:T}_{11}\texttt{.t}_{12})\ \texttt{v}_2 \rightarrow_{cmp} [\texttt{x} \mapsto \texttt{v}_2]\texttt{t}_{12} \qquad \text{(E-APPABS)}$$

$$\texttt{pred 0} \rightarrow_{cmp} \texttt{0} \qquad \text{(E-PREDZERO)}$$

$$\texttt{pred succ nv}_1 \rightarrow_{cmp} \texttt{nv}_1 \qquad \text{(E-PREDSUCC)}$$

$$\{\texttt{v}_1\texttt{,v}_2\}\texttt{.1} \rightarrow_{cmp} \texttt{v}_1 \qquad \text{(E-PAIRBETA1)}$$

$$\{\texttt{v}_1\texttt{,v}_2\}\texttt{.2} \rightarrow_{cmp} \texttt{v}_2 \qquad \text{(E-PAIRBETA2)}$$

Big-step Semantic Rules:

$$\texttt{v} \Downarrow \texttt{v} \qquad \text{(B-VALUE)}$$

$$\frac{\texttt{t}_1 \Downarrow (\lambda\texttt{x:T}_{11}\texttt{.t}_{12}) \qquad \texttt{t}_2 \Downarrow \texttt{v}_2 \qquad [\texttt{x} \mapsto \texttt{v}_2]\texttt{t}_{12} \Downarrow \texttt{v}}{\texttt{t}_1\ \texttt{t}_2 \Downarrow \texttt{v}} \qquad \text{(B-APP)}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{nv}_1}{\texttt{succ t}_1 \Downarrow \texttt{succ nv}_1} \qquad \text{(B-SUCC)}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{0}}{\texttt{pred t}_1 \Downarrow \texttt{0}} \qquad \text{(B-PREDZERO)}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{succ nv}_1}{\texttt{pred t}_1 \Downarrow \texttt{nv}_1} \qquad \text{(B-PREDSUCC)}$$

$$\frac{\texttt{t}_1 \Downarrow \texttt{v}_1 \qquad \texttt{t}_2 \Downarrow \texttt{v}_2}{\{\texttt{t}_1\texttt{,t}_2\} \Downarrow \{\texttt{v}_1\texttt{,v}_2\}} \qquad \text{(B-PAIR)}$$

$$\frac{\texttt{t} \Downarrow \{\texttt{v}_1\texttt{,v}_2\}}{\texttt{t.1} \Downarrow \texttt{v}_1} \qquad \text{(B-PROJ1)}$$

$$\frac{\texttt{t} \Downarrow \{\texttt{v}_1\texttt{,v}_2\}}{\texttt{t.2} \Downarrow \texttt{v}_2} \qquad \text{(B-PROJ2)}$$