Fundamentals of Programming Languages

Assignment 2

Cost Semantics and Higher-Order Recursion

Guilherme João Correia Lopes
fc52761

2022/2023

1 Big-step semantics with cost

Assuming all the problem context given for the questions in group 1, we have the following as the answer for each question:

A. The big step evaluation rules

$$\textsc{B-Succ}\ \frac{\texttt{t1} \Downarrow^{n} \texttt{nv1}}{\texttt{succ t1} \Downarrow^{n+1} \texttt{succ nv1}} \qquad \textsc{B-PredZero}\ \frac{\texttt{t1} \Downarrow^{n} \texttt{0}}{\texttt{pred t1} \Downarrow^{n+1} \texttt{0}} \qquad \textsc{B-PredSucc}\ \frac{\texttt{t1} \Downarrow^{n} \texttt{succ nv1}}{\texttt{pred t1} \Downarrow^{n+1} \texttt{nv1}}$$

B. Example of reduction

$$\frac{*\text{(lacking space for } 2^{nd} \text{ B-APP, see below)} \qquad \frac{}{\texttt{2} \Downarrow^{0} \texttt{2}}\ \textsc{B-Value} \qquad \frac{\frac{\frac{\frac{\frac{}{\texttt{0} \Downarrow^{0} \texttt{0}}\ \textsc{B-Value}}{\texttt{succ 0} \Downarrow^{0+1} \texttt{succ 0}}\ \textsc{B-Succ}}{\texttt{succ(succ 0)} \Downarrow^{1+1} \texttt{succ(succ 0)}}\ \textsc{B-Succ}}{\texttt{pred 2} \Downarrow^{1+2} \texttt{succ 0}}\ \textsc{B-PredSucc}}{\texttt{pred(pred 2)} \Downarrow^{1+3} \texttt{0}}\ \textsc{B-PredSucc}}{\lambda\texttt{f}.\lambda\texttt{x}.\texttt{f(fx) pred 2} \Downarrow^{1+0+4} \texttt{0}}\ \textsc{B-App}$$

$$* \quad \frac{\frac{}{\lambda\texttt{f}.\lambda\texttt{x}.\texttt{f(fx)} \Downarrow^{0} \lambda\texttt{f}.\lambda\texttt{x}.\texttt{f(fx)}}\ \textsc{B-Value} \qquad \frac{}{\texttt{pred} \Downarrow^{0} \texttt{pred}}\ \textsc{B-Value} \qquad \frac{}{\lambda\texttt{x}.\texttt{pred(pred x)} \Downarrow^{0} \lambda\texttt{x}.\texttt{pred(pred x)}}\ \textsc{B-Value}}{\lambda\texttt{f}.\lambda\texttt{x}.\texttt{f(fx) pred} \Downarrow^{0+0+0+1} \lambda\texttt{x}.\texttt{pred(pred x)}}\ \textsc{B-App}$$

We can conclude that we have $applyTwice$ $\texttt{pred 2} \Downarrow^{\texttt{k}} \texttt{v}$ for: cost $k = 5$; value $v = 0$;

C. Evaluation returns a value

Induction on the rules for $t \Downarrow^{n} v$

1. Induction on B-VALUE: $v \Downarrow^{0} v$
   This is straightforward, because the rule tells that every value reduces to a value in 0 steps.

2. Induction on B-APP:
Similarly to the B-SUCC, PREDZERO and B-PREDSUCC cases below, the rule B-APP tells us that $t \Downarrow^n v$ is true if it is confirmed to be the case that $t_1 \Downarrow^n \lambda x.t_1'$ and $t_2 \Downarrow^m v_2$ and $[x \to v_2]t_1' \Downarrow^o v_1$. In this case, we have to explore 3 subcases for each premise.

   (a) Subcase $t_1 \Downarrow^n \lambda x.t_1'$
   This is somewhat straightforward, because t_1 reduces to $\lambda x.t_1'$, and $\lambda x.t_1'$ is a value.

   (b) Subcase $t_2 \Downarrow^m v_2$
   This is straightforward, because t_2 reduces to a value.

   (c) Subcase $[x \to v_2]t_1' \Downarrow^o v_1$
   This is also straightforward, because the substitution $[x \to v_2]t_1'$ reduces to a value.

   We can then see that, when applying B-APP, every subcase meets the requirements, so we can then conclude that $v1$ is a value and have $t \Downarrow^n v$ for B-APP.

3. Induction on B-SUCC:
The rule B-SUCC tells us that $t \Downarrow^n v$ is true if it is confirmed to exist a $t1 \Downarrow^n nv1$. If so, by applying the induction hypothesis, for $t1 \Downarrow^n nv1$, we conclude that $nv1$ is a value, and so we have $t \Downarrow^n v$ for B-SUCC.

4. Induction on B-PREDZERO:
The rule B-PREDZERO tells us that $t \Downarrow^n v$ is true if it is confirmed to exist a $t1 \Downarrow^n 0$. If so, by applying the induction hypothesis, for $t1 \Downarrow^n 0$, we conclude that 0 is a value, and so we have $t \Downarrow^n v$ for B-PREDZERO.

5. Induction on B-PREDSUCC:
Similarly to the B-SUCC and PREDZERO cases, the rule B-PREDSUCC tells us that $t \Downarrow^n v$ is true if it is confirmed to exist a $t1 \Downarrow^n succ\, nv1$. If so, by applying the induction hypothesis, for $t1 \Downarrow^n succ\, nv1$, we conclude that $succ\, nv1$ is a value (proved in the B-SUCC case), and so we have $t \Downarrow^n v$ for B-PREDSUCC.

D. The cost of big step and small step semantics coincide

1. Proof by rule induction:

   (a) Case B-VALUE:
   The rule says that a value reduces to a value in 0 steps, in other words, $v \Downarrow^0 v$, so it is equal to $v \to^0 v$ according to the reflexibility of the multi-step evaluation.

   (b) Case B-APP:
   $t = t_1 t_2,\ v = v_1$ and $k = n + m + o + 1$
   The rule says that $t \Downarrow^k v$ only if we have the following:

      i. $t_1 \Downarrow^n \lambda x.t_1'$, so by the induction hypothesis, we have $t_1 \to^n \lambda x.t_1'$.
      ii. $t_2 \Downarrow^m v_2$, so by the induction hypothesis, we have $t_2 \to^m v_2$.
      iii. $[x \to v_2]t_1' \Downarrow^o v_1$, so by the induction hypothesis, we have $[x \to v_2]t_1' \to^o v_1$.

   So we can conclude that if $t \Downarrow^k v$, then $t \to^k v$.

   (c) Case B-SUCC:
   $t = succ\, t_1,\ v = succ\, nv_1$ and $k = n + 1$
   The rule says that $t \Downarrow^k v$ only if we have $t_1 \Downarrow^n nv_1$. By the induction hypothesis, having $t_1 \Downarrow^n nv_1$ then we have $t_1 \to^n nv_1$, so we can conclude that if $t \Downarrow^k v$, then $t \to^k v$.

   (d) Case B-PREDZERO:
   $t = pred\, t_1,\ v = 0$ and $k = n + 1$
   The rule says that $t \Downarrow^k v$ only if we have $t_1 \Downarrow^n 0$. By the induction hypothesis, having $t_1 \Downarrow^n 0$ then we have $t_1 \to^n 0$, so we can conclude that if $t \Downarrow^k v$, then $t \to^k v$.

(e) Case B-PREDSUCC:
$t = pred\, t_1,\ v = nv_1$ and $k = n + 1$
The rule says that $t \Downarrow^k v$ only if we have $t_1 \Downarrow^n succ\, nv_1$. By the induction hypothesis, having $t_1 \Downarrow^n succ\, nv_1$ then we have $t_1 \rightarrow^n succ\, nv_1$, so we can conclude that if $t \Downarrow^k v$, then $t \rightarrow^k v$.

2. Proof by induction on k:

(a) Case $k = 0$:
The only case in which $k = 0$ is when we have the rule B-VALUE $v \rightarrow^0 v$, which says that a value reduces to a value in 0 steps, so by induction hypothesis we conclude $v \Downarrow^0 v$.

(b) Case $k = n$

i. Case B-APP:
$t = t_1 t_2$ and $v = v_1$
The rule says that $t \rightarrow^k v$ only if we have the following:
A. $t_1 \rightarrow^p \lambda x.t_1'$, so by the induction hypothesis, we have $t_1 \Downarrow^p \lambda x.t_1'$.
B. $t_2 \rightarrow^m v_2$, so by the induction hypothesis, we have $t_2 \Downarrow^m v_2$.
C. $[x \rightarrow v_2]t_1' \rightarrow^o v_1$, so by the induction hypothesis, we have $[x \rightarrow v_2]t_1' \Downarrow^o v_1$.
So we can conclude that if $t \rightarrow^k v$, then $t \Downarrow^k v$.

ii. Case B-SUCC:
$t = succ\, t_1$ and $v = succ\, nv_1$
The rule says that $t \rightarrow^k v$ only if we have $t_1 \rightarrow^{k-1} nv_1$. By the induction hypothesis, having $t_1 \rightarrow^{k-1} nv_1$ then we have $t_1 \Downarrow^{k-1} nv_1$, so we can conclude that if $t \rightarrow^k v$, then $t \Downarrow^k v$.

iii. Case B-PREDZERO:
$t = pred\, t_1$ and $v = 0$
The rule says that $t \rightarrow^k v$ only if we have $t_1 \rightarrow^{k-1} 0$. By the induction hypothesis, having $t_1 \rightarrow^{k-1} 0$ then we have $t_1 \Downarrow^{k-1} 0$, so we can conclude that if $t \rightarrow^k v$, then $t \Downarrow^k v$.

iv. Case B-PREDSUCC:
$t = pred\, t_1$ and $v = nv_1$
The rule says that $t \rightarrow^k v$ only if we have $t_1 \rightarrow^{k-1} succ\, nv_1$. By the induction hypothesis, having $t_1 \rightarrow^{k-1} succ\, nv_1$ then we have $t_1 \Downarrow^{k-1} succ\, nv_1$, so we can conclude that if $t \rightarrow^k v$, then $t \Downarrow^k v$.

2 Godel's system T

Assuming all the problem context given for the questions in group 2, we have the following as the answer for each question:

A. Example of reduction

$$\dfrac{\mathtt{t} \rightarrow \mathtt{t}' \qquad \mathtt{t}' \rightarrow^* \mathtt{t}''}{\mathtt{t} \rightarrow^* \mathtt{t}''}\ \text{E-TRANSITIVITY} \qquad\qquad \dfrac{}{\mathtt{t} \rightarrow^* \mathtt{t}}\ \text{E-REFLEXIVITY}$$

For the sake of space and readability, it will only be expressed the $t \rightarrow t'$ derivation directly, in multiple one-step transitions, correspondent to what the derivation tree using E-TRANSITIVITY and E-REFLEXIVITY rules would look like.

$$(\lambda \mathtt{n} : \mathsf{Nat}.\mathtt{rec}(0; \mathtt{x}.\mathtt{y}.(\mathtt{succ}\ \mathtt{x}) + \mathtt{y})(\mathtt{n}))(\mathtt{succ}(\mathtt{succ}(\mathtt{succ}\ 0))) \rightarrow$$

$$\mathtt{rec}(0; \mathtt{x}.\mathtt{y}.(\mathtt{succ}\ \mathtt{x}) + \mathtt{y})(\mathtt{succ}(\mathtt{succ}(\mathtt{succ}\ 0))) \rightarrow$$

$$(\texttt{succ}(\texttt{succ}(\texttt{succ } 0))) + (\texttt{rec}(0; \texttt{x.y.}(\texttt{succ x}) + \texttt{y})(\texttt{succ}(\texttt{succ } 0))) \to$$

$$(\texttt{succ}(\texttt{succ}(\texttt{succ } 0))) + (\texttt{succ}(\texttt{succ } 0)) + (\texttt{rec}(0; \texttt{x.y.}(\texttt{succ x}) + \texttt{y})(\texttt{succ } 0)) \to$$

$$(\texttt{succ}(\texttt{succ}(\texttt{succ } 0))) + (\texttt{succ}(\texttt{succ } 0)) + (\texttt{succ } 0) + (\texttt{rec}(0; \texttt{x.y.}(\texttt{succ x}) + \texttt{y})(0)) \to$$

$$(\texttt{succ}(\texttt{succ}(\texttt{succ } 0))) + (\texttt{succ}(\texttt{succ } 0)) + (\texttt{succ } 0) + 0 \to$$

$$(\texttt{succ}(\texttt{succ}(\texttt{succ}(\texttt{succ}(\texttt{succ}(\texttt{succ } 0))))))$$

B. A definition for +

$\texttt{sum} = \lambda \texttt{n}.\lambda \texttt{m}.\texttt{rec}(\texttt{m}; \texttt{x.y.succ y})(\texttt{n})$

C. Typing the recursor

$$\frac{\Gamma \vdash \texttt{t}_0 : \mathsf{T} \qquad \Gamma, \texttt{x} : \mathsf{Nat}, \texttt{y} : \mathsf{T} \vdash \texttt{t}_1 : \mathsf{T} \qquad \Gamma \vdash \texttt{t} : \mathsf{Nat}}{\Gamma \vdash \texttt{rec}(\texttt{t}_0; \texttt{x.y.t}_1)(\texttt{t}) : \mathsf{T}} \text{ T-REC}$$

D. A typing derivation

Assuming the existence of the arithmetic expressions typing rules for numbers (e.g. T-ZERO) and of the simply typed lambda-calculus typing rules (e.g. T-VAR and T-ABS).

$$\dfrac{\dfrac{\dfrac{}{\texttt{n} : \mathsf{Nat} \vdash 0 : \mathsf{Nat}} \text{ T-ZERO} \qquad \texttt{x} : \mathsf{Nat}, \texttt{y} : \mathsf{Nat}, \texttt{n} : \mathsf{Nat} \vdash (\texttt{succ x}) + \texttt{y} : \mathsf{Nat} \qquad \dfrac{\texttt{n} : \mathsf{Nat} \in \texttt{n} : \mathsf{Nat}}{\texttt{n} : \mathsf{Nat} \vdash \texttt{n} : \mathsf{Nat}} \text{ T-VAR}}{\texttt{n} : \mathsf{Nat} \vdash \texttt{rec}(0; \texttt{x.y.}(\texttt{succ x}) + \texttt{y})(\texttt{n}) : \mathsf{Nat}} \text{ T-REC}}{\vdash \lambda \texttt{n} : \mathsf{Nat}.\texttt{rec}(0; \texttt{x.y.}(\texttt{succ x}) + \texttt{y})(\texttt{n}) : \mathsf{Nat} \to \mathsf{Nat}} \text{ T-ABS}$$

A term t is typable (or well typed) if there is some T such that t : T. With this, it is observable, that for the term S there is a T such that S: T, because, as the derivation shows, S: Nat $\to$ Nat, in other words, T= Nat $\to$ Nat.

E. Progress

[Proof] By induction on a derivation of $t : T$. The T-ZERO and T-ABS cases are immediate, since t in these cases is a value. For the other cases, we argue as follows.

1. Case T-SUCC:
   $\texttt{t} = \texttt{succ } \texttt{t}_1$ and $\texttt{t}_1 : \mathsf{Nat}$
   By induction hypothesis, either t_1 is a value or else there is some t_1' such that $\texttt{t}_1 \to \texttt{t}_1'$. If t_1 is a value, then the canonical forms lemma assures us that it must be a numeric value, in which case so is t. On the other hand, if $\texttt{t}_1 \to \texttt{t}_1'$, then, by E-SUCC, $\texttt{succ } \texttt{t}_1 \to \texttt{succ } \texttt{t}_1'$.

2. Case T-VAR:
   This case cannot occur, because t is closed.

3. Case T-APP:
   Using the Inversion of Typing Relation, we know that $t = t_1 t_2$, $\Gamma \vdash t_1 : T_{11} \to T_{12}$ and $\Gamma \vdash t_1 : T_{11}$. By induction hypothesis, either t_1 is a value or else there is some t_1' such that $\texttt{t}_1 \to \texttt{t}_1'$. By induction hypothesis, either t_2 is a value or else there is some t_2' such that $\texttt{t}_2 \to \texttt{t}_2'$. With that said, we have the following:

   (a) If $\texttt{t}_1 \to \texttt{t}_1'$, then apply E-APP1.

   (b) If t_1 is a value and $\texttt{t}_2 \to \texttt{t}_2'$, then apply E-APP2.

   (c) If t_1 is a value and t_2 is also a value, we know that $\vdash t_1 : T_{11} \to T_{12}$ and the canonical form tells that $t_1 = \lambda x.t_1$, then we apply E-APPABS.

4. Case T-REC:
   $\mathtt{t} = \mathtt{rec}(\mathtt{t}_0; \mathtt{x.y.t}_1)(\mathtt{t}_2),\ \Gamma \vdash \mathtt{t}_0 : \mathsf{T},\ \Gamma, x\colon \mathsf{Nat}, y\colon \mathsf{T} \vdash \mathtt{t}_1 : \mathtt{T}$ and $\Gamma \vdash \mathtt{t}_2 : \mathsf{Nat}$
   By induction hypothesis, either t_2 is a value or else there is some t_2' such that $\mathtt{t}_2 \to \mathtt{t}_2'$. If t_2 is a value, then the canonical forms lemma assures us that it must be a numeric value, in other words, either 0 or $succ\ nv$, and one of the rules E-REC-Z or E-REC-S applies to t. On the other hand, if $\mathtt{t}_2 \to \mathtt{t}_2'$, then, by E-REC-A, $\mathtt{rec}(\mathtt{t}_0; \mathtt{x.y.t}_1)(\mathtt{t}_2) \to \mathtt{rec}(\mathtt{t}_0; \mathtt{x.y.t}_1)(\mathtt{t}_2')$.

F. Preservation

[Proof] By induction on a derivation of $t : T$. At each step of the induction, we assume that the desired property holds for all subderivations (i.e., that if $\mathtt{s} : \mathsf{S}$ and $\mathtt{s} \to \mathtt{s}'$, then $\mathtt{s}' : \mathsf{S}$ whenever $\mathtt{s} : \mathsf{S}$ is proved by a subderivation of the present one) then and proceed by case analysis on the final rule in the derivation.

1. Case T-ZERO:
   $\mathtt{t} = 0$ and $\mathtt{T} = \mathsf{Nat}$
   If the last rule in the derivation is T-ZERO, then we know from the form of this rule that t must be a value and $\mathtt{T}$ must be : Nat. But if t is a value, then it cannot be the case that $\mathtt{t} \to \mathtt{t}'$ for any t', and the requirements of the theorem are vacuously satisfied.

2. Case T-SUCC:
   $\mathtt{t} = \mathtt{succ}\ \mathtt{t}_1, \mathtt{T} = \mathsf{Nat}$ and $\mathtt{t}_1 : \mathsf{Nat}$
   By inspecting the evaluation rules, we see that there is just one rule, E-SUCC, that can be used to derive $\mathtt{t} \to \mathtt{t}'$. The form of this rule tells us that $\mathtt{t}_1 \to \mathtt{t}_1'$. Since we also know $\mathtt{t}_1 : \mathsf{Nat}$, we can apply the induction hypothesis to obtain $\mathtt{t}_1' : \mathsf{Nat}$, from which we obtain $\mathtt{succ}(\mathtt{t}_1') : \mathsf{Nat}$, i.e., $\mathtt{t}' : \mathtt{T}$, by applying rule T-SUCC.

3. Case T-VAR:
   $\mathtt{t} = \mathtt{x}$ and $\mathtt{x} : \mathsf{T}$
   If the last rule in the derivation is T-VAR, then we know from the form of this rule that t must be a value and $\mathtt{T}$ must be : T. But if t is a value, then it cannot be the case that $\mathtt{t} \to \mathtt{t}'$ for any t', and the requirements of the theorem are vacuously satisfied.

4. Case T-ABS:
   $\mathtt{t} = \lambda\mathtt{x} : \mathsf{T}_1.\mathtt{t}_2,\ \mathtt{T} = \mathsf{T}_1 \to \mathsf{T}_2$ and $\mathtt{t}_2 : \mathsf{T}_2$
   If the last rule in the derivation is T-ABS, then we know from the form of this rule that t must be a value and $\mathtt{T}$ must be : $\mathsf{T}_1 \to \mathsf{T}_2$. But if t is a value, then it cannot be the case that $\mathtt{t} \to \mathtt{t}'$ for any t', and the requirements of the theorem are vacuously satisfied.

5. Case T-APP:
   If the last rule in the derivation is T-APP, then, using the Inversion of Typing Relation, we know that $t = t_1 t_2,\ \mathsf{T} = \mathsf{T}_{12},\ \Gamma \vdash t_1 : T_{11} \to T_{12}$ and $\Gamma \vdash t_2 : T_{11}$. By inspecting the evaluation rules, we find that there are three rules, E-APP1, E-APP2 and E-APPABS, that can be used to derive $\mathtt{t} \to \mathtt{t}'$.

   (a) E-APP1: $\mathtt{t}_1 \to \mathtt{t}_1',\ \mathtt{t}' = \mathtt{t}_1'\mathtt{t}_2$
   Since we know $\Gamma \vdash t_1 : \mathsf{T}_{11} \to\ \mathsf{T}_{12}$, we can apply the induction hypothesis to obtain $\Gamma \vdash t_1' : \mathsf{T}_{11} \to\ \mathsf{T}_{12}$, from which we obtain $\Gamma \vdash \mathtt{t}_1'\mathtt{t}_2 : \mathsf{T}$, i.e., $\mathtt{t}' : \mathsf{T}$, by applying rule T-APP.

   (b) E-APP2: $\mathtt{t}_2 \to \mathtt{t}_2',\ \mathtt{t}' = \mathtt{v}_1\mathtt{t}_2'$
   Since we know $\Gamma \vdash t_2 : \mathsf{T}_{11}$, we can apply the induction hypothesis to obtain $\Gamma \vdash t_2' : \mathsf{T}_{11}$, from which we obtain $\Gamma \vdash \mathtt{v}_1\mathtt{t}_2' : \mathsf{T}$, i.e., $\mathtt{t}' : \mathsf{T}$, by applying rule T-APP.

   (c) E-APPABS: $\mathtt{t}' = \Gamma \vdash [\mathtt{x} \to \mathtt{t}_2]\mathtt{t}_1'$
   We know that $\vdash t_1 : \mathsf{T}_{11} \to \mathsf{T}_{12}$ and the canonical form tells that $t_1 = \lambda x.t_1'$
   $\Gamma, \mathtt{x} : \mathsf{T}_{11} \vdash \mathtt{t}_1' : \mathsf{T}_{12}$
   t_2 is a value $\Gamma \vdash \mathtt{t}_2 : \mathsf{T}_{11}$
   We obtain $\Gamma \vdash [x \to t_2]t_1'$: T, i.e., $\mathtt{t}' : \mathsf{T}$, by applying the Preservation of Types Under Substitution, in which its result preserves the type T.

6. Case T-REC:
   $\mathtt{t} = \mathtt{rec}(\mathtt{t}_0; \mathtt{x.y.t}_1)(\mathtt{t}_2),\ \Gamma \vdash \mathtt{t}_0 : \mathsf{T},\ \Gamma, x{:}\ \mathsf{Nat}, y{:}\ \mathsf{T} \vdash \mathtt{t}_1 : \mathsf{T}$ and $\Gamma \vdash \mathtt{t}_2 : \mathsf{Nat}$
   By inspecting the evaluation rules, we see that there are three rules, E-REC-A, E-REC-Z and E-REC-S, that can be used to derive $\mathtt{t} \to \mathtt{t}'$.

   (a) E-REC-A: $\mathtt{t}_2 \to \mathtt{t}_2',\ \mathtt{t}' = \mathtt{rec}(\mathtt{t}_0; \mathtt{x.y.t}_1)(\mathtt{t}_2')$
   Since we know $\Gamma \vdash t_2 : \mathsf{Nat}$, we can apply the induction hypothesis to obtain $\Gamma \vdash t_2' : \mathsf{Nat}$, from which we obtain $\mathtt{rec}(\mathtt{t}_0; \mathtt{x.y.t}_1)(\mathtt{t}_2') : \mathsf{T}$, i.e., $\mathtt{t}' : \mathsf{T}$, by applying rule T-REC.

   (b) E-REC-Z: $\mathtt{t}_2 = 0,\ \mathtt{t}' = \mathtt{t}_0$
   If $t \to t'$ is derived using E-REC-Z, then from the form of this rule we see that t_2 must be 0 and the resulting term t' is t_0. This mean we are finished, since we know (by de assumptions of the T-REC case) that t_0: T, which is what we need.

   (c) E-REC-S: $\mathtt{t}_2 = \mathtt{succ\,nv},\ \mathtt{t}' = \Gamma \vdash [\mathtt{x} \to \mathtt{nv}][\mathtt{y} \to \mathtt{s}]\mathtt{t}_1$
   If $t \to t'$ is derived using E-REC-S, then from the form of this rule we have the following:

      i. First Substitution:
      $\Gamma, \mathtt{x} : \mathsf{Nat} \vdash \mathtt{t}_1 : \mathsf{T}$
      $t_2 = \mathtt{succ\,nv}$ and $\Gamma \vdash \mathtt{t}_2 : \mathsf{Nat}$
      We obtain $\Gamma \vdash [x \to nv]t_1$: T, i.e., $\mathtt{t}' : \mathsf{T}$, by applying the Preservation of Types Under Substitution, in which its result preserves the type T.

      ii. Second Substitution:
      Assuming t' to be the result of the first substitution.
      $\Gamma, \mathtt{y} : \mathsf{T} \vdash \mathtt{t}' : \mathsf{T}$
      $s = rec(t_0; x.y.t_1)(nv)$ and s: T
      We obtain $\Gamma \vdash [y \to s]t'$: T, i.e., $\mathtt{t}'' : \mathsf{T}$, by applying the Preservation of Types Under Substitution, in which its result preserves the type T.