Demand Unchanged Variable Analysis Technical Report

Chahyun Kang and Kimball Germane

Brigham Young University, Provo UT 84606, USA
chi2eut@byu.edu kimball@cs.byu.edu

1 Semantics

1.1 Demand Operational Semantics (De Bruijn)

$\Sigma \vdash e \Downarrow_d v$

$$\textsc{Ref-Zero}\ \frac{\Sigma \vdash e_1 \Downarrow_d v}{(e_0\, e_1) :: \Sigma \vdash x^0 \Downarrow_d v} \qquad \textsc{Ref-Succ}\ \frac{\Sigma \vdash e_0 \Downarrow_d (\lambda.e, \Sigma_0) \qquad \Sigma_0 \vdash x^n \Downarrow_d v}{(e_0\, e_1) :: \Sigma \vdash x^{n+1} \Downarrow_d v}$$

$$\textsc{Lam}\ \frac{}{\Sigma \vdash \lambda.e \Downarrow_d (\lambda.e, \Sigma)}$$

$$\textsc{App}\ \frac{\Sigma \vdash e_0 \Downarrow_d (\lambda.e, \Sigma_0) \qquad \Sigma \vdash e_1 \Downarrow_d v_1 \qquad (e_0\, e_1) :: \Sigma \vdash e \Downarrow_d v}{\Sigma \vdash (e_0\, e_1) \Downarrow_d v}$$

1.2 Demand Operational Semantics (names)

$\Sigma \vdash C[e] \Downarrow_d v$

$$\textsc{Bind-Rat}\ \frac{x, \Sigma \vdash C[(e_0\, e_1)] \Uparrow_d v}{x, \Sigma \vdash C[([e_0]\, e_1)] \Uparrow_d v}$$

$$\textsc{Bind-Ran}\ \frac{x, \Sigma \vdash C[(e_0\, e_1)] \Uparrow_d v}{x, \Sigma \vdash C[(e_0\, [e_1])] \Uparrow_d v} \qquad \textsc{Bind-Zero}\ \frac{\Sigma \vdash C_0[(e_0\, [e_1])] \Downarrow_d v}{x, C_0[(e_0\, e_1)] :: \Sigma \vdash C[\lambda x.[e]] \Uparrow_d v}$$

$$\textsc{Bind-Succ}\ \frac{x \neq y \qquad \Sigma \vdash C_0[([e_0]\, e_1)] \Downarrow_d (C_\lambda[\lambda z.e_\lambda], \Sigma_\lambda) \qquad x, \Sigma_\lambda \vdash C[\lambda y.e] \Uparrow_d v}{x, C_0[(e_0\, e_1)] :: \Sigma \vdash C[\lambda y.[e]] \Uparrow_d v}$$

$$\textsc{Ref}\ \frac{x, \Sigma \vdash C[x] \Uparrow_d v}{\Sigma \vdash C[x] \Downarrow_d v} \qquad \textsc{Lam}\ \frac{}{\Sigma \vdash C[\lambda x.e] \Downarrow_d (C[\lambda x.e], \Sigma)}$$

$$\textsc{App}\ \frac{\begin{array}{c}\Sigma \vdash C[([e_0]\, e_1)] \Downarrow_d (C_\lambda[\lambda.e], \Sigma_0) \\ \Sigma \vdash C[(e_0\, [e_1])] \Downarrow_d v_1 \qquad C[(e_0\, e_1)] :: \Sigma \vdash C_\lambda[\lambda x.[e]] \Downarrow_d v\end{array}}{\Sigma \vdash C[(e_0\, e_1)] \Downarrow_d v}$$

Theorem 1. *Modulo syntactic contexts, for all Σ, $C[e]$, v, $\Sigma \vdash e \Downarrow_d v \Leftrightarrow \Sigma \vdash C[e] \Downarrow_d v$.*

1.3 Demand Evaluation

$\Sigma \sim \rho$

$$\frac{}{\epsilon \sim \epsilon} \qquad \frac{\Sigma \vdash C[(e_0\, e_1)] \Downarrow_d (C_\lambda[\lambda x.e], \Sigma_0) \qquad \Sigma_0 \sim \rho}{C[(e_0\, e_1)] :: \Sigma \sim (C[(e_0\, e_1)] :: \Sigma) :: \rho}$$

$\rho \vdash C[e] \Downarrow_e v \;\; \rho \vdash C[e] \Rightarrow_e c$

$$\textsc{Bind-Rat}\ \frac{x, \rho \vdash C[(e_0\ e_1)] \Uparrow_e v}{x, \rho \vdash C[([e_0]\ e_1)] \Uparrow_e v} \qquad \textsc{Bind-Ran}\ \frac{x, \rho \vdash C[(e_0\ e_1)] \Uparrow_e v}{x, \rho \vdash C[(e_0\ [e_1])] \Uparrow_e v}$$

$$\textsc{Bind-Zero}\ \frac{}{x, \rho \vdash C[\lambda x.[e]] \Uparrow_e (C[\lambda x.[e]], \rho)} \qquad \textsc{Bind-Succ}\ \frac{x \neq y \qquad x, \rho \vdash C[\lambda y.e] \Uparrow_d v}{x, \Sigma :: \rho \vdash C[\lambda y.[e]] \Uparrow_d v}$$

$$\textsc{Ref}\ \frac{\begin{array}{c} x, \rho_x \vdash C_x[x] \Uparrow_d (C_\lambda[\lambda x.[e]], (C[(e_0\ e_1)] :: \Sigma) :: \rho_\lambda) \\ \rho_\lambda \vdash C_\lambda[\lambda x.e] \Rightarrow_e (C[([e_0]\ e_1)], \Sigma :: \rho_0) \end{array} \qquad \Sigma :: \rho_0 \vdash C[(e_0\ [e_1])] \Downarrow_e v}{\rho_x \vdash C_x[x] \Downarrow_e v}$$

$$\textsc{Lam}\ \frac{}{\rho \vdash C[\lambda x.e] \Downarrow_e (C[\lambda x.e], \rho)}$$

$$\textsc{App}\ \frac{\begin{array}{c} \Sigma :: \rho \vdash C[([e_0]\ e_1)] \Downarrow_e (C_\lambda[\lambda x.e], \rho_\lambda) \\ \Sigma :: \rho \vdash C[(e_0\ [e_1])] \Downarrow_e v_1 \end{array} \qquad (C[(e_0\ e_1)] :: \Sigma) :: \rho_\lambda \vdash C_\lambda[\lambda x.[e]] \Downarrow_e v}{\Sigma :: \rho \vdash C[(e_0\ e_1)] \Downarrow_e v}$$

$$\textsc{Find-Rat}\ \frac{x, \rho \vdash C[([e_0]\ e_1)] \Rightarrow_e (C_x[x], \rho_x)}{x, \rho \vdash C[(e_0\ e_1)] \Rightarrow_e (C_x[x], \rho_x)} \qquad \textsc{Find-Ran}\ \frac{x, \rho \vdash C[(e_0\ [e_1])] \Rightarrow_e (C_x[x], \rho_x)}{x, \rho \vdash C[(e_0\ e_1)] \Rightarrow_e (C_x[x], \rho_x)}$$

$$\textsc{Find-Bod}\ \frac{\begin{array}{c} x \neq y \qquad \rho_\lambda \vdash C_\lambda[\lambda y.e] \Rightarrow_e (C[([e_0]\ e_1)], \Sigma :: \rho) \\ x, (C[(e_0\ e_1)] :: \Sigma) :: \rho_\lambda \vdash C_\lambda[\lambda y.[e]] \Rightarrow_e (C_x[x], \rho_x) x, \rho \vdash C[(e_0\ [e_1])] \Rightarrow_e (C_x[x], \rho_x) \end{array}}{x, \rho_\lambda \vdash C_\lambda[\lambda y.e] \Rightarrow_e (C_x[x], \rho_x)}$$

$$\textsc{Rat}\ \frac{}{\rho \vdash C[([e_0]\ e_1)] \Rightarrow_e (C[([e_0]\ e_1)], \rho)} \qquad \textsc{Ran}\ \frac{\begin{array}{c} \Sigma :: \rho \vdash C[([e_0]\ e_1)] \Downarrow_e (C_\lambda[\lambda x.e], \rho_\lambda) \\ x, (C[(e_0\ e_1)] :: \Sigma) :: \rho_\lambda \vdash C_\lambda[\lambda x.[e]] \Rightarrow_e c \end{array}}{\Sigma :: \rho \vdash C[(e_0\ [e_1])] \Rightarrow_e c}$$

$$\textsc{Bod}\ \frac{\rho_\lambda \vdash C_\lambda[\lambda x.e] \Rightarrow_e (C[([e_0]\ e_1)], \rho) \qquad \rho \vdash C[(e_0\ e_1)] \Rightarrow_e c}{(C[(e_0\ e_1)] :: \Sigma) :: \rho_\lambda \vdash C_\lambda[\lambda x.[e]] \Rightarrow_e c}$$

Theorem 2. *For all Σ, Σ_v, ρ, ρ_v, $C[e]$, $C_\lambda[\lambda x.e]$, if $\Sigma \sim \rho$, then $\Sigma \vdash C[e] \Downarrow_d (C_\lambda[\lambda x.e], \Sigma_v) \Leftrightarrow \rho \vdash C[e] \Downarrow_e (C_\lambda[\lambda x.e], \rho_v)$ where $\Sigma_v \sim \rho_v$.*

2 Proofs

Theorem 3. *If $\rho \sim \rho_d$ then $\rho \vdash e \Downarrow v$, $\sigma \leftrightarrow \rho_d \vdash e \Downarrow_d v$ where $v \sim v_d$*

$$v_d \ ::= \ (\lambda x.e, \ \Sigma) \mid n$$

$$\text{Const}\frac{}{\Sigma \ \vdash \ n \ \Downarrow_d \ n} \qquad \text{Lam}\frac{}{\Sigma \ \vdash \ \lambda x.e \ \Downarrow_d \ (\lambda x.e, \ \Sigma)}$$

$$\text{Ref0}\frac{\Sigma \ \vdash \ e_1 \ \Downarrow_d \ v_d}{(app \ e_0 \ e_1) :: \Sigma \ \vdash \ x^0 \Downarrow_d \ v_d} \qquad \text{Refn}\frac{\Sigma \ \vdash \ e_0 \ \Downarrow_d \ (\lambda x.e, \ \Sigma') \quad \Sigma' \ \vdash \ x^n \ \Downarrow_d \ v}{(app \ e_0 \ e_1) :: \Sigma \ \vdash \ x^{n+1} \ \Downarrow_d \ v_d}$$

$$\text{App}\frac{\Sigma \ \vdash \ e_0 \Downarrow_d \ (\lambda x.e, \ \Sigma') \quad \Sigma \ \vdash \ e_1 \Downarrow_d \ v_d 0 \quad (app \ e_0 \ e_1) :: \Sigma \ \vdash \ e \ \Downarrow_d \ v_d}{\Sigma \ \vdash \ (app \ e_0 \ e_1) \Downarrow_d \ v_d}$$

Fig. 1. Pure Demand Operational Semantics Rules

Proof. This is a proof.

Theorem 4 (Fundamental UVA Theorem). *If* $\rho \ \vdash \ e \ \Downarrow_d, \ (\lambda x.e, \ \rho_0), \ X$ *then* $\forall y \in [free(e) \cap free(\lambda x.e)] \setminus X, \ \rho(y) = \rho_0(y)$

Proof. This is a proof.

Theorem 5 (Fundamental UVA Theorem For Demand Inf-CFA). *If* $\rho \ \vdash \ e \ \Downarrow_{ev}, \ (\lambda x.e, \ \rho_0), \ X$
then $\forall y \in [free(e) \cap free(\lambda x.e)] \setminus X, \ \rho(y) = \rho_0(y)$

Proof. This is a proof.

Theorem 6. *If* $\rho_\infty \ \vdash \ e \Downarrow_\infty, \ v, \ X$ *then* $|\rho_\infty| \sqsubseteq \rho_{\infty-1}$ *and* $\rho_{\infty-1} \ \vdash \ e \ \Downarrow_{\infty-1} v, \ X$ *s.t.* $|v| \sqsubseteq v$ *and* $X \sqsubseteq X$

Proof. We prove by using theorem 7. Because the semantics are identical other than an additional parameter for changed set, the relation stays the same.

Theorem 7 (Demand Evaluation → Demand Evaluation Without Argument). *If* $\rho_d = \rho_{woa}$ *then* $\rho_d \ \vdash \ C[e] \ \Downarrow_d \ C'[\lambda x.e], \ \rho'_d \ \rightarrow \ \rho_{woa} \ \vdash \ C[e] \ \Downarrow_d^{woa} C'[\lambda x.e], \ \rho'_{woa}$
s.t. $\rho'_d = \rho'_{woa}$

Proof. This is proven by induction on $\Downarrow_d$. Two semantics are identical except for omitting an argument evaluation in Demand Evaluation Without Argument semantics for application evaluation case. Thus, if demand evaluation yields a result, it meets all the conditions necessary for demand evaluation without arguments to work.

Theorem 8 (Demand Evaluation Without Argument → Demand ∞-CFA). *If* $\rho_d \sim \rho$ *then* $\rho_d \ \vdash \ C[e] \ \Downarrow_d \ C'[\lambda x.e], \ \rho'_d \ \rightarrow \ \rho \ \vdash \ C[e] \ \Downarrow_\infty C'[\lambda x.e], \ \rho'$
where $\rho'_d \sim \rho'$

Proof. We prove the theorem by induction on the $\Downarrow$.

Lam Case: This is a base case where environments stay the same, so the case proof goes through straightforwardly.

App Case: The first conditions can be proven by using the base case. For the environment extension for the second conditions, both semantics are using the same app function to build the extended environment, so the relation still holds. We can then use the base case to prove the second conditions.

Ref Case: We first use the bind lemma to prove that the resulting expressions and environments between two semantics correspond. For the second conditions, the demand ∞-CFA's call has the same condition as demand evaluation's trace and constraint. Finally, we use the base case to prove the last conditions.

Lemma 1 (Demand Eval Without Argument $\rightarrow$ Demand ∞-CFA Trace). *If* $\rho_d \sim \rho_\infty$, *then* $\rho_d \vdash C[e] \Rightarrow_{tr} C'[(e_0\ e_1)],\ \rho'_d \leftrightarrow \rho_\infty \vdash C[e] \Rightarrow_{tr\infty} C'[(e_0\ e_1)],\ \rho'_\infty$ *where* $\rho'_d \sim \rho'_\infty$

Proof. We prove by induction on $\Rightarrow_{tr}$.

Fun Case: This is a base case, which the expression and the environment stays the same. The proof goes through straightforwardly.

Arg Case: For the first condition, we use the main theorem. For the second condition, we use the find lemma. Finally, we use the hypothesis from the base case to prove the last condition's relation.

Bod Case: We simply use the base case hypothesis for both conditions since they are both trace relations.

Lemma 2 (Demand Eval Without Argument $\rightarrow$ Demand ∞-CFA Bind). *If* $\rho_d \sim \rho_\infty$, *then* $n,\ \rho_d \vdash C[x^n] \Rightarrow_{db} C'[\lambda x.[e]],\ \rho'_d \leftrightarrow \rho_\infty \vdash C[x],\ x \Rightarrow_b C'[\lambda x.[e]],\ \rho'_\infty$ *where* $\rho'_d \sim \rho'_\infty$

Proof. We prove by showing the relation between predefining the count n for each variable according to de Bruijn indices. Each variable's index is predefined by its depth from its binding lambda procedure. We define it as such: $depth(C,\ x) = n$ Because the indices were defined according to depth, given a same variable, it will find the correct lambda expression from the variable site.

Lemma 3 (Demand Eval Without Argument $\rightarrow$ Demand ∞-CFA Find). *If* $\rho_d \sim \rho_\infty$, *then* $0,\ \rho_d \vdash C[\lambda x.[e]],\ x \Rightarrow_f C'[x^n],\ \rho'_d \leftrightarrow \rho_\infty \vdash C[\lambda x.[e]]\ x \Rightarrow_{f\infty} C'[x],\ \rho'_\infty$ *where* $\rho'_\infty \sim \rho'_{\infty-1}$ *and* $depth(C',\ x) = n$

Proof. We prove by showing the relation between predefining the count n for each variable according to de Bruijn indices. Each variable's index is predefined by its depth from its binding lambda procedure. We define it as such: $depth(C,\ x) = n$ Because the indices were defined according to depth, given a same variable, it will find the correct expression site that contains the variable in search. Also, we assume all variables are unique or alphabetized, which removes any confusions.

Theorem 9 (Demand ∞-CFA $\rightarrow$ Demand $\infty - 1$-CFA). *If* $\rho_\infty = \rho_{\infty-1}$ *then* $\rho_\infty \vdash C[e] \Downarrow_\infty C_v[v]_\infty,\ \rho'_\infty \rightarrow \rho_{\infty-1} \vdash C[e] \Downarrow_{\infty-1} C_v[v]_{\infty-1},\ \rho'_{\infty-1}$ *s.t.* $\rho'_\infty \sim \rho'_{\infty-1}$

Proof. We prove the theorem by induction on the $\Downarrow$ relation. Two semantics are nearly identical except the function app is limited to $\infty - 1$ in $\Downarrow_{\infty-1}$. We use $\rho_\infty \sim \rho_{\infty-1}$ to relate an infinite to a bound environment.

Lam Case: Lambda case is a base case, where the environment stays the same for both semantics. The case proof goes through straightforwardly.

App Case: We use the base case to prove the first condition for each semantics. Then both semantics extend their environments using app, but $app_{\infty-1}$ is simply bound. Thus extended environments' relation still holds, so we can use the base case.

Ref Case: First, we use the bind lemma to show that the resulting expression and environment correspond. Then, we use the call lemma to show that the resulting expression and environment correspond. Finally, we use the base case to prove the last condition.

Lemma 4 (Demand ∞-CFA $\rightarrow$ Demand $\infty - 1$-CFA Trace). *If* $\rho_\infty \sim \rho_{\infty-1}$*, then* $\rho_\infty \vdash C[e] \Rightarrow_{tr\infty} C'[(e_0\ e_1)],\ \rho'_\infty \leftrightarrow \rho_{\infty-1} \vdash C[e] \Rightarrow_{tr\infty-1} C'[(e_0\ e_1)],\ \rho'_{\infty-1}$ *where* $\rho'_\infty \sim \rho'_{\infty-1}$

Proof. We prove by induction on $\rightarrow_{tr}$.

Fun Case: This is a base case, which the environment and the expression stays the same for both semantics. The proof goes through straightforwardly.

Arg Case: For the first condition, we use the main evaluation theorem. Then, we use the find lemma to show that the environment stays related. Finally, we use the base case to prove the last conditions.

Bod Case: We simply use the given hypothesis from the base case for two conditions.

Lemma 5 (Demand ∞-CFA $\rightarrow$ Demand $\infty - 1$-CFA Bind). *If* $\rho_\infty \sim \rho_{\infty-1}$*, then* $\rho_\infty \vdash C[x],\ x \Rightarrow_b C'[\lambda x.[e]],\ \rho'_\infty \leftrightarrow \rho_{\infty-1} \vdash C[x],\ x \Rightarrow_b C'[\lambda x.[e]],\ \rho'_{\infty-1}$ *where* $\rho'_\infty \sim \rho'_{\infty-1}$

Proof. This is a trivial proof since both semantics for bind operator are identical. They are equivalent.

Lemma 6 (Demand ∞-CFA $\rightarrow$ Demand $\infty - 1$-CFA Find). *If* $\rho_\infty \sim \rho_{\infty-1}$*, then* $\rho_\infty \vdash C[\lambda x.[e]],\ x \Rightarrow_f C'[x],\ \rho'_\infty \leftrightarrow \rho_{\infty-1} \vdash C[\lambda x.[e]]\ x \Rightarrow_f C'[x],\ \rho'_{\infty-1}$ *where* $\rho'_\infty \sim \rho'_{\infty-1}$

Proof. This is a trivial proof since both semantics for find operator are identical. They are equivalent.

$$\text{Lam}\ \frac{}{\rho \vdash C[\lambda x.e] \Downarrow_d (C[\lambda x.e], \rho)}$$

$$\text{App}\ \frac{\begin{array}{c}\rho \vdash C[([e_0]\ e_1)] \Downarrow_d C'[\lambda x.e], \rho_0 \\ \rho \vdash C[(e_0\ [e_1])] \Downarrow_d C_1[v_1], \rho_1 \\ app(C[(e_0\ e_1)], \rho, \rho_0) \vdash C'[\lambda x.[e]] \Downarrow_d C_v[v], \rho_v\end{array}}{\rho \vdash C[(e_0\ e_1)] \Downarrow_d C_v[v], \rho_v}$$

$$\text{Ref}\ \frac{\begin{array}{c}n, \rho \vdash C[x^n] \Uparrow_b C_x[\lambda x.[e]], (C'[(e_0\ e_1)] :: \Sigma) :: \rho\prime \\ \rho' \vdash C_x[\lambda x.e] \Rightarrow_{tr} C'[(e_0\ e_1)], \rho'' \\ fst(\rho'\prime) = \Sigma \\ \rho'' \vdash C'[(e_0\ [e_1])] \Downarrow_d C_v[v], \rho_v\end{array}}{\rho \vdash C[x^n] \Downarrow_d C_v[v], \rho_v}$$

$$\frac{}{0, \rho \vdash C[\lambda x.[e]] \Uparrow_b C[\lambda x.[e]], \rho} \qquad \frac{n, \rho \vdash C[\lambda x.e] \Uparrow_b C_b[b], \rho_b}{n+1, \rho \vdash C[\lambda x.[e]] \Uparrow_b C_b[b], \rho_b}$$

$$\frac{n, \rho \vdash C[(e_0\ e_1)] \Uparrow_b C_b[b], \rho_b}{n, \rho \vdash C[[e_0]\ e_1] \Uparrow_b C_b[b], \rho_b} \qquad \frac{n, \rho \vdash C[(e_0\ e_1)] \Uparrow_b C_b[b], \rho_b}{n, \rho \vdash C[e_0\ [e_1]] \Uparrow_b C_b[b], \rho_b}$$

$$\text{Fun}\ \frac{}{\rho \vdash C[([e_0]\ e_1)] \Rightarrow_{tr} C[(e_0\ e_1)], \rho}$$

$$\text{Arg}\ \frac{\begin{array}{c}\rho \vdash C[([e_0]\ e_1)] \Downarrow_d C_0[\lambda x.e], \rho_0 \\ 0, app(C[(e_0\ e_1)], \rho, \rho_0), \vdash C_0[\lambda x.[e]] \Rightarrow_f C_x[x^n], \rho_x \\ \rho_x \vdash C_x[x^n] \Rightarrow_{tr} C_c[c], \rho_c\end{array}}{\rho \vdash C[(e_0\ [e_1])] \Rightarrow_{tr} C_c[c], \rho_c}$$

$$\text{Bod}\ \frac{\begin{array}{c}\rho \vdash C[\lambda x.e] \Rightarrow_{tr} C'[(e_0\ e_1)], \rho' \\ \rho' \vdash C'[(e_0\ e_1)] \Rightarrow_{tr} C_c[c], \rho_c\end{array}}{\rho \vdash C[\lambda x.[e]] \Rightarrow_{tr} C_c[c], \rho_c}$$

$$\frac{n, \rho \vdash C[([e_0]\ e_1)] \Rightarrow_f C_f[e], \rho_f}{n, \rho \vdash C[(e_0 e_1)] \Rightarrow_f C_f[e], \rho_f} \qquad \frac{n, \rho \vdash C[(e_0\ [e_1])] \Rightarrow_f C_f[e], \rho_f}{n, \rho \vdash C[(e_0 e_1)] \Rightarrow_f C_f[e], \rho_f}$$

$$\frac{\begin{array}{c}\rho \vdash C[\lambda.e] \Rightarrow_{tr} C'[(e_0\ e_1)], \rho' \\ n+1, app(C'[(e_0\ e_1)], \rho', \rho) \vdash C[\lambda.[e]] \Rightarrow_f C_f[e], \rho_f\end{array}}{n, \rho \vdash C[\lambda x.e] \Rightarrow_f C_f[e], \rho_f}$$

$$\frac{}{n, \rho \vdash C[x^n] \Rightarrow_f C[x^n], \rho}$$

Fig. 2. Demand Evaluation Rules

$$\text{Lam}\ \frac{}{\rho \ \vdash\ C[\lambda x.e] \ \Downarrow_d \ (C[\lambda x.e],\ \rho)}$$

$$\text{App}\ \frac{\begin{array}{c}\rho \ \vdash\ C[([e_0]\ e_1)] \ \Downarrow_d \ C'[\lambda x.e],\ \rho_0 \\ app(C[(e_0\ e_1)],\ \rho,\ \rho_0) \ \vdash\ C'[\lambda x.[e]] \ \Downarrow_d \ C_v[v],\ \rho_v\end{array}}{\rho \ \vdash\ C[(e_0\ e_1)] \ \Downarrow_d \ C_v[v],\ \rho_v}$$

$$\text{Ref}\ \frac{\begin{array}{c}n,\ \rho \ \vdash\ C[x^n] \ \Uparrow_b \ C_x[\lambda x.[e]],\ (C'[(e_0\ e_1)] :: \Sigma) :: \rho\prime \\ \rho' \ \vdash\ C_x[\lambda x.e] \ \Rightarrow_{tr} \ C'[(e_0\ e_1)],\ \rho'' \\ fst(\rho'\prime) = \Sigma \\ \rho'' \ \vdash\ C'[(e_0\ [e_1])] \ \Downarrow_d \ C_v[v],\ \rho_v\end{array}}{\rho \ \vdash\ C[x^n] \ \Downarrow_d \ C_v[v],\ \rho_v}$$

$$\frac{}{0,\ \rho \ \vdash\ C[\lambda x.[e]] \ \Uparrow_b \ C[\lambda x.[e]],\ \rho} \qquad \frac{n,\ \rho \ \vdash\ C[\lambda x.e] \ \Uparrow_b \ C_b[b],\ \rho_b}{n+1,\ \rho \ \vdash\ C[\lambda x.[e]] \ \Uparrow_b \ C_b[b],\ \rho_b}$$

$$\frac{n,\ \rho \ \vdash\ C[(e_0\ e_1)] \ \Uparrow_b \ C_b[b],\ \rho_b}{n,\ \rho \ \vdash\ C[[e_0]\ e_1] \ \Uparrow_b \ C_b[b],\ \rho_b} \qquad \frac{n,\ \rho \ \vdash\ C[(e_0\ e_1)] \ \Uparrow_b \ C_b[b],\ \rho_b}{n,\ \rho \ \vdash\ C[e_0\ [e_1]] \ \Uparrow_b \ C_b[b],\ \rho_b}$$

$$\text{Fun}\ \frac{}{\rho \ \vdash\ C[([e_0]\ e_1)] \ \Rightarrow_{tr} \ C[(e_0\ e_1)],\ \rho}$$

$$\text{Arg}\ \frac{\begin{array}{c}\rho \ \vdash\ C[([e_0]\ e_1)] \ \Downarrow_d \ C_0[\lambda x.e],\ \rho_0 \\ 0,\ app(C[(e_0\ e_1)],\ \rho,\ \rho_0),\ \vdash\ C_0[\lambda x.[e]] \ \Rightarrow_f \ C_x[x^n],\ \rho_x \\ \rho_x \ \vdash\ C_x[x^n] \ \Rightarrow_{tr} \ C_c[c],\ \rho_c\end{array}}{\rho \ \vdash\ C[(e_0\ [e_1])] \ \Rightarrow_{tr} \ C_c[c],\ \rho_c}$$

$$\text{Bod}\ \frac{\begin{array}{c}\rho \ \vdash\ C[\lambda x.e] \ \Rightarrow_{tr} \ C'[(e_0\ e_1)],\ \rho' \\ \rho' \ \vdash\ C'[(e_0\ e_1)] \ \Rightarrow_{tr} \ C_c[c],\ \rho_c\end{array}}{\rho \ \vdash\ C[\lambda x.[e]] \ \Rightarrow_{tr} \ C_c[c],\ \rho_c}$$

$$\frac{n,\ \rho \ \vdash\ C[([e_0]\ e_1)] \ \Rightarrow_f \ C_f[e],\ \rho_f}{n,\ \rho \ \vdash\ C[(e_0 e_1)] \ \Rightarrow_f \ C_f[e],\ \rho_f} \qquad \frac{n,\ \rho \ \vdash\ C[(e_0\ [e_1])] \ \Rightarrow_f \ C_f[e],\ \rho_f}{n,\ \rho \ \vdash\ C[(e_0 e_1)] \ \Rightarrow_f \ C_f[e],\ \rho_f}$$

$$\frac{\begin{array}{c}\rho \ \vdash\ C[\lambda.e] \ \Rightarrow_{tr} \ C'[(e_0\ e_1)],\ \rho' \\ n+1,\ app(C'[(e_0\ e_1)],\ \rho',\ \rho) \ \vdash\ C[\lambda.[e]] \ \Rightarrow_f \ C_f[e],\ \rho_f\end{array}}{n,\ \rho \ \vdash\ C[\lambda x.e] \ \Rightarrow_f \ C_f[e],\ \rho_f}$$

$$\frac{}{n,\ \rho \ \vdash\ C[x^n] \ \Rightarrow_f \ C[x^n],\ \rho}$$

Fig. 3. Demand Evaluation Rules Without Argument Eval

$$\text{Lam}\ \frac{}{\rho \ \vdash\ C[\lambda x.e] \ \Downarrow_{ev}\ C[\lambda x.e],\ \rho}$$

$$\text{App}\ \frac{\begin{array}{c}\rho \ \vdash\ C[(app\ [e_0]\ e_1)],\ \Downarrow_\infty\ C'[\lambda x.e],\ \rho' \\ app_\infty(C[(e_0\ e_1)]) \ \vdash\ C'[\lambda x.[e]]\ \Downarrow_\infty\ C_v[e_v],\ \rho_v\end{array}}{C[(e_0\ e_1)],\ \rho\ \Downarrow_\infty\ C_v[e_v],\ \rho_v}$$

$$\text{Ref}\ \frac{\begin{array}{c}\rho \ \vdash\ C[x],\ x\ \Uparrow_b\ C'[\lambda x.[e]],\ \rho' \\ \rho' \ \vdash\ C'[(\lambda x.[e]),\ \rho'\ \Downarrow_{c\infty}\ C''[(e_0\ e_1)],\ \rho'' \\ \rho''\ \vdash\ C''[(e_0\ [e_1])]\ \Downarrow_\infty\ C_v[e_v],\ \rho_v\end{array}}{\rho\ \vdash\ C[x]\ \Downarrow_{ev}\ C_v[e_v],\ \rho_v}$$

$$\text{Fun}\ \frac{}{\rho\ \vdash\ C[([e_0]\ e_1)]\ \Downarrow_{tr\infty}\ C[(e_0\ e_1)],\ \rho}$$

$$\text{Arg}\ \frac{\begin{array}{c}\rho\ \vdash\ C[([e_0]\ e_1)]\ \Downarrow_\infty\ C'[\lambda x.e],\ \rho' \\ app(C[(e_0\ e_1)],\ \rho,\ \rho')\ \vdash\ C'[\lambda x.[e]],\ x\ \Rightarrow_f\ C_x[x],\ \rho_x \\ \rho_x\ \vdash\ C_x[x]\ \Rightarrow_{tr\infty}\ C_c[e_c],\ \rho_c\end{array}}{\rho\ \vdash\ C[(app\ e_0\ [e_1])]\ \Rightarrow_{tr}\ C_c[e_c],\ \rho_c}$$

$$\text{Bod}\ \frac{\begin{array}{c}\rho\ \vdash\ C[\lambda x.e]\ \Rightarrow_{tr\infty}\ C'[(e_0\ e_1)],\ \rho' \\ \rho'\ C'[(e_0\ e_1)]\ \Rightarrow_{tr\infty}\ C_c[c],\ \rho_c\end{array}}{\rho\ \vdash\ C[\lambda x.[e]]\ \Rightarrow_{tr\infty}\ C_c[c],\ \rho_c}$$

$$\text{Known Call}\ \frac{\begin{array}{c}\rho\ \vdash\ C[\lambda x.e]\ \Rightarrow_{tr\infty}\ C_c[(app\ e_0\ e_1)],\ \rho' \\ \Sigma' = C_c[(app\ e_0\ e_1)] :: fst(\rho') \qquad \Sigma' = \Sigma\end{array}}{\Sigma :: \rho\ \vdash\ C[\lambda x.[e]]\ \Downarrow_{call}\ C_c[(app\ e_0\ e_1)],\ \rho'}$$

$$\text{Unknown Call}\ \frac{\begin{array}{c}\rho\ \vdash\ C[\lambda x.e]\ \Downarrow_{tr}\ C_c[(app\ e_0\ e_1)],\ \rho' \\ \Sigma' = C_c[(app\ e_0\ e_1)] :: fst(\rho') \qquad \Sigma' \sqsubset \Sigma\end{array}}{\Sigma' :: \rho \xRightarrow{Refines} \Sigma :: \rho}$$

Fig. 4. Demand ∞-CFA Rules