# COMPUTATIONAL HIGHER TYPE THEORY (CHTT)

## ROBERT HARPER

### Lecture Notes of Week 2 by Stephanie Balzer and Yue Niu

## 1   Setting the Scene

Last week, we explored *closed term computation* for the simply typed lambda calculus (STLC) with the goal to prove that all well-typed STLC programs terminate. Our exploration lead us to rediscover Tait's 1967 *hereditary termination* $\mathsf{HT}_A(M)$ of a term $M$ at type $A$ and *hereditary terminating substitution* $\mathsf{HT}_\Gamma(\gamma)$ of closing substitution mapping $\gamma$ at context $\Gamma$ to strengthen our inductive hypothesis. The predicates $\mathsf{HT}_A(M)$ and $\mathsf{HT}_\Gamma(\gamma)$ amount to *behavioral* invariants, indexed by a type and typing context, respectively. Depending on the source, different terms are used in the literature to refer to methods based on such behavioral invariants. Most commonly used is the term *hereditary*. Tait himself used the term *computability*, whereas some people refer to such methods as *Tait's Method*. Statman 1985 introduced the term *logical relation*, which underlines the idea of a behavioral invariant. Moreover, the term *Girard's method* can be found in the literature as well as the term *Reducibility Method*, which is most likely due to Girard himself.

In this week's lectures, we are going to generalize the results of the previous week to *open term computation*. Before doing so, however, we first have to finish our proof of hereditary termination.

## 2   Closed Term Computation Continued

Our main result of last week is the proof or Theorem 1, which states that well-typed terms are hereditarily terminating under substitution by hereditarily terminating maps:

**Theorem 1** (Hereditary termination). *If $\Gamma \vdash M : A$ and $\gamma : \cdot \to \Gamma$ so that $\mathsf{HT}_\Gamma(\gamma)$ then $\mathsf{HT}_A(\hat{\gamma}(M))$.*

Given this result, we should move on to proving that hereditary termination actually implies termination. Before doing so, however, we first have to finish our proof Theorem 1, which relies on a lemma known as *head expansion* or *reverse execution*, which we are going to prove next:

**Lemma 2** (Head expansion). *If $\mathsf{HT}_A(M')$ and $M \mapsto M'$ then $\mathsf{HT}_A(M)$.*

*Proof.* By induction on the structure of $A$.

- $A = b$
  Suppose $\mathsf{HT}_b(M')$ and $M \mapsto M'$. We need to show $\mathsf{HT}_b(M)$. By the definition of hereditary termination, it suffices to show that $M \Downarrow$. This is clear since $M' \Downarrow$ and $M \mapsto M'$.

- $A = A_1 \to A_2$
  Suppose $\mathsf{HT}_{A_1 \to A_2}(M')$ and $M \mapsto M'$. We need to show $\mathsf{HT}_{A_1 \to A_2}(M)$. Now suppose $\mathsf{HT}_{A_1}(N)$, and we need to show $\mathsf{HT}_{A_2}(M\ N)$. By the definition of hereditary termination and by our assumption, we have $\mathsf{HT}_{A_2}(M'\ N)$. Note that $M\ N \mapsto M'\ N$ by our assumption and the compuation rule. Now applying the IH[1], we have $\mathsf{HT}_{A_2}(M\ N)$. □

Let's now return to proving our ultimate goal, namely that hereditary termination implies termination of terms in the STLC:

**Theorem 3** (Termination). *If $\mathsf{HT}_A(M)$ then $M\ \mathsf{term}_\beta$.*

---

[1] Note that the IH can be applied because the induction is on the structure of $A$, not on terms. Whereas the terms in the hereditary termination predicate get bigger, the types get smaller.

***Proof Attempt.***   By induction on the structure of $A$.

- $A = b$

  Suppose $\mathsf{HT}_b(M)$. Done by definition of hereditary termination.

- $A = A_1 \to A_2$

  Suppose $\mathsf{HT}_{A_1 \to A_2}(M)$. We need to show $M$ $\mathsf{term}_\beta$. We are stuck here.

<span style="color:red">X</span>

The question is what we should do at this point. One possibility is to basically "give up" and weaken the theorem to expect termination to hold only for observables, i.e., for terms of the base type. Alternatively, we could strengthen the induction hypothesis even further by changing the defintion of $\mathsf{HT}_A(M)$ to also insist on $M$ $\mathsf{term}_\beta$.

Another option is to change the definition of hereditary termination entirely and use a *positive* formulation rather than a *negative* one. Whereas our current negative formulation is phrased in terms of an implication, the positive formulation requires the term to be an actual lambda. *Positive hereditary termination* $\mathsf{HT}_A^+(M)$ of a term $M$ at type $A$ is defined as follows:

$$\mathsf{HT}_b^+(M) \triangleq M \mapsto^* c \quad (\exists c : b)$$
$$\mathsf{HT}_{A \to B}^+(M) \triangleq M \mapsto^* \lambda x{:}A.\,M_2 \text{ and } \forall M_1.\ \mathsf{HT}_A^+(M_1) \implies \mathsf{HT}_{A_2}^+([M_1/x]M_2)$$

The positive formulation is also known as the *method of canonical forms*. Using a positive formulation of hereditary termination, the proof of the introductory rule is straightforward, whereas the elimination rule takes some work, relying on head expansion, in particular.

We can rephrase the above definition of $\mathsf{HT}_A^+(M)$ by introducing the notion of a *hereditary value* $\mathsf{HV}_A(V)$ of a value $V$ at type $A$

$$\mathsf{HT}_b^+(M) \triangleq M \mapsto^* V \text{ value such that } \mathsf{HV}_A(V)$$

and then give a definition of a hereditary value that differs depending on whether a *call-by-value* (cbv) or *call-by-name* (cbn) semantics is used:

$$\mathsf{HV}_b(V) \triangleq V = c : b$$
$$\mathsf{HV}_{A_1 \to A_2}^{\mathsf{cbv}}(V) \triangleq V = \lambda x{:}A.\,M_2 \text{ and } \forall V_1.\mathsf{HV}_{A_1}^{\mathsf{cbv}}(V_1) \implies \mathsf{HT}_{A_2}([V_1/x]M_2)$$
$$\mathsf{HV}_{A_1 \to A_2}^{\mathsf{cbn}}(V) \triangleq V = \lambda x{:}A.\,M_2 \text{ and } \forall M_1.\mathsf{HT}_{A_1}(M_1) \implies \mathsf{HT}_{A_2}([M_1/x]M_2)$$

In both interpretations a hereditary value at a base type is simply a value, whereas at a function type the insistance on a value as an argument is only made for a call-by value semantics, but not for a call-by-name semantics. From this perspective, the negative formulation of hereditary termination only really makes sense in the context of a call-by-name semantics. Only then, function arguments can be unevaluated computations that permit the weakened termination theorem discussed earlier that insists on termination for base types only. For a call-by-value semantics, on the other hand, a positive formulation of hereditary termination is required for termination, as this formulation is phrased in terms of values.

In this case study of closed term computation we have assumed a call-by-name evaluation semantics. For a call-by-value evaluation semantics, our main fundamental Theorem 1 needs to be phrased in terms of hereditary values, with $\mathsf{HV}_\Gamma^{\mathsf{cbv}}(\gamma)$ correspondingly defined:

**Theorem 4** (Call-by-value hereditary termination). *If* $\Gamma \vdash M : A$ *and* $\gamma : \cdot \to \Gamma$ *so that* $\mathsf{HV}_\Gamma^{\mathsf{cbv}}(\gamma)$ *then* $\mathsf{HT}_A^+(\hat{\gamma}(M))$.

## 2.1   Summary

The introduction of behavioral invariants has allowed us to express the *semantic* property of *syntax*. Specifically, we have shown that well-type terms terminate. To talk about the semantic properties of terms, we introduce the following notation:

$$
\begin{array}{lll}
A \text{ type} & \text{iff} & A, B ::= b \mid A \to B \\
\gamma \in \Gamma & \text{iff} & \mathsf{HT}_\Gamma(\gamma) \\
M \in A & \text{iff} & \mathsf{HT}_A(M) \\
\Gamma \gg M \in A & \text{iff} & \gamma \in \Gamma \implies \hat{\gamma}(M) \in A
\end{array}
$$

Note that the "membership" relation has a computational flavor; it is a behavorial condition on $M$ (or $\gamma$), which says that $M$ (or $\gamma$) satisfies the specification $A$ (or $\Gamma$).

Now, we can state our main fundamental Theorem 1 as follows:

**Theorem 5** (Hereditary termination – semantic formulation)**.** *If* $\Gamma \vdash M : A$*, then* $\Gamma \gg M \in A$*.*

This is in some sense a soundness theorem (in the language of formal logics), which means that the formally derivable terms are actually true (according to the computational specification). Therefore, we can think of formal systems as a way of *accessing* the *truth*. Note that we make no claims that $\Gamma \gg M \in A$ be decidable, as it is fruitless to expect the truth to be decidable in general. [2]

# 3   Open Term Computation

Next, we move on to *open term computation*, for which we would like to prove *normalization*. In this new setting, we interpret the judgment $\Gamma \vdash M : A$ as a mapping on open terms — rather than a mapping on closed terms, as we have done previously. In this setting, we adopt a computational view on variables, treating them as indeterminates.

A term $M$ is *normalizing*, $M \text{ norm}_\beta$, if it has some terminating sequence of beta-reductions. Recall the beta-contraction relation, $P \text{ contr}_\beta P'$:

$$
(\lambda x{:}A.\, M)\, N \text{ contr}_\beta [N/x]M
$$

Relying on beta-contraction, we formulate beta-reduction by the following rules:

$$
\frac{M \text{ contr}_\beta N}{M \mapsto_\beta N}
\qquad
\frac{M \mapsto_\beta M'}{M\, N \mapsto_\beta M'\, N}
\qquad
\frac{N \mapsto_\beta N'}{M\, N \mapsto_\beta M\, N'}
\qquad
\frac{M \mapsto_\beta M'}{\lambda x{:}A.\, M \mapsto_\beta \lambda x{:}A.\, M'}
$$

Further, we define beta-normal form, $N \text{ nf}_\beta$:

$$
N \text{ nf}_\beta \triangleq N \not\mapsto_\beta
$$

Where $N \not\mapsto_\beta$ means that no beta-reduction rule applies to $N$. Finally, we can define when a term is beta-normalizing, $M \text{ norm}_\beta$:

$$
M \text{ norm}_\beta \triangleq \exists N. M \mapsto_\beta^* N \text{ and } N \text{ nf}_\beta
$$

Now we can state the fundamental theorem for normalization:

**Theorem 6** (Normalization)**.** *If* $\Gamma \vdash M : A$ *then* $M \text{ norm}_\beta$*.*

---

[2]See Derek Dreyer's Milner Award Lecture for further discussion.

Attempting to prove Theorem 6 by induction on typing, we will get stuck on proving the case of function application, as we already did when attempting to prove termination last week. Similarly, we introduce a stronger notion for normalization, *hereditary normalization*:

$$\mathsf{HN}_b^\Delta(M) \triangleq M \ \mathrm{norm}_\beta$$
$$\mathsf{HN}_{A \to B}^\Delta(M) \triangleq \forall N. \ \mathsf{HN}_A^\Delta(N) \implies \mathsf{HN}_B^\Delta(M \ N)$$

The above definition relies on *hereditary normalizing substitution*, which generalizes hereditary terminating substitution to open terms:

$$\mathsf{HN}_\Gamma^\Delta(\gamma) \triangleq \forall x : A \in \Gamma. \ \mathsf{HN}_A^\Delta(\gamma(x))$$

As before, $\gamma$ defines the mapping $\gamma : \Delta \to \Gamma$ such that for each $x : A \in \Gamma$ we have $\Delta \vdash \gamma(x) : A$. The superscript $\Delta$ in $\mathsf{HN}_\Gamma^\Delta(\gamma)$ draws attention to the fact that open terms can be substituted for variables, contrasting hereditary terminating substitution $\mathsf{HT}_\Gamma(\gamma)$, where $\Delta$ is empty. Recall that $\hat{\gamma}(M)$, or $\hat{M}$, for short, denotes the application of substitution to the term $M$, which is defined for our language as follows:

$$\hat{\gamma}(c) \triangleq c$$
$$\hat{\gamma}(x) \triangleq \gamma(x)$$
$$\hat{\gamma}(MN) \triangleq \hat{\gamma}(M)\hat{\gamma}(N)$$
$$\hat{\gamma}(\lambda x{:}A.\, M) \triangleq \lambda x{:}A.\, \hat{\gamma}(M) \quad (x \notin \mathsf{dom}\ \gamma)$$

Given the definitions of $\mathsf{HN}_A^\Delta(M)$ and $\mathsf{HN}_\Gamma^\Delta(\gamma)$ we can now state our fundamental theorem of normalization:

**Theorem 7** (Hereditary normalization)**.** *If* $\Gamma \vdash M : A$ *and* $\mathsf{HN}_\Gamma^\Delta(\gamma)$, *then* $\mathsf{HN}_A^\Delta(\hat{\gamma}(M))$.

Our fundamental theorem uses a negative formulation of hereditary normalization, which is the only possible choice in the presence of indeterminates. This raises the question what how to accommodate positive types, such as sums.

To prove normalization for the simply typed lambda calculus, we additionally need the following two lemmas, whose proofs mutually rely on each other:

**Lemma 8.** *If* $\mathsf{HN}_A^\Delta(M)$ *then* $M \ \mathrm{norm}_\beta$.

**Lemma 9.** $\mathsf{HN}_\Gamma^\Gamma(\mathsf{id}(\Gamma))$.

In the following, we develop the proof of our fundamental Theorem 7 and the two lemmas above. As usual, we develop the proof step-by-step, starting with failed attempts that lead us to refine what we have already stated.

### 3.1 Proving Normalization

#### 3.1.1 Proof of Fundamental Theorem

***Proof of Theorem 7.*** By induction on typing.

- $\Gamma', x : A \vdash x : A$
  Suppose $\mathsf{HN}_{\Gamma',x:A}^\Delta(\gamma)$. We need to show $\mathsf{HN}_A^\Delta(\hat{\gamma}(x))$, which follows directly from our assumption.

- $\Gamma \vdash c : b$
  Suppose $\mathsf{HN}_\Gamma^\Delta(\gamma)$. We need to show that $\mathsf{HN}_b^\Delta(c)$, which is to show that $c \ \mathrm{norm}_\beta$. Since $c \ \mathrm{nf}_\beta$, we have $c$ normalizing in 0 steps.

- $$\frac{\Gamma \vdash M : A \to B \qquad \Gamma \vdash N : A}{\Gamma \vdash M \ N : B}$$

  Applying the IH, we have $\mathsf{HN}_{A \to B}^\Delta(\hat{\gamma}(M))$ and $\mathsf{HN}_A^\Delta(\hat{\gamma}(N))$. By definition of hereditary normalization, we have $\mathsf{HN}_B^\Delta(\hat{\gamma}(M) \ \hat{\gamma}(N))$. Since $\hat{\gamma}(M \ N) = \hat{\gamma}(M) \ \hat{\gamma}(N)$, we have $\mathsf{HN}_B^\Delta(\hat{\gamma}(M \ N))$.

- $$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x{:}A.\, M : A \to B}$$

  Suppose $\mathsf{HN}^\Delta_\Gamma(\gamma)$. We need to show that $\mathsf{HN}^\Delta_{A \to B}(\hat{\gamma}(\lambda x{:}A.\, M))$. Thus, suppose $\mathsf{HN}^\Delta_A(N)$. It suffices to show that $\mathsf{HN}^\Delta_B(\hat{\gamma}(\lambda x{:}A.\, M)\ N)$. Notice that with $N$, we can extend $\gamma$ to be $\gamma' = \gamma[x \mapsto N]$, and since $N$ is hereditarily normalizing at $A$, it follows that $\mathsf{HN}^\Delta_{\Gamma, x:A}(\gamma')$. Applying the IH, we have that $\mathsf{HN}^\Delta_B(\hat{\gamma}'(M))$. Since substitution is commutative, $\hat{\gamma}'(M) = [N/x]\hat{\gamma}(M)$, and $\hat{\gamma}(\lambda x{:}A.\, M)\ N = \lambda x{:}A.\, \hat{\gamma}(M)\ N$. Further, $\lambda x{:}A.\, \hat{\gamma}(M)\ N \mapsto [N/x]\hat{\gamma}(M)$ by the computation rule. The result then follows from head expansion. □

### 3.1.2 Attempts at Proving Lemmas 8 and 9

The proof of Lemmas 8 and 9 goes by simultaneous induction. It will turn out that we will need to further strengthen Lemma 9 in order to get the induction go through, but let's see first where we fail.

***Proof Attempt of Lemma 8.*** By induction on the structure of $A$.

- $A = b$
  Immediate from definition of hereditary normalization.

- $A = A_1 \to A_2$
  Suppose $\mathsf{HN}^\Delta_A(M)$. We need to show $M$ $\mathrm{norm}_\beta$. Now we are in a bind. If we can somehow use Lemma 9 to obtain a hereditarily normalizing input of type $A_1$, then we could apply the inductive hypothesis to possibly obtain the result. For now, we will assume that there is a way to create a context $\Gamma'$ such that $x : A_1 \in \Gamma'$ (see Section 3.1.3 for solution). By Lemma 9 we have $\mathsf{HN}^{\Gamma'}_{\Gamma'}(\mathsf{id}(\Gamma'))$, in particular, $\mathsf{HN}^{\Gamma'}_{A_1}(x)$. By the definition of hereditary normalization, we further have that $\mathsf{HN}^{\Gamma'}_{A_2}(M\ x)$. Now applying the IH, we see that $M\ x$ $\mathrm{norm}_\beta$, and by Lemma 12 we have $M$ $\mathrm{norm}_\beta$.

  <span style="color:red">X</span>

Postponing the issues with this proof, we will move on to proving Lemma 9, and resolve everything after presenting the two proofs.

***Proof Attempt of Lemma 9.*** Let $x : A \in \Gamma$ be arbitrary. Proceed by induction on the structure of $A$.

- $A = b$
  We need to show $\mathsf{HN}^\Gamma_b(x)$, which holds since $x$ $\mathrm{nf}_\beta$.

- $A = A_1 \to A_2$
  We need to show $\mathsf{HN}^\Gamma_{A_1 \to A_2}(x)$. Suppose $\mathsf{HN}^\Gamma_{A_1}(M_1)$, and it suffices to show that $\mathsf{HN}^\Gamma_{A_2}(x\ M_1)$. Now it would be nice to apply the IH to complete the proof, which is unfortunately not strong enough.

  <span style="color:red">X</span>

If we knew that a variable of a function type is hereditarily normalizing when applied to a beta-normal term, we could complete the above proof. So, let's refine Lemma 9 with this idea, resulting in Lemma 10:

**Lemma 10.** *For all $k$, if $x : A_1 \to \cdots \to A_k \to A \in \Gamma$ and for each $i \le k$, $\Gamma \vdash M_i : A_i$ and $M_i$ $\mathrm{norm}_\beta$, then $\mathsf{HN}^\Gamma_A(x\ M_1\ \ldots M_k)$.*

Note that we need each argument to be *normalizing* instead of merely *hereditarily normalizing*. Before proving Lemma 10, we first return to the issue raised in the proof of Lemma 8.

### 3.1.3 Resolving Issues with Kripke Semantics

In the proof of Lemma 8, we have assumed that we can create a context $\Gamma'$ out of thin air, which allowed us to supply the argument to the function term. For this to be indeed the case, we need a mechanism to *allocate* a fresh variable. Once we have such a mechanism, however, the question comes up whether hereditary normalization

is *stable* under such context extensions. To guarantee that this is indeed the case, we change the definition of hereditary normalization to make use of the notion of a *Kripke* semantics. If we think of the typing context as a model of the "world", then Kripke semantics suggests that we can stipulate hereditary normalization to hold in all context extensions, or "future worlds". Viewed logically, this would be the admissibility of weakening with respect to hereditary normalization.

Thus for our definition of hereditary normalization, we change the higher order terms to expect "world extensions":

$$\mathsf{HN}^{\Delta}_{A_1 \to A_2}(M) \triangleq \text{ if } \Delta' \geq \Delta \text{ and } \mathsf{HN}^{\Delta'}_{A_1}(M_1) \text{ then } \mathsf{HN}^{\Delta'}_{A_2}(M\ M_1)$$

Note that $\geq$ is a pre-order on contexts $\Delta$, and thus is reflexive and transitive. In general, we get the following *monotonicity* guarantee:

**Remark 11** (Monotonicity)**.** *If $\mathsf{HN}^{\Delta}_{A}(M)$ and $\Delta' \geq \Delta$, then $\mathsf{HN}^{\Delta'}_{A}(M)$.*

We now return to proving our lemmas, accounting for the new definition of hereditary normalization. The proof of our main fundamental theorem remains valid.

### 3.1.4 Proofs of Lemmas 8 and 10

*Proof of Lemma 8.* By induction on the structure of $A$, simultaneously with the proof of Lemma 10.

- $A = b$
  Immediate from definition of hereditary normalization.

- $A = A_1 \to A_2$
  Suppose $\mathsf{HN}^{\Delta}_{A}(M)$. We need to show $M\ \mathrm{norm}_\beta$. Let $\Delta' = \Delta, x : A_1$. By IH on Lemma 10 (with $k = 0$), we have $\mathsf{HN}^{\Delta'}_{A_1}(x)$. By the definition of hereditary termination, we have $\mathsf{HN}^{\Delta'}_{A_2}(M\ x)$. Applying the IH, we have $M\ x\ \mathrm{norm}_\beta$, and by Lemma 12 we have $M\ \mathrm{norm}_\beta$. □

*Proof of Lemma 10.* By induction on the structure of $A$, simultaneously with the proof of Lemma 8.

- $A = b$
  We need to show $x\ M_1\ \ldots M_k\ \mathrm{norm}_\beta$, which holds since each $M_i$ is normalizing, and thus the term does not beta-reduce.

- $A = A_{k+1} \to A_{k+2}$
  We need to show $\mathsf{HN}^{\Gamma}_{A}(x\ M_1\ \ldots M_k)$. Let $\Gamma' \geq \Gamma$ and $\mathsf{HN}^{\Gamma'}_{A_{k+1}}(M_{k+1})$. It suffices to show that $\mathsf{HN}^{\Gamma'}_{A_{k+2}}(x\ M_1\ \ldots M_k\ M_{k+1})$. By IH on Lemma 8, we have that $M_{k+1}\ \mathrm{norm}_\beta$, and we obtain the result by applying the IH with $k + 1$. □

Finally, we have to state and prove Lemma 12, which we relied upon in the proofs of Lemmas 8 and 10:

**Lemma 12.** *If $M\ x\ \mathrm{norm}_\beta$, then $M\ \mathrm{norm}_\beta$.*

*Proof.* Let $M\ x \mapsto^*_\beta N$ and $N\ \mathrm{nf}_\beta$. Proceed with induction on the length of the sequence to beta-normal form.

- $M\ x \mapsto^{(0)}_\beta N$
  Then $M\ x\ \mathrm{nf}_\beta$, and $M\ \mathrm{nf}_\beta$, and $M$ also beta-normalizes in 0 steps.

- $M\ x \mapsto^{(k+1)}_\beta N$
  Proceed by cases on the first step.
  - $\lambda x{:}a.\ M'\ x \mapsto_\beta M'$
    Then $M'$ beta-normalizes in $k$ steps. The same $k$ steps will also normalize $M$ performed under the lambda.

  - $M\ x \mapsto_\beta M'\ x$
    By IH, $M'\ \mathrm{norm}_\beta$, and it follows that $M\ \mathrm{norm}_\beta$.

- $M\ x \mapsto_\beta M'\ N$
  Impossible.

- $\lambda x{:}A.\ M' \mapsto_\beta \lambda x{:}A.\ M''$
  Impossible.

<div align="right">□</div>

## 3.2   Remarks

We conclude these notes with providing a restatement of Lemma 10 using evaluation contexts and a note on strong normalization.

### 3.2.1   Lemma 10 with evaluation context

Recall the definition of evaluation contexts:

$$\mathcal{E} ::= \cdot \mid \mathcal{E}\ M$$

We now further characterize evaluation contexts as mappings between types, as detailed in Chapter 46 of Harper [2016]:

$$\frac{}{\cdot : (\Gamma \triangleright A) \rightsquigarrow (\Gamma \triangleright A)} \qquad \frac{\mathcal{E} : (\Gamma \triangleright A) \rightsquigarrow (\Gamma' \triangleright A_1 \to A_2) \qquad \Gamma \vdash M : A_1}{\mathcal{E}\ M : (\Gamma \triangleright A) \rightsquigarrow (\Gamma' \triangleright A_2)}$$

Where $\mathcal{E} : (\Gamma \triangleright A) \rightsquigarrow (\Gamma' \triangleright A')$ can be read as if $\Gamma \vdash M : A$, then $\Gamma' \vdash \mathcal{E}\{M\} : A'$.

Further, we can define when evalutaion contexts are beta-normalizing:

$$\frac{}{\cdot\ \mathrm{norm}_\beta} \qquad \frac{M\ \mathrm{norm}_\beta \qquad \mathcal{E}\ \mathrm{norm}_\beta}{\mathcal{E}\ M\ \mathrm{norm}_\beta}$$

Now, Lemma 10 can be formulated as follows:

**Lemma 13.** *If $\mathcal{E} : (\Gamma \triangleright C) \rightsquigarrow (\Gamma \triangleright A)$ and $\mathcal{E}\ \mathrm{norm}_\beta$, then $\mathsf{HN}_A^{\Gamma,x:C}(\mathcal{E}\{x\})$*

*Proof.* Proceed by induction on the structure of $A$.

- $A = b$
  Need to show that $\mathsf{HN}_b^\Gamma(x)$, which follows since variables are beta-normal.

- $A = A_1 \to A_2$
  Proceed with nested induction on context typing and normalization.
  - $\mathcal{E} = \cdot$
    Let $\Gamma' \geq \Gamma, x : A$. Suppose $\mathsf{HN}_{A_1}^{\Gamma'}(M)$. It suffices to show $\mathsf{HN}_{A_2}^{\Gamma'}(x\ M)$. Note $M\ \mathrm{norm}_\beta$ from IH on Lemma 8. Now apply the outer IH with $\cdot\ M : (\Gamma \triangleright A) \rightsquigarrow (\Gamma \triangleright A_2)$ to obtain $\mathsf{HN}_A^{\Gamma,x:A}(\cdot\ M\{x\})$, and result follows from monotonicity of hereditary normalization.

  - $\mathcal{E} = \mathcal{E}'\ M$ because $\mathcal{E}' : (\Gamma \triangleright C) \rightsquigarrow (\Gamma \triangleright A' \to A), \Gamma \vdash M : A'$ and $M\ \mathrm{norm}_\beta$, $\mathcal{E}'\ \mathrm{norm}_\beta$
    We need to show that $\mathsf{HN}_A^{\Gamma,x:C}(\mathcal{E}\{x\})$. Let $\Gamma' \geq \Gamma, x : C$, and suppose $\mathsf{HN}_{A_1}^{\Gamma'}(N)$. We need to show $\mathsf{HN}_{A_2}^{\Gamma'}(\mathcal{E}\{x\}\ N)$. Since $\mathcal{E}\{x\}\ N = \mathcal{E}\ N\{x\}$, it suffices to show $\mathsf{HN}_{A_2}^{\Gamma'}(\mathcal{E}\ N\{x\})$. By the definition of context typing, we have $\mathcal{E}\ N : (\Gamma \triangleright C) \rightsquigarrow (\Gamma \triangleright A_2)$. In addition, by IH on Lemma 8, we have $N\ \mathrm{norm}_\beta$, consequently $\mathcal{E}\ N\ \mathrm{norm}_\beta$. Lastly, we obtain the result by applying the outer IH on $\mathcal{E}\ N$.

<div align="right">□</div>

Note that the proof is an induction over the lexicographical ordering on the structures of $(A,\ \mathcal{E} : (\Gamma \rhd C) \rightsquigarrow (\Gamma \rhd C),\ \mathcal{E}\ \mathrm{norm}_\beta)$. In particular, when appealing to the inductive hypothesis, the size of structurese relating to $\mathcal{E}$ increases in some cases, but only when $A$ is correspondingly decreasing. We can recover Lemma 10 by instantiating Lemma 13 with the empty evaluation context.

### 3.2.2    Strong Normalization

In these notes, we have proved normalization for the simply typed lambda calculus. Occasionally, *strong normalization* is shown to hold for a calculus, which guarantees that there are no infinite reduction sequences. The value of asserting strong normalization for a calculus is that it enables reasoning by transfinite induction on reductions.

## References

Robert Harper. *Practical Foundations for Programming Languages*. Cambridge University Press, 2nd edition, 2016.

Richard Statman. Logical relations and the typed $\lambda$-calculus. *Information and Control*, 65(2/3):85–97, 1985.

William W. Tait. Intensional Interpretations of Functionals of Finite Type I. *Journal of Symbolic Logic*, 1967.