CIS 7000-1 Homework 4

NAME: FILL IN HERE

December 1, 2025

1 Syntax-directed type-and-effect systems

In class, we defined a type-and-effect system that tracks potential nontermination. Actually, we defined *two* different type systems: an initial version containing a sub-effecting rule, and then a syntax-directed version, which we proved equivalent to the original version.

For this problem, finish the definition of the type system shown below. In particular, you must update the typing rules VO-ABS, TO-APP, TO-IFZ, TO-PRJ1, TO-PRJ2, and TO-CASE defined in hw4.ott, by adding premises and (potentially) modifying the conclusion of the rule. Your type system should consider the effect as an "output" of the judgment and calculate the minimal effect that follows from the rules according to the effect ordering. In some rules, you may need to use the operator $\varepsilon_1 \sqcup \varepsilon_2$, that calculates the least upper bound of two effects. This operator returns $\bot$ if both inputs are $\bot$, and DIV otherwise.

$\boxed{\Gamma \vdash_{OUT} e \overset{\varepsilon}{\in} \tau}$ *(In context Γ, e has type τ and effect ε)*

$$\textsc{to-ret}\ \dfrac{\Gamma \vdash v \in \tau}{\Gamma \vdash_{OUT} \mathbf{ret}\, v \overset{\bot}{\in} \tau}$$

$$\textsc{to-let}\ \dfrac{\Gamma \vdash_{OUT} e_1 \overset{\varepsilon_1}{\in} \tau_1 \qquad \Gamma, x:\tau_1 \vdash_{OUT} e_2 \overset{\varepsilon_2}{\in} \tau}{\Gamma \vdash_{OUT} \mathbf{let}\, x \;=\; e_1\, \mathbf{in}\, e_2 \overset{\varepsilon_1 \oplus \varepsilon_2}{\in} \tau}$$

$$\textsc{to-app}\ \dfrac{\texttt{FILL IN HERE}}{\Gamma \vdash_{OUT} v_1\ v_2 \overset{\varepsilon}{\in} \tau_2}$$

$$\textsc{to-ifz}\ \dfrac{\texttt{FILL IN HERE}}{\Gamma \vdash_{OUT} \mathbf{case}\, v\, \mathbf{of}\, \{0 \Rightarrow e_1;\ \mathbf{S}\, x \Rightarrow e_2\} \overset{\varepsilon}{\in} \tau}$$

$$\textsc{to-prj1}\ \dfrac{\texttt{FILL IN HERE}}{\Gamma \vdash_{OUT} \mathbf{prj}_1 v \overset{\varepsilon}{\in} \tau_1}$$

$$\textsc{to-prj2}\ \dfrac{\texttt{FILL IN HERE}}{\Gamma \vdash_{OUT} \mathbf{prj}_1 v \overset{\varepsilon}{\in} \tau_1}$$

$$\textsc{to-case}\ \dfrac{\texttt{FILL IN HERE}}{\Gamma \vdash_{OUT} \mathbf{case}\, v\, \mathbf{of}\, \{\mathbf{inj}_1 x_1 \Rightarrow e_1; \mathbf{inj}_2 x_2 \Rightarrow e_2\} \overset{\varepsilon}{\in} \tau}$$

$$\textsc{to-unfold}\ \dfrac{\Gamma \vdash v \in \mu\alpha.\tau}{\Gamma \vdash_{OUT} \mathbf{unfold}\, v \overset{\mathsf{DIV}}{\in} \tau[\mu\alpha.\tau/\alpha]}$$

$\boxed{\Gamma \vdash v \in \tau}$ *(In context Γ, v has type τ)*

$$\textsc{vo-var}\ \dfrac{x : \tau \in \Gamma}{\Gamma \vdash x \in \tau}$$

$$\textsc{vo-zero}\ \dfrac{}{\Gamma \vdash 0 \in \mathbf{Nat}}$$

$$\textsc{vo-succ}\ \dfrac{\Gamma \vdash v \in \mathbf{Nat}}{\Gamma \vdash \mathbf{S}\, v \in \mathbf{Nat}}$$

$$\textsc{vo-pair}\ \dfrac{\Gamma \vdash v_1 \in \tau_1 \qquad \Gamma \vdash v_2 \in \tau_2}{\Gamma \vdash (v_1, v_2) \in \tau_1 \overset{\varepsilon}{\times} \tau_2}$$

$$\textsc{vo-inj1}\ \dfrac{\Gamma \vdash v_1 \in \tau_1}{\Gamma \vdash \mathbf{inj}_1 v_1 \in \tau_1 + \tau_2}$$

$$\textsc{vo-inj2}\ \dfrac{\Gamma \vdash v_2 \in \tau_2}{\Gamma \vdash \mathbf{inj}_2 v_2 \in \tau_1 + \tau_2}$$

$$\textsc{vo-abs}\ \dfrac{\texttt{FILL IN HERE}}{\Gamma \vdash \lambda x.e \in \tau_1 \xrightarrow{\varepsilon} \tau_2}$$

$$\textsc{vo-rec}\ \dfrac{\tau\ \mathsf{ok} \qquad \Gamma, x:\tau \vdash v \in \tau}{\Gamma \vdash \mathbf{rec}\, x.v \in \tau}$$

$$\textsc{vo-fold}\ \dfrac{\Gamma \vdash v \in \tau[\mu\alpha.\tau/\alpha]}{\Gamma \vdash \mathbf{fold}\, v \in \mu\alpha.\tau}$$

Your syntax-directed system should satisfy the following two lemmas, connecting it to the original type system. For this problem, you do not need to typeset these proofs, you only need to present the rules. You may wish to look at the Rocq definitions and proofs in the module to see how to use that tool to experiment with your rules.

Lemma 1.1 (Syntax-directed system implies original system)**.** $\Gamma \vdash_{OUT} e \overset{\varepsilon}{\in} \tau$ implies $\Gamma \vdash e \overset{\varepsilon}{\in} \tau$.

Lemma 1.2 (Original system implies new syntax directed system)**.** If $\Gamma \vdash e \overset{\varepsilon}{\in} \tau$ then there exists some ε_1 such that $\varepsilon_1 <: \varepsilon$ and $\Gamma \vdash_{OUT} e \overset{\varepsilon_1}{\in} \tau$.

2 Running time as an effect

Design a type-and-effect system for running time. The time folder contains Rocq starter code for your use.

Your type-and-effect judgment, written $\Gamma \vdash e \overset{\varepsilon}{\in} \tau$, should assign every term some effect ε, where effect annotations should be either some natural number k, or DIV to indicate potential nontermination:

$$\varepsilon := k \mid \mathsf{DIV}$$

1. *Effect algebra*: what are the definitions of $\bot$, $\varepsilon_1 <: \varepsilon_2$ and $\varepsilon_1 \oplus \varepsilon_2$? Make sure that these definitions satisfy the properties of a pre-ordered monoid. (You don't need to typeset the proofs of these properties.)

$$\begin{aligned} \bot &= \\ \varepsilon_1 \oplus \varepsilon_2 &= \\ \varepsilon_1 <: \varepsilon_2 &= \end{aligned}$$

2. *Type system*: The typing rules of this language are the same as that of the type-and-effect language found in Chapter 7 of the lecture notes, except that effect tracking must be modified to count execution steps.

   Write the modified versions of rules TIE-RET, TIE-LET, TIE-APP, TIE-PRJ1, TIE-PRJ2, TIE-IFZ, and TIE-CASE.

3. *Example*: Your type system should *not* just use DIV as the effect for everything. Show the typing derivation for a term that type checks with effect 3 that does not use the effect subsumption rule (i.e. a term that takes exactly three steps to reduce to a value).

4. *Effect soundness*: Prove the following theorem. In your answer, you do not need to typeset the entire proof. Merely, identify what induction principles you use and explicitly state any helper lemmas that your proof relies on. (You do not need to define a logical relation for this problem.)

   Lemma 2.1 (Finite Step Soundness)**.** If $\vdash e \overset{k}{\in} \tau$ then there exists some $j \le k$, and value v, such that $e \rightsquigarrow^j \textbf{ret}\ v$.

3 Soundness of monadic type system

We described a monadic type system for tracking nontermination, called MON. Prove that terms with nonboxed types always terminate using a logical relation. (This logical relation does not need to be step-indexed.) If you are working in Rocq, you can use the definitions in the modal/modal.v module.

Lemma 3.1 (Monadic soundness)**.** If $\vdash e \in \tau$ and τ is not a box type, then there exists some v such that $e \rightsquigarrow^* \textbf{ret}\, v$.

For this problem, you need to typeset the definition of the logical relation and state and prove the semantic soundness lemmas for rules T-LET-NB, T-BIND, and T-PURE.