CT-λ, a cyclic simply typed λ-calculus with fixed points

Stefano Berardi

Abstract

...

1 Definition of CT-λ

We define pseudo-terms of CT-λ: the well-formed terms of CT-λ will be a proper subset.

Definition 1.1 (pseudo-terms CT-λ)

1. *The types of* CT-λ *are all types we can define with* $\mathbf{N}$ *and* $\to$.
2. *The pseudo-terms of* CT-λ *are all possibly infinite trees we can define with* x^T *(*T *type),* $\lambda x^T.t$, $\mathtt{ap}(t,u)$, 0, $S(t)$ *(successor of* t*),* $\mathtt{cond}x.(f,g)$, $\mathtt{name\text{-}in}(f,x)$ *with* $x \notin \mathrm{FV}(f)$.
3. *The scope of the binder* $\lambda x^T.t$ *is* t. *The scope of the binder* $\mathtt{cond}x.(f,g)$ *is* g *(*f *is not in the scope of* $\mathtt{cond}x.(f,g)$*).*

When $t = S^n(0)$ *we say that* t *is a numeral.*

As an example we have the term $t = \mathtt{cond}x.(0,(\lambda y.t)(0))$. We say that a tree is regular if it has finitely many subtrees. t is an infinite regular tree, with subtrees t, 0, $(\lambda y.t)(0)$, $\lambda y.t$.

The typing rules are the usual ones but for `cond` and `name-in`.

Assume Γ, Δ are sequents of length n, $m$respectively. Suppose $f : \{1,\dots,n\} \to \Delta$ is any injection.

1. Structural rule $\mathtt{struct}(f)$. If $t : \Gamma \vdash T$ then $t : \Delta \vdash T$
2. If $t : \Gamma \vdash T$ and Γ' is a renaming of Γ then $t : \Gamma' \vdash T$
3. If $x \notin \Gamma$ and $f : \Gamma \vdash \mathbf{N} \to T$ then $f(x) : \Gamma, x : \mathbf{N} \vdash T$.
4. If $x : A \in \Gamma$ then $x : \Gamma \vdash A$.
5. If $\Gamma \vdash f : A \to B$ and $a : \Gamma \vdash A$ then $f(a) : \Gamma \vdash B$.
6. If $\Gamma, x : A \vdash b : B$ then $\lambda x : A.b : \Gamma \vdash A \to B$.
7. $0 : \Gamma \vdash \mathbf{N}$
8. If $t : \Gamma \vdash \mathbf{N}$ then $S(t) : \Gamma \vdash \mathbf{N}$.
9. `cond`. If $f : \Gamma, x : T \vdash T$, $x \notin \mathrm{FV}(f)$ and $g : \Gamma, x : T \vdash T$ then $\mathtt{cond}x.(f,g) : \Gamma \vdash \mathbf{N} \to T$.

`name-in` is a particular case of `ap`, when we replace $x : \mathbf{N}$ with some $a : \Gamma \vdash \mathbf{N}$ then we replace `name-in` with `ap`.

As an example if $t = \mathtt{cond}x.(0,(\lambda x.t)(0))$ is as above, then $t : \mathbf{N} \to \mathbf{N}$.

Our goal is to provide a set of well-formed term for CT-λ and interpret them as total functionals. Our first step is to provide reduction rules for CT-λ.

Definition 1.2 (reduction rules for CT-λ)

1. β*:* $(\lambda x^A.b)(a) \mapsto_\beta b[a/x]$ *and* $\mathtt{cond}x.(f,g)(0) \mapsto_{\mathtt{cond}} f$, $\mathtt{cond}x.(f,g)(S(t)) \mapsto_{\mathtt{cond}} g[t/x]$.
2. $\mapsto$ *is the context and transitive closure of* $\mapsto_\beta$ *and* $\mapsto_{\mathtt{cond}}$*, in which we forbid reductions in proper subterms of any* $\mathtt{cond}x.(f,g)$.

When $t \mapsto u$ *we say that* t *safely reduces to* u. *We call unsafe a reduction inside any* $\mathtt{cond}x.(f,g)$.

As an example if $t = \texttt{cond}x.(0, (\lambda x.t)(0))$ is as above, then t is normal, because all redexes in t are of the form $(\lambda x.t)(0)$ and inside a `cond`.

The reason for forbiding reductions inside $\texttt{cond}x.(f, g)$ is that thorugh `cond` we will-express fixed-point equations. Therefore reductions for `cond` can easily loop and they are "unsafe". For this reason, we allow the minimum possible reductions concerning $\texttt{cond}x.(f, g)$, those of the form $\texttt{cond}x.(f, g)(0) \mapsto_{\texttt{cond}} f$ and $\texttt{cond}x.(f, g)(S(t)) \mapsto_{\texttt{cond}} g[t/x]$ for maximal `cond`-expression, and no reduction inside the arguments f, g of `cond`.

As an example, if n is any numeral, then $t(n) = \texttt{cond}x.(0, (\lambda x.t)(0))(2) : \mathbf{N}$ has infinitely many β-redexes inside `cond`, therefore infinitely many unsafe reductions are possible. There are only finitely many safe reduction from $t(n)$ instead: $t(n)$ reduces to to $t(n-1)$ for n times, until we get $t(0)$, then 0 and we stop.

Indeed, we have $t(0) = \texttt{cond}x.(0, (\lambda x.t)(0))(0) = 0$, therefore $t(1) = \texttt{cond}x.(0, (\lambda x.t)(0))(1) = (\lambda x.t)(0)(0) = t(0) = 0$, then $t(2) = \texttt{cond}x.(0, (\lambda x.t)(0))(2) = (\lambda x.t)(0)(1) = t(1) = 0$, and so forth.

We define the well-formed terms of CT-λ: for them we will prove strong normalization, church-rosser for terms of type $\mathbf{N}$, and the fact that every term of type $\mathbf{N}$ is a numeral. As a consequence, well-formed terms CT-λ will be interpreted as total functionals.

The first step in defining well-formed terms is defining a correspondence between atoms in the proof that t is well-typed. We need first the notion of list of types and atomic types for a term.

Definition 1.3 (Integer maps) *1. The type list of t is $\vec{C} = \vec{A}, \vec{B}$.*

2. An atom index of t is any $j \in \{1, \dots, n+m\}$ such that $C_j = \mathbf{N}$.

3. We define $\texttt{ins}(a, x) = x + 1$ if $x \geq a + 1$, and $\texttt{ins}(a, x) = x$ if $x \geq a$.

We now define the atom correspondence in CT-λ.

Definition 1.4 (Atom correspondence in CT-λ) *Assume $\vec{A} = A_1, \dots, A_n$, $\vec{B} = B_1, \dots, B_m$ and $t : \vec{x} : \vec{A} \vdash \vec{B} \to \mathbf{N}$ is a pseudo-term.*

1. *The type list of t is $\vec{C} = \vec{A}, \vec{B}$. An atom index of t is any $j \in \{1, \dots, n+m\}$ such that $C_j = \mathbf{N}$.*
2. *For each atom index k in t, each immediate subterms t' of t each atom index k' in t' we define the relation: "k', t' the successor of k, t". We require:*
   (1) if t is obtained by a rule $\texttt{struct}(f)$ then $k = f(k')$ if $k' \leq n$ and $k = k' - n + m$ if $k \geq n + 1$.
   (2) $k' = \texttt{ins}(n, k)$ if $t = f(a)$, $a : \mathbf{N}$ and $t' = f$.
   (3) $k = k'$ in all other cases.

From the atom correspondence we define the global trace condition and the notion of well-formed term.

Definition 1.5 (Global trace condition and well-formed terms in CT-λ) *1. A path π in t is any list $t_0, \dots, t_n$ of subterms of t such that $t_0 = t$ and each t_{i+1} is an immediate subterm of t_i.*

2. *Assume $\pi = t_0, \dots, t_n$ is a branch of t. A trace of π consists of an integer list $k_m, \dots, k_n$ such that for each $i = m, \dots, n$: k_i is an atom index of t_i, and if $i + 1 \leq n$ then k_{i+1}, t_{i+1} is the successor of k_i, t_i.*
3. *Assume $\tau = k_m, \dots, k_n$ is a trace of $\pi = t_0, \dots, t_n$ and $i = m, \dots, n$. τ is progressing in i if $t_i = \texttt{cond}x.(f, g)$ for some f, g, and k_i is the index of the first argument the `cond`-rule, otherwise τ is not progressing in i.*
4. *t satisfies the global trace condition if for all infinite paths π in t there is some infinitely progressing path τ in π.*
5. *Well-formed terms are all terms which are regular trees (having finitely many subtrees), and satisfying the global trace condition.*

2 Examples of terms of CT-λ

2.1 The sum map

As a first example of term of CT-λ we can provide a well-formed term computing the sum on $\mathbf{N}$.
. . . . . .

2.2 The iterator

A second example. We define a term $\mathtt{It}$ of CT-λ computing the iteration of maps on $\mathbf{N}$. We define a normal term $\mathtt{It} : (\mathbf{N} \to \mathbf{N}), \mathbf{N}, \mathbf{N} \to \mathbf{N}$ such that $\mathtt{It}(f, n, a) = f^n(a)$ for all $n \in \mathbf{N}$. We have to to solve the equations $\mathtt{It}(f, a, 0) = a$ and $\mathtt{It}(f, a, S(t)) = f(\mathtt{It}(f, a, t))$. We solve them with $\mathtt{It} = \lambda f, a, x.i[a, f, x]$ with $i[a, f, x] = (\mathtt{cond}x.(a, f(i[a, f, x])))(x)$.

The term is well-typed and regular by definition. We check the global trace condition. We mark the last unnamed argument $\mathbf{N}$ of $\mathtt{It}$. The mark moves to x in $i[a, f, x]$. Through a $\mathtt{name\text{-}in}$-rule the mark moves to the unique unnamed argument $\mathbf{N}$ of $\mathtt{cond}x.(a, f(i[a, f, x]))$. The mark either moves to the name x in the context of a and there it stops, or it progresses and moves to x in the context of $f(i[f, a, x])$, then in the context of $i[f, a, x] : \mathbf{N}$. Now we loop: if the path continues forever, then after infinitely many step the mark from which we started progresses infinitely many times, each time it crosses a $\mathtt{cond}$-rule.

2.3 The Interval Map

A third example. We simulate interval with two variables $\mathtt{nil} : \alpha$ and $\mathtt{cons} : \mathbf{N}, \alpha \to \alpha$. We recursively define a notation for maps by $[] = \mathtt{nil}$ and $[a, \vec{a}] = \mathtt{cons}(a, [\vec{a}])$. We add no elimination rules for lists, though, only the variables $\mathtt{nil}$ and $\mathtt{cons}$.

We will define a term $\mathtt{I}$ with one argument $f : \mathbf{N} \to \mathbf{N}$) and three argument $a, x, y : \mathbf{N}$, such that

$$\mathtt{I}(f, a, n, m) = [f^n(a), f^{n+1}(a), \dots, f^{n+m}(a)]$$

for all $n, m \in \mathbf{N}$. We have to to solve the recursive equations

$$\mathtt{I}(f, a, x, 0) = \mathtt{cons}(f^n(a), \mathtt{nil}) \qquad \mathtt{I}(f, a, x, S(t)) = \mathtt{cons}(f^n(a), \mathtt{I}(f, a, x, t))$$

We solve them with $\mathtt{I} = \lambda f, a.v$, where $v : \mathbf{N}, \mathbf{N} \to \mathbf{N}$, $v = \lambda x.\mathtt{cond}y.(\ \mathtt{cons}(w, \mathtt{nil}),\ \mathtt{cons}(w, v(S(x), y))\ )$ and $w = \mathtt{It}(f, a, x)$.

The term is well-typed and regular by definition. We check the global trace condition. We mark the last unnamed argument $\mathbf{N}$ of $\mathtt{I}$. The mark moves to the last unnamed argument $\mathbf{N}$ of $v : \mathbf{N}, \mathbf{N} \to \mathbf{N}$. We unfold v to $\lambda x.\mathtt{cond}y.(\ \mathtt{cons}(w, \mathtt{nil}),\ \mathtt{cons}(w, v(S(x), y))\ )$ The mark progresses and moves to the name y in the context of the body of the λ-abstraction, then to y in the context of $\mathtt{cons}(w, \mathtt{nil})$ or of $\mathtt{cons}(w, v(S(x), y))$, then in the context of $\mathtt{nil}$ and stops, or in the context of $w = \mathtt{It}(f, a, x) : \mathbf{N}$, for which we already checked the global trace condition, or in the context of $v(S(x), y)) : \alpha$.

Then the mark moves from the named argument y of $v(S(x)) : \mathbf{N} \to \mathbf{N}$ to the unique unnamed argument $\mathbf{N}$ of $v(S(x))$, and eventually to the last argument of $v : \mathbf{N}, \mathbf{N} \to \mathbf{N}$. From v we loop: after infinitely many step either we reached some w and we find some infinite progressing trace inside it, or the mark from which we started progresses infinitely many times through v. In both cases we have the global trace condition.

We have infinitely many nested β-reduction $(\lambda x. \dots)(S(x))$. We can remove all of them in a single step. Inside the β-redex number n we obtain a sub-term $w[S(x)/x] \dots [S(x)/x]$ (substitution repeated n times). The result is $w[S^n(x)] = \mathtt{It}(f, a, S^n(x))$. The nested substitution produce new β-reductions $\mathtt{It}(f, a, S^n(x)) = (\lambda x.i[f, a, x])(S^n(x))$ for all $n \in \mathbf{N}$. This is a non-regular term: we have infinitely many pairwise different sub-terms $x, S(x), S(S(x)), S(S(S(x))), \dots$. We need infinitely many steps to normalize all $\mathtt{It}(f, a, S^n(x))$ to $f^n(i[f, a, x])$, even if we allow to reduce all $\beta, \mathtt{cond}$-redexes at the same time. Also the normal form is not regular: it contains all terms $f^n(i[f, a, x])$ for $n \in \mathbf{N}$, hence infinitely many pairwise different terms. These infinite sub-terms are of a particulary simple form, though. They are obtained by the repeating n times the assignment $z := f(z)$, then applying $z := i[f, a, x]$ once to the result.

Apparently, $\mathtt{I}$ is some term of CT-λ which cannot be normalized in finite time, not even if we allow infinite parallel reductions without any "safety" restriction. The normal form is produced *only in the*

limit and it is *not regular*. If we allow to reduce infinitely many nested β-redexes in one step, also the intermediate steps of the infinite reduction of `I` are not regular.

3 Weak normalization for closed terms of`CT-`λ of type N

In this section we prove that every closed term of`CT-`λ (that is, well-typed, regular and with the global trace condition) normalizes with finitely many "safe" steps to some numeral $n \in \mathbf{N}$.

. . . . . .

4 Uniqueness of normal form for closed terms of `CT-`λ of type N

In this section we prove a weak form of confluence: normal form for all closed terms of `CT-`λ of type $\mathbf{N}$ is unique.

By the weak normalization result proved in **S**3, we will deduce: for all for all closed terms t of `CT-`λ of type $\mathbf{N}$, there is some $n \in \mathbf{N}$ such that there is a safe reduction $t \mapsto n$, and all normal form of t are equal to n.

We define a notion $\sim$ of equivalence for well-typed terms with the same type. Assume $t, u : A$ are *closed* terms of `CT-`λ. (that is, well-typed, regular and with the global trace condition).

We define an equivalence $\sim_0$ for closed terms, by induction on the type A of t, u.

1. Assume $A = \mathbf{N}$. Then $t \sim_0 u$ if and only if for all normal $t', u' : \mathbf{N}$, if $t \mapsto t'$ and $u \mapsto u'$ then $t' = u'$.
2. Assume $A = B \to C$. Then $t \sim_0 u$ if and only if for all closed $b, c : B$ if $b \sim_0 c$ then $t(b) \sim_0 u(c)$.

We will prove that for all closed terms t of `CT-`λ we have $t \sim_0 t$. If $A = \mathbf{N}$ this means that the normal form of all closed terms of `CT-`λ of type $\mathbf{N}$ is unique, which is our goal.

We first define $t \sim u$ for *any* terms of `CT-`λ with the same type. $t \sim u$ if and only if for any two substitution $\sigma = [\vec{t}/\vec{x}]$ and $\tau = [\vec{u}/\vec{x}]$, if $\vec{t} \sim_0 \vec{u}$ and $\sigma(t)$, $\sigma(u)$ are closed then $\sigma(t) \sim_0 \sigma(u)$.

We will prove that for all terms t of `CT-`λ we have $t \sim t$. In the case of closed terms t we can choose $\sigma = \tau =$ the empty sbustitution and we obtain $t \sim_0 t$, which is our goal.

Assume that $t : \mathbf{N}$ is a closed term t of `CT-`λ and $n \in \mathbf{N}$.

1. We say that n is a value of t if $t \mapsto n$ (if t safely reduces to n).
2. We say that t is confluent if $t \sim_0 t$.

Assume $\vec{x} : \vec{A}$ and $t[\vec{x}] : \vec{D} \to \mathbf{N}$. Assume $\vec{a} : \vec{A}, \vec{d} : \vec{D}$ and $\vec{b} : \vec{A}, \vec{e} : \vec{D}$ are vectors of closed terms of `CT-`λ. We say that these four vectors are a counterexample to t if $\vec{a}, \vec{d} \sim_0 \vec{b}, \vec{e}$ and $\neg(t[\vec{a}](\vec{d}) \sim_0 t[\vec{b}](\vec{e}))$.

We have $t \sim t$ if and only if there is no counter-example for t.

By the weak normalization result proved in **S**3, all closed term t of `CT-`λ of type $\mathbf{N}$ have a value. If t is confluent, then the value is unique. For confluent closed term t of `CT-`λ of type $\mathbf{N}$, "safe" reduction preserves the value. Indeed, if $t \mapsto u$ then u is a closed term of `CT-`λ of type $\mathbf{N}$, therefore u has a value m, which is also a value of t. By confluence of t we conclude that $n = m$.

If t is as above, with value n, and $t \mapsto S(u)$, then $n > 0$, u is a confluent closed term t of `CT-`λ of type $\mathbf{N}$, and the value of u is $n - 1$. Indeed,if $u \mapsto u'$ and u' is normal, then $t \mapsto S(u')$, therefore $S(u') = n$ by confluence of t, and we conclude that $n > 0$ and $u' = n - 1$. Thus, the normal form of u is unique.

We will prove the following.

Theorem 4.1 *Assume t is any well-typed term such that $\neg(t \sim t)$. Then there is some infinite path $\pi = \{t_i | i \in \mathbf{N}\}$ of t with some infinite sequence $\sigma = \{\sigma_i | i \in \mathbf{N}\}$, with each σ_i counter-example for t_i, and with the value of σ compatible with the trace conditions of π.*

As a corollary from the theorem we will conclude: if t satisfies the global trace condition, then there is no such σ, therefore $t \sim t$, as we wished to show.

We prove that for any term $(t \sim t)$ with counter-example $\vec{a}, \vec{d} \sim_0 \vec{b}, \vec{e}$ and $\neg(t[\vec{a}](\vec{d}) \sim_0 t[\vec{b}](\vec{e}))$ we can find some immediate subterm t' and with a counter-example $\vec{a'}, \vec{d'} \sim_0 \vec{b'}, \vec{e'}$ and $\neg(t'[\vec{a'}](\vec{d'}) \sim_0 t'[\vec{b'}](\vec{e'}))$, *compatible with trace condition.*

5 appendix

```
To: kmr@is.sci.toho-u.ac.jp (Daisuke Kimura)
Re: proof of Weak Normalization to an integer for CT-lambda
Fri, 22 Mar 2024 08:25:57 +0100

    By the way, I re-checked the weak curry-howard proof, now i think that the proof
does not require the property p-->q, a-->b ==> p[a/x]-->q[b/x] and can be completed
with the notion of safe reduction.
but in fact it would be more interesting to prove full church-rosser for Circular T-lambda,
as anupam does for his circular T.

    About strong normalization, we can prove it for "safe" reductions, those inside no cond.
More in general, we know that we can have infinite reduction sequences, because we can
have infinitely many redexes. However, for any infinite reduction sequence sigma, I conjecture
we can prove a kind of stabilization of the term. After some reduction step, the term only
changes inside some cond nested k times.

    Namely, I conjecture that

"for any cyclic lambda term t, any infinite reduction sequence (sigma(n)|n in N) with sigma(0)=t,
any k in N, there is a n0 in N such that for all n>=n0, the terms sigma(n) and sigma(n0)
coincide on all branches with at most k times cond."

    Best, Stefano
```

We say that a node is in the $k, \texttt{cond}$-level if it has less than k nodes `cond` in its branch.

We say that an infinite reduction sequence is is fair for nodes of $k, \texttt{cond}$-level in the following case: for all $i \in \mathbf{N}$, all $k > 0$, there is some $j \geq i$ such that at step j either all nodes in the $k, \texttt{cond}$-level are normal, or one of them is reduced at step j.

Conjecture. An infinite reduction sequence σ is normalizing in the limit if and only if for all $k \in \mathbf{N}$, $k > 0$, σ is fair for the task of reducing nodes in the $k, \texttt{cond}$-level.