NCPO goes Beta-Eta-Long Normal Form

Johannes Niederhauser
University of Innsbruck, Innsbruck, Austria

Aart Middeldorp
University of Innsbruck, Innsbruck, Austria

Abstract

We adapt our recently introduced higher-order reduction order NCPO to $\beta\eta$-long normal forms which makes it directly applicable to Nipkow's higher-order rewrite systems.

2012 ACM Subject Classification Theory of computation → Equational logic and rewriting; Theory of computation → Lambda calculus

Keywords and phrases higher-order rewriting, termination, computability path order

1 Introduction

We recently introduced NCPO [8,9], a reduction order for proving termination of higher-order rewriting on $\beta\eta$-normal forms where matching modulo $\beta\eta$ is employed. This formalism is very similar to Nipkow's higher-order rewrite systems (HRSs) [7] which work with $\beta\eta$-long normal forms instead. As higher-order recursive path orders like NCPO usually have a hard time dealing with lambda abstractions, we defined it for $\beta\eta$-normal forms which typically contain fewer lambda abstractions than $\beta\eta$-long normal forms. However, this design choice excludes the direct applicability of NCPO to HRSs. For NCPO's cousin NHORPO [5], this was resolved by putting it into a suitable wrapper which makes it applicable to any orientation of η. However, there are benefits of directly working on $\beta\eta$-long normal forms: For both NCPO and NHORPO, termination can only be guaranteed for terms where each function symbol is applied to as many arguments as required by its arity. This is not a problem for $\beta\eta$-long normal forms since function symbols (and variables) are always maximally applied. Moreover, choosing the largest possible arity for each function symbol potentially enhances the power of NCPO and NHORPO. Finally, $\beta\eta$-long normal forms facilitate an easier definition of the important concept of nonversatile terms. Hence, we will directly adapt the definition of NCPO to $\beta\eta$-long normal forms despite the potentially increased number of lambda abstractions which the order has to deal with. Our preliminary results suggest that the resulting reduction order is a powerful and lightweight termination method for HRSs.

2 Preliminaries

In this paper, we consider higher-order rewriting on simply-typed lambda terms [1,3]. Given a set $\mathcal{S}$ of sorts, we define the set of simple types $\mathcal{T}$ as usual and follow the convention that the function space constructor $\to$ is right-associative, i.e., $a \to b \to c$ denotes $a \to (b \to c)$. Throughout this text, lowercase letters $a, b, c, \ldots$ denote sorts while upper case letters $T, U, V, \ldots$ denote arbitrary types. For each type $U \in \mathcal{T}$ we consider an infinite set of variables $\mathcal{V}_U$ as well as a set of function symbols $\mathcal{F}_U$ where $\mathcal{V}_U \cap \mathcal{F}_U = \varnothing$ and $\mathcal{V}_U \cap \mathcal{V}_V = \mathcal{F}_U \cap \mathcal{F}_V = \varnothing$ for $V \neq U$. We denote the set of all variables and function symbols by $\mathcal{V} = \bigcup\{\mathcal{V}_U \mid U \in \mathcal{T}\}$ and $\mathcal{F} = \bigcup\{\mathcal{F}_U \mid U \in \mathcal{T}\}$, respectively. The set of simply-typed lambda terms of a type U (Λ_U) is defined as follows: $\mathcal{V}_U \cup \mathcal{F}_U \subseteq \Lambda_U$, if $x \in \mathcal{V}_U$ and $s \in \Lambda_V$ then $\lambda x.s \in \Lambda_{U \to V}$, and if $s \in \Lambda_{U \to V}$ and $t \in \Lambda_U$ then $st \in \Lambda_V$. We follow the convention that application is left-associative, i.e., stu denotes $(st)u$. A *term* is a member of the set $\Lambda = \bigcup\{\Lambda_U \mid U \in \mathcal{T}\}$ and we define the function $\tau \colon \Lambda \to \mathcal{T}$ as $\tau(s) = U$ if $s \in \Lambda_U$.

We write $\mathsf{FV}(s)$ for the set of free variables of a term s. The term $s[x/t]$ denotes the term where all free occurrences of x have been replaced by t without capturing the free variables of t (*capture-avoiding substitution*). Due to the fact that infinitely many variables are available for each type, this can always be done by renaming the bound variables accordingly (*α-renaming*). In the remainder, we abstract away from this technicality by viewing lambda terms as equivalence classes modulo α-renaming. Every term s has a unique β-normal form which we denote by $s{\downarrow_\beta}$. Given a β-normal form s, we write $s{\uparrow^\eta}$ for its unique η-expanded form. Put together, every term s has a unique $\beta\eta$-long normal form $s{\updownarrow^\eta_\beta} = s{\downarrow_\beta}{\uparrow^\eta}$. Terms in $\beta\eta$-long normal form do not contain partial applications, so we can use syntactic sugar where we assign the maximal possible arity to each $h \in \mathcal{F} \cup \mathcal{V}$ and write $h(t_1, \dots, t_n)$ instead of $ht_1 \dots t_n$. Rewriting to β-normal form and the set of $\beta\eta$-long normal forms are denoted by $\to^!_\beta$ and $\mathsf{NF}(\beta\eta^{-1}) \subseteq \Lambda$, respectively.

A $\beta\eta$-long normal substitution σ is a mapping from variables to $\beta\eta$-long normal forms of the same type where $\mathcal{D}\mathsf{om}(\sigma) = \{x \mid \sigma(x) \neq x{\uparrow^\eta}\}$ is finite. We often write σ as a set of variable bindings. Given a substitution $\sigma = \{x_1 \mapsto t_1, \dots, x_n \mapsto t_n\}$, we define $s\sigma$ as the simultaneous capture-avoiding substitution $s[x_1/t_1, \dots, x_n/t_n]$. The free variables of a substitution are defined as follows: $\mathsf{FV}(\sigma) = \bigcup\{\mathsf{FV}(\sigma(x)) \mid x \in \mathcal{D}\mathsf{om}(\sigma)\}$. A substitution is said to be *away from* a finite set of variables X if $(\mathcal{D}\mathsf{om}(\sigma) \cup \mathsf{FV}(\sigma)) \cap X = \varnothing$.

Contexts C are lambda terms which contain exactly one occurrence of the special symbol $\Box$ which can assume any type. We write $C[s]$ for the lambda term C where $\Box$ is replaced by s without employing capture-avoiding substitution. A binary relation $R \subseteq \Lambda \times \Lambda$ is *monotone* if $s \mathrel{R} t$ implies $C[s] \mathrel{R} C[t]$ for every context C. Furthermore, we say that a term t is a *subterm* of s ($s \trianglerighteq t$) if there exists a context C such that $s = C[t]$ and define the proper subterm relation $s \rhd t$ as $s \trianglerighteq t$ and $s \neq t$. Since we view lambda terms as equivalence classes modulo α-renaming, the subterm relation is also defined modulo α-renaming. Hence, we have e.g. $\lambda x.t \rhd t[x/z]$ for a fresh variable z.

A pair of terms $\ell \to r$ with $\ell, r \in \mathsf{NF}(\beta\eta^{-1})$ and $\tau(\ell) = \tau(r) \in \mathcal{S}$ constitutes a *rule* if $\mathsf{FV}(r) \subseteq \mathsf{FV}(\ell)$ and ℓ is not of the form $x(t_1, \dots, t_n)$. A *higher-order rewrite system* (HRS) is a set of rules. Given an HRS $\mathcal{R}$, its *rewrite relation* $\to_\mathcal{R}$ is defined as follows: There is a *rewrite step* $s \to_\mathcal{R} t$ if there exist a rule $\ell \to r \in \mathcal{R}$, a $\beta\eta$-long normal substitution σ and a context C such that $s = C[\ell\sigma{\downarrow_\beta}] \in \mathsf{NF}(\beta\eta^{-1})$ and $t = C[r\theta{\downarrow_\beta}]$. Note that our definition of HRSs is equivalent to the original one given in [7]. In particular, $\to_\mathcal{R} \subseteq \mathsf{NF}(\beta\eta^{-1}) \times \mathsf{NF}(\beta\eta^{-1})$. Hence, both rules and rewrite steps only consider terms in their unique $\beta\eta$-long normal form whereas matching is performed modulo $\beta\eta$. We say that an HRS $\mathcal{R}$ is *terminating* if $\to_\mathcal{R}$ is well-founded. We now define the notion of $\beta\eta$-long normal higher-order reduction orders.

▶ **Definition 1.** *A $\beta\eta$-*long normal higher-order reduction order *is a pair $(>, \sqsupset)$ which satisfies the following conditions: $\sqsupset \subseteq \Lambda \times \Lambda$ is a well-founded relation, $\sqsupset$ is monotone, $\to_\beta \subseteq \sqsupset$, and $s > t$ implies $s\sigma{\downarrow_\beta} \sqsupset^+ t\sigma{\downarrow_\beta}$ for all $s, t \in \mathsf{NF}(\beta\eta^{-1})$ and $\beta\eta$-long normal substitutions σ. We often refer to the last condition as $\beta\eta$-*long normal stability*. An* HRS *$\mathcal{R}$ is* compatible *with a $\beta\eta$-long normal higher-order reduction order $(>, \sqsupset)$ if $\ell >^+ r$ for all $\ell \to r \in \mathcal{R}$.*

As in [5,9], the intuition behind this definition is that $>$ will be used to orient the rules of HRSs while relying on the termination proof of its plain variant $\sqsupset$. Despite calling $(>, \sqsupset)$ an order, we do not demand transitivity of any of its components. In the context of higher-order rewriting, this is standard as $\sqsupset$ contains β-reduction which is not transitive. By taking the identity substitution, we can see that $\beta\eta$-long normal stability implies $> \subseteq \sqsupset^+$.

▶ **Theorem 2.** *If an* HRS *$\mathcal{R}$ is compatible with a $\beta\eta$-long normal higher-order reduction order $(>, \sqsupset)$, then $\mathcal{R}$ is terminating.*

Proof. For a proof by contradiction, consider an infinite rewrite sequence $s_1 \to_{\mathcal{R}} s_2 \to_{\mathcal{R}} \cdots$. By definition, $s_1 = C[\ell\sigma{\downarrow_\beta}]$ and $s_2 = C[r\sigma{\downarrow_\beta}]$ for some $\ell \to r \in \mathcal{R}$ where σ is a $\beta\eta$-long normal substitution and C is a context such that $C[\ell\sigma{\downarrow_\beta}] \in \mathsf{NF}(\beta\eta^{-1})$. By assumption, $\ell >^+ r$, so $\ell\sigma{\downarrow_\beta} \sqsupset^+ r\sigma{\downarrow_\beta}$ follows from $\beta\eta$-long normal stability and we obtain $C[\ell\sigma{\downarrow_\beta}] \sqsupset^+ C[r\sigma{\downarrow_\beta}]$ by monotonicity of $\sqsupset$. Hence, we can transform the infinite sequence $s_1 \to_{\mathcal{R}} s_2 \cdots$ into the infinite descending sequence $s_1 \sqsupset^+ s_2 \cdots$ which contradicts well-foundedness of $\sqsupset$. Thus, $\mathcal{R}$ is terminating. ◀

3 The Beta-Eta-Long-Normal Computability Path Order

We start by adapting the notion of nonversatile terms from [5, 9] to $\beta\eta$-long normal forms. Intuitively, a term in $\beta\eta$-long normal form is nonversatile if the application of any $\beta\eta$-long normal substitution together with the subsequent β-normalization only affects its proper subterms. As this is needed in the inductive proof of $\beta\eta$-long normal stability, a comparison $s > t$ in our order is only defined for nonversatile s. As opposed to [5, 9], we can directly define the class of nonversatile terms since we work on $\beta\eta$-long normal forms: Fully applied function symbols and lambda abstractions are *nonversatile* while fully applied variables are *versatile*. The set of nonversatile terms is denoted by $\Lambda_{\mathsf{nv}} \subseteq \mathsf{NF}(\beta\eta^{-1})$.

Besides the usual inference rules on terms, reduction orders derived from HORPO [4] require appropriate orders on types. The following definition recalls the notion of admissible type orders from CPO [2].

▶ **Definition 3.** *We define the left (right) argument relation on types $\rhd_l$ ($\rhd_r$) as follows: $T \to U \rhd_l T$ ($T \to U \rhd_r U$). An order $\succ_{\mathcal{T}}$ on types is admissible if $\rhd_r \subseteq \succ_{\mathcal{T}}$, $\gtrdot = (\succ_{\mathcal{T}} \cup \rhd_l)^+$ is well-founded, and if $T \to U \succ_{\mathcal{T}} V$ then $U \succeq_{\mathcal{T}} V$ or $V = T \to U'$ with $U \succ_{\mathcal{T}} U'$. Given a type T and $a \in \mathcal{S}$ we write $a \gtrdot_{\mathcal{S}} T$ ($a \geqslant_{\mathcal{S}} T$) if $a \gtrdot b$ ($a \geqslant b$) for every $b \in \mathcal{S}$ occurring in T.*

▶ **Lemma 4** (Lemma 2.3 from [2]). *Given a well-founded order $\succ_{\mathcal{S}}$ on sorts, let $\succ_{\mathcal{T}}$ be the smallest order on types containing $\succ_{\mathcal{S}}$ and $\rhd_r$ such that $V \succ_{\mathcal{T}} V'$ implies $U \to V \succ_{\mathcal{T}} U \to V'$ for all U, V and V'. Then, $>_{\mathcal{T}}$ is admissible.*

Unlike their first-order versions, higher-order recursive path orders do not enjoy the subterm property. Thus, if we want to recursively define $f(\bar{t}) > v$ by $t_i \geqslant v$ for some i, we usually have to check whether $\tau(t_i) \succeq_{\mathcal{T}} \tau(v)$ holds for the given admissible type order $\succ_{\mathcal{T}}$. In particular, this means that establishing $s > t$ by choosing some $s \rhd s'$ and showing $s' \geqslant t$ is only possible if we check that there is a weak decrease in the admissible type order for all intermediate terms in the recursive definition of $\rhd$. CPO extends HORPO by allowing these checks to be dismissed for the special case of accessible subterms which are determined by type-related properties. To this end, we use the sets $\mathcal{P}\mathsf{os}^+(T)$ and $\mathcal{P}\mathsf{os}_a(T)$ of *positive sort positions* and *positions of* $a \in \mathcal{S}$ in a type $T \in \mathcal{T}$ as defined in [2, Definition 7.2] in the following definition of accessible arguments of function symbols and basic sorts as given in [2, Definitions 7.3 & 7.4].

▶ **Definition 5.** *With every $f \in \mathcal{F}_{T_1 \to \cdots \to T_n \to a}$ we associate a set $\mathsf{Acc}(f)$ of* accessible arguments *of f such that $i \in \mathsf{Acc}(f)$ implies $a \geqslant_{\mathcal{S}} T_i$ and $\mathcal{P}\mathsf{os}_a(T_i) \subseteq \mathcal{P}\mathsf{os}^+(T_i)$ for all $1 \leqslant i \leqslant n$. Furthermore, we say that $a \in \mathcal{S}$ is* basic *if the following conditions hold: $T \prec_{\mathcal{T}} a$ implies that T is a basic sort, and for all $f \in \mathcal{F}_{T_1 \to \cdots \to T_n \to a}$ and $i \in \mathsf{Acc}(f)$, $T_i = a$ or T_i is a basic sort.*

Note that the condition $T \prec_{\mathcal{T}} a$ is straightforward to check with the admissible type order from Lemma 4 as only sorts can be smaller than sorts. Next, we define the notion of

$$
\begin{array}{rl}
\langle \mathcal{F}\rhd \rangle & f(\bar{t}) >^{b,X} v \text{ if } f \in \mathcal{F} \text{ and } t_i \unrhd_{\mathsf{b}}^{\mathsf{nv}} \cdot \unrhd_{\mathsf{a}} \cdot \geqslant_{\tau}^{b} v \text{ for some } i \\
\langle \mathcal{F}{=}_{\mathsf{mul}} \rangle & f(\bar{t}) >^{b,X} g(\bar{u}) \text{ if } f \simeq_{\mathcal{F}} g,\ \mathsf{stat}(f) = \mathsf{mul} \text{ and } \bar{t}\ (>_{\tau}^{b} \cup \rhd_{@}^{X} \cdot \to_{\beta}^{!})_{\mathsf{mul}}\ \bar{u} \\
\langle \mathcal{F}{=}_{\mathsf{lex}} \rangle & f(\bar{t}) >^{b,X} g(\bar{u}) \text{ if } f \simeq_{\mathcal{F}} g,\ \mathsf{stat}(f) = \mathsf{lex} \text{ and} \\
& \quad \exists i.\, t_i >_{\tau}^{b} \cup \rhd_{@}^{X} \cdot \to_{\beta}^{!} u_i \text{ and } \forall j < i.\, t_j = u_j \text{ and } \forall j > i.\, f(\bar{t}) >^{b,X} t_j \\
\langle \mathcal{F}\succ \rangle & f(\bar{t}) >^{b,X} g(\bar{u}) \text{ if } f \in \mathcal{F},\ f \succ_{\mathcal{F}} g \text{ and } f(\bar{t}) >^{b,X} u_i \text{ for all } i \\
\langle \mathcal{F}\mathcal{V} \rangle & f(\bar{t}) >^{1,X} y(\bar{u}) \text{ if } f \in \mathcal{F},\ f(\bar{t}) >^{0,X} y{\uparrow}^{\eta} \text{ and } f(\bar{t}) >^{1,X} u_i \text{ for all } i \\
\langle \mathcal{F}\lambda \rangle & f(\bar{t}) >^{b,X} \lambda y.v \text{ if } f \in \mathcal{F},\ f(\bar{t}) >^{b,X\cup\{z\}} v[y/z],\ \tau(y) = \tau(z) \text{ and } z \text{ fresh} \\
\langle \mathcal{F}\mathcal{X} \rangle & f(\bar{t}) >^{b,X} y{\uparrow}^{\eta} \text{ if } f \in \mathcal{F} \text{ and } y \in X \\
\langle \lambda\rhd \rangle & \lambda x.t >^{b,X} v \text{ if } t[x/z] \geqslant_{\tau}^{b,X} v,\ \tau(x) = \tau(z) \text{ and } z \text{ fresh} \\
\langle \lambda{=} \rangle & \lambda x.t >^{b,X} \lambda y.v \text{ if } t[x/z] >^{b,X} v[y/z],\ \tau(x) = \tau(y) = \tau(z) \text{ and } z \text{ fresh} \\
\langle \lambda{\neq} \rangle & \lambda x.t >^{b,X} \lambda y.v \text{ if } \lambda x.t >^{b,X} v[y/z],\ \tau(x) \neq \tau(y) = \tau(z) \text{ and } z \text{ fresh}
\end{array}
$$

Figure 1 Rules of NCPO-LNF.

nonversatilely accessible subterms. The definition closely follows the one given in [2, Definition 7.5] but with appropriate modifications regarding nonversatility and $\beta\eta$-long normal forms.

▶ **Definition 6.** *We write $s \rhd_{\mathsf{b}}^{\mathsf{nv}} t$ if t is a subterm of $s \in \Lambda_{\mathsf{nv}}$ reachable by nonversatile intermediate terms in the recursive definition of $\rhd$, t is of basic sort and $\mathsf{FV}(t) \subseteq \mathsf{FV}(s)$, i.e., no bound variables have become free. Furthermore, $s \rhd_{\mathsf{a}} t$ if $s = f(s_1, \ldots, s_n)$ and $s_j \unrhd_{\mathsf{a}} t$ for $j \in \mathsf{Acc}(f)$. Note that $\rhd_{\mathsf{b}}^{\mathsf{nv}}, \rhd_{\mathsf{a}} \subseteq \mathsf{NF}(\beta\eta^{-1}) \times \mathsf{NF}(\beta\eta^{-1})$. A term t is* nonversatilely accessible *in a nonversatile term s if $s \rhd_{\mathsf{b}}^{\mathsf{nv}} t$ or $s \rhd_{\mathsf{a}} t$.*

The following definition, adapted from [2, Definition 7.8], introduces the notion of structurally smaller terms with respect to a set of variables X. It is an important ingredient of CPO with accessible subterms as it facilitates a way of using the set X with which the order is parameterized in places where it would otherwise lead to non-termination.

▶ **Definition 7.** *Let X be a finite set of variables. We say that a term t is* structurally smaller *than a term $s \in \mathsf{NF}(\beta\eta^{-1})$, written $s \rhd_{@}^{X} t$, if there are $a \in \mathcal{S}$, $x_1, \ldots, x_n \in X$ and $u \in \Lambda_{T_1 \to \cdots \to T_n \to a}$ such that $t = u x_1 \cdots x_n$, $s \rhd_{\mathsf{a}} u$, $\tau(s) = a$, $\tau(x_i) = T_i$,and $\mathcal{P}\mathsf{os}_a(T_i) = \varnothing$ for all $1 \leqslant i \leqslant n$. Here, $\rhd_{@}^{X} \subseteq \mathsf{NF}(\beta\eta^{-1}) \times \Lambda$.*

In the following, let $\succ_{\mathcal{T}}$ be an admissible order on types and $\succsim_{\mathcal{F}}$ a preorder on $\mathcal{F}$ called *precedence* with a well-founded strict part $\succ_{\mathcal{F}} = \succsim_{\mathcal{F}} \setminus \precsim_{\mathcal{F}}$ and the equivalence relation $\simeq_{\mathcal{F}} = \succsim_{\mathcal{F}} \cap \precsim_{\mathcal{F}}$. Furthermore, for every $f \in \mathcal{F}$ we fix a status $\mathsf{stat}(f) \in \{\mathsf{mul}, \mathsf{lex}\}$ such that symbols equivalent in $\simeq_{\mathcal{F}}$ have the same status. We are now able to lift CPO with accessible subterms and small symbols [2] to an appropriate component of a $\beta\eta$-long normal higher-order reduction order.

▶ **Definition 8.** *Given a finite set X of variables and $b \in \{0, 1\}$, the order $>^{b,X} \subseteq \mathsf{NF}(\beta\eta^{-1}) \times \mathsf{NF}(\beta\eta^{-1})$ is inductively defined in Figure 1. Furthermore, $s >_{\tau}^{b,X} t$ if $s >^{b,X} t$ and $\tau(s) \succeq_{\mathcal{T}} \tau(t)$, and $>^{b}$ ($>_{\tau}^{b}$) is a shorthand for $>^{b,\varnothing}$ ($>_{\tau}^{b,\varnothing}$).*

Note that $s >^{b,X} t$ is well-defined by induction on (b, s, t) with respect to the well-founded order $(>_{\mathbb{N}}, \rhd, \rhd)_{\mathsf{lex}}$. We refer to $>_{\tau}^{1}$ as the $\beta\eta$-long normal computability path order (NCPO-LNF) and use the symbol $\sqsupset$ with the same decorations as $>$ (except for b) to denote CPO with accessible subterms. The definition of $>^{X}$ is an adaption of the corresponding definition for NCPO to $\beta\eta$-long normal forms. In particular, the modifications of the original rules

Table 1 Experimental results.

problem	NCPO-LNF	NCPO	WANDA
[9, Example 6] (extended)	✓	×	✓
[9, Example 7]	✓	✓	✓
[9, Example 8]	✓	✓	✓
[2, Example 5.2]	✓	✓	✓
[2, Example 7.1]	✓	✓	×
[2, Example 8.19]	✓	✓	✓
[5, Example 7.1]	✓	✓	✓
[5, Example 7.2]	×	×	✓
[5, Example 7.3]	✓	✓	✓
`neutr.p`	×	×	✓
`neutrN.p`	✓	✓	✓

$\langle \mathcal{F}@\rangle$ and $\langle \mathcal{FV}\rangle$ justify renaming them to $\langle \mathcal{FV}\rangle$ and $\langle \mathcal{FX}\rangle$, respectively. Moreover, the rule $\langle \lambda \rhd \eta\rangle$ of NCPO has been removed and we do not need cases for applications on the left-hand side as terms of the form $y(\bar{t})$ are versatile. Furthermore, we removed the rule $\langle \lambda \mathcal{V}\rangle$ because for variables of functional type, $\lambda x.t >^X y{\uparrow}^\eta$ cannot be directly simulated in CPO which we need for the proof of $\beta\eta$-long normal stability. We also removed the extension of CPO to small symbols: Since terms are in $\beta\eta$-long normal form, applicative uncurried systems, the main motivation behind small symbols in [2], do not have to be considered anymore. Without applications we would still have the $\langle \lambda \mathcal{F}_\mathsf{s}\rangle$ rule which allows us to discard small symbols when comparing to an abstraction in NCPO, but we could not find an example where this is needed the setting of $\beta\eta$-long normal forms. More generally, $\lambda x.t >^X g(\bar{u})$ can only be a recursive comparison of $f(\lambda x.t_1, t_2, \ldots, t_n) >^X g(\bar{u})$ for some $f \in \mathcal{F}$ as $\tau(g(\bar{u})) \in \mathcal{S}$. If $f \succsim_\mathcal{F} g$ then g can be discarded anyway, so we need $g \succ_\mathcal{F} f$ which means that f is also a small symbol. However, $g \succ_\mathcal{F} f$ can only be enforced by a goal $g(\bar{v}) > f(\bar{s})$ resolved with CPO's $\langle \mathcal{F}_\mathsf{s}{=}\rangle$ which requires $g(\bar{v}) >_\tau s_1$. Since $\tau(g(\bar{v})) \in \mathcal{S}$ and $\tau(s_1) = U \to V$, this is impossible with the admissible type ordering from Lemma 4 which we implemented.

▶ **Theorem 9.** *The pair $(>^1_\tau, \sqsupset_\tau)$ is a $\beta\eta$-long normal higher-order reduction order.*

A prototype implementation of NCPO-LNF is available at GitHub.[1] Table 1 compares NCPO-LNF with NCPO and the state of the art higher-order termination tool WANDA [6] on the small set of problems used in our original NCPO paper [9]. Here, ✓ stands for a successful termination proof and × denotes that termination could not be established. Note that NCPO-LNF proves termination of the corresponding HRS while WANDA's soundness is limited to PRSs and NCPO operates on a $\beta\eta$-normal flavor of HRSs. Since HRSs are restricted to rules operating on sorts, we have adapted [9, Example 6] accordingly which explains why NCPO cannot handle it in contrast to our results reported in [9]. Overall, our experimental results give evidence that NCPO-LNF performs slightly better than NCPO for some interesting HRSs. In order to strengthen this evidence, NCPO and NCPO-LNF should be evaluated on larger problem databases such as TPDB and COPS[2] in the future.

[1] `https://github.com/niedjoh/hrsterm`
[2] `https://ari-cops.uibk.ac.at/COPS`

References

1 Henk P. Barendregt. Lambda calculi with types. In Samson Abramsky, Dov M. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*, volume 2, pages 117–309. Oxford University Press, 1992. doi:10.1093/oso/9780198537618.003.0002.

2 Frédéric Blanqui, Jean-Pierre Jouannaud, and Albert Rubio. The computability path ordering. *Logical Methods in Computer Science*, 11(4):3:1–3:45, 2015. doi:10.2168/LMCS-11(4:3)2015.

3 Alonzo Church. A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5(2):56–68, 1940. doi:10.2307/2266170.

4 Jean-Pierre Jouannaud and Albert Rubio. Polymorphic higher-order recursive path orderings. *Journal of the ACM*, 54(1):1–48, 2007. doi:10.1145/1206035.1206037.

5 Jean-Pierre Jouannaud and Albert Rubio. Normal higher-order termination. *ACM Transactions on Computational Logic*, 16(2):13:1–13:38, 2015. doi:10.1145/2699913.

6 Cynthia Kop. WANDA - a higher order termination tool (system description). In Zena M. Ariola, editor, *Proc. 5th International Conference on Formal Structures for Computation and Deduction*, volume 167 of *Leibniz International Proceedings in Informatics*, pages 36:1–36:19, 2020. doi:10.4230/LIPIcs.FSCD.2020.36.

7 Richard Mayr and Tobias Nipkow. Higher-order rewrite systems and their confluence. *Theoretical Computer Science*, 192(1):3–29, 1998. doi:10.1016/S0304-3975(97)00143-6.

8 Johannes Niederhauser and Aart Middeldorp. The computability path order for beta-eta-normal higher-order-rewriting (full version). *CoRR*, abs/2505.20121, 2025. doi:10.48550/arXiv.2505.20121.

9 Johannes Niederhauser and Aart Middeldorp. The computabiliy path order for beta-eta-normal higher-order rewriting. In *Proc. 30th International Conference on Automated Deduction*, 2025. To appear.