Short proofs of normalization for simply-typed lambda-calculus and Gödel's system T

Ralph Matthes[a], CNRS (Université Paris VII, PPS)
(joint work with Felix Joachimski, München)

Séminaire de logique
May 30, 2002
Université de Savoie, Chambéry

[a] matthes@pps.jussieu.fr,
Homepage: http://www.pps.jussieu.fr/~matthes/

Abstract:

(Joint work with Felix Joachimski.)

Strong normalization for simply-typed lambda-calculus has been established in many different ways. The semantical approach of using logical predicates à la Tait or Girard has been most successful in the treatment of extensions such as by impredicativity or by primitive recursion. Nevertheless, it is valuable to have proofs which exclusively use syntactic—or proof-theoretic—ideas in order to establish strong normalization of lambda calculi, which will be done in the talk for simply-typed lambda-calculus and its extension by primitive recursion over the naturals (a.k.a. Gödel's system T).

The proof for simply-typed lambda-calculus is based on the inductive characterization SN of strongly normalizing terms by van Raamsdonk and Severi and argues exclusively by induction on types, terms and the definition of SN.

(continued)

The extension to Gödel's T must go beyond formalizability in Peano arithmetic. This is achieved by introducing an infinitely-branching inductive rule into a variant Ω of the appropriate generalization of SN for T. Induction on Ω and on natural numbers are the only new ingredients needed to complete the proof.

For the full paper containing also the treatment of permutative conversions for sum types with case analysis and those for a variant of lambda-calculus in the spirit of von Plato, see
`http://www.tcs.informatik.uni-muenchen.de/~matthes/papers/sn.html`

λ-calculus: Statics

Raw terms:

$$\Lambda \ni r, s ::= x \mid rs \mid \lambda x r.$$

Inductive characterization:

$$\Lambda \ni r, s ::= x\vec{r} \mid \lambda x r \mid (\lambda x r) s \vec{s},$$

Normal forms:

$$\mathrm{NF} \ni r ::= x\vec{r} \mid \lambda x r.$$

Types:

$$\rho, \sigma ::= \iota \mid \rho \to \sigma.$$

Type assignment:

$$\frac{r : \rho \to \sigma \qquad s : \rho}{rs : \sigma} \qquad \frac{r : \sigma \qquad x : \rho}{\lambda x r : \rho \to \sigma}$$

λ-calculus: Dynamics

Reduction relation $\to$, defined inductively:

$$(\lambda x r)s \to r_x[s]$$

$$r \to r' \implies \lambda x r \to \lambda x r', rs \to r's, sr \to sr'.$$

Intuitively, a term r is strongly normalizing iff there are no infinite reduction sequences starting with r.

We take the more proof-theoretic definition

$$\frac{\forall r'. r \to r' \Rightarrow r' \Downarrow}{r \Downarrow}$$

$r \Downarrow$ is equivalent with r's strong normalization (bar induction!)

The set SN (appeared first in van Raamsdonk/Severi 1995) is defined inductively

$$\frac{\vec{r} \in \mathrm{SN}}{x\vec{r} \in \mathrm{SN}}\,(\mathrm{Var}) \qquad \frac{r \in \mathrm{SN}}{\lambda x r \in \mathrm{SN}}\,(\lambda) \qquad \frac{r_x[s]\vec{s} \in \mathrm{SN} \qquad s \in \mathrm{SN}}{(\lambda x r) s\vec{s} \in \mathrm{SN}}\,(\beta)$$

Observation: SN is syntax-directed.

Theorem $r \Downarrow \iff r \in \mathrm{SN}$.

For the proof of strong normalization, we only need that $r \in \mathrm{SN}$ implies $r \Downarrow$, i. e., that the defining rules of SN are closure rules of the set $\mathrm{WF}_{\rightarrow} := \{r \mid r \Downarrow\}$ of strongly normalizing terms, hence *soundness*.

Why? Because we then only need to show that every well-typed term is in SN.

The other direction is interesting for untyped terms but very easy to establish. Therefore only soundness:

(Var). Assume $\vec{r}\Downarrow$. Since each reduction on $x\vec{r}$ has to take place in one of the $\vec{r}$, we also get $x\vec{r}\Downarrow$.

(λ). Each reduction on λxr actually occurs in r, so the hypothesis $r\Downarrow$ yields $\lambda xr\Downarrow$.

(β). Show $(\lambda xr)s\vec{s}\Downarrow$ by main induction on $s\Downarrow$ and side induction on $r_x[s]\vec{s}\Downarrow$. We have to prove $t\Downarrow$ for any immediate reduct t:

$(\lambda xr)s\vec{s}\to(\lambda xr')s\vec{s}$. Then $r_x[s]\vec{s}\to r'_x[s]\vec{s}$ by substitutivity, hence by side induction hypothesis $(\lambda xr')s\vec{s}\Downarrow$.

$(\lambda xr)s\vec{s}\to(\lambda xr)s'\vec{s}$. Then $r_x[s]\vec{s}\to^* r_x[s']\vec{s}$ by compatibility, hence also $r_x[s']\vec{s}\Downarrow$. The main induction hypothesis yields $(\lambda xr)s'\vec{s}\Downarrow$.

$(\lambda xr)s\vec{s}\to(\lambda xr)s\vec{s'}$. Then $r_x[s]\vec{s}\to r_x[s]\vec{s'}$, hence by side induction hypothesis $(\lambda xr)s\vec{s'}\Downarrow$.

$(\lambda xr)s\vec{s}\to r_x[s]\vec{s}\Downarrow$ by assumption.

“Show $(\lambda xr)s\vec{s}\Downarrow$ by main induction on $s\Downarrow$ and side induction on $r_x[s]\vec{s}\Downarrow$.”

Main induction on $s\Downarrow$ amounts to showing $\rightarrow$-progressivity of the set

$$M := \{s \mid \forall r, \vec{s}. r_x[s]\vec{s}\Downarrow \Rightarrow (\lambda xr)s\vec{s}\Downarrow\}.$$

We hence may assume that any one-step-reduct of s is in M. For $s \in M$ we now do side induction on $r_x[s]\vec{s}\Downarrow$, i.e., we have to show $\rightarrow$-progressivity of the set

$$N := \{t' \mid t'\Downarrow \wedge \forall r, \vec{s}. t' = r_x[s]\vec{s} \Rightarrow (\lambda xr)s\vec{s}\Downarrow\}.$$

Observe that s is already fixed, and that we forced N to contain only strongly normalizing terms t' (the same “induction loading” is needed for primitive recursion as opposed to iteration only).

The work to be done is exactly the same as on the previous slide, hence that kind of reasoning is fully justified.

Remarks

Tait 1975 gave the first proof of strong normalization of second-order lambda-calculus. There, he considered the soundness of rule (β) as an essential ingredient. Clearly, it can be proven more easily by reference to hypothetical infinite reduction sequences. The current proof is much easier to carry out in theorem provers (and has been done in Isabelle by Tobias Nipkow and Stefan Berghofer).

The idea of *van Raamsdonk/Severi 1995* was to put these closure properties into a separate definition, hence improving modularity of the arguments.

Lemma $r \in \mathrm{SN}, s^\rho \in \mathrm{SN} \implies (i)\ rs \in \mathrm{SN},\ (ii)\ r_x[s] \in \mathrm{SN}.$

Proof by simultaneous induction on ρ and side induction on the generation of $r \in \mathrm{SN}$. Cases are analyzed according to the derivation of $r \in \mathrm{SN}$. We start with the easy cases.

(λ). $\lambda yr \in \mathrm{SN}$ is generated from $r \in \mathrm{SN}$. (i). For $(\lambda yr)s \in \mathrm{SN}$ it suffices to show $r_y[s] \in \mathrm{SN}$. But this is exactly the side induction hypothesis for (ii). (ii). Side induction hypothesis (ii) yields $r_x[s] \in \mathrm{SN}$ and consequently $\lambda y r_x[s] \in \mathrm{SN}$.

(β). $(\lambda yr)t\vec{t} \in \mathrm{SN}$ is obtained from $r_y[t]\vec{t} \in \mathrm{SN}$ and $t \in \mathrm{SN}$. Therefore both side induction hypotheses apply and yield (i) $r_y[t]\vec{t}s \in \mathrm{SN}$ and (ii) $(r_y[t]\vec{t})_x[s], t_x[s] \in \mathrm{SN}$, respectively. From these we obtain the assertions $(\lambda yr)t\vec{t}s \in \mathrm{SN}$ (i) and $((\lambda yr)t\vec{t})_x[s] \in \mathrm{SN}$ (ii) by definition of SN.

the difficult case

(Var). $r \equiv y\vec{r} \in \mathrm{SN}$ is generated from $\vec{r} \in \mathrm{SN}$. (i). $\vec{r} \in \mathrm{SN}$ together with $s \in \mathrm{SN}$ shows $y\vec{r}s \in \mathrm{SN}$, using (Var). (ii). Side induction hypothesis (ii) yields $r_k[x := s] \in \mathrm{SN}$ for each r_k in $\vec{r}$. Now if $y \not\equiv x$ we get $y\vec{r}_x[s] \in \mathrm{SN}$ at once. Otherwise we need $s\vec{r}_x[s] \in \mathrm{SN}$, but this follows by multiple applications of the main induction hypothesis for (i), since each of the arguments in the nested application has a subtype of ρ.

Clearly, this argument involves an additional induction on the length of the list $\vec{r}$.

By induction on (well-typed) terms, one shows that every term is in SN, hence strongly normalizing.

A proof variant, suggested by René David

Instead of proving closure of SN under substitution and under application, prove only closure under substitution, hence only prove:

Lemma $r, s^\rho \in \mathrm{SN} \Longrightarrow r_x[s] \in \mathrm{SN}$

Its proof will need the following proposition: $r \in \mathrm{SN} \Longrightarrow rx \in \mathrm{SN}$.

The proposition can be proved by induction on $r \in \mathrm{SN}$, the lemma by the same principle as before. We do not even need the embedded induction on the length of the list of arguments.

Closure under application is favourably proven by using $rs = (rx)_x[s]$ with fresh variable x. (For purposes of proof-theoretic analysis, the representation $rs = (xs)_x[r]$ is not well-suited.)

the critical case

Only the variable case with a non-trivial elimination $(xt^{\rho_0})^{\rho_1}\vec{t}$ on the same variable x that is substituted by s. Then $\rho = \rho_0 \to \rho_1$.

- By the side induction hypothesis we have $t_x[s], \vec{t}_x[s] \in \mathrm{SN}$. Consequently also $z\vec{t}_x[s] \in \mathrm{SN}$ for a fresh variable $z : \rho_1$.
- By the proposition we have $sy^{\rho_0} \in \mathrm{SN}$ for a new y, hence by induction hypothesis for ρ_0 also $st_x[s] \in \mathrm{SN}$.
- The induction hypothesis for ρ_1 shows $(xt\vec{t})_x[s] = (z\vec{t}_x[s])_z[st_x[s]] \in \mathrm{SN}$.

Why the efforts for the simultaneous proof of (i) and (ii)?

This will only be important for the extensions by term constructs with permutative conversions such as ΛJ (the variant of natural deduction advocated by Jan *von Plato*) or *case analysis* (presuming sum types resp. disjunction).

(In fact, one can push the "single-threaded" proof through, but will have the whole proof of closure under application in one of the cases of the proof of closure under substitution, hence reducing modularity of the overall proof.)

There is a more fine-grained analysis by Arnold Beckmann where the application rule is put into the definition of a set SN_k where k denotes the maximal rank of the type of r in the applications rs put into SN_k. Then *rank reduction* can be analyzed quantitatively. One may say that the shown proof does the cut-elimination to cut-rank 0 in one pass.

Gödel's system T

in the version of "Proofs and Types" of Girard, Taylor and Lafont, i. e., in a λ-calculus formulation with primitive recursion as opposed to iteration and with a recursion rule applying not only to numerals.

Add a basic type N for the natural numbers, a constant $0 : \mathsf{N}$ and a unary function symbol $\mathcal{S} : \mathsf{N} \to \mathsf{N}$ for the successor.

Usually, one would add for every type ρ a constant R_ρ (the recursor for functions from naturals into type ρ) such that $R_\rho r^{\mathsf{N}} s^\rho t^{\mathsf{N}\to\rho\to\rho} : \rho$. The new reduction rules: $R0st \to s$ and $R(\mathcal{S}r)st \to tr(Rrst)$.

We slightly modify this syntax to the elimination rule and reductions

$$\frac{r : \mathsf{N} \qquad s : \rho \qquad t : \mathsf{N} \to \rho \to \rho}{r(s,t) : \rho}$$

$$0(s,t) \to s \qquad (\mathcal{S}r)(s,t) \to tr(r(s,t))$$

Elimination notation

We repeatedly used $\vec{r}$ in our treatment of simply-typed λ-calculus as is usual. Now, there is a second kind of application, namely N-elimination $r(s,t)$. This suggests a special meta-syntax for eliminations:

$$R, S ::= r \mid (r, s).$$

With this notation we can characterize terms by

$$\mathsf{T} \ni r, s ::= x\vec{R} \mid \lambda x r \mid 0 \mid \mathcal{S} r \mid 0(r,s)\vec{S} \mid (\mathcal{S} r)(s,t)\vec{S} \mid (\lambda x r) s \vec{S}$$

and normal forms by

$$\mathrm{NF} \ni r, s ::= x\vec{R} \mid \lambda x r \mid 0 \mid \mathcal{S} r \quad \text{with } R ::= r \mid (r, s).$$

Observe that we always try to analyze the natural numbers argument of the recursions—also if recursions are nested this way. This has not been needed in the classical works of Sanchis, Diller and Howard.

Definition of SN for T

$(r,s) \in \mathrm{SN}$ is an abbreviation for $r \in \mathrm{SN}$ *and* $s \in \mathrm{SN}$.

$$\frac{}{0 \in \mathrm{SN}}\,(0) \qquad \frac{r \in \mathrm{SN}}{\mathcal{S}r \in \mathrm{SN}}\,(\mathcal{S}) \qquad \frac{r \in \mathrm{SN}}{\lambda x r \in \mathrm{SN}}\,(\lambda)$$

and for recursion

$$\frac{r\vec{S} \in \mathrm{SN} \qquad s \in \mathrm{SN}}{0(r,s)\vec{S} \in \mathrm{SN}}\,(R_0) \qquad \frac{tr(r(s,t))\vec{S} \in \mathrm{SN}}{(\mathcal{S}r)(s,t)\vec{S} \in \mathrm{SN}}\,(R_{\mathcal{S}})$$

The rules (Var) and (β) now allow for N–eliminations.

$$\frac{\vec{R} \in \mathrm{SN}}{x\vec{R} \in \mathrm{SN}}\,(\mathrm{Var}) \qquad \frac{r_x[s]\vec{S} \in \mathrm{SN} \qquad s \in \mathrm{SN}}{(\lambda x r)s\vec{S} \in \mathrm{SN}}\,(\beta)$$

One *could* avoid the elimination notation in favour of evaluation contexts, and this was the notation in which the present characterization had first been expressed (on Good Friday 1996).

Towards a proof of normalization

Soundness of SN is no harder than previously for simply-typed λ-calculus except for the use of course-of-value induction than induction only (due to the duplication of r and t in the recursion rule).

From this alone, it is easy to redo a Tait-style computability proof without mentioning the reduction relation any further: one shows that every well-typed term is in SN.

SN is also complete, i. e., all strongly normalizing terms belong to SN. This is again simple to see.

As a corollary we get: SN is closed under the subterm relation. (Will be needed deep inside the proof.)

The method for simply-typed λ-calculus cannot carry over

Since normalization of Gödel's T implies consistency of Peano's arithmetic (this is the raison d'être of the system, introduced as the terms for the Dialectica interpretation), we have to do a proof beyond formalizability in arithmetic. Formally, the inductive definition of SN is not part of Peano arithmetic but can be easily arithmetized since it is finitely branching (see Troelstra 1973). Therefore, the previous method *cannot* work. It *does* not work since the variable rule with the general eliminations breaks the proof of closure under substitution, e. g. in $(x(s,t))_x[\mathcal{S}y]$ with appropriate t.

The solution is (as in the papers by Sanchis, Diller and Howard) to introduce an infinitary rule.

The set Ω

is generated by the same rules as SN, where (Var) and $(R_{\mathcal{S}})$ are restricted

$$\frac{\vec{r}\ \in \Omega}{x\vec{r}\ \in \Omega}\,(\mathrm{Var}) \qquad \frac{tn(n(s,t))\vec{T}\ \in \Omega}{(\mathcal{S}n)(s,t)\vec{T}\ \in \Omega}\,(R_{\mathcal{S}})$$

and the following infinitary ω-rule is added.

$$\frac{r \in \Omega \qquad \cdots\ n(s,t)\vec{T} \in \Omega\ \cdots_{n\in\mathbb{N}}}{r(s,t)\vec{T} \in \Omega}\,(\omega)$$

Then N-eliminations of a variable can only enter Ω through the ω-rule. Thus, in the untyped calculus not studied here, the term $x(\lambda z.zz, y)(\lambda z.zz)$ would be in $\mathrm{SN} \setminus \Omega$.

Ω is closed under substitution

The well-known lemma $r \in \Omega, s^\rho \in \Omega \implies$ (i) $rs \in \Omega$, (ii) $r_x[s] \in \Omega$. or its (cuter) brother with (ii) only can be established as usual (since the possible N-eliminations only matter for the variable rule, and here in Ω, they are excluded).

From the lemma, we get (as earlier for SN) that every (typable) term is in Ω. (In the case of the ω-rule, we need an obvious side induction on the index n.)

It has to be admitted that the proof based on (ii) only already appeared in Sanchis' article, submitted in 1965. But from his arguments, it is not possible to see that terms in Ω are strongly normalizing in the full formulation of the system with recursion also for the successor of an arbitrary term (even for the successor of a variable). For Sanchis' reduction, it is not hard to see that the rule (ω) is a closure rule of the set of strongly normalizing terms.

Injecting Ω into SN

The rules of Ω are restrictions of SN–rules, except for (ω) which is (after renaming Ω to SN) a weakened version of the following rule of *value expansion*

$$\frac{r \in \mathrm{SN} \qquad |r|(s,t)\vec{S} \in \mathrm{SN}}{r(s,t)\vec{S} \in \mathrm{SN}}\,(*)$$

where the *numeral* $|r|$ is defined recursively for $r^{\mathsf{N}} \in \mathrm{SN}$ by

$$\begin{array}{rcl}
|x\vec{R}| & := & 0, \\
|0| & := & 0, \\
|\mathcal{S}r| & := & \mathcal{S}|r|, \\
|(\lambda x r)s\vec{S}| & := & |r_x[s]\vec{S}|, \\
|0(r,s)\vec{S}| & := & |r\vec{S}|, \\
|(\mathcal{S}r)(s,t)\vec{S}| & := & |tr(r(s,t))\vec{S}|.
\end{array}$$

the final step

In order to show $\Omega \subseteq \mathrm{SN}$, it suffices that $(*)$ is admissible in SN.

A more general statement has to be shown with substitution not only into the head position (and consequently even for simultaneous substitution):

Lemma (Value expansion)

$\vec{s}^{\mathsf{N}} \in \mathrm{SN}, r[\vec{x} := |\vec{s}|] \in \mathrm{SN} \Longrightarrow r[\vec{x} := \vec{s}] \in \mathrm{SN}.$

Proof by induction on the generation of $r^{\vec{n}} \in \mathrm{SN}$ where we abbreviate $r^{\vec{t}} := r_{\vec{x}}[\vec{t}]$ and set $\vec{n} := |\vec{s}|$. We have to show $r^{\vec{s}} \in \mathrm{SN}$.

The proof for the cases of recursion (initial and step case) are both quite complicated and need auxiliary statements in their proofs:

Value expansion, cont'd

These embedded statements are

$$\forall \hat{s}^{\mathsf{N}} \in \mathrm{SN}. |\hat{s}| = 0 \Longrightarrow \hat{s}(r_0^{\vec{s}}, r_1^{\vec{s}})\vec{R}^{\vec{s}} \in \mathrm{SN}$$

and

$$\forall \hat{s}^{\mathsf{N}}. |\hat{s}| = \mathcal{S}m \Longrightarrow \hat{s}(s^{\vec{s}}, t^{\vec{s}})\vec{R}^{\vec{s}} \in \mathrm{SN}$$

and are both proved (in their context, using the main induction hypothesis) by side induction on $\hat{s} \in \mathrm{SN}$.

Conclusions

- no need for geometric intuition: only well-founded parts matter
- no reduct analysis after having established soundness of SN
- Sanchis' analysis extended to current formulation of T
  - needs another concept of vector of trailing arguments
  - value expansion as key concept to escape reduction strategies
  - cannot work with the characterization SN alone

Further Work

- extends to systems with intersection types *and* permutative conversions
- elimination notation most fruitful for impredicative encodings and computability predicates proofs for inductive types

Finis