Under consideration for publication in Math. Struct. in Comp. Science

Sequent Combinators: A Hilbert System for the Lambda Calculus

HEALFDENE GOGUEN[1] and JEAN GOUBAULT-LARRECQ[2]

[1] *Department of Computer Science,*
The King's Buildings, University of Edinburgh,
Edinburgh, EH9 3JZ, Scotland
[2] *G.I.E. Dyade, INRIA Rocquencourt,*
Domaine de Voluceau, B.P.105,
F-78153 Le Chesnay Cedex, France

Received November 1998, Revised April 1999

This paper introduces Hilbert systems for λ-calculus, called sequent combinators, addressing many of the problems of Hilbert systems that have led to the more widespread adoption of natural deduction systems in computer science. This suggests that Hilbert systems, with their uniform approach to meta-variables and substitution, may be a more suitable framework than λ-calculus for type theories and programming languages.
Two calculi are introduced here. The calculus SKIn captures λ-calculus reduction faithfully, is confluent even in the presence of meta-variables, is normalizing but not strongly normalizing in the typed case, and standardizes. The sub-calculus SKInT captures λ-reduction in slightly less obvious ways, and is a language of proof-terms not directly for intuitionistic logic, but for a fragment of S4 that we name near-intuitionistic logic. To our knowledge, SKInT is the first confluent, first-order calculus to capture λ-calculus reduction fully and faithfully and be strongly normalizing in the typed case. In particular, no calculus of explicit substitutions has yet achieved this goal.

1. Introduction

Hypothetical reasoning is an important tool in logic. However, although most logics include mechanisms for hypothetical reasoning, there is an important divide in how this reasoning is admitted. One approach is to introduce basic principles of reasoning and then prove in the meta-theory that the deduction theorem holds, saying that if B is provable under hypothesis A then $A \Rightarrow B$ is provable: we shall call such systems *Hilbert systems.* The other approach is to include the deduction theorem as a rule of inference, thereby making hypothetical reasoning a first-class construct. We call such systems *natural deduction systems*, after Gentzen (Gentzen, 1969).

This distinction lifts smoothly to type theory with first-class proof objects. Curry's combinators with application (Curry and Feys, 1958) encode Hilbert's axioms with modus ponens, and the abstraction algorithm captures the deduction theorem. Church's typed λ-calculus (Church, 1940) encodes Gentzen's natural deduction.

Hilbert systems have some important advantages over natural deduction systems. The meta-theoretic notion of substitution is simple textual substitution in Hilbert systems, making them easier to understand, to implement, and to reason about. Indeed, Church (Church, 1956) seems to have justified internalizing the deduction theorem as λ-abstraction on the basis of Hilbert systems and the deduction theorem.

However, natural deduction systems have come to dominate computer science for several reasons. First, it is thought that the size of derivations or proof objects explodes using the deduction theorem in Hilbert systems. Secondly, meta-properties such as the deduction theorem are commonly brought into the object language; systems of explicit substitution (Abadi et al., 1991) further internalize substitution and weakening. Thirdly, λ-calculus reduction is stronger and more natural than combinator reduction: the ξ-equality of λ-calculus, which allows equality under λ-abstraction or in hypothetical proofs, is not captured naturally by reduction on combinators. Finally, the duality between introduction and elimination for the logical constants in natural deduction systems does not appear in Hilbert systems.

This paper aims to demonstrate that these defects can be overcome by using a new Hilbert system, sequent combinators. We first introduce SKIn and show that λ-calculus reduction is preserved by translation to this calculus. Together with the benefits of easier implementations, this suggests that Hilbert systems can be used to underlie type-theory based proof assistants and functional programming language implementations.

With respect to the final point, we can take the view underlying Martin-Löf's Logical Framework and consider the defined calculus as a meta-language for defining logics or type theories. This approach recovers the duality between introduction and elimination, while the meta-language remains basic, since abstraction is replaced with families of constants whose behavior can be understood directly. We leave this point as the subject of future work.

We show that the reduction of SKIn is confluent and standardizes, but is slightly too permissive: it is weakly but not strongly normalizing in the typed case, like $\lambda\sigma$ (Abadi et al., 1991). We therefore introduce another, very close calculus, SKInT, which has the required properties of confluence, standardization, and termination. However, SKInT no longer interprets full λ-reduction by the simple translation used for SKIn. But, as it interprets more than call-by-value λ-reduction, there are a number of alternative translations that make SKInT interpret full λ-reduction, obtained by using any translation of the λ-calculus into the call-by-value λ-calculus (Plotkin, 1975).

1.1. *Sequent Combinators*

Following Gentzen, we may interpret intuitionistic sequents $A_1, \ldots, A_m \vdash B$ either as the formula $A_1 \wedge \cdots \wedge A_m \Rightarrow B$, the *Hilbert style*, or as B under assumptions $A_1 \ldots A_m$, the *natural deduction style*. We adopt the Hilbert-style interpretation to define a new language of combinators, where the sequent above is interpreted as $A_1 \Rightarrow \ldots \Rightarrow A_m \Rightarrow B$, and currying is implicit. The language has term constructors corresponding to inference rules similar to those of sequent calculus, rather than constants or combinators corresponding to axioms in Hilbert's formal systems. We show that this language is sound and complete for the λ-calculus, including the equalities α, β and ξ, and extends to η.

The intuition underlying Curry's combinators is that they represent particular λ-terms, or functions. The basic operations of our sequent combinators are just Curry's: identity I, projection K, and composition S, but they are indexed by a fixed number of arguments. Hence, our combinator $\mathrm{K}_m(M)$, corresponding to K, takes the first m arguments and passes them on to M, but ignores the $m + 1$st argument. This is in contrast to K's simpler behavior of taking two arguments and returning the first. Our combinators for I and S are also parameterized by a number of arguments. Section 3.2 gives a more formal intuition for the meaning of the combinators, by mapping them to λ-terms.

Alternatively, the sequent combinators we introduce corresponding to I, K and S are similar to the rules of inference for variables, thinning and substitution for λ-calculus.

Sequent combinators have important meta-theoretic properties. Unlike with combinators, our equalities are naturally interpreted as reductions, and full λ-calculus reduction is preserved by translation into sequent combinators. Unlike systems of explicit substitution, we do not need any operation of currying.

1.2. *Related Work*

Our approach has many similarities to that of Curry, but there are important differences. Curry models the λ-calculus using a basic set of combinators, S and K, plus an invisible binary application operator, together with equations to explain their behavior. However, S and K are so basic that complex equations are needed to capture the rules η expressing weak extensionality and especially ξ mentioned above. Although these equations are easy to verify, they are difficult to understand in the modern treatment of type theory and term rewriting, where we treat equalities as reduction rules from left to right.

Our work improves Curry's here because the equalities over our combinators can easily be understood as reduction rules, rewriting from left to right. The combinators S and K, as well as the basic operation of application, are captured as particular instances of our combinators. Soundness of our system for Curry's equations follows straightforwardly because both our calculus and his are sound and complete for λ-calculus. Furthermore, the relationship between reduction in λ-calculus and our sequent combinators is much stronger than that with Curry's combinators. Reduction in the λ-calculus is preserved by translation to SKIn, and many strategies are also preserved by the same simple translation into the related calculus SKInT. This presents us with new opportunities to study various questions of interest in the λ-calculus without having to cope with the difficulties of handling bound variables and α-renaming.

A disadvantage of our representation is that we introduce families of combinators indexed by natural numbers. This is more complicated than the simple language with only two or three basic combinators plus application.

Combinators have been used in implementations of functional programming following Turner (Turner, 1979), leading to such notions as supercombinators (Hughes, 1982), lambda-lifting (Johnsson, 1984), and director strings (Kennaway and Sleep, 1988)—which are, by the way, at least as complex as our combinators with integer indices. However, these implementations do not consider the problem of equality of expressions involving the abstraction operator, and in particular the ξ-rule is not sound. This is usually not a problem in functional language implementation, but is both an outstanding theoretical problem for λ-calculus and a practical problem for efficient implementations of type-theoretic proof assistants.

Systems of explicit substitution (pioneered by Abadi et al. (Abadi et al., 1991), which many others followed) and categorical combinators (Curien, 1993) move substitution from the meta-theory to the object language. In our formulation, the meta-theoretic substitution is simple textual substitution, rather than the complex substitution of λ-calculus (see Section 2). And while our system satisfies equalities such as β and ξ with respect to the meta-theoretic notions of abstraction and substitution, the actual reductions of the system do not use the meta-theoretic definitions.

Our systems are also related to Kamareddine and Ríos' λse (Kamareddine and Ríos, 1997). Our systems and theirs involve indexed operations for thinning and substitution, and therefore look superficially alike. The main differences are the presence of λ and application operators in λse and their absence in SKIn and SKInT, and a more general thinning operator in λse allowing thinning by a context rather than a single variable. This latter feature of λse has not been explored yet for sequent combinators, although it may lead to interesting systems lying between SKIn and SKInT. The other differences, such as different indexing conventions, are inessential, and arise from λse being an explicit treatment of de Bruijn variables, whereas our work is a generalization of combinators.

Categorical logic, in particular as developed in (Lambek and Scott, 1989), has several of the features we mention in the introduction. In particular, substitution is internalized, and the relationship between the object and meta-substitutions is clear. Furthermore, logical constants are presented as usual in category theory as limit diagrams. However, there are problems with categorical logic from the point of view of the intensional type theory used in modern proof assistants. First, λ-reduction is not captured by reduction in categorical logic. Furthermore, the size of derivations seems to increase significantly under the abstraction algorithm. Finally, the extensionality inherent to category theory—and required to capture λ-equality—is too strong for more sophisticated type theories; for example, extensionality with absurdity and type universes leads to non-termination and undecidability of type checking (see e.g. (Goguen and Luo, 1993)).

1.3. *Overview*

The paper is organized as follows. Section 2 introduces the typing rules for the simply-typed λ-calculus. Section 3 introduces the combinators, typing and reduction rules for them, then studies the relationship of this system with the λ-calculus. Section 4 explores the SKInT calculus and its properties. It is first shown that, whereas the logical reading of the λ-calculus and SKIn is given by intuitionistic logic, the natural logical reading of SKInT is given by near-intuitionistic logic, a fragment of S4 that can also be seen as a relaxation on the Kripke semantics of intuitionistic logic. Section 4 proceeds to scrutinize SKInT as a calculus, and various properties of SKInT reduction are established: strong normalization of simply-typed terms; confluence, even in the untyped case and in the presence of meta-variables; and standardization. Section 5 relies on the techniques developed for SKInT to establish similar, but sometimes weaker properties for SKIn: weak termination, confluence, and standardization again. Although SKInT has fewer reduction rules than SKIn, we can use it instead of SKIn to implement λ-calculus reduction, and different ways of doing so are examined in Section 6. Section 7 discusses further work, and we conclude in Section 8.

2. Simply-Typed λ-Calculus

We assume a denumerable set V of variables $x, y, z, \ldots$, and a non-empty set O of so-called *base types*, with the variable o ranging over O. The types, pre-contexts and terms are introduced by the grammar:

$$\begin{array}{rcl} A, B, C \in Ty & ::= & o \mid A \Rightarrow B \\ \Gamma, \Delta, \Phi \in C & ::= & () \mid \Gamma, x{:}A \\ M, N, P \in T & ::= & x \mid M(N) \mid \lambda x.M \end{array}$$

Parentheses will be used liberally to disambiguate terms, types, etc. A pre-context $x_1{:}A_1, \ldots, x_m{:}A_m$ is a context if the x_is are distinct. We shall also write $M(N_1, \ldots, N_k)$ instead of $M(N_1)\ldots(N_k)$, for $k \geq 0$.

Substitution in the λ-calculus is complex, because it has to avoid variable capture. Assuming a fixed procedure for producing fresh variables z, we may define M with N substituted for x, written $[N/x]M$, as follows:

$$\begin{array}{rll} [P/x]x & =_{\mathrm{df}} P & [P/x]y \quad =_{\mathrm{df}} y \quad (y \not\equiv x) \\ [P/x]M(N) & =_{\mathrm{df}} ([P/x]M)([P/x]N) & \\ [P/x](\lambda x.M) & =_{\mathrm{df}} \lambda x.M & \\ [P/x](\lambda y.M) & =_{\mathrm{df}} \lambda y.[P/x]M & (x \not\equiv y \text{ and } y \text{ not free in } P) \\ [P/x](\lambda y.M) & =_{\mathrm{df}} \lambda z.[P/x][z/y]M & (z \text{ not free in } M,\, P,\, z \not\equiv x \text{ and } y \text{ free in } P) \end{array}$$

The compatible closure R^c of a relation R extends R to subterms rather than simply outermost constructors. Notably, it includes the rule ξ, that if $M R^c N$ then also $(\lambda x.M) R^c (\lambda x.N)$. Two terms M and N are α-equivalent, $M \equiv N$, if they are related by the compatible closure of the following rule:

$$\lambda x.M \quad \alpha \quad \lambda y.[y/x]M \quad (y \text{ not occurring in } M)$$

The judgment form for this calculus is as usual, $\Gamma \vdash M : A$. The rules of inference for the calculus are as follows:

$$\frac{\Gamma \text{ context} \qquad x{:}A \notin \Gamma}{\Gamma, x{:}A \vdash x : A} \quad (\textsc{Var})$$

$$\frac{\Gamma_0, \Gamma_1 \vdash M : A \qquad z \notin dom(\Gamma_0, \Gamma_1)}{\Gamma_0, z{:}C, \Gamma_1 \vdash M : A} \quad (\textsc{Thin})$$

$$\frac{\Gamma, x{:}A \vdash M : B}{\Gamma \vdash \lambda x.M : A \Rightarrow B} \quad (\lambda)$$

$$\frac{\Gamma \vdash M : A \Rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash M(N) : B} \quad (\text{App})$$

$$\frac{\Gamma_0, z{:}C, \Gamma_1 \vdash M : A \quad \Gamma_0 \vdash N : C}{\Gamma_0, \Gamma_1 \vdash [N/z]M : A} \quad (\text{Subst})$$

$$\frac{\Gamma \vdash M : A \quad M \equiv N}{\Gamma \vdash N : A} \quad (\alpha)$$

Strictly speaking, rule (Subst) can be shown to be admissible, and need not be included. Similarly, the rule (Thin) can be restricted to weakening, or simple context extension, and allowed to apply only to variables. However, we have chosen this form to emphasize the relationship with the combinators to be introduced in the next section.

The notions of $\beta\eta$-equality and reduction on terms in the λ-calculus are as usual:

$$\begin{array}{lll} (\lambda x.M)(N) & \beta & [N/x]M \\ \lambda x.(M(x)) & \eta & M \qquad (x \text{ not free in } M) \end{array}$$

Reduction $M \triangleright N$ is the compatible closure of β and α; $\triangleright^+$ is the transitive closure of $\triangleright$, and $\triangleright^*$ its reflexive transitive closure; $M = N$ is the least equivalence relation containing $\triangleright$. We similarly define $\triangleright_\eta$, $\triangleright_\eta^+$ and $=_\eta$ as the corresponding closures of β plus η plus α.

3. Introducing SKIn

Let $m, n, p, \ldots$ denote natural numbers. Types and contexts in SKIn are as in the λ-calculus. Variable-free contexts, represented as lists of types, and SKIn-terms are defined by:

$$\begin{array}{rcl} X, Y, Z \in L & ::= & () \mid X, A \\ M, N, P \in T & ::= & x \mid \mathrm{I}_m \mid \mathrm{K}_m(M) \mid \mathrm{S}_m(M, N) \end{array}$$

If X is a variable-free context, then its length $|X|$ is simply its length as a list.

The semantical idea behind SKIn—and this holds also in the untyped case—is given by the following informal correspondence between SKIn-terms and λ-terms:

$$\begin{array}{rcl} \mathrm{I}_m & \sim & \lambda x_0. \ldots .\lambda x_{m-1}.\lambda x_m.x_m \\ \mathrm{S}_m(M, N) & \sim & \lambda x_0. \ldots .\lambda x_{m-1}.M(x_0, \ldots, x_{m-1}, N(x_0, \ldots, x_{m-1})) \\ \mathrm{K}_m(M) & \sim & \lambda x_0. \ldots .\lambda x_{m-1}.\lambda x_m.M(x_0, \ldots, x_{m-1}) \end{array}$$

It should be apparent that I_m, S_m, K_m generalize Curry's combinators I, S and K respectively.

The judgment for this calculus is $\vdash_\Gamma M : A$. Intuitively, the usual sequent calculus judgment $X \vdash A$, where the proof contains no free variables, translates to $\vdash_{()} M : X \to A$ for some term M, where $X \to A$ is defined as:

$$() \to A =_{\mathrm{df}} A \quad (X, B) \to A =_{\mathrm{df}} X \to (B \Rightarrow A)$$

We write the rules of inference in a way that is suggestive of the sequent calculus, where $\to$ informally plays the role of $\vdash$ in sequent calculus (Γ is a fixed context):

$$(\mathrm{SI}_m) \quad \mathrm{S}_m(\mathrm{I}_m, P) \rhd P \qquad (\mathrm{SK}_m) \quad \mathrm{S}_m(\mathrm{K}_m(M), P) \rhd M$$

$$\begin{array}{llll}
(\mathrm{S}_m\mathrm{I}_n) & \mathrm{S}_m(\mathrm{I}_n, P) \rhd \mathrm{I}_{n-1} & (\mathrm{K}_m\mathrm{I}_n) & \mathrm{K}_m(\mathrm{I}_{n-1}) \rhd \mathrm{I}_n \\
(\mathrm{S}_m\mathrm{K}_n) & \mathrm{S}_m(\mathrm{K}_n(M), P) \rhd \mathrm{K}_{n-1}(\mathrm{S}_m(M, P)) & (\mathrm{K}_m\mathrm{K}_n) & \mathrm{K}_m(\mathrm{K}_{n-1}(M)) \rhd \mathrm{K}_n(\mathrm{K}_m(M)) \\
(\mathrm{S}_m\mathrm{S}_n) & \mathrm{S}_m(\mathrm{S}_n(M, N), P) & (\mathrm{K}_m\mathrm{S}_n) & \mathrm{K}_m(\mathrm{S}_{n-1}(M, P)) \\
& \quad \rhd \mathrm{S}_{n-1}(\mathrm{S}_m(M, P), \mathrm{S}_m(N, P)) & & \quad \rhd \mathrm{S}_n(\mathrm{K}_m(M), \mathrm{K}_m(P))
\end{array}$$

Fig. 1. SKIn reduction rules (for every $0 \leq m < n$)

$$\frac{x{:}A \in \Gamma}{\vdash_\Gamma x \,:\, () \to A} \qquad (\text{Par})$$

$$\frac{}{\vdash_\Gamma \mathrm{I}_{|X|} \,:\, (X, A) \to A} \qquad (\mathrm{I})$$

$$\frac{\vdash_\Gamma M \,:\, X \to A}{\vdash_\Gamma \mathrm{K}_{|X|}(M) \,:\, (X, B) \to A} \qquad (\mathrm{K})$$

$$\frac{\vdash_\Gamma M \,:\, (X, A) \to B \quad \vdash_\Gamma N \,:\, X \to A}{\vdash_\Gamma \mathrm{S}_{|X|}(M, N) \,:\, X \to B} \qquad (\mathrm{S})$$

We can see how the usual combinators are represented in this calculus. Application $M(N)$ is simply $\mathrm{S}_0(M, N)$. On the other hand, K corresponds informally to the SKIn term $\mathrm{K}_1(\mathrm{I}_0)$ (this follows easily by using the translation of Definition 3.7 on the term $\lambda x.\lambda y.x$); and S corresponds informally to $\mathrm{S}_3(\mathrm{K}_1(\mathrm{I}_0), \mathrm{I}_1)$ (this again follows by using the translation on the term $\lambda x.\lambda y.\mathrm{S}_1(x, y)$, or on $\lambda x.\lambda y.\lambda z.(xz)(yz)$ and normalizing).

3.1. *Equality and Reduction*

Reduction in SKIn is defined by the rules in Figure 1, where we use the same notation $\rhd$ for the reduction relation as in the λ-calculus, since no ambiguity can arise.

We use the same notations as in Section 2 for the various closures of sequent combinator reduction, where the intended reduction $\rhd$ is by reading the equalities from left to right. As in the λ-calculus, these rules are untyped.

Similar to the λ-calculus, we have both syntactic equality $M \equiv N$ and provable equality $M = N$. However, in the combinator language $\equiv$ is exactly syntactic identity, rather than α-equivalence in the λ-calculus.

We can already state a few easy results about reduction:

Proposition 3.1 (Weak Church-Rosser). The reduction relation $\rhd$ of SKIn is locally confluent.

Proof. There are 19 critical pairs, all joinable, which were generated and checked automatically by a variant of the Knuth-Bendix algorithm. We give one case, where the top term may rewrite either to the left- or the right-hand term, by the rule labeling the corresponding arrow, and the diagram can be closed by the dotted arrows leading to the common reduct.

$$\begin{tikzcd}[column sep=small]
 & \mathrm{K}_m(\mathrm{S}_{n-1}(\mathrm{K}_{p-1}(M), P)) \arrow[dl, "(\mathrm{S}_{n-1}\mathrm{K}_{p-1})"'] \arrow[dr, "(\mathrm{K}_m\mathrm{S}_n)"] & \\
\mathrm{K}_m(\mathrm{K}_{p-2}(\mathrm{S}_{n-1}(M, P))) \arrow[dr, dotted, "(\mathrm{K}_m\mathrm{K}_{p-1}), (\mathrm{K}_m\mathrm{S}_n)"'] & & \mathrm{S}_n(\mathrm{K}_m(\mathrm{K}_{p-1}(M)), \mathrm{K}_m(P)) \arrow[dl, dotted, "(\mathrm{K}_m\mathrm{K}_p), (\mathrm{S}_n\mathrm{K}_p)"] \\
 & \mathrm{K}_{p-1}(\mathrm{S}_n(\mathrm{K}_m(M), \mathrm{K}_m(P))) &
\end{tikzcd}$$

□

SKIn is in fact confluent (Theorem 5.4), but we lack the necessary tools to show it at this point of the paper.

Proposition 3.2 (Subject Reduction). If $\vdash_\Gamma M : A$ and $M \triangleright N$ then $\vdash_\Gamma N : A$.

Proof. Straightforward. □

Weaker versions of the thinning equalities have appeared in the literature (Peyton-Jones, 1986), e.g. $S(K(M), K(N)) = K(M(N))$.

Unfortunately, although SKIn preserves λ-calculus reduction, we have the following negative result:

Proposition 3.3. SKIn is not strongly normalizing on typed terms.

Proof. This is just as for $\lambda\sigma$ (Melliès, 1995). Let M_n be the sequence of terms defined as:

$$\begin{aligned} &M_0 =_{\mathrm{df}} \mathrm{I}_1 \qquad M_1 =_{\mathrm{df}} \mathrm{I}_1 \\ &M_{n+2} =_{\mathrm{df}} \mathrm{S}_0(\mathrm{K}_0(M_n), M_{n+1}) \end{aligned}$$

and let $N_n =_{\mathrm{df}} \mathrm{S}_0(\mathrm{K}_0(\mathrm{S}_0(\mathrm{K}_0(\mathrm{I}_1), M_n)), M_{n+1})$, of type $(A, B) \to B$, for any A, B. Then we get the following infinite reduction:

$$\begin{aligned} N_n \equiv\, & \mathrm{S}_0(\mathrm{K}_0(\mathrm{S}_0(\mathrm{K}_0(\mathrm{I}_1), M_n)), M_{n+1}) \\ \triangleright\, & \mathrm{S}_0(\mathrm{S}_1(\mathrm{K}_0(\mathrm{K}_0(\mathrm{I}_1)), \mathrm{K}_0(M_n)), M_{n+1}) \\ \triangleright\, & \mathrm{S}_0(\mathrm{S}_0(\mathrm{K}_0(\mathrm{K}_0(\mathrm{I}_1)), M_{n+1}), \mathrm{S}_0(\mathrm{K}_0(M_n), M_{n+1})) \\ \equiv\, & \mathrm{S}_0(\mathrm{S}_0(\mathrm{K}_0(\mathrm{K}_0(\mathrm{I}_1)), M_{n+1}), M_{n+2}) \\ \triangleright\, & \mathrm{S}_0(\mathrm{S}_0(\mathrm{K}_1(\mathrm{K}_0(\mathrm{I}_1)), M_{n+1}), M_{n+2}) \\ \triangleright\, & \mathrm{S}_0(\mathrm{K}_0(\mathrm{S}_0(\mathrm{K}_0(\mathrm{I}_1), M_{n+1})), M_{n+2}) \equiv N_{n+1} \end{aligned}$$

□

However, we shall see that, just like $\lambda\sigma$ (Goubault-Larrecq, 1998b), SKIn normalizes weakly in the typed case (Theorem 5.3).

$$
\begin{aligned}
[\![x]\!](P_0,\ldots,P_{n-1}) &=_{\mathrm{df}} x(P_0,\ldots,P_{n-1})\\
[\![\mathrm{I}_m]\!](P_0,\ldots,P_{n-1}) &=_{\mathrm{df}} \begin{cases} P_m(P_{m+1},\ldots,P_{n-1}) & n>m\\ \lambda x_n\ldots\lambda x_m.x_m & n\le m\end{cases}\\
[\![\mathrm{K}_m(M)]\!](P_0,\ldots,P_{n-1}) &=_{\mathrm{df}} \begin{cases} [\![M]\!](P_0,\ldots,P_{m-1},P_{m+1},\ldots,P_{n-1}) & n>m\\ \lambda x_n\ldots\lambda x_m.[\![M]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{m-1}) & n\le m\end{cases}\\
[\![\mathrm{S}_m(M,N)]\!](P_0,\ldots,P_{n-1}) &=_{\mathrm{df}} \begin{cases} [\![M]\!](P_0,\ldots,P_{m-1},[\![N]\!](P_0,\ldots,P_{m-1}),P_m,\ldots,P_{n-1}) & n\ge m\\ \lambda x_n\ldots\lambda x_{m-1}.[\![M]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{m-1}, & \\ \qquad [\![N]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{m-1})) & n<m\end{cases}
\end{aligned}
$$

Fig. 2. Interpretation of sequent combinators as λ-terms

3.2. *Completeness*

We first show that SKIn is complete for the λ-calculus, to motivate our combinators. By *completeness*, we mean that we may interpret SKIn-terms as λ-terms in such a way that types are preserved, and that equalities are preserved. Our interpretation of SKIn-terms, which is an elaboration of the informal semantics of SKIn-terms presented earlier, is shown in Figure 2, as a map from lists of λ-terms $(P_0,\ldots,P_{n-1})$ $(n\ge 0)$ to λ-terms $[\![M]\!](P_0,\ldots,P_{n-1})$. We use the notation $P_i,\ldots,P_{j-1}$ when $i\le j$ to denote the sequence of all P_k, $i\le k<j$; when $i=j$, this sequence is just empty. This interpretation has the important properties of preserving equality and typing:

Lemma 3.4 (Completeness for Typing). If $\vdash_\Gamma M : A$ then $\Gamma\vdash[\![M]\!]() : A$.

Proof. More generally, we show that whenever $\vdash_\Gamma M : (A_0,\ldots,A_{n-1})\to A$, and $\Gamma\vdash P_0 : A_0,\ldots,\Gamma\vdash P_{n-1} : A_{n-1}$, then $\Gamma\vdash[\![M]\!](P_0,\ldots,P_{n-1}) : A$, by induction on the structure of the typing derivation for M.

This is clear when M is a variable x. When $M\equiv\mathrm{I}_m$, then either $n>m$, in which case we must have $A_m=(A_{m+1},\ldots,A_{n-1})\to A$, and then $[\![M]\!](P_0,\ldots,P_{n-1})\equiv P_m(P_{m+1},\ldots,P_{n-1})$ indeed has type A; or $n\le m$, and then A is of the form $(X,B)\to B$, with $|X|=m-n$, so $[\![M]\!](P_0,\ldots,P_{n-1})\equiv\lambda x_n\ldots\lambda x_m.x_m$ has type $(X,B)\to B$, i.e., A.

When $M\equiv\mathrm{K}_m(N)$ and $n>m$, the typing derivation for M must end in rule (K), whose premise must be $\vdash_\Gamma N : (A_0,\ldots,A_{m-1},A_{m+1},\ldots,A_{n-1})\to A$; by induction hypothesis, $\Gamma\vdash[\![N]\!](P_0,\ldots,P_{m-1},P_{m+1},\ldots,P_{n-1}) : A$, hence $\Gamma\vdash[\![M]\!](P_0,\ldots,P_{n-1}) : A$. When $M\equiv\mathrm{K}_m(N)$ and $n\le m$, the premise must be $\vdash_\Gamma N : (A_0,\ldots,A_{n-1},A_n,\ldots,A_{m-1})\to B$, where $A=(A_n,\ldots,A_{m-1},A_m)\to B$, so by induction hypothesis, weakening and application $\Gamma,x_n{:}A_n,\ldots,x_m{:}A_m\vdash[\![N]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{m-1}) : B$; this implies the claim.

When $M\equiv\mathrm{S}_m(N,P)$ and $n\ge m$, the typing derivation for M must end in rule (S), with premises $\vdash_\Gamma N : (A_0,\ldots,A_{m-1},B,A_m,\ldots,A_{n-1})\to A$ and $\vdash_\Gamma P : (A_0,\ldots,A_{m-1})\to B$ for some type B; by induction hypothesis, $\Gamma\vdash[\![P]\!](P_0,\ldots,P_{m-1}) : B$ and therefore also $\Gamma\vdash[\![N]\!](P_0,\ldots,P_{m-1},[\![P]\!](P_0,\ldots,P_{m-1}),P_m,\ldots,P_{n-1}) : A$, hence the claim. If $M\equiv\mathrm{S}_m(N,P)$ and $n<m$, then the premises must be $\vdash_\Gamma N : (A_0,\ldots,A_{n-1},A_n,\ldots,A_{m-1},B)\to C$ and $\vdash_\Gamma P : (A_0,\ldots,A_{n-1},A_n,\ldots,A_{m-1})\to B$, with $A=$

$(A_n, \ldots, A_{m-1}) \to C$; by induction hypothesis, weakening and application we must then have:

$$\Gamma, x_n{:}A_n, \ldots, x_{m-1}{:}A_{m-1} \vdash [\![P]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1}) : B$$
$$\Gamma, x_n{:}A_n, \ldots, x_{m-1}{:}A_{m-1} \vdash [\![N]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1}, [\![P]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1})) : C$$

This proves the claim. □

In the following theorem, equalities in SKIn translate to equalities involving the η rule in the λ-calculus. We shall refine this result, in order to drop the need for η, in Theorem 3.20.

Theorem 3.5 (Completeness for Equality). If $M = N$ in SKIn, then $[\![M]\!]() =_\eta [\![N]\!]()$.

More generally, if $M \rhd N$ in SKIn, then $[\![M]\!]() \rhd_\eta^* [\![N]\!]()$ in the λ-calculus.

Even more generally, if $M \rhd N$ in SKIn, then $[\![M]\!](P_0, \ldots, P_{n-1}) \rhd_\eta^* [\![N]\!](P_0, \ldots, P_{n-1})$ in the λ-calculus, for any $n \geq 0$ and λ-terms $P_0, \ldots, P_{n-1}$.

Proof. By structural induction on the depth of the contracted redex in M, then by case analysis on the reduction rule that we used. See A for details.

The need for the η rule arises when we translate (SI_m) redexes. Consider for example $\mathrm{S}_1(\mathrm{I}_1, x)$, where x is a variable. Then:

$$[\![\mathrm{S}_1(\mathrm{I}_1, x)]\!]() \equiv \lambda x_0.[\![\mathrm{I}_1]\!](x_0, [\![x]\!](x_0)) \equiv \lambda x_0.[\![x]\!](x_0) \equiv \lambda x_0.x(x_0)$$

and this requires the use of η to be reduced to $x \equiv [\![x]\!]()$. □

3.3. *Soundness*

Soundness extends Curry's programme of using combinators as a bound-variable-free language for the λ-calculus. There are several steps to this:

- Functional Abstraction. This involves defining the abstraction operation $[x]M$ by structural induction on M (Definition 3.6).
- Substitution. This is the standard operation of textual substitution $M[x \leftarrow P]$.
- Soundness. We show that abstraction and substitution are coherent, and that reduction is preserved by abstraction. Hence, the reduction and equality of λ-calculus are preserved by the translation to sequent combinators (Theorem 3.14, Corollary 3.15).

The abstraction algorithm corresponds to the deduction theorem for intuitionistic implicational logic: if from Γ, A we can prove B then we can prove $A \Rightarrow B$ from Γ. Technically, because the combinators include information about their list of arguments, the algorithm is particularly easy to define.

Definition 3.6 (Abstraction). Let $[x]M$ be defined inductively by:

$$\begin{array}{rcl} [x]x =_{\mathrm{df}} \mathrm{I}_0 & & [x]y =_{\mathrm{df}} \mathrm{K}_0(y) \quad (y \not\equiv x) \\ [x]\mathrm{I}_m & =_{\mathrm{df}} & \mathrm{I}_{1+m} \\ [x]\mathrm{K}_m(M) & =_{\mathrm{df}} & \mathrm{K}_{1+m}([x]M) \\ [x]\mathrm{S}_m(M, N) & =_{\mathrm{df}} & \mathrm{S}_{1+m}([x]M, [x]N) \end{array}$$

Notice that, unlike some definitions of abstraction in combinatory logic, we do not need a special case to define abstraction $[x]M$ for terms M where x does not occur free, i.e., as $\mathrm{K}_0(M)$. This is because our calculus is sufficiently expressive that $\mathrm{K}_0(M) \triangleright^* [x]M$ holds, using the thinning reductions.

We invite the reader to check that the corresponding abstraction algorithm for λ-terms *in de Bruijn notation* is even simpler, and in essence does nothing, as noticed by G. Dowek. In fact, the abstraction algorithm is essentially a translation to de Bruijn notation.

We can now translate untyped λ-terms into the combinator language:

Definition 3.7. Define P^* inductively on the structure of the λ-term P:

$$\begin{array}{rcl} x^* & =_{\mathrm{df}} & x \\ (P(Q))^* & =_{\mathrm{df}} & \mathrm{S}_0(P^*, Q^*) \\ (\lambda x.P)^* & =_{\mathrm{df}} & [x](P^*) \end{array}$$

Our notion of substitution $M[x \leftarrow N]$ is the trivial, first-order, textual notion.

The usual development of combinators then translates into our setting without difficulty. The contribution of our language is that ξ-reduction holds: if $M \triangleright N$ then $[x]M \triangleright [x]N$.

Lemma 3.8. $\mathrm{S}_0([x]M, N) \triangleright^+ M[x \leftarrow N]$.

Proof. By structural induction on M. If $M \equiv x$, then $\mathrm{S}_0([x]M, N) \equiv \mathrm{S}_0(\mathrm{I}_0, N) \triangleright N \equiv M[x \leftarrow N]$ by rule (SI_0). If $M \equiv y \not\equiv x$, then $\mathrm{S}_0([x]M, N) \equiv \mathrm{S}_0(\mathrm{K}_0(y), N) \triangleright y \equiv M[x \leftarrow N]$ by rule (SK_0). Otherwise, M is of the form $f_m(M_1, \dots, M_n)$ for some operator f_m, $m \geq 0$, $f \in \{\mathrm{S}, \mathrm{K}, \mathrm{I}\}$ (when f is I, then $n = 0$), so $\mathrm{S}_0([x]M, N) \equiv \mathrm{S}_0(f_{m+1}([x]M_1, \dots, [x]M_n), N) \triangleright f_m(\mathrm{S}_0([x]M_1, N), \dots, \mathrm{S}_0([x]M_n, N))$ (by rule $(\mathrm{S}_0 f_{m+1})$) $\triangleright^* f_m(M_1[x \leftarrow N], \dots, M_n[x \leftarrow N])$ (by induction hypothesis) $\equiv M[x \leftarrow N]$. ☐

Lemma 3.9. In SKIn, $[x]M \equiv [y](M[x \leftarrow y])$ if $y \notin \mathrm{FV}(M)$.

Proof. By structural induction on M. ☐

Lemma 3.10. In SKIn, if $M \triangleright N$, then $[x]M \triangleright [x]N$.

Proof. By induction on the depth of the redex that gets contracted in M.

Base case: M is itself the contracted redex. If $M \triangleright N$ by (SI_m), then $[x]M \triangleright [x]N$ by (SI_{m+1}): indeed, we must have $M \equiv \mathrm{S}_m(\mathrm{I}_m, P)$ and $N \equiv P$, so $[x]M \equiv \mathrm{S}_{m+1}(\mathrm{I}_{m+1}, [x]P) \triangleright [x]P \equiv [x]N$ by (SI_{m+1}). Similarly, abstraction on x translates (SK_m) to (SK_{m+1}), $(\mathrm{S}_m\mathrm{I}_p)$ to $(\mathrm{S}_{m+1}\mathrm{I}_{p+1})$, and so on.

Induction case: M is not the contracted redex. So $M \equiv f_m(M_1, \dots, M_n)$ for some operator f_m and the contracted redex is inside some M_i, $1 \leq i \leq n$. Let M_i contract to N_i, and N_j be M_j for every $j \neq i$. Then $[x]M \equiv f_{m+1}([x]M_1, \dots, [x]M_i, \dots, [x]M_n) \triangleright f_{m+1}([x]M_1, \dots, [x]N_i, \dots, [x]M_n)$ (by induction hypothesis), and by definition, the latter is equal to $f_{m+1}([x]N_1, \dots, [x]N_i, \dots, [x]N_n)$, i.e. to $[x]N$. ☐

Lemma 3.11. For every SKIn-term M, for every variable y not free in M, $\mathrm{K}_0(M) \triangleright^* [y]M$.

Proof. By structural induction on M. If M is a variable x, then $[y]M \equiv \mathrm{K}_0(x)$; indeed, y is not free in M, so $y \not\equiv x$. It follows that in this case, $\mathrm{K}_0(M) \equiv [y]M$. If M is any other term, then M has the form $f_m(M_1, \ldots, M_n)$ for some operator f_m and some $n \geq 0$; then $\mathrm{K}_0(M) \triangleright f_{m+1}(\mathrm{K}_0(M_1), \ldots, \mathrm{K}_0(M_n))$ by rule $(\mathrm{K}_0 f_{m+1})$, and by induction hypothesis, the latter reduces to $f_{m+1}([y]M_1, \ldots, [y]M_n) \equiv [y]M$. □

Lemma 3.12. For all distinct variables x and y, and for all SKIn-terms M and N such that y is not free in N, $([y]M)[x \leftarrow N] \triangleright^* [y](M[x \leftarrow N])$.

Proof. By structural induction on M. If $M \equiv y$, then $([y]M)[x \leftarrow N] \equiv \mathrm{I}_0[x \leftarrow N] \equiv \mathrm{I}_0 \equiv [y]M \equiv [y](M[x \leftarrow N])$, where $M \equiv M[x \leftarrow N]$ since $x \not\equiv y$. If $M \equiv x$, then $([y]M)[x \leftarrow N] \equiv (\mathrm{K}_0(x))[x \leftarrow N]$ (since $y \not\equiv x$) $\equiv \mathrm{K}_0(N) \triangleright^* [y]N$ (by Lemma 3.11) $\equiv [y](M[x \leftarrow N])$. If M is any other term, then it is of the form $f_m(M_1, \ldots, M_n)$, and $([y]M)[x \leftarrow N] \equiv f_{m+1}([y]M_1, \ldots, [y]M_n)[x \leftarrow N] \equiv f_{m+1}(([y]M_1)[x \leftarrow N], \ldots, ([y]M_n)[x \leftarrow N]) \triangleright^* f_{m+1}([y](M_1[x \leftarrow N]), \ldots, [y](M_n[x \leftarrow N]))$ (by induction hypothesis) $\equiv [y](M[x \leftarrow N])$. □

Lemma 3.13. For all λ-terms M and N and all variables x, $M^*[x \leftarrow N^*] \triangleright^* ([N/x]M)^*$.

Proof. By induction on the size of M.

If $M \equiv x$, then $M^*[x \leftarrow N^*] \equiv x[x \leftarrow N^*] \equiv N^* \equiv ([N/x]M)^*$. If M is another variable y, then $M^*[x \leftarrow N^*] \equiv y \equiv y^* \equiv ([N/x]M)^*$.

If M is an application $M_1(M_2)$, then $M^*[x \leftarrow N^*] \equiv (\mathrm{S}_0(M_1^*, M_2^*))[x \leftarrow N^*] \equiv \mathrm{S}_0(M_1^*[x \leftarrow N^*], M_2^*[x \leftarrow N^*]) \triangleright^* \mathrm{S}_0(([N/x]M_1)^*, ([N/x]M_2)^*)$ (by induction hypothesis) $\equiv ([N/x]M)^*$.

If M is an abstraction $\lambda x.M_1$ then $(\lambda x.M_1)^*[x \leftarrow N^*] \equiv (\lambda x.M_1)^*$ (since $x \notin \mathrm{FV}((\lambda x.M_1)^*) \equiv ([N/x](\lambda x.M_1))^*$. And if M is an abstraction $\lambda y.M_1$, then $M^*[x \leftarrow N^*] \equiv ([y]M_1^*)[x \leftarrow N^*] \equiv ([z](M_1^*[y \leftarrow z]))[x \leftarrow N^*]$ (for $z \notin \mathrm{FV}(N^*)$, by Lemma 3.9) $\triangleright^* [z]((M_1^*)[y \leftarrow z][x \leftarrow N^*])$ (by Lemma 3.12) $\triangleright^* [z]([N/x][z/y]M_1)^*$ (by Lemma 3.10 and induction hypothesis—notice that the size of M_1 and $[z/y]M_1$ are equal) $\equiv ([N/x](\lambda y.M_1))^*$. □

Theorem 3.14 (Soundness for Reduction). If $M \triangleright N$ then $M^* \triangleright^+ N^*$.

Proof. By structural induction on M, using Lemma 3.8 and Lemma 3.13 for β, and Lemma 3.10 for ξ. □

Corollary 3.15 (Soundness for Equality). If $M = N$ then $M^* = N^*$.

Preservation of typing is straightforward:

Lemma 3.16. For every SKIn-term N, if $\vdash_{\Gamma, x:A} N : B$, then $\vdash_\Gamma [x]N : A \Rightarrow B$.

Proof. By structural induction on N. If $N \equiv x$, then B is A, and $[x]N \equiv \mathrm{I}_0$ is of type $A \Rightarrow A$. If N is some other variable y, then $y{:}B$ is in Γ, so $\vdash_\Gamma \mathrm{K}_0(y) : A \Rightarrow B$, and the claim follows since $[x]N \equiv \mathrm{K}_0(y)$. If $N \equiv \mathrm{I}_m$, then B is of the form $(A_0, \ldots, A_{m-1}, A_m) \rightarrow$

A_m, and $[x]N \equiv \mathrm{I}_{m+1}$, and the claim follows since $\vdash_\Gamma \mathrm{I}_{m+1} : (A, A_0, \ldots, A_{m-1}, A_m) \rightarrow A_m$. If $N \equiv \mathrm{K}_m(P)$, then B is of the form $(A_0, \ldots, A_{m-1}, A_m) \rightarrow A_{m+1}$, and $\vdash_{\Gamma,x:A} P : (A_0, \ldots, A_{m-1}) \rightarrow A_{m+1}$; by induction hypothesis, $\vdash_\Gamma [x]P : (A, A_0, \ldots, A_{m-1}) \rightarrow A_{m+1}$, so $\vdash_\Gamma \mathrm{K}_{m+1}([x]P) : (A, A_0, \ldots, A_{m-1}, A_m) \rightarrow A_{m+1}$; when $N \equiv \mathrm{S}_m(P, Q)$, the argument is similar. □

Lemma 3.17. If $\Gamma \vdash M : A$ then $\vdash_\Gamma M^* : A$.

Proof. By structural induction on M, where $\Gamma \vdash M : A$. If M is a variable x, then $x{:}A$ is in Γ, so $\vdash_\Gamma x : A$, that is, $\vdash_\Gamma M^* : A$. If M is an application $P(Q)$, then necessarily $\Gamma \vdash P : B \Rightarrow A$ and $\Gamma \vdash Q : B$ for some type B. By induction hypothesis, $\vdash_\Gamma P^* : B \Rightarrow A$ and $\vdash_\Gamma Q^* : B$, so $\vdash_\Gamma \mathrm{S}_0(P^*, Q^*) : A$, that is, $\vdash_\Gamma M^* : A$. If M is an abstraction $\lambda x.N$, then necessarily A is of the form $B \Rightarrow C$, with $\Gamma, x{:}B \vdash N : C$, so by induction hypothesis $\vdash_{\Gamma,x:B} N^* : C$. By Lemma 3.16, then, $\vdash_\Gamma [x]N^* : B \Rightarrow C$, therefore $\vdash_\Gamma M^* : A$. □

In this section, we show that the map $M \mapsto M^*$ is a conservative embedding of λ-terms in SKIn modulo conversion: that is, $M^* = N^*$ if and only if $M = N$. One direction is soundness. The other direction requires the following auxiliary notion of well-stagedness.

Definition 3.18. Define the *level* $lev(M)$ of the SKIn-term M by:

$$\begin{array}{rcl} lev(x) & = & 0 \\ lev(\mathrm{I}_m) & = & m+1 \\ lev(\mathrm{S}_m(M,N)) & = & \max(m, lev(M)-1) \\ lev(\mathrm{K}_m(M)) & = & \max(m, lev(M))+1 \end{array}$$

A SKIn-term is *well-staged* if and only if all its subterms of the form $\mathrm{S}_m(M,N)$ are such that $lev(M) \geq m$, $lev(N) \geq m$, and all its subterms of the form $\mathrm{K}_m(M)$ are such that $lev(M) \geq m$.

Intuitively, $lev(M)$ is the number of lambdas in front of $[\![M]\!]()$; in particular, $lev(P^*)$, where $P = \lambda x_1 \ldots \lambda x_n.Q$ and Q is not an abstraction, is n. More generally, for any non-abstraction subterm Q of P, Q^* will appear abstracted n times, where there are n bound variables in the scope of Q—hence as a term of level n. Now, when we go down the term P, the number of bound variables in scope can only increase. The notion of well-stagedness is intended to represent this monotonicity property.

Then:

Lemma 3.19. Let M be a λ-term. Then M^* is well-staged.

Recall that we shall show later on (Theorem 5.4) that SKIn is confluent. Taking this as an assumption for the moment, we can refine Theorem 3.5 in the case of well-staged terms:

Theorem 3.20 (Completeness for Equality (2)). For any well-staged terms M and N, if $M = N$ in SKIn, then $[\![M]\!]() = [\![N]\!]()$, provided that SKIn is confluent.

In general, if M is well-staged and $M \triangleright N$ in SKIn, then N is also well-staged and $[\![M]\!]() \triangleright^* [\![N]\!]()$ in the λ-calculus.

Even more generally, if M is well-staged and $M \rhd N$ in SKIn, then $[\![M]\!](P_0, \ldots, P_{n-1}) \rhd^* [\![N]\!](P_0, \ldots, P_{n-1})$ in the λ-calculus, for any $n \geq 0$ and λ-terms $P_0, \ldots, P_{n-1}$.

Proof. Full details are given in B. Note that the example we gave in Theorem 3.5 to show that we needed the η rule, $\mathrm{S}_1(\mathrm{I}_1, x)$, is *not* well-staged. □

The following corollary does not follow from Theorem 3.5, since the latter would only show $[\![M^*]\!]() =_\eta [\![N^*]\!]()$. We need this refinement in order to show conservativity.

Corollary 3.21. If SKIn is confluent then $M^* = N^*$ implies $[\![M^*]\!]() = [\![N^*]\!]()$.

Lemma 3.22. For any λ-term P, $P \rhd^* [\![P^*]\!]()$.

Theorem 3.23 (Conservativity (1)). Assume that SKIn is confluent.

Then, the map $M \mapsto M^*$ is a conservative embedding of λ-terms in SKIn modulo conversion. That is, $M^* = N^*$ if and only if $M = N$.

Proof. The if direction follows by Soundness.

The only-if direction follows because $M = [\![M^*]\!]() = [\![N^*]\!]() = N$, by Corollary 3.21 and Lemma 3.22. □

Lemma 3.22 implies that, modulo conversion, $[\![-]\!]()$ is a left inverse to $-^*$. The curious reader might wonder whether it is also a right inverse to $-^*$. An anonymous referee has suggested that this might be the case, and in fact that we might have $M \rhd^* ([\![M]\!]())^*$ for every SKIn-term M. Unfortunately, this does not work: take $M =_{\mathrm{df}} \mathrm{S}_1(\mathrm{I}_1, x)$, then $([\![M]\!]())^* \equiv (\lambda x_0 . x(x_0))^* \equiv \mathrm{S}_1(\mathrm{K}_0(x), \mathrm{I}_0)$. Then M only reduces to x, which is not the same normal form as $\mathrm{S}_1(\mathrm{K}_0(x), \mathrm{I}_0)$. In fact, the best we can hope for here is that $M =_\eta ([\![M]\!]())^*$, which we conjecture.

If M is well-staged, we conjecture that indeed $M \rhd^* ([\![M]\!]())^*$. However, we have not yet been able to prove this. If this conjecture holds, then there would be a simple confluence proof of SKIn restricted to well-staged terms: if the well-staged term M reduces both to M_1 and to M_2 in SKIn, then by Theorem 3.20 $[\![M]\!]()$ reduces both to $[\![M_1]\!]()$ and to $[\![M_2]\!]()$ in the λ-calculus. Since the latter is confluent, the latter two have a common reduct P, so $([\![M_1]\!]())^*$ and $([\![M_2]\!]())^*$ both reduce to P^* in SKIn, by Theorem 3.14. The conjecture would imply that $([\![M_1]\!]())^*$ and $([\![M_2]\!]())^*$ would be reducts of M_1 and M_2 respectively, so that P^* would be a common reduct of M_1 and M_2. This was suggested to us by the same referee; we had abandoned this line of argument since it is impossible to extend this to a proof that the whole of SKIn was confluent.

One might also wonder whether, conversely to Theorem 3.23, the mapping $M \mapsto [\![M]\!]()$ defines a conservative embedding of SKIn inside the λ-calculus. The answer is no, and the same counter-example still works: $\mathrm{S}_1(\mathrm{I}_1, x)$ and $\mathrm{S}_1(\mathrm{K}_0(x), \mathrm{I}_0)$ are not convertible in SKIn, whereas $[\![\mathrm{S}_1(\mathrm{I}_1, x)]\!]() \equiv \lambda x_0 . x(x_0) \equiv [\![\mathrm{S}_1(\mathrm{K}_0(x), \mathrm{I}_0)]\!]()$. However, the conjecture above implies that the subset of all well-staged SKIn-terms is conservatively embedded inside the λ-calculus.

4. SKInT: Terminating Sequent Combinators, and Near-Intuitionistic Logic

We now introduce SKInT. This can be seen as a variant on SKIn, where the rule $(\mathrm{K}_m\mathrm{S}_{m+1})$: $\mathrm{K}_m(\mathrm{S}_m(M,N)) \triangleright \mathrm{S}_{m+1}(\mathrm{K}_m(M),\mathrm{K}_m(N))$, i.e. rule $(\mathrm{K}_m\mathrm{S}_n)$ where $n = m+1$, is dropped. This is enough to ensure strong normalization in the typed case; in the example of Proposition 3.3 in particular, this defeats the first reduction. This also facilitates the design of the η rules: the only η-rules that we need in SKInT are $\mathrm{S}_{m+1}(\mathrm{K}_m(M),\mathrm{I}_m) \triangleright_\eta M$, $m \geq 0$.

The rule $(\mathrm{K}_m\mathrm{S}_{m+1})$ is needed in the proof of Soundness for Reduction, Theorem 3.14, and in particular in Lemma 3.12 allowing reduction to pass under the binder. For example, it is necessary to show that $((\lambda x.\lambda y.x)(a(b)))^* = \mathrm{S}_0(\mathrm{K}_1(\mathrm{I}_0),\mathrm{S}_0(a,b)) \triangleright^+ \mathrm{K}_0(\mathrm{S}_0(a,b)) \triangleright \mathrm{S}_1(\mathrm{K}_0(a),\mathrm{K}_0(b)) = (\lambda y.a(b))^*$.

This discussion only justifies SKInT on ad hoc grounds. However, SKInT arose from deeper considerations on the modal logic S4. In fact, SKInT will happen to have very few properties in common with SKIn: presenting it as a variant of SKIn can be seen as a mere convenience. SKInT is indeed a natural sub-calculus of the $\lambda\mathtt{ev}Q$-calculus (Goubault-Larrecq, 1996a), a calculus of proof terms for the minimal modal logic S4. This calculus basically implements a well-behaved Lisp-like eval/quote mechanism, presented as an infinite tower of calculi (Wand and Friedman, 1986): in $\lambda\mathtt{ev}Q$, $\mathrm{S}_m(M,N)$ is $\mathtt{ev}^{m+1}MN$ (eval M in stack N, at level $m+1$ of the tower), $\mathrm{K}_m(M)$ is $Q^{m+1}M$ (`kwote`, at level $m+1$), and I_m is the empty stack id^{m+1} at level $m+1$. The connection between SKInT and $\lambda\mathtt{ev}Q$ is not arbitrary either, and arises from the translation of intuitionistic logic to S4 induced by their respective Kripke semantics (the translation $[A]$ of the formula A being defined by $[o] =_{\mathrm{df}} \Box o$, $[A \Rightarrow B] =_{\mathrm{df}} \Box([A] \Rightarrow [B])$).

We start therefore by giving a semantical account of SKInT through *near-intuitionistic logic*, a fragment of S4 which can also be seen as a relaxation on the usual Kripke semantics of intuitionistic logic. This will occupy Section 4.1. We shall then investigate the meta-mathematical properties of SKInT: termination properties in Section 4.2, confluence in Section 4.3, and standardization in Section 4.4.

Our intention in this section is only to motivate SKInT as a natural system in its own right, and as a platform for developing results for SKInT. We leave the full development of SKInT to another paper.

4.1. *Near-Intuitionistic Logic and the λ_{clos}-Calculus*

Definition 4.1. A *near-intuitionistic*, or *S4*, *frame* is a triple $(\mathcal{W},\leq,\rho)$, where $\mathcal{W}$ is a non-empty set of so-called *worlds* w, $\leq$ is a pre-order on $\mathcal{W}$ (a reflexive and transitive relation), and ρ is a valuation mapping base types $o \in O$ to subsets of $\mathcal{W}$.

The relation $\models_{(\mathcal{W},\leq,\rho)}$, or $\models$ when the frame is clear, is defined by:

$$\begin{array}{rcll} w \models o & \text{iff} & w \in \rho(o) & \qquad (o \in O) \\ w \models A \Rightarrow B & \text{iff} & \forall w' \cdot w \leq w' \supset w' \models A \supset w' \models B & \end{array}$$

where $\supset$ is ordinary implication.

We call *near-intuitionistic logic* the logic whose formulas are types, and such that the true formulas are those that hold in every frame and under every valuation.

Frames are defined as for the modal logic S4; the only difference with intuitionistic frames is that we do not require $\rho(o)$ to be upwards-closed, i.e. to be such that $w \in \rho(o)$ implies $w' \in \rho(o)$ for every w' such that $w \leq w'$. That is, this is a variant of intuitionistic logic where propositions in a world may fail to remain true in all subsequent worlds. Alternatively, Definition 4.1 can be seen as the semantics of a particular fragment of S4, where $A \Rightarrow B$ means the same as the S4 formula $\Box(A \supset B)$.

The typing rules of SKInT are as for SKIn, except for rule (K), which is now restricted to the case where A is an arrow type. We name the restricted rule (KT). Here is, then, the whole set of typing rules for SKInT:

$$\frac{x{:}A \in \Gamma}{\vdash_\Gamma x : () \to A} \quad (\textsc{Par})$$

$$\frac{}{\vdash_\Gamma \mathrm{I}_{|X|} : (X, A) \to A} \quad (\mathrm{I})$$

$$\frac{\vdash_\Gamma M : (X, C) \to A}{\vdash_\Gamma \mathrm{K}_{|X|}(M) : (X, B, C) \to A} \quad (\mathrm{KT})$$

$$\frac{\vdash_\Gamma M : (X, A) \to B \qquad \vdash_\Gamma N : X \to A}{\vdash_\Gamma \mathrm{S}_{|X|}(M, N) : X \to B} \quad (\mathrm{S})$$

We shall see that the typing rules of SKInT define a sound and complete deduction system for near-intuitionistic logic. Although we could prove this directly, it will be instructive to go through another language of proof-terms instead, λ_{clos}, and to show, first, that the latter is a language of proof-terms for near-intuitionistic logic, and second, that it is isomorphic to SKInT (modulo convertibility, and in the presence of η rules). As λ_{clos} is very close to the λ-calculus, this will weave a more precise web of relationships between these languages. This will also show that, in a certain sense, SKInT-reduction, as λ_{clos}-reduction, represents cut-elimination in a sequent system for near-intuitionistic logic.

The λ_{clos}*-calculus* is inspired by the variant of Bierman-de Paiva's λ_{S4} (Bierman and de Paiva, 1992) used in (Goubault-Larrecq, 1996b).

Definition 4.2. The λ_{clos}-pre-terms s, t, ..., are defined as:

$$t ::= x \mid t(t) \mid \langle x \cdot t, \theta \rangle$$

where θ is an explicit *substitution*, i.e. a finite mapping $\{x_1 := t_1, \ldots, x_k := t_k\}$ from variables x_i to λ_{clos}-terms t_i, $1 \leq i \leq k$. Terms of the form $\langle x \cdot t, \theta \rangle$ are called *closures*; $x \cdot t$ is the *code* part of the closure, x is its *argument*, t is its *body* and θ is the *environment* part.

The variable x and the variables in the domain of θ are bound in the body t of any closure $\langle x \cdot t, \theta \rangle$.

The λ_{clos}*-terms* are those pre-terms whose closure subterms $\langle x \cdot t, \theta \rangle$ have the property that every variable free in t, except x, is in the domain of θ.

Intuitively, closures are pairs of a piece of code computing the value of t when given a value for x as argument in the environment θ; the role of θ is to map the free variables of t

(except x) to their respective values while evaluating t. We might have written $\langle \lambda x \cdot t, \theta \rangle$ to remind the reader of the usual λ-notation, but the λ sign is redundant with the brackets, so we have dispensed with it in order not to clutter the notation.

Substitution is defined as usual, but the constraint on closures in λ_{clos}-terms entails that substitution on closures only operates on the environment, not the body. Formally:

Definition 4.3. Let $\sigma =_{\mathrm{df}} \{y_1 := u_1, \dots, y_p := u_p\}$ be a substitution. The *application* $s\sigma$ of σ to the λ_{clos}-term s is defined by:

$$\begin{array}{rcll} y_i\sigma & =_{\mathrm{df}} & u_i & (1 \le i \le p) \\ y\sigma & =_{\mathrm{df}} & y & (y \not\equiv y_i, 1 \le i \le p) \\ (s(t))\sigma & =_{\mathrm{df}} & s\sigma(t\sigma) & \\ \langle x \cdot t, \{x_1 := t_1, \dots, x_k := t_k\}\rangle\sigma & =_{\mathrm{df}} & \langle x \cdot t, \{x_1 := t_1\sigma, \dots, x_k := t_k\sigma\}\rangle & \end{array}$$

The fact that the substitution σ is not applied to the body t of the closure in the last line is correct: the only free variables in t are either the bound variable x, or are bound by the environment part of the closure.

We adopt Barendregt's convention on variable naming, so that no variable is bound at two distinct occurrences or occurs both bound and free in a term; we shall sometimes violate this condition (like in the definition of $\langle x \cdot t \rangle$ below) in the name of increased readability, under the convention that the intended term is obtained by some obvious α-renaming. Finally, α-equivalent terms are equated, where α-equivalence is the smallest congruence $\equiv$ such that:

$$\begin{array}{ll} (\alpha) & \langle x' \cdot t\{x := x', x_1 := x'_1, \dots, x_k := x'_k\}, \{x'_1 := t_1, \dots, x'_k := t_k\}\rangle \\ & \equiv \langle x \cdot t, \{x_1 := t_1, \dots, x_k := t_k\}\rangle \end{array}$$

where x, x_1, ..., x_k are pairwise distinct, and so are x', x'_1, ..., x'_k. That is, not only is the name of the argument x irrelevant, but so are the names of the bound variables x_1, ..., x_n in the domain of the environment.

To make the notation somewhat more transparent, we define $\langle x \cdot t \rangle$ as an abbreviation for $\langle x \cdot t, \{x_1 := x_1, \dots, x_k := x_k\}\rangle$, where x_1, ..., x_k are the free variables of t except x. We call terms $\langle x \cdot t \rangle$ *abstractions*; it follows that every closure $\langle x \cdot t, \theta \rangle$ can be written as an abstraction on which the substitution θ has been applied: $\langle x \cdot t, \theta \rangle \equiv \langle x \cdot t \rangle \theta$.

The reduction relation of λ_{clos} is defined as the compatible closure of:

$$\begin{array}{lll} (\beta) & \langle x \cdot t, \theta \rangle(s) & \triangleright \ t(\theta \cup \{x := s\}) \\ (\iota) & \multicolumn{2}{l}{\langle x \cdot t, \{x_1 := \langle y \cdot s, \{y_1 := s_1, \dots, y_p := s_p\}\rangle, x_2 := t_2, \dots, x_k := t_k\}\rangle,} \\ & & \triangleright \ \langle x \cdot t\{x_1 := \langle y \cdot s, \{y_1 := y_1, \dots, y_p := y_p\}\rangle\} \\ & & \quad \{y_1 := s_1, \dots, y_p := s_p, x_2 := t_2, \dots, x_k := t_k\}\rangle \\ (0) & \multicolumn{2}{l}{\langle x \cdot t, \{x_1 := t_1, x_2 := t_2, \dots, x_k := t_k\}\rangle} \\ & & \triangleright \ \langle x \cdot t, \{x_2 := t_2, \dots, x_k := t_k\}\rangle \qquad (x_1 \text{ not free in } t) \\ (2) & \multicolumn{2}{l}{\langle x \cdot t, \{x_1 := t_1, x_2 := t_2, \dots, x_k := t_k\}\rangle} \\ & & \triangleright \ \langle x \cdot t\{x_1 := x_2\}, \{x_2 := t_2, \dots, x_k := t_k\}\rangle \qquad (\text{if } t_1 \equiv t_2) \end{array}$$

where $\{x_1 := t_1, \dots, x_k := t_k\} \cup \{x := s\}$ is defined as $\{x_1 := t_1, \dots, x_k := t_k, x := s\}$, when $x \not\equiv x_i$, for every i, $1 \le i \le k$.

Some comments are in order. Rule (β) is essentially just ordinary β-reduction for closures. Due to the variable naming convention, it would be equivalent to write $t\theta\{x := s\}$ or $t\{x := s\}\theta$ instead of $t(\theta \cup \{x := s\})$.

The idea of rule (ι) is that whenever some free variable x_1 in t (except x) is mapped to a closure $\langle y \cdot s, \{y_1 := s_1, \ldots, y_p := s_p\}\rangle$, we can push the code part $y \cdot s$ of x_1 inside the body t of the closure itself, retaining the environment part $y_1 := s_1, \ldots, y_p := s_p$ in the environment of the outer closure; this can be thought as an inlining rule, as used in compilers for functional languages. Note that we might have chosen any x_i instead of x_1, but since environments are functions, bindings $x_i := t_i$ commute freely: choosing x_1 does not entail any loss of generality. Logicians will also notice from the typing rules to come that this is the box-under-box rule of S4 or of linear logic.

Rule (0) is a garbage collection rule: it expresses that there is no need to keep a binding $x_1 := t_1$ in the environment when the code part does not refer to x_1. Rule (2) is a contraction rule: if x_1 and x_2 are bound to the same term t_1 in the environment, we can eliminate one binding, say $x_1 := t_1$, and replace x_1 by x_2 in the body t of the code part.

Our first result, which we shall use later on, is that λ_{clos} defines, in a natural way, a superset of the call-by-value λ-calculus (Plotkin, 1975):

Theorem 4.4. Define the following translation function from λ-terms M to λ_{clos}-terms $M^\sharp$ as follows:

$$\begin{array}{rcl} x^\sharp & =_{\mathrm{df}} & x \\ (M(N))^\sharp & =_{\mathrm{df}} & M^\sharp(N^\sharp) \\ (\lambda x.M)^\sharp & =_{\mathrm{df}} & \langle x \cdot M^\sharp\rangle \end{array}$$

Let $\triangleright_V$ denote the notion of reduction in the λ_V-calculus, a.k.a. the call-by-value λ-calculus. This is the compatible closure of the rule:

$$(\beta_V) \quad (\lambda x.M)(V) \triangleright_V [V/x]M$$

where V is a *value*, i.e. any non-application term.

Then $M \triangleright_V N$ implies $M^\sharp \triangleright^+ N^\sharp$ in λ_{clos}.

Proof. First, $((\lambda x.M)(V))^\sharp \triangleright M^\sharp\{x := V^\sharp\}$ in λ_{clos}, by rule (β). It remains to show that $M^\sharp\{x := V^\sharp\} \triangleright^* ([V/x]M)^\sharp$, by structural induction on M. The only non-trivial case is when M is a λ-abstraction $\lambda y \cdot N$: then $M^\sharp\{x := V^\sharp\} \equiv \langle y \cdot N^\sharp\rangle\{x := V^\sharp\}$. If x is not free in N, hence in $N^\sharp$, then the result is obvious. Otherwise, assume x free in N, hence also in $N^\sharp$, by an easy structural induction on N. If V is a variable, then the result follows by α-renaming. If V is a λ-abstraction $\lambda z.P$, then $V^\sharp \equiv \langle z \cdot P^\sharp\rangle$ is a closure; then $\langle y \cdot N^\sharp\rangle\{x := V^\sharp\} \equiv \langle y \cdot N^\sharp\rangle\{x := \langle z \cdot P^\sharp\rangle\} \triangleright \langle y \cdot N^\sharp\{x := \langle z \cdot P^\sharp\rangle\}\rangle$ (by (ι)) $\triangleright^* \langle y \cdot ([\lambda z.P/x]N)^\sharp\rangle$ (by induction hypothesis) $\equiv \langle y \cdot ([V/x]N)^\sharp\rangle \equiv ([V/x]M)^\sharp$. □

Through the $M \mapsto M^\sharp$ translation, the λ_{clos}-calculus defines a *strict* superset of λ_V. Indeed, let x, y, and z be three variables, then $(\lambda x.x)(y(z))$ and $y(z)$ are not convertible in λ_V, but $((\lambda x.x)(y(z)))^\sharp \equiv \langle x \cdot x\rangle(y(z)) \triangleright y(z) \equiv (y(z))^\sharp$ in λ_{clos}.

We shall also consider the following η-rule:

Definition 4.5. The $\lambda_{clos\eta}$-calculus is defined by the reduction relation $\triangleright_\eta$, the compat-

$$\overline{\Gamma, x{:}A \vdash x \,:\, A} \quad (\textsc{Ax})$$

$$\frac{\Gamma \vdash s \,:\, A \Rightarrow B \quad \Gamma \vdash t \,:\, A}{\Gamma \vdash s(t) \,:\, B} \quad (\textsc{App})$$

$$\frac{\begin{array}{c}\Gamma \vdash t_1 \,:\, B_1 \Rightarrow C_1 \quad \ldots \quad \Gamma \vdash t_k \,:\, B_k \Rightarrow C_k \\ x_1{:}B_1 \Rightarrow C_1, \ldots, x_k{:}B_k \Rightarrow C_k, x{:}B \vdash s \,:\, C\end{array}}{\Gamma \vdash \langle x \cdot s, \{x_1 := t_1, \ldots, x_k := t_k\}\rangle \,:\, B \Rightarrow C} \quad (\textsc{Clos})$$

Fig. 3. Typing λ_{clos}

ible closure of (β), (ι), (0), (2) and the following η-rule:

$$(\eta) \quad \langle x \cdot z(x), \theta\rangle \triangleright z\theta$$

where z is a variable other than x.

Notice that z here is a *variable*, not any term with x not free in it, as in the λ-calculus.

The simple typing rules for λ_{clos} are as shown in Figure 3.

The typing rule for closures can be viewed as a combination of a rule to type abstractions:

$$\frac{\Gamma^{\Rightarrow}, x{:}B \vdash s \,:\, C}{\Gamma^{\Rightarrow}, \Delta \vdash \langle x \cdot s\rangle \,:\, B \Rightarrow C} \quad (\textsc{Abs})$$

where $\Gamma^{\Rightarrow}$ is restricted to be an *arrowed* context, i.e. one mapping each variable x_i to an arrow type $B_i \Rightarrow C_i$, and Δ is any context, that we use to build in weakening; and of a rule (Cut) to type (implicit) substitutions:

$$\frac{\Gamma \vdash t \,:\, A \quad \Gamma, x{:}A \vdash s \,:\, B}{\Gamma \vdash s\{x := t\} \,:\, B} \quad (\textsc{Cut})$$

The latter is the famous cut rule of sequent calculi, and is easily shown to be admissible in the system of Figure 3, by structural induction on s.

Our first point is to show that the natural deduction system, i.e. the typing system of Figure 3, considering types as formulae, and λ_{clos}-terms as proof-terms, is the proof-theoretical counterpart of the semantics of Definition 4.1:

Theorem 4.6. The deduction rules of Figure 3 are sound and complete for the semantics of Definition 4.1. That is, given a context $x_1{:}A_1, \ldots, x_n{:}A_n$ and a type A_0, there is a λ_{clos}-term t such that $x_1{:}A_1, \ldots, x_n{:}A_n \vdash t \,:\, A_0$ is derivable in the system of Figure 3 if and only if, for every near-intuitionistic frame $(\mathcal{W}, \leq, \rho)$, for every $w \in W$ such that $w \models A_1$ and $\ldots$ and $w \models A_n$, $w \models A_0$, where $\models$ abbreviates $\models_{(\mathcal{W},\leq,\rho)}$.

Proof. Soundness: by structural induction on t. The only non-trivial case is for closures $\langle x \cdot s, \{x_1 := t_1, \ldots, x_k := t_k\}\rangle$: soundness works in this case because any true *arrow* type must remain true in all later worlds.

Completeness: by an argument à la Smullyan (Goubault-Larrecq and Mackie, 1997). Assume that there is no λ_{clos}-term t such that $\Gamma_0 \vdash t : A_0$ is derivable. We then build a frame and a world w_0 such that w_0 satisfies Γ_0 but $w_0 \not\models A_0$; worlds are maximally consistent sets of signed formulas, $+A$ or $-A$. This parallels the standard construction

$$\frac{}{\Gamma, x{:}A \vdash x \ : \ A} \qquad \text{(Ax)}$$

$$\frac{\Gamma, y{:}C \vdash t \ : \ A \quad \Gamma \vdash s \ : \ B}{\Gamma, x{:}B \Rightarrow C \vdash t\{y := x(s)\} \ : \ A} \qquad (\Rightarrow\text{L})$$

$$\frac{\Gamma^{\Rightarrow}, x{:}B \vdash t \ : \ C}{\Gamma^{\Rightarrow}, \Delta \vdash \langle x \cdot t \rangle \ : \ B \Rightarrow C} \qquad (\Rightarrow\text{R})$$

$$\frac{\Gamma \vdash t \ : \ A \quad \Gamma, x{:}A \vdash s \ : \ B}{\Gamma \vdash s\{x := t\} \ : \ B} \qquad (\textsc{Cut})$$

Fig. 4. Sequent calculus for near-intuitionistic logic

of worlds as maximally consistent sets of formulas, except that we use signed formulas to remedy the lack of negation in our language.

□

The point now is that the reduction rules of λ_{clos} encode cut-elimination in near-intuitionistic logic. To make this precise, consider the Gentzen system of Figure 4, where we decorate types on the left with variables and types on the right with λ_{clos}-terms, to show the correspondence with the natural deduction system of Figure 3. It is understood that $\Gamma^{\Rightarrow}$ denotes an arrowed context, while Γ and Δ denote arbitrary contexts.

Theorem 4.7 (Cut Elimination). The following statements are equivalent:

— (i). $\Gamma \vdash s \ : \ A$ is derivable in the system of Figure 3;
— (ii). $\Gamma \vdash s \ : \ A$ is derivable in the system of Figure 4.

Moreover, if any of them holds, then there is a λ_{clos}-term t such that $s \rhd^* t$ and $\Gamma \vdash t \ : \ A$ is derivable in the system of Figure 4 without rule (Cut).

Proof. The equivalence between (i) and (ii) is easy and fairly standard. The second part of the Theorem follows from the following facts: subject reduction holds for simply typed λ_{clos}-reduction; simply typed λ_{clos}-reduction is strongly normalizing; and every simply typed λ_{clos}-normal term is the proof term of some cut-free proof in the system of Figure 4. We can also show that the simply-typed $\lambda_{clos\eta}$-calculus is strongly normalizing. □

We can also show:

Lemma 4.8. The λ_{clos}-calculus is confluent. The simply-typed $\lambda_{clos\eta}$-calculus is confluent.

Proof. Let $(0\iota 2)$ denote the group of rules (0), (ι) and (2). Every reduction on $(0\iota 2)$ terminates, and $(0\iota 2)$ is locally confluent, so by Newman's Lemma $(0\iota 2)$ is confluent. More generally, we can show that λ_{clos} enjoys the finite developments property: contracting only those (β)-redexes that are residuals of redexes in the initial term of the reduction always terminates, and defines a confluent notion of reduction. Standard arguments then show that this implies the confluence of λ_{clos} (Barendregt, 1984). As far as the $\lambda_{clos\eta}$-calculus is concerned, although we believe it to be confluent, it seems rather painful to show this because of the non-left-linear rule (2). We shall only need the result in the

$$
\begin{aligned}
x^\star &=_{\mathrm{df}} x\\
(s(t))^\star &=_{\mathrm{df}} \mathrm{S}_0(s^\star, t^\star)\\
\langle x \cdot s, \{x_1 := s_1, \ldots, x_k := s_k\}\rangle^\star &=_{\mathrm{df}}\\
&\quad ([x]s^\star)[x_1 \leftarrow s_1^\star, \ldots, x_k \leftarrow s_k^\star]
\end{aligned}
\qquad
\begin{aligned}
[x]x &=_{\mathrm{df}} \mathrm{I}_0\\
[x]y &=_{\mathrm{df}} \mathrm{K}_0(y) \quad (y \not\equiv x)\\
[x]\mathrm{I}_m &=_{\mathrm{df}} \mathrm{I}_{m+1}\\
[x]\mathrm{S}_m(u, v) &=_{\mathrm{df}} \mathrm{S}_{m+1}([x]u, [x]v)\\
[x]\mathrm{K}_m(u) &=_{\mathrm{df}} \mathrm{K}_{m+1}([x]u)
\end{aligned}
$$

Fig. 5. Translation from λ_{clos} to SKInT

$$
\begin{aligned}
x^\circ &=_{\mathrm{df}} x\\
\mathrm{I}_m{}^\circ &=_{\mathrm{df}} \langle x_0 \cdot \langle x_1 \cdot \ldots \langle x_{m-1} \cdot\\
&\qquad \langle z \cdot z\rangle\rangle \ldots\rangle\rangle\\
(\mathrm{S}_m(M, N))^\circ &=_{\mathrm{df}} \mathcal{S}_m(M^\circ, N^\circ)\\
(\mathrm{K}_m(M))^\circ &=_{\mathrm{df}} \mathcal{K}_m(M^\circ)
\end{aligned}
\qquad
\begin{aligned}
\mathcal{S}_0(s, t) &=_{\mathrm{df}} s(t)\\
\mathcal{S}_{m+1}(s, t) &=_{\mathrm{df}} \langle z \cdot \mathcal{S}_m(x(z), y(z)),\\
&\qquad \{x := s, y := t\}\rangle\\
\mathcal{K}_0(s) &=_{\mathrm{df}} \langle z \cdot x, \{x := s\}\rangle\\
\mathcal{K}_{m+1}(s) &=_{\mathrm{df}} \langle z \cdot \mathcal{K}_m(x(z)), \{x := s\}\rangle
\end{aligned}
$$

Fig. 6. A translation from SKInT to λ_{clos}

typed case, however; since $\lambda_{clos\eta}$ terminates in the typed case, it is enough to check the critical pairs. □

Now we come, at last, to the reason why we introduced λ_{clos} in the first place. The following theorem shows in particular that SKInT is a complete proof-term language for near-intuitionistic logic, which was the main point of this section.

Theorem 4.9. Let $s \mapsto s^\star$ be the map from λ_{clos}-terms to SKInT-terms shown in Figure 5. Conversely, let $u \mapsto u^\circ$ be the map from SKInT-terms to λ_{clos}-terms of Figure 6. Then:

(i) If $\Gamma \vdash s : A$ is derivable in the system of Figure 3, then $\vdash_\Gamma s^\star : A$ is derivable in SKInT;
(ii) If $\vdash_\Gamma M : A$ is derivable in SKInT, then $\Gamma \vdash M^\circ : A$ is derivable in the system of Figure 3;
(iii) If $s \triangleright t$ in λ_{clos}, resp. $\lambda_{clos\eta}$, then $s^\star \triangleright^* t^\star$ in SKInT, resp. SKInT$_\eta$;
(iv) If $M \triangleright^* N$ in SKInT, resp. SKInT$_\eta$, then $M^\circ = N^\circ$ in $\lambda_{clos\eta}$;
(v) For every λ_{clos}-term s, $s^{\star\circ} = s$ in λ_{clos};
(vi) For every SKInT-term M, $M^{\circ\star} \triangleright_\eta^* M$ in SKInT$_\eta$;
(vii) For every λ_{clos}-normal, resp. $\lambda_{clos\eta}$-normal, λ_{clos}-term s, $s^\star$ is a SKInT-normal, resp. SKInT$_\eta$-normal, SKInT-term.

Proof. Tedious series of computations. Note that (iv) and (v) also hold in the simply-typed case, because λ_{clos} and $\lambda_{clos\eta}$ are confluent in the simply-typed case (Lemma 4.8) and enjoy subject reduction. □

In other words, (iii)–(vi) tell us that, up to $\lambda_{clos\eta}$-convertibility on one side, and up to SKInT$_\eta$-convertibility on the other, the λ_{clos}-terms and the SKInT-terms are isomorphic; the isomorphism is given by the pair of inverse functions $s \mapsto s^\star$ and $u \mapsto u^\circ$. This isomorphism also preserves types in both directions, by (i) and (ii). And one half of it (the map $s \mapsto s^\star$) even preserves normal forms literally, by (vii).

In particular, the typing rules of SKInT are sound and complete for near-intuitionistic logic, and SKInT-normal forms correspond to cut-free proofs.

4.2. *Termination*

SKInT is more well-behaved than SKIn, and most of our proofs of properties of SKIn go through an analysis of SKInT first.

Definition 4.10 (Σ, ΣT). Let Σ denote the sub-calculus of SKIn from Section 3.1 minus the rule (SI_m), $m \geq 0$.

Similarly, let ΣT denote the sub-calculus of SKInT minus the rule (SI_m), $m \geq 0$.

To mimick calculi of explicit substitutions, Σ is the sub-calculus in charge of propagating substitutions downwards, while (SI_m): $\mathrm{S}_m(\mathrm{I}_m, M) \triangleright M$ is the β-rule (more precisely, the βI-rule). This can be seen most easily in the proof of Theorem 3.5 in A: through the $[\![\text{-}]\!]()$ translation, every rule of Σ translates to an equality in the λ-calculus, and the only rules that may translate to proper reductions in the λ-calculus are (SI_m), $m \geq 0$. In the other direction, notice that $((\lambda x.x)(P))^* \equiv \mathrm{S}_0(\mathrm{I}_0, P^*)$ is an (SI_0)-redex, which reduces to P^*, thus doing a very simple case of βI-reduction. A β-reduction that is not a βI-reduction is typically $(\lambda x.y)(P) \triangleright y$, which gets translated into SKIn as $\mathrm{S}_0(\mathrm{K}_0(y), P^*) \triangleright y$ by (SK_0), not (SI_0). In general, β-contraction is simulated in SKIn (see the proof of Theorem 3.14) by propagating redexes down to the variables that must be replaced: for example, $((\lambda x.x(y))(P))^* \equiv \mathrm{S}_0(\mathrm{S}_1(\mathrm{I}_0, \mathrm{K}_0(y)), P^*)$ rewrites to $\mathrm{S}_0(\mathrm{S}_0(\mathrm{I}_0, P^*), \mathrm{S}_0(\mathrm{K}_0(y), P^*)) \equiv (((\lambda x.x)(P))((\lambda x.y)(P)))^*$, which then rewrites to $(P(y))^*$ by one βI and one β step. As the reader may see, the analogy is not perfect: SKIn tends to blur the distinction between β-redexes $(\lambda x.P)(Q)$ and explicit substitutions $P[x := Q]$.

ΣT is, similarly, SKInT without this same βI-rule, or equivalently Σ without the rule $\mathrm{K}_m(\mathrm{S}_m(M, N)) \triangleright \mathrm{S}_{m+1}(\mathrm{K}_m(M), \mathrm{K}_m(N))$. Σ_η, ΣT_η, SKIn_η, SKInT_η denote the corresponding calculi with their respective η-rules.

Let's now examine termination.

Lemma 4.11. ΣT and ΣT_η terminate.

Proof. This holds even in the untyped case. The proof is technical, see C for the full proof. We merely give an idea of the proof here.

We first show that the three rules $(\mathrm{K}_m\mathrm{I}_n)$, $(\mathrm{K}_m\mathrm{K}_n)$ and $(\mathrm{K}_m\mathrm{S}_{n+1})$ (the K *group*) terminate by using a translation from SKInT-terms M to SKInT-terms $[\![M]\!]_q$ that attempts to predict the effects of pushing K_m down the term on indices of operators, and then using a recursive path ordering (Dershowitz, 1987) on the translated terms. We then show that the six rules obtained by adding $(\mathrm{S}_m\mathrm{I}_n)$, $(\mathrm{S}_m\mathrm{K}_n)$ and $(\mathrm{S}_m\mathrm{S}_n)$ (the S *group*) to the K group define a terminating rewrite system, by using a mapping $M \mapsto [\![M]\!]_e\gamma$ from SKInT-terms to $\mathbb{N}$, parameterized by sequences γ of non-negative integers; we show that $[\![[\![M]\!]_q]\!]_e\gamma > [\![[\![N]\!]_q]\!]_e\gamma$ whenever $M \triangleright N$ by some rule in the S-group, while $[\![[\![M]\!]_q]\!]_e\gamma \geq [\![[\![N]\!]_q]\!]_e\gamma$ when $M \triangleright N$ by some rule in the K-group. The $M \mapsto [\![M]\!]_e\gamma$ translation resembles the $M \mapsto [\![M]\!](P_0, \ldots, P_{n-1})$ translation, and depends monotonically on the indices of operators: the rules of the S group then decrease these indices. We then define another translation $M \mapsto [\![M]\!]'\gamma$ from SKInT-terms to $\mathbb{N}$, such that $[\![M]\!]'\gamma > [\![N]\!]'\gamma$ whenever $M \rhd N$ by rule (SK_m), and $[\![M]\!]'\gamma \geq [\![N]\!]'\gamma$ if $M \rhd N$ by some rule in the S or K group. This translation is even closer to the $M \mapsto [\![M]\!](P_0, \dots, P_{n-1})$ translation. This finishes the proof that ΣT terminates.

ΣT_η then terminates because the η-rule $\mathrm{S}_{m+1}(\mathrm{K}_m(M), \mathrm{I}_m)$ quasi-commutes over ΣT and clearly terminates on its own (see (Dershowitz, 1987)). □

Notice that Σ does not terminate, since the counter-example of Proposition 3.3 only uses rules in Σ.

In the typed case, we again have:

Proposition 4.12 (Subject Reduction). If $\vdash_\Gamma M : A$ and $M \rhd N$ in SKInT then $\vdash_\Gamma N : A$.

Proof. Straightforward. Remember that we use (KT) instead of (K) to type terms of the form $\mathrm{K}_m(M)$, and that $(\mathrm{K}_m\mathrm{S}_n)$ is not a reduction rule of SKInT if $n = m + 1$. □

At last, we justify our claim that SKInT is better behaved than SKIn, as far as termination is concerned:

Theorem 4.13 (Strong Normalization). The simply-typed SKInT and SKInT$_\eta$ calculi are strongly normalizing.

Proof. The idea is to translate SKInT-terms to λ-terms by the $M \mapsto [\![M]\!](P_0, \dots, P_{n-1})$ translation. This translation preserves types, and the direction of reduction, but due to the fact that the translations of I_m and $\mathrm{K}_m(M)$ do not depend on some of the P_i's, some reductions in SKInT translate as equalities; e.g., $[\![\mathrm{S}_0(\mathrm{K}_0(M), N)]\!](P_0, \dots, P_n) \equiv [\![M]\!](P_0, \dots, P_n)$, so any reductions in N are completely ignored by the translation. To correct this, we change the translation of terms of the form $\mathrm{K}_m(M)$ to the following:

$$[\![\mathrm{K}_m(M)]\!]_\bullet(P_0, \dots, P_{n-1}) =_{\mathrm{df}} [\![M]\!]_\bullet(P_0, \dots, P_{m-1}, P_m \oplus P_{m+1}, P_{m+2}, \dots, P_{n-1})$$

where $\oplus$ is an operator that we add to the λ-calculus, so that $P \oplus Q$ is semantically equivalent to Q; the trick is that, in $P \oplus Q$, we still see all reductions occurring in P. For technical reasons, we shall also need a few less important additional operators.

Formally, let the *typed $\lambda\oplus$-calculus* be the calculus whose terms are those defined by the grammar:

$$s, t ::= x_A \mid t(t) \mid \lambda x_A.t \mid \epsilon(t) \mid \iota(t) \mid t \oplus t$$

where A ranges over simple types, submitted to the typing rules of Figure 7. Notice that any $\lambda\oplus$-term has exactly one typing derivation, hence exactly one type.

We omit type indices on variables when they should be obvious. Moreover, to make the notation lighter, we assume that $\oplus$ is right-associative, that is, $s \oplus t \oplus r$ denotes

$$\frac{}{\vdash x_A : A} \qquad \frac{\vdash s : A_1 \Rightarrow A_2 \quad \vdash t : A_1}{\vdash s(t) : A_2} \qquad \frac{\vdash s : A_2}{\vdash \lambda x_{A_1}.s : A_1 \Rightarrow A_2}$$

$$\frac{\vdash s : A}{\vdash \epsilon(s) : A} \qquad \frac{\vdash s : A}{\vdash \iota(s) : A} \qquad \frac{\vdash s : A_1 \quad \vdash t : A_2}{\vdash s \oplus t : A_2}$$

Fig. 7. Typing the $\lambda\oplus$-terms

$$\begin{array}{rcl}
[\![\mathrm{S}_m(M,N)]\!]_\bullet(s_0,\ldots,s_{n-1}) & =_{\mathrm{df}} & [\![M]\!]_\bullet(s_0,\ldots,s_{m-1}, \\
 & & \quad \epsilon(\lambda z_m \ldots \lambda z_{p-1}.[\![N]\!]_\bullet(s_0,\ldots,s_{m-1},z_m,\ldots,z_{p-1})), \\
 & & \quad s_m,\ldots,s_{n-1}) \\
[\![\mathrm{K}_m(M)]\!]_\bullet(s_0,\ldots,s_{n-1}) & =_{\mathrm{df}} & [\![M]\!]_\bullet(s_0,\ldots,s_{m-1},s_m \oplus s_{m+1},s_{m+2},\ldots,s_{n-1}) \\
[\![\mathrm{I}_m]\!]_\bullet(s_0,\ldots,s_{n-1}) & =_{\mathrm{df}} & \iota(s_0 \oplus \ldots \oplus s_{m-1} \oplus s_m)(s_{m+1},\ldots,s_{n-1}) \\
[\![x]\!]_\bullet(s_0,\ldots,s_{n-1}) & =_{\mathrm{df}} & x(s_0,\ldots,s_{n-1})
\end{array}$$

Fig. 8. The $[\![\_]\!]_\bullet\_$ interpretation $(m \geq 0)$.

$s \oplus (t \oplus r)$. The reduction rules are as follows:

$$\begin{array}{llcll llcl}
(\beta) & (\lambda x.s)(t) & \triangleright & [t/x]s & & (\epsilon) & \epsilon(s) & \triangleright & s \\
(\eta) & \lambda x.s(x) & \triangleright & s & (x \text{ not free in } s) & (\iota) & \iota(s) & \triangleright & s \\
(\oplus-) & s \oplus t & \triangleright & t & & (\oplus) & (s \oplus t) \oplus r & \triangleright & s \oplus (t \oplus r)
\end{array}$$

The $[\![\_]\!]_\bullet$ translation maps typed SKInT-terms M, of type $(A_0,\ldots,A_{n-1}) \to o$, where o is a base type, and n typed $\lambda\oplus$-terms $s_0, \ldots, s_{n-1}$ of respective types $A_0, \ldots, A_{n-1}$, to a term $[\![M]\!]_\bullet(s_0,\ldots,s_{n-1})$ of type o. It is defined in Figure 8. A sequence $s_i,\ldots,s_j$ denotes the sequence of elements s_k with $i \leq k$ and $k \leq j$, for all i and j such that $i \leq j+1$; in particular, if $i = j+1$, $s_i,\ldots,s_j$ denotes the empty sequence. Check that the definition is meaningful. In particular, in the definition of $[\![\mathrm{K}_m(M)]\!]_\bullet(s_0,\ldots,s_{n-1})$, typing constraints (namely, rule (I)) entail that $n-1 \geq m+1$, so that s_{m+1} is indeed present in the argument list.

We now show that the typed $\lambda\oplus$-calculus is strongly normalizing; that $M \triangleright N$ implies $[\![M]\!]_\bullet(s_0,\ldots,s_{n-1}) \triangleright^* [\![N]\!]_\bullet(s_0,\ldots,s_{n-1})$ for any sequence $s_0, \ldots, s_{n-1}$ for which the left-hand side makes sense; that moreover when $M \triangleright_\eta N$ by rule (SI_m) or by $(\eta\mathrm{S}_m)$, $[\![M]\!]_\bullet(s_0,\ldots,s_{n-1}) \triangleright^+ [\![N]\!]_\bullet(s_0,\ldots,s_{n-1})$. See D for details.

So only finitely many instances of (SI_m) or $(\eta\mathrm{S}_m)$ can occur in a reduction. Since ΣT terminates by Lemma 4.11, the typed SKInT and SKInT$_\eta$-calculi terminate. □

It is a bit unfortunate that we had to introduce yet another calculus, the $\lambda\oplus$-calculus, just to prove this result. But, for one, λ_{clos} cannot be used, as Theorem 4.9 *(iv)* only shows that $M \triangleright N$ in SKInT implies $M^\circ = N^\circ$ in $\lambda_{clos\eta}$, not $M^\circ \triangleright^+ N^\circ$. A similar translation to the call-by-value λ-calculus is even more out of the question: recall that the latter simulates strictly less reductions than λ_{clos} through the $M \mapsto M^\sharp$ translation.

4.3. *Confluence*

Lemma 4.14. ΣT is confluent.

Proof. Among the critical pairs of SKIn from Proposition 3.1, the critical pairs of SKInT are as for SKIn, except that the cases involving rule $(\mathrm{K}_m\mathrm{S}_n)$ with $n = m + 1$ are dropped. All these critical pairs are joined by rules of SKInT: so SKInT is locally confluent. Among the critical pairs of SKInT, those of ΣT are joined by rules in ΣT, so ΣT is locally confluent as well. By Lemma 4.11 and Newman's Lemma, ΣT is confluent. $\square$

The confluence of ΣT is almost equivalent to the finite developments theorem (Barendregt, 1984), but not quite, since βI-reductions (reductions by some instance of (SI_m)) are left out of ΣT. This is however close enough, so that confluence of SKInT follows easily.

Theorem 4.15 (Confluence). SKInT and SKInT_η are confluent.

Proof. By a method due to Yokouchi (Yokouchi, 1989; Hardin and Lévy, 1989). Let βI denote the rewrite relation induced by the rule schemes (SI_m), $m \geq 0$. Let $\beta I_{||}$ be the following notion of parallel reduction:

$$\frac{}{M \triangleright_{\beta I_{||}} M} \qquad \frac{P \triangleright_{\beta I_{||}} Q}{\mathrm{S}_m(\mathrm{I}_m, P) \triangleright_{\beta I_{||}} Q} \qquad \frac{M \triangleright_{\beta I_{||}} N \quad P \triangleright_{\beta I_{||}} Q}{\mathrm{S}_m(M, P) \triangleright_{\beta I_{||}} \mathrm{S}_m(N, Q)} \qquad \frac{M \triangleright_{\beta I_{||}} N}{\mathrm{K}_m(M) \triangleright_{\beta I_{||}} \mathrm{K}_m(N)}$$

We claim that:

(i)
$$\begin{tikzcd} M \arrow[r, "\beta I_{||}"] \arrow[d, "\Sigma T"'] & P \arrow[d, dotted, "\Sigma T^*"] \\ N \arrow[r, dotted, "\Sigma T^* \beta I_{||}"'] & Q \end{tikzcd}$$

where solid lines denote universally quantified reductions, dashed lines denote existentially quantified reductions conditioned on the former, ΣT denotes one ΣT step, $\beta I_{||}$ denotes one $\beta I_{||}$ step, ΣT^* denotes zero, one or more ΣT steps, and $\Sigma T^* \beta I_{||}$ denotes zero, one or more ΣT steps followed by one $\beta I_{||}$ step. Indeed, because all rules are left-linear, we only have to consider the critical pairs. There are 2 of them, which parallel two critical pairs in Proposition 3.1. Let $0 \leq m < n$:

1If $M \equiv \mathrm{S}_m(\mathrm{S}_n(\mathrm{I}_n, M_1), N_1)$, $N \equiv \mathrm{S}_{n-1}(\mathrm{S}_m(\mathrm{I}_n, N_1), \mathrm{S}_m(M_1, N_1))$ by $(\mathrm{S}_m\mathrm{S}_n)$, and $P \equiv \mathrm{S}_m(M_1', N_1')$ with $N_1 \triangleright_{\beta I_{||}} N_1'$, $M_1 \triangleright_{\beta I_{||}} M_1'$. Then $N \triangleright_{\Sigma T} \mathrm{S}_{n-1}(\mathrm{I}_{n-1}, \mathrm{S}_m(M_1, N_1)) \triangleright_{\beta I_{||}} P$.

2If $M \equiv \mathrm{K}_m(\mathrm{S}_n(\mathrm{I}_n, M_1))$, $N \equiv \mathrm{S}_{n+1}(\mathrm{K}_m(\mathrm{I}_n), \mathrm{K}_m(M_1))$ by rule $(\mathrm{K}_m\mathrm{S}_{n+1})$, and $P \equiv \mathrm{K}_m(M_1')$ with $M_1 \triangleright_{\beta I_{||}} M_1'$. Then we have $N \triangleright_{\Sigma T} \mathrm{S}_{n+1}(\mathrm{I}_{n+1}, \mathrm{K}_m(M_1)) \triangleright_{\beta I_{||}} P$.

We then claim:

$$(ii)\qquad \begin{tikzcd} M \arrow[r, "\beta I_{||}"] \arrow[d, "\Sigma T^*"'] & P \arrow[d, dotted, "\Sigma T^*"] \\ N \arrow[r, dotted, "\Sigma T^* \beta I_{||}"'] & Q \end{tikzcd}$$

We show (ii) by induction on $\nu(M)$, the length of the longest ΣT reduction starting from M (by Lemma 4.11). If $\nu(M) = 0$, then the result is clear. Otherwise, let the first reduction step from M to N rewrite M to M_1, with $\nu(M_1) < \nu(M)$. Claim (ii) follows from the following diagram, where the induction hypothesis applies since $\nu(M_2) < \nu(M)$:

$$\begin{tikzcd} M \arrow[rr, "\beta I_{||}"] \arrow[d, "\Sigma T"'] \arrow[drr, phantom, "\text{by } (i)"] & & P \arrow[d, dotted, "\Sigma T^*"] \\ M_1 \arrow[r, dotted, "\Sigma T^*"] \arrow[d, "\Sigma T^*"'] \arrow[dr, phantom, "\text{by Lemma 4.14}"] & M_2 \arrow[r, dotted, "\beta I_{||}"] \arrow[d, dotted, "\Sigma T^*"] \arrow[dr, phantom, "\text{by induction}"] & P_1 \arrow[d, dotted, "\Sigma T^*"] \\ N \arrow[r, dotted, "\Sigma T^*"'] & N_2 \arrow[r, dotted, "\Sigma T^* \beta I_{||}"'] & Q \end{tikzcd}$$

Using Lemma 4.14, it follows immediately that:

$$(iii)\qquad \begin{tikzcd} M \arrow[r, "\Sigma T^* \beta I_{||}"] \arrow[d, "\Sigma T^*"'] & P \arrow[d, dotted, "\Sigma T^*"] \\ N \arrow[r, dotted, "\Sigma T^* \beta I_{||}"'] & Q \end{tikzcd}$$

Now, notice that: (iv) $\beta I_{||}$ is strongly confluent, because it is the parallelization of a left-linear system, βI, with no critical pairs.

We now claim that $\Sigma T^* \beta I_{||}$ is strongly confluent, that is:

$$\begin{tikzcd} M \arrow[r, "\Sigma T^* \beta I_{||}"] \arrow[d, "\Sigma T^* \beta I_{||}"'] & P \arrow[d, dotted, "\Sigma T^* \beta I_{||}"] \\ N \arrow[r, dotted, "\Sigma T^* \beta I_{||}"'] & Q \end{tikzcd}$$

This is by induction on $\nu(M)$ again. If the reductions starting from M both rewrite by $\beta I_{||}$ (in particular if $\nu(M)=0$), then this follows from the fact that $\beta I_{||}$ is strongly confluent, by (iv). Otherwise:

$$(v) \qquad \begin{tikzcd}
M \arrow[r, "\Sigma T^*"] \arrow[d, "\Sigma T*"'] \arrow[dr, phantom, "\text{by Lemma 4.14}"] & N_2 \arrow[r, "\Sigma T^* \beta I_{||}"] \arrow[d, dotted, "\Sigma T^*"] \arrow[dr, phantom, "\text{by } (iii)"] & P \arrow[d, dotted, "\Sigma^*"] \\
N_1 \arrow[r, dotted, "\Sigma T^*"] \arrow[d, "\Sigma T^* \beta I_{||}"'] \arrow[dr, phantom, "\text{by } (iii)"] & N_3 \arrow[r, dotted, "\Sigma T^* \beta I_{||}"] \arrow[d, dotted, "\Sigma T^* \beta I_{||}"] \arrow[dr, phantom, "\text{by induction}"] & N_5 \arrow[d, dotted, "\Sigma T^* \beta I_{||}"] \\
N \arrow[r, dotted, "\Sigma T^*"'] & N_4 \arrow[r, dotted, "\Sigma T^* \beta I_{||}"'] & Q
\end{tikzcd}$$

where by assumption there is at least one ΣT step from M to N_1 or to N_2; the induction hypothesis applies since $\nu(M_3) \leq \min(\nu(M_1), \nu(M_2)) < \nu(M)$.

Finally, the reflexive transitive closures of $\Sigma T^* \beta I_{||}$ and of SKInT are exactly the same, so by (v) SKInT is confluent.

As far as SKInT_η is concerned, let $\beta\eta I$ denote the rewrite relation induced by the rule schemes (SI_m) and $(\eta \mathrm{S}_m)$, $m \geq 0$. Let $\beta\eta I_{||}$ be the following notion of parallel reduction:

$$\frac{}{M \rhd_{\beta\eta I_{||}} M} \quad \frac{}{\mathrm{S}_{m+1}(\mathrm{K}_m(M), \mathrm{I}_m) \rhd_{\beta\eta I_{||}} M} \quad \frac{P \rhd_{\beta\eta I_{||}} Q}{\mathrm{S}_m(\mathrm{I}_m, P) \rhd_{\beta\eta I_{||}} Q}$$
$$\frac{M \rhd_{\beta\eta I_{||}} N \quad P \rhd_{\beta\eta I_{||}} Q}{\mathrm{S}_m(M, P) \rhd_{\beta\eta I_{||}} \mathrm{S}_m(N, Q)} \quad \frac{M \rhd_{\beta\eta I_{||}} N}{\mathrm{K}_m(M) \rhd_{\beta\eta I_{||}} \mathrm{K}_m(N)}$$

We claim that:

$$(i') \qquad \begin{tikzcd}
M \arrow[r, "\beta\eta I_{||}"] \arrow[d, "\Sigma T"'] & P \arrow[d, dotted, "\Sigma T^*"] \\
N \arrow[r, dotted, "\Sigma T^* \beta\eta I_{||}"'] & Q
\end{tikzcd}$$

As for (i), we only have to consider the critical pairs. There are 5 new ones, which are the same as the critical pairs for adding η to SKInT. Let $0 \leq m < n$. We give as an example the only one for which we need parallel η reduction. In this case, $M \equiv \mathrm{S}_{m+1}(\mathrm{K}_m(\mathrm{S}_n(M_1, N_1)), \mathrm{I}_m)$, $N \equiv \mathrm{S}_{m+1}(\mathrm{S}_{n+1}(\mathrm{K}_m(M_1), \mathrm{K}_m(N_1)), \mathrm{I}_m)$ by rule $(\mathrm{K}_m \mathrm{S}_{n+1})$, and $P \equiv \mathrm{S}_n(M_1, N_1)$ by $\beta\eta I_{||}$. By $(\mathrm{S}_{m+1}\mathrm{S}_{n+1})$, then, N rewrites to $\mathrm{S}_n(\mathrm{S}_{m+1}(\mathrm{K}_m(M_1), \mathrm{I}_m), \mathrm{S}_{m+1}(\mathrm{K}_m(N_1), \mathrm{I}_m))$, and then to P by $\beta\eta I_{||}$. The rest of the argument of the SKInT case then goes through, replacing $\beta I_{||}$ by $\beta\eta I_{||}$ and (i) by (i'). $\square$

Notice that confluence holds even in the untyped case, and also in the presence of free (meta-)variables. In contrast, confluence in the presence of free variables holds for rather few calculi of explicit substitutions; one example where it does hold is $\lambda\sigma_{\Uparrow}$ (Hardin and Lévy, 1989).

4.4. *Standardization*

Standardization implies that if a term has a normal form then there is a simple, effective way of computing it. In the λ-calculus, this strategy consists in always reducing the leftmost outermost β-redex.

In SKInT, and as we shall see in SKIn as well, this strategy reduces the redexes on the spine first, where the spine is the leftmost branch of the term. Formally:

Definition 4.16. Let the *spines* S be the sequences defined by:

$$S ::= x \mid \mathrm{I}_m \mid \mathrm{S}_m S \mid \mathrm{K}_m S \qquad (m \geq 0)$$

The *arity* of a spine is the number of operators of the form S_m, $m \geq 0$, in it. For any spine S of arity n, we let $S[M_1, \ldots, M_n]$ be:

$$\begin{array}{c} x[] =_{\mathrm{df}} x \qquad \mathrm{I}_m[] =_{\mathrm{df}} \mathrm{I}_m \\ (\mathrm{S}_m S)[M_1, \ldots, M_n] =_{\mathrm{df}} \mathrm{S}_m(S[M_1, \ldots, M_{n-1}], M_n) \\ (\mathrm{K}_m S)[M_1, \ldots, M_n] =_{\mathrm{df}} \mathrm{K}_m(S[M_1, \ldots, M_n]) \end{array}$$

Every term M can be written in a unique way as $S[M_1, \ldots, M_n]$: the spine S is the sequence of operators along the leftmost branch of M, read top-down, and $M_1, \ldots, M_n$ (the *arguments* of M) are the second arguments of operators S_m on the spine, read bottom-up. Iterating this decomposition of spine and arguments allows us to see terms as trees of spines.

Spine reductions and standard reductions are then defined as follows.

Definition 4.17. A redex is called a *spine redex* in M if and only if it occurs on the spine of M. Define the relation of *spine reduction* $\triangleright^s$ by $M \triangleright^s N$ if and only if $M \triangleright N$ by contracting a spine redex in M, and let $\triangleright^{s*}$ be its reflexive transitive closure. A term that has no spine redex is called *spine-normal*.

Define $M \triangleright^{std*} N$ by induction on N viewed as a tree of spines, if and only if $M \triangleright^{s*} S[M_1, \ldots, M_n]$, and $N \equiv S[N_1, \ldots, N_n]$, where $M_i \triangleright^{std*} N_i$ for each i, $1 \leq i \leq n$.

We then use a technique due to René David to show standardization.

Theorem 4.18 (Standardization). If $M \triangleright^* N$ in SKIn (resp. SKInT), then $M \triangleright^{std*} N$.

Proof. By induction on M_1 as a tree of spines, check that $N \triangleright^{std*} M_1 \triangleright M_2$ implies $N \triangleright^{std*} M_2$, permuting rules if necessary. By induction on the length of reductions, then, the result follows. Details are given in E. □

So, to normalize a (normalizable) term, compute one of its spine-normal forms, then reduce its arguments in the same way, recursively. This allows us to build reduction machines for SKIn and SKInT in any of the standard ways (Peyton-Jones, 1986). For instance, represent terms by the following data structure, which makes the spines explicit. Let $\mathcal{V}$ be a set of (meta-)variables, and let:

$$\begin{array}{rcll} \mathcal{L} & =_{\mathrm{df}} & \mathcal{V} + \mathbb{N} & \text{leaves} \\ \mathcal{R} & =_{\mathrm{df}} & (\mathbb{N} \times (\mathcal{T} + \{\text{-}\}))^* & \text{rakes} \\ \mathcal{T} & =_{\mathrm{df}} & \mathcal{L} \times \mathcal{R} & \text{terms} \end{array}$$

where - is a distinguished element, $+$ denotes disjoint sum, and S^* denotes the set of all lists of elements of the set S. The leaves ℓ in $\mathcal{L}$ are either meta-variables $x \in \mathcal{V}$, or integers m representing I_m, and are the bottoms of spines. Rakes in $\mathcal{R}$ are partial spines, together with their arguments; intuitively, an element of the list of the form $(m, \text{-})$ denotes $\mathrm{K}_m(\_)$, and an element of the form (m, t) with $t \in \mathcal{T}$ denotes $\mathrm{S}_m(\_, t)$. Formally, denote by ϵ the empty list, by $r_1.r_2$ the concatenation of r_1 and r_2, and write $[c]$ for the list with one element c. When c is a couple (ℓ, r), let $[\ell, r]$ denote $[(\ell, r)]$. The terms t denote SKInT-terms $\gamma(t)$, where:

$$\begin{array}{rcll} \gamma & : & \mathcal{T} \to \mathrm{SKInT} & \\ \gamma(x, r) & =_{\mathrm{df}} & \gamma_1(x, r) & (x \in \mathcal{V}) \\ \gamma(m, r) & =_{\mathrm{df}} & \gamma_1(\mathrm{I}_m, r) & (m \in \mathbb{N}) \end{array}$$

$$\begin{array}{rcl} \gamma_1 & : & \mathrm{SKInT} \times \mathcal{R} \to \mathrm{SKInT} \\ \gamma_1(M, [m, \text{-}].r) & =_{\mathrm{df}} & \gamma_1(\mathrm{K}_m(M), r) \\ \gamma_1(M, [m, t].r) & =_{\mathrm{df}} & \gamma_1(\mathrm{S}_m(M, \gamma(t)), r) \end{array}$$

We now define a reduction relation $\leadsto$ between triples $up, down, \ell$, where $up, down \in \mathcal{R}$ and $\ell \in \mathcal{L}$. Such triples denote terms $(\ell, rev(down).up)$, where rev is the list reversing function, together with the position on this spine where we shall attempt to detect a possible redex. Let c denote any element of $\mathbb{N} \times (\mathcal{T} \cup \{\text{-}\})$, $i, j \in \mathbb{N}$, $\ell, \ell' \in \mathcal{L}$, $r \in \mathcal{R}$, and $t, t' \in \mathcal{T}$. The rules are given in Figure 9, where the $\langle$default$\rangle$ rule applies only if none of the previous rules does. The idea is that, as every rule of SKInT is of the form $f(g(\ldots), \ldots)$ for some operators f and g, we detect the corresponding redexes by letting up code the part of the spine containing f and all operators above it, and letting $down$ code the part of the spine containing g and all operators below it.

The following facts are easily proved:

Proposition 4.19.

- *(i)* If $up, down, \ell \leadsto up', down', \ell'$ by some rule other than $\langle$default$\rangle$, then it holds: $\gamma(\ell, rev(down).up) \rhd^s \gamma(\ell', rev(down').up')$;
- *(ii)* If $up, down, \ell \leadsto up', down', \ell'$ by $\langle$default$\rangle$, then it holds: $\gamma(\ell, rev(down).up) \equiv \gamma(\ell', rev(down').up')$;
- *(iii)* If $\gamma(\ell, rev(down))$ is spine-normal and $up, down, \ell \leadsto^* up', down', \ell'$, then again $\gamma(\ell', rev(down'))$ is spine-normal;
- *(iv)* If $\gamma(\ell, rev(down))$ is spine-normal and $\gamma(\ell, rev(down).up) \rhd^{s*} \gamma(\ell', rev(down').up')$ by some bottommost spine-reduction, then $up, down, \ell \leadsto^* up', down', \ell'$;
- *(v)* Every $\leadsto$-normal triple is of the form $\epsilon, down, \ell$ for some rake $down$ and some leaf ℓ.

$(\mathrm{K}_i\mathrm{I}_{j+1})$	$[i,\text{-}].up,$	ϵ, j	$\leadsto$	$up,$	$\epsilon, j+1$	$(i \leq j)$
$(\mathrm{S}_i\mathrm{I}_j)$	$[i,t].up,$	ϵ, j	$\leadsto$	$up,$	$\epsilon, j-1$	$(i < j)$
(SI_i)	$[i,(\ell,r)].up,$	ϵ, j	$\leadsto$	$r.up,$	ϵ, ℓ	$(i = j)$
$(\mathrm{K}_i\mathrm{K}_{j+1})$	$[i,\text{-}].up,$	$[j,\text{-}].down, \ell$	$\leadsto$	$[i,\text{-}].[j+1,\text{-}].up,$	$down, \ell$	$(i \leq j)$
$(\mathrm{K}_i\mathrm{S}_{j+1})$	$[i,\text{-}].up,$	$[j,(\ell',r')].down, \ell$	$\leadsto$	$[i,\text{-}].[j+1,(\ell',r'.[i,\text{-}])].up,$	$down, \ell$	$(i < j)$
$(\mathrm{S}_i\mathrm{K}_j)$	$[i,t].up,$	$[j,\text{-}].down, \ell$	$\leadsto$	$[i,t].[j-1,\text{-}].up,$	$down, \ell$	$(i < j)$
(SK_i)	$[i,t].up,$	$[j,\text{-}].down, \ell$	$\leadsto$	$up,$	$down, \ell$	$(i = j)$
$(\mathrm{S}_i\mathrm{S}_j)$	$[i,t].up,$	$[j,(\ell',r')].down, \ell$	$\leadsto$	$[i,t].[j-1,(\ell',r'.[i,t])].up,$	$down, \ell$	$(i < j)$
$\langle$default$\rangle$	$[c].up,$	$down, \ell$	$\leadsto$	$up,[c].down, \ell$		

Fig. 9. The SKInT spine reduction machine

That is, $\leadsto$ implements bottommost spine-rewriting: given an element (ℓ, r) of $\mathcal{T}$ such that $\gamma(\ell, r) \equiv M$, get the $\leadsto$-normal form of r, ϵ, ℓ (if it exists). Write this normal form ϵ, r', ℓ': then $\gamma(\ell', r')$ is a spine-normal reduct of M. Recursively applying this procedure to the elements of $\mathcal{T}$ in r' provides a normalization procedure for M, implementing $\triangleright^{std*}$.

This procedure terminates as soon as the standard reduction starting from M that always reduces bottommost spine-redexes on a given spine itself terminates: this is because rule $\langle$default$\rangle$ can only be applied finitely many times in a row. We claim that the restriction to bottommost spine-redexes is not essential, as any spine reduction strategy applies the (SI_m) exactly the same number of times. This implies that the procedure above actually finds the normal form of M whenever it exists. Proving the claim requires a study of the notion of solvability of SKInT-terms, and will be the subject of another paper (Goubault-Larrecq, 1998a).

Finally, note that, as far as implementations are concerned, we may represent integers in rakes as machine integers. Indeed, these integers are bounded above by the number of nested abstractions in the term that we reduce, provided that this term is closed. For example, a term represented on a machine with a 32-bit address range will necessarily feature (much) less that 2^{32} nested abstractions, and integers coded on 32 bits are enough. That is, there is no need for arbitrary precision integers, even under paranoid correctness criteria.

5. **Properties of** SKIn

The study of SKInT allows us to infer similar, albeit sometimes weaker, properties for SKIn: weak termination in Section 5.1, confluence in Section 5.2, and standardization in Section 5.3.

5.1. *Weak Termination of* SKIn

SKIn is harder to deal with than SKInT, because the counter-example to its termination also shows that Σ does not terminate. So we cannot just replay the proofs that we gave for SKInT. We instead pick reduction strategies in SKIn that do terminate.

Split Σ in two parts, the SKT group:

$$\begin{array}{rrcl}
(\mathrm{SK}_m) & \mathrm{S}_m(\mathrm{K}_m(M), N) & \rhd & M \\
(\mathrm{S}_m\mathrm{K}_n) & \mathrm{S}_m(\mathrm{K}_n(M), N) & \rhd & \mathrm{K}_{n-1}(\mathrm{S}_m(M, N)) \\
(\mathrm{S}_m\mathrm{S}_n) & \mathrm{S}_m(\mathrm{S}_n(M, N), P) & \rhd & \mathrm{S}_{n-1}(\mathrm{S}_m(M, P), \mathrm{S}_m(N, P)) \\
(\mathrm{K}_m\mathrm{K}_n) & \mathrm{K}_m(\mathrm{K}_{n-1}(M)) & \rhd & \mathrm{K}_n(\mathrm{K}_m(M)) \\
(\mathrm{K}_m\mathrm{S}_{n+1}) & \mathrm{K}_m(\mathrm{S}_n(M, N)) & \rhd & \mathrm{S}_{n+1}(\mathrm{K}_m(M), \mathrm{K}_m(N))
\end{array}$$

where $0 \leq m < n$, and the rest, namely rules $(\mathrm{S}_m\mathrm{I}_n)$, $(\mathrm{K}_m\mathrm{I}_n)$ and $(\mathrm{K}_m\mathrm{S}_{m+1})$. Observe that:

Lemma 5.1. *SKT is confluent and terminating.*

Proof. SKT is locally confluent: the critical pairs are a subset of those considered in Proposition 3.1, where again the cases involving rule $(\mathrm{K}_m\mathrm{S}_n)$ with $n = m + 1$ are dropped. Furthermore, SKT is a subsystem of ΣT, so it terminates by Lemma 4.11. By Newman's Lemma, SKT is confluent. $\square$

In particular, every SKIn-term P has a unique SKT-normal form, which we denote by $\downarrow(P)$. We then build a variant of Σ minus SKT as the following I *group*:

$$\begin{array}{rrcl}
(\mathrm{K}_m\mathrm{I}_n) & \mathrm{K}_m(\mathrm{I}_{n-1}) & \rhd & \mathrm{I}_n \\
(\mathrm{S}_m\mathrm{I}_n) & \mathrm{S}_m(\mathrm{I}_n, N) & \rhd & \mathrm{I}_{n-1} \\
(\mathrm{K}_m\mathrm{S}'_{m+1}) & \mathrm{K}_m(\mathrm{S}_m(M, N)) & \rhd & \mathrm{S}_{m+1}(\mathrm{K}_m(M), \downarrow(\mathrm{K}_m(N)))
\end{array}$$

where $0 \leq m < n$.

A key observation is that if M is SKT-normal, and M rewrites by I to N, then N is SKT-normal. Since I restricted to SKT-normal terms terminates (see F), it follows:

Lemma 5.2. *Σ normalizes weakly. More precisely, all SKT-eager Σ-reductions terminate, where a reduction is SKT-eager if and only if it only reduces I-redexes in SKT-normal forms.*

Theorem 5.3 (Weak Normalization). *All simply-typed SKIn-terms are weakly normalizing for the SKIn notion of reduction.*

Proof. By a technique similar to (Goubault-Larrecq, 1998b), but closer to (Goubault-Larrecq, 1997). We use a slight variant of the $M \mapsto [\![M]\!]$ interpretation of Figure 2, the $M \mapsto [\![M]\!]_\circ$ translation, where only the clause for I_m changes, namely:

$$[\![\mathrm{I}_m]\!]_\circ(P_0, \ldots, P_{n-1}) \quad =_{\mathrm{df}} \quad \begin{cases} I(P_m)(P_{m+1}, \ldots, P_{n-1}) & n > m \\ \lambda x_n \ldots \lambda x_m . I(x_m) & n \leq m \end{cases}$$

where $I =_{\mathrm{df}} \lambda x.x$. All other clauses are as in Figure 2, replacing $[\![\_]\!]$ by $[\![\_]\!]_\circ$ everywhere. It is easy to replay the proofs of A. Therefore, Theorem 3.5 carries over, that is, $M \rhd N$ implies $[\![M]\!](P_0, \ldots, P_{n-1}) \rhd^*_\eta [\![N]\!](P_0, \ldots, P_{n-1})$. Moreover, βI-redexes translate to non-empty reductions, thanks to the introduction of I: if $M \equiv \mathrm{S}_k(\mathrm{I}_k, N)$, then either $n \geq k$

and:

$$\begin{aligned}
&[\![M]\!]_\circ(P_0, \dots, P_{n-1}) \\
&\equiv [\![\mathrm{I}_k]\!]_\circ(P_0, \dots, P_{k-1}, [\![N]\!]_\circ(P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}) \\
&\equiv I([\![N]\!]_\circ(P_0, \dots, P_{k-1}))(P_k, \dots, P_{n-1}) \\
&\rhd ([\![N]\!]_\circ(P_0, \dots, P_{k-1}))(P_k, \dots, P_{n-1}) \\
&\rhd^* [\![N]\!]_\circ(P_0, \dots, P_{n-1})
\end{aligned}$$

or $n < k$ and:

$$\begin{aligned}
&[\![M]\!]_\circ(P_0, \dots, P_{n-1}) \\
&\equiv \lambda x_n \dots \lambda x_{k-1} . [\![\mathrm{I}_k]\!]_\circ(P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}, [\![N]\!]_\circ(P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1})) \\
&\equiv \lambda x_n \dots \lambda x_{k-1} . I([\![N]\!]_\circ(P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1})) \\
&\rhd \lambda x_n \dots \lambda x_{k-1} . [\![N]\!]_\circ(P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}) \\
&\rhd_\eta^* [\![N]\!]_\circ(P_0, \dots, P_{n-1})
\end{aligned}$$

Define $M \succ_\circ N$ (resp. $M \succeq_\circ N$), where M, N are SKIn-terms, if and only if $[\![M]\!]_\circ(P_0, \dots, P_{n-1}) \rhd_\eta^+ [\![N]\!]_\circ(P_0, \dots, P_{n-1})$ (resp. $\rhd_\eta^*$) for all λ-terms $P_0, \dots, P_{n-1}$. We have just seen that $\mathrm{S}_k(\mathrm{I}_k, N) \succ_\circ N$, while in general $M \rhd N$ implies $M \succeq_\circ N$. But the strict inequality $\mathrm{S}_k(\mathrm{I}_k, N) \succ_\circ N$ does not pass to context in general.

Formally, recall that a context $\mathcal{C}$ is a term with one hole []. $\mathcal{C}[M]$ denotes $\mathcal{C}$ with the hole replaced by the term M. Contexts are used to pinpoint where a given redex occurs. In general, $M \succeq_\circ N$ implies $\mathcal{C}[M] \succeq_\circ \mathcal{C}[N]$, but the same does not hold for $\succ_\circ$. The *safe contexts* are those such that whenever $M \succ_\circ N$, then $\mathcal{C}[M] \succ_\circ \mathcal{C}[N]$.

For each $m \geq 0$, let T_m be the set of terms defined by:

— $\mathrm{S}_k(M, N) \in T_m$ if and only if $m \geq k$ and $M \in T_{m+1}$, or $m < k$, $M \in T_k$ and $N \in T_m$;
— $\mathrm{K}_k(M) \in T_m$ if and only if $m > k$ and $M \in T_{m-1}$;
— $\mathrm{I}_k \in T_m$ if and only if $m \geq k$;
— every variable is in T_m.

This is a valid definition of $M \in T_m$ by structural induction on M. The *syntactically safe contexts* are defined by the grammar:

$$\mathcal{S} ::= [] \mid \mathrm{S}_m(\mathcal{S}, T) \mid \mathrm{S}_m(T_m, \mathcal{S}) \mid \mathrm{K}_m(\mathcal{S})$$

where T denotes the set of all SKIn-terms. It is fairly easy to show that all syntactically safe contexts are safe, and that for every Σ-normal term of the form $\mathcal{C}[M]$, $\mathcal{C}$ is syntactically safe (see F, Lemma F.8 and Lemma F.9). It follows that whenever $M \rhd N$ by rule (SI_m), and M is Σ-normal, then $M \succ_\circ N$.

On the other hand, recall that $M \rhd N$ implies $M \succeq_\circ N$ for any other rule. So, provided that $\succ_\circ$ is well-founded on typed SKIn-terms, every Σ-eager reduction must be finite, where a reduction is *Σ-eager* if and only if it is of the form:

$$M_0 \rhd_\Sigma^* M_1 \rhd_{\beta I} M_2 \rhd_\Sigma^* \dots \rhd_{\beta I} M_{2i} \rhd_\Sigma^* M_{2i+1} \rhd_{\beta I} M_{2i+2} \rhd_\Sigma^* \dots$$

where $\rhd_{\beta I}$ is one-step reduction by (SI_m), $m \geq 0$, $\rhd_\Sigma$ is reduction by the rules in group Σ, and $M_1, M_3, \dots, M_{2i+1}, \dots$, are Σ-normal. Note that Σ-eager reduction is a family of normalization strategies, by Lemma 5.2.

It remains to show that $\succ_\circ$ is well-founded on typed SKIn-terms. If $M \succ_\circ N$ and M is of type A in some typing context, then $[\![M]\!]_\circ() \rhd_\eta^+ [\![N]\!]_\circ()$, of type A in the same typing context: we conclude by noticing that $\beta\eta$-reduction terminates on typed λ-terms. $\square$

5.2. *Confluence of* SKIn

We are finally in position to show that:

Theorem 5.4 (Confluence). SKIn and Σ are confluent.

Proof. We apply Hardin's interpretation method (Hardin, 1989) several times. This method is as follows. Let R and R' be two reduction relations, such that R' is normalizing and confluent. Let $R'(M)$ denote the unique R'-normal form of M. R then induces a reduction relation R_0 on R'-normal forms, defined as the smallest such that $M\ R\ N$ implies $R'(M)\ R_0\ R'(N)$. Then, whenever R_0 is confluent on R'-normal forms, and R_0 is a sub-relation of $(R \cup R')^*$, $R \cup R'$ is confluent. This is a simple diagram chase:

$$\begin{tikzcd}
 & & M \arrow[dll, "(R\cup R')^*"'] \arrow[d, dotted, "R'^*"] \arrow[drr, "(R\cup R')^*"] & & \\
M_1 \arrow[d, dotted, "R'^*"'] & & R'(M) \arrow[dll, dotted, "R_0^*"] \arrow[drr, dotted, "R_0^*"'] & & M_2 \arrow[d, dotted, "R'^*"] \\
R'(M_1) \arrow[drr, dotted, "R_0^*"'] & & & & R'(M_2) \arrow[dll, dotted, "R_0^*"] \\
 & & N & &
\end{tikzcd}$$

To show the Theorem, we first show that SK, the group of rules consisting of SKT plus the rules ($\mathrm{K}_m\mathrm{S}_{m+1}$), $m \geq 0$ (alternatively, Σ minus the rules involving I_m), is confluent: take R to be the group of rules ($\mathrm{K}_m\mathrm{S}_{m+1}$), $m \geq 0$, R' to be SKT, then R_0 is the group I_0 of all rules ($\mathrm{K}_m\mathrm{S}'_{m+1}$): $\mathrm{K}_m(\mathrm{S}_m(M,N)) \rhd \mathrm{S}_{m+1}(\mathrm{K}_m(M), \downarrow(\mathrm{K}_m(N)))$, $m \geq 0$. It follows that Σ is confluent: take R to be SK, R' to be the group of rules ($\mathrm{S}_m\mathrm{I}_n$), ($\mathrm{K}_m\mathrm{I}_n$), $0 \leq m < n$, and R_0 is SK again. Similarly, SKIn is confluent: take R to be SK, R' to be the group of rules ($\mathrm{S}_m\mathrm{I}_n$), ($\mathrm{K}_m\mathrm{I}_n$), (SI_m), $0 \leq m < n$. The details are in G. $\square$

By Theorem 3.23, then:

Corollary 5.5 (Conservativity). The map $M \mapsto M^*$ is a conservative embedding of λ-terms in SKIn modulo conversion. That is, $M^* = N^*$ if and only if $M = N$.

5.3. *Standardization of* SKIn

We just recall that we have shown in Theorem 4.18, Section 4.4 that SKIn standardizes. It follows that a simple modification to the machine of Figure 9 yields a machine for spine reduction in SKIn: in the ($\mathrm{K}_i\mathrm{S}_{j+1}$) row, replace the $i < j$ condition by $i \leq j$. In this case, however, it is not obvious that $\rightsquigarrow$ terminates as soon as the starting triple denotes a term that has a spine-normal form. This is the case provided the following conjecture holds: for every SKIn-term M, if M has a spine-normal form, then bottommost spine-reduction terminates.

6. Translating the λ-calculus to λ_{clos} and SKInT

At this point of the paper, we are faced with a dilemma: SKIn interprets the λ-calculus faithfully, is confluent and standardizes, but it only normalizes weakly in the typed case. This makes it a delicate choice for an implementation of λ-reduction, just as $\lambda\sigma$, $\lambda\sigma_{\Uparrow}$ and the other non-strongly-normalizing calculi of explicit substitutions: every such implementation must first be shown to obey one of the weak normalization strategies of the calculus. On the other hand, there is no such problem with SKInT, which basically has all the good properties of the λ-calculus—termination, confluence, standardization—and also has a natural η-rule. However, we have seen in Section 4.1 that SKInT was related not to the λ-calculus and intuitionistic logic, but rather to the λ_{clos}-calculus and near-intuitionistic logic.

Our goal in this section is to show that there are numerous bridges between the λ and λ_{clos}-calculi, and that therefore SKInT is a valid choice for implementing the λ-calculus itself. The main point to be developed here is that SKInT is not the culprit for not interpreting full λ-reduction: the $M \mapsto M^*$ translation of Definition 3.7, taking λ-terms as input, is the problem, and is too simple-minded.

We shall explore a few ways to do this, all based on translations from the call-by-value λ-calculus to SKInT. Recall that λ_{clos}, which proves the same equalities as SKInT (Theorem 4.9), proves strictly more equalities than the call-by-value λ-calculus. We shall use the following:

Lemma 6.1. The following claims hold:

(i) For every λ-term M, $M^* \equiv M^{\sharp\star}$.
(ii) $M \rhd_V^* N$ in the λ_V-calculus implies $M^* \rhd^* N^*$ in SKInT.

Proof. (i) is obvious. If $M \rhd_V N$, then $M^\sharp \rhd^+ N^\sharp$ in λ_{clos} by Theorem 4.4, so $M^{\sharp\star} \rhd^* N^{\sharp\star}$ by Theorem 4.9 (iii); so $M^* \rhd^* N^*$ by (i): (ii) follows by induction on the length of the reduction.

□

On the typing side, we have:

Proposition 6.2. For every arrowed context $\Gamma^{\Rightarrow}$, for every λ-term M such that $\Gamma^{\Rightarrow} \vdash M : A$, $\Gamma^{\Rightarrow} \vdash M^\sharp : A$ and $\vdash_{\Gamma^{\Rightarrow}} M^* : A$.

Indeed, the first claim is obvious, while the second follows by Theorem 4.9 and the fact that $M^* \equiv M^{\sharp\star}$. Notice that the context $\Gamma^{\Rightarrow}$ needs to be arrowed for us to use the typing rule (Abs).

6.1. *Plotkin's Translation*

There are many translations from the λ-calculus to the λ_V-calculus. *À tout seigneur, tout honneur*, we shall first examine Plotkin's call-by-name CPS (continuation passing style) transform (Plotkin, 1975). This is defined as:

$$\begin{array}{rcl} \mathcal{P}_n(x) & =_{\mathrm{df}} & \lambda k.x(k) \\ \mathcal{P}_n(\lambda x.M) & =_{\mathrm{df}} & \lambda k.k(\lambda x.\mathcal{P}_n(M)) \\ \mathcal{P}_n(M(N)) & =_{\mathrm{df}} & \lambda k.\mathcal{P}_n(M)(\lambda m.m(\mathcal{P}_n(N), k)) \end{array}$$

Recall that $M \rhd N$ in the λ-calculus implies $\mathcal{P}_n(M) \rhd^+ \mathcal{P}_n(N)$ in the λ_V-calculus. Let $\mathcal{P}_n(M)^*$ be defined as $(\mathcal{P}_n(M))^*$. It follows, using Lemma 6.1, that $M \mapsto \mathcal{P}_n(M)^*$ defines an embedding of the λ-calculus, with β-reduction, inside SKInT. It is unknown at the moment whether this embedding is conservative.

As far as typing is concerned, it is well-known that $M \mapsto \mathcal{P}_n(M)$ gives rise to typings obtained by double negation transformations in the style of Kolmogorov (Griffin, 1990; Murthy, 1991). Let $\perp$ denote any type, let $\neg A$ denote $A \Rightarrow \perp$, define $\mathcal{P}_n(A)$ as $\neg\neg(A)_0$, where $(A)_0$ is defined by:

$$\begin{array}{rcl} (o)_0 & =_{\mathrm{df}} & o \quad (o \in O) \\ (A \Rightarrow B)_0 & =_{\mathrm{df}} & \mathcal{P}_n(A) \Rightarrow \mathcal{P}_n(B) \end{array}$$

We extend these notations to contexts $\mathcal{P}_n(\Gamma)$, and let $\mathcal{P}_n(x_1{:}A_1, \ldots, x_n{:}A_n)$ denote $x_1{:}\mathcal{P}_n(A_1), \ldots, x_n{:}\mathcal{P}_n(A_n)$. Then, $\Gamma \vdash M : A$ implies $\mathcal{P}_n(\Gamma) \vdash \mathcal{P}_n(M) : \mathcal{P}_n(A)$. Since $\mathcal{P}_n(\Gamma)$ is arrowed by construction, by Proposition 6.2, this also implies $\vdash_{\mathcal{P}_n(\Gamma)} \mathcal{P}_n(M)^* : \mathcal{P}_n(A)$. It follows that reducing $\mathcal{P}_n(M)^*$, when M is a typed λ-term, always terminates, by Theorem 4.13.

6.2. *The L Translation*

A translation that will be conservative is the following *L-translation*:

$$L(M) =_{\mathrm{df}} \lambda z.L_z(M) \qquad \begin{array}{rcl} L_P(x) & =_{\mathrm{df}} & x(P) \\ L_P(\lambda x.M) & =_{\mathrm{df}} & \lambda x.L_P(M) \\ L_P(M(N)) & =_{\mathrm{df}} & L_P(M)(L(N)) \end{array}$$

where z is a fresh variable and P is any term. The idea is that, as $L(M)$ is an abstraction, $L_P(M(N))$ is the application of a term to a value, so β-redexes translate to (β_V)-redexes this way. The index P in $L_P(M)$ is somewhat reminiscent of a world index in Kripke semantics. Let $L^\sharp(M)$ denote $(L(M))^\sharp$, and $L^*(M)$ denote $(L(M))^*$.

Theorem 6.3 (Soundness). SKInT is sound for λ-reduction along L^*, that is: $M \rhd^* N$ in the λ-calculus implies $L^*(M) \rhd^* L^*(N)$, and $M = N$ implies $L^*(M) = L^*(N)$.

Proof. Easy computations. See H. □

It is an easy exercise to compute an explicit definition of L^*, without going through $M \mapsto M^*$ and L. Define $L^*(M)$ as $[z]L^*_z(M)$, where L^*_P is defined as follows, for all terms P:

$$\begin{array}{rcl} L^*_P(x) & =_{\mathrm{df}} & \mathrm{S}_0(x, P) \\ L^*_P(M(N)) & =_{\mathrm{df}} & \mathrm{S}_0(L^*_P(M), L^*(N)) \\ L^*_P(\lambda x.M) & =_{\mathrm{df}} & [x]L^*_P(M) \end{array}$$

Lemma 6.4 (Soundness for Typing). Define $L(A)$ on types A by $L(A) =_{\mathrm{df}} \top \Rightarrow L_\top(A)$, where:

$$\begin{array}{rcl} L_\top(o) & =_{\mathrm{df}} & o \quad (o \in O) \\ L_\top(A \Rightarrow B) & =_{\mathrm{df}} & L(A) \Rightarrow L_\top(B) \end{array}$$

where $\top$ is any given, fixed type. Extend this to contexts by $L(x_1{:}A_1, \ldots, x_n{:}A_n) =_{\mathrm{df}} x_1{:}L(A_1), \ldots, x_n{:}L(A_n)$.

If $\Gamma \vdash M : A$, then $\vdash_{L(\Gamma)} L^*(M) : L(A)$.

Proof. Assume $\Gamma \vdash M : A$. We show that: (i) $L(\Gamma), \Gamma' \vdash L(M) : L(A)$ for any context Γ', and that: (ii) $L(\Gamma), \Gamma'' \vdash L_z(M) : L_\top(A)$, for any context Γ'' such that $L(\Gamma), \Gamma'' \vdash z : \top$. This is by structural induction on M. Since (i) trivially follows from (ii) (take $\Gamma'' =_{\mathrm{df}} \Gamma', z{:}\top$), we concentrate on (ii). If M is a variable x, then $x{:}A$ is in Γ, so (ii) is clear. If M is an abstraction $\lambda x.N$, then A is of the form $B \Rightarrow C$, and $\Gamma, x{:}B \vdash N : C$, so by induction hypothesis (ii) $L(\Gamma), x{:}L(B) \vdash L_z(N) : L_\top(C)$, therefore (ii) holds by rule (Abs). If M is an application $N(P)$, then for some type B, $\Gamma \vdash N : B \Rightarrow A$ and $\Gamma \vdash P : B$. Let Γ'' be any context such that $L(\Gamma), \Gamma'' \vdash z : \top$. By induction hypothesis (ii), $L(\Gamma), \Gamma'' \vdash L_z(N) : L(B) \Rightarrow L_\top(A)$; by induction hypothesis (i), $L(\Gamma), \Gamma'' \vdash L(P) : L(B)$. So $L(\Gamma), \Gamma'' \vdash L_z(N)(L(P)) : L_\top(A)$, showing (ii).

Now from (i), in particular $L(\Gamma) \vdash L(M) : L(A)$. The Lemma follows from Proposition 6.2, using the fact that $L(\Gamma)$ is arrowed. □

Theorem 6.5. The map $M \mapsto L^*(M)$ is a conservative embedding of λ-terms in SKInT modulo conversion. That is, $L^*(M) = L^*(N)$ if and only if $M = N$. This holds also in the simply-typed case.

Proof. The if direction is soundness, so consider the only if direction. If $L^*(M) = L^*(N)$, then since SKInT is confluent, $L^*(M) \rhd^* P$ and $L^*(N) \rhd^* P$ for some SKInT-term P. Since the SKInT rules are all SKIn rules, and since $L^*(M)$ and $L^*(N)$ are well-staged by Lemma 3.19, Theorem 3.20 entails that $[\![L^*(M)]\!]() \rhd^* [\![P]\!]()$ and $[\![L^*(N)]\!]() \rhd^* [\![P]\!]()$ in the λ-calculus (not the λ_V-calculus). By Lemma 3.22, $L(M) \rhd^* [\![L^*(M)]\!]()$ and $L(N) \rhd^* [\![L^*(N)]\!]()$, so $L(M) = L(N)$ in the λ-calculus.

It remains to show that when $L(M) = L(N)$ in the λ-calculus, then $M = N$ in the λ-calculus as well. For convenience, we shall consider two λ-calculi, differing only in the set of variables they are based on, the source λ-calculus and the target λ-calculus. Conceptually, L maps terms in the source language to the target language. Then, split the set of variables in two infinite, disjoint sets X and Z, and assume that X is the set of variables in source λ-terms, and $X \cup Z$ is the set of variables in target λ-terms. We can reformulate the definition of L as follows: $L(M) =_{\mathrm{df}} \lambda z.L_z(M)$, with $z \in Z$, while $L_z(x) =_{\mathrm{df}} x(z)$ for $x \in X$, $L_z(M(N)) =_{\mathrm{df}} L_z(M)(L(N))$, and $L_z(\lambda x.M) =_{\mathrm{df}} \lambda x.L_z(M)$, where $x \in X$.

The λ-terms that are in the range of L and L_z respectively have particular syntactic forms. It is easy to see that $L(M) \in V$ and that $L_z(M) \in T_z$, where the sets V and T_z, $z \in Z$, are described by the following grammars:

$$\begin{array}{lcl} V & ::= & \lambda z.T_z \mid x \\ T_z & ::= & V(z) \mid T_z(V) \mid \lambda x.T_z \end{array}$$

where z ranges over Z, x over X. We shall write $s, t, \ldots$, for terms in T_z, for any z, and $u, v, \ldots$, for terms in V, to give a visual cue of the domains of terms.

We define an translation L^{-1} that works as a left-inverse to L as follows. L^{-1} maps terms in V to source λ-terms, and is defined by $L^{-1}(\lambda z.t) =_{\mathrm{df}} L_z^{-1}(t)$ and $L^{-1}(x) =_{\mathrm{df}} x$ for every $x \in X$, where L_z^{-1} is the function from T_z to the set of source λ-terms defined by:

$$\begin{array}{lcl} L_z^{-1}(u(z)) & =_{\mathrm{df}} & L^{-1}(u) \\ L_z^{-1}(s(v)) & =_{\mathrm{df}} & L_z^{-1}(s)(L^{-1}(v)) \\ L_z^{-1}(\lambda x.s) & =_{\mathrm{df}} & \lambda x.L_z^{-1}(s) \end{array}$$

L^{-1} and L_z^{-1} are well-defined by mutual recursion. In particular, observe that L_z^{-1} is well-defined on applications: indeed, no term in T_z can be both of the form $u(z)$ with $u \in V$, $z \in Z$, and of the form $s(v)$ with $s \in T_z$ and $v \in V$, since otherwise we would have $z \equiv v$, which is impossible since $Z \cap T_z = \emptyset$.

A straightforward structural induction on M then shows that $L^{-1}(L(M)) \equiv M$ and $L_z^{-1}(L_z(M)) \equiv M$. Moreover, we can show that (see H):

— if $u \in V$ and $u \rhd^* v$, then $v \in V$ (Lemma H.5);
— if $u \in V$ and $u \rhd^* v$, then $L^{-1}(u) \rhd^* L^{-1}(v)$ (Lemma H.6).

Now, if $L(M) = L(N)$, then for some λ-term P, $L(M) \rhd^* P$ and $L(N) \rhd^* P$. By the above, and since $L(M)$ and $L(N)$ are in V, $L^{-1}(L(M)) \rhd^* L^{-1}(P)$ and $L^{-1}(L(N)) \rhd^* L^{-1}(P)$; in other words, $L^{-1}(P)$ is a common reduct of both $M \equiv L^{-1}(L(M))$ and $N \equiv L^{-1}(L(N))$, so $M = N$.

Since we have shown $M = N$ by exhibiting a common reduct of M and N, and since subject reduction holds for the simply-typed λ-calculus, the claim is also proved in the simply-typed case. $\square$

One could also notice that L^* is an injective function, which maps β-normal λ-terms to normal SKInT-terms (use Theorem 4.9), so that L^* preserves weak normalization: for every weakly λ-normalizing M, $L^*(M)$ is weakly SKInT-normalizing. But this result, and preservation of strong normalization as well, can be proved in a far more general setting, using disciplines of conjunctive types. This will be published elsewhere (Goubault-Larrecq, 1998a).

In short, SKInT is a confluent first-order calculus, even on terms with free variables, implementing λ-reduction fully and faithfully, which terminates in the typed case. To our knowledge, this is the first calculus that has all of these properties together.

6.3. *What About No Translation?*

Until now, we were looking for embeddings of the λ-calculus inside SKInT. This is useful in general, for example in implementing type theory based proof assistants; in fact, this is necessary for unification algorithms in the style of (Dowek et al., 1995). However, if the goal is just to implement reduction machines, we do not need as much. The representations used for optimal λ-reduction, notably (Lamping, 1990; Gonthier et al., 1992a), only have the following property: there is a translation f from λ-terms to some target language, and a converse translation g, called the *read-back function*, such that if M normalizes to N in the λ-calculus, then $f(M)$ normalizes to some term t in the target language such that $g(t) \equiv N$. But $M \rhd^* N$ does not imply $f(M) \rhd^* f(N)$, since reductions in the target language only simulate reductions that reduce all the redexes in a given Lévy family as a whole; this is precisely the purpose of optimal λ-reduction. Furthermore, the normal form t may not be the image by f of a normal λ-term N. In fact, some of the reductions from M to N are not accomplished in the target language, but rather by the read-back function g.

Exactly the same happens with SKInT: take for f the $M \mapsto M^*$ translation, and for f the $M \mapsto [\![M]\!]()$ translation. Reduction in SKInT does not implement Lévy-optimal λ-reductions this way; our only point is to show that the read-back model of computation described above works with SKInT instead of SKIn, so SKInT is a viable alternative to SKIn. Our claim follows from the following result:

Lemma 6.6. If M is a normal SKInT-term, then $[\![M]\!]()$ is a normal λ-term.

Proof. More generally, we show that for every normal SKInT-term M with $lev(M) = m$, for all normal λ-terms $P_0, \ldots, P_{n-1}$ such that P_i is not a λ-abstraction for any i, $i < m$, then $[\![M]\!](P_0, \ldots, P_{n-1})$ is a normal λ-term. Observe that the condition on P_i means informally that the m first arguments of the interpretation function are not abstractions. We shall also use the following easily verified facts: (i) $\mathrm{K}_m(M_1)$ is SKInT-normal if and only if M_1 is and $lev(M_1) \leq m$; (ii) $\mathrm{S}_m(M_1, M_2)$ is SKInT-normal if and only if M_1 and M_2 are, and $lev(M_1) \leq m$.

If M is a variable, this is obvious. If M is of the form I_k, then by assumption $lev(M) = k + 1 = m$. If $n > k$, that is, if $n \geq m$, then $[\![M]\!](P_0, \ldots, P_{n-1}) \equiv P_k(P_{k+1}, \ldots, P_{n-1})$ is normal, since by assumption P_k, $k < m$, is not a λ-abstraction. If $n \leq k$, then $[\![M]\!](P_0, \ldots, P_{n-1}) \equiv \lambda x_n \ldots \lambda x_{m-1}.x_{m-1}$ is normal.

If M is of the form $\mathrm{K}_k(M_1)$, then we have two cases. Observe that, by (i), $lev(M_1) \leq k$, so $m = lev(M) = \max(k, lev(M_1)) + 1 = k + 1$, and therefore $lev(M_1) \leq m - 1$. If $n > k$, then $[\![M]\!](P_0, \ldots, P_{n-1}) \equiv [\![M_1]\!](P_0, \ldots, P_{k-1}, P_{k+1}, \ldots, P_{n-1})$. By assumption the $lev(M_1)$ first elements in $P_0, \ldots, P_{k-1}, P_{k+1}, \ldots, P_{n-1}$, which are among the $m - 1$ first, and therefore also among the m first in $P_0, \ldots, P_{n-1}$, are not λ-abstractions. So the induction hypothesis applies, and $[\![M]\!](P_0, \ldots, P_{n-1})$ is therefore normal. If $n \leq k$, then

$[\![M]\!](P_0, \ldots, P_{n-1}) \equiv \lambda x_n \ldots \lambda x_k \cdot [\![M_1]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1})$. Since $m = k + 1$, $n < m$, so $P_0, \ldots, P_{n-1}$ are not λ-abstractions: the induction hypothesis applies again, showing the claim.

If M is of the form $\mathrm{S}_k(M_1, M_2)$, then we again consider two cases. Observe that by (*ii*) $lev(M_1) \leq k$, so $m = lev(M) = \max(k, lev(M_1) - 1) = k$, in particular $lev(M_1) \leq m$. If $n \geq k$, then $[\![M]\!](P_0, \ldots, P_{n-1}) \equiv [\![M_1]\!](P_0, \ldots, P_{k-1}, [\![M_2]\!](P_0, \ldots, P_{k-1}), P_k, \ldots, P_{n-1})$. But then the induction hypothesis applies immediately, since $n \geq k$ implies $n \geq m$, and therefore the m first arguments to the interpretation function are $P_0, \ldots, P_{m-1}$, which are not abstractions by assumption. If $n < k$, then $[\![M]\!](P_0, \ldots, P_{n-1}) \equiv \lambda x_n \ldots \lambda x_{k-1} \cdot [\![M_1]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1}, [\![M_2]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1}))$. The m first arguments to the interpretation of M_1 are then exactly $P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1}$, since $m = k$, and are therefore not λ-abstractions; so by induction hypothesis the claim is proved. $\square$

Now, for every λ-term M, if M^* normalizes in SKInT, say to the SKInT-normal form N, then $M \rhd^* [\![M^*]\!]()$ (by Lemma 3.22) $\rhd^* [\![N]\!]()$ (by Lemma 3.19 and Theorem 3.20, using the fact that SKInT is a sub-calculus of SKIn), and $[\![N]\!]()$ is λ-normal by Lemma 6.6. Reducing M^* in SKInT (not SKIn) using any reduction strategy, and then reading back the normal form $[\![N]\!]()$, is then an effective procedure for normalizing M. It will be a consequence of results shown elsewhere (Goubault-Larrecq, 1998a) that $M \mapsto M^*$ preserves weak and strong normalization, so this procedure applies to every normalizable λ-term M.

7. Further Work

Although our current strong normalization proof is complicated, we believe that it can be considerably simplified; at least the fact that substitution in SKIn and SKInT is essentially first-order substitution (recall that $\mathrm{S}_0([x]M, N) \rhd^+ M[x \leftarrow N]$) should simplify the models for proofs of strong normalization, compared to traditional λ-calculi (Girard et al., 1989; Goguen, 1995).

The form of our judgments and rules is intuitively similar to the sequent calculus, but the actual rules are different, instead resembling the rules for the combinators S and K. Following the sequent calculus more closely is a natural continuation of this work. This also leads to the study of linear calculi, with a formal similarity between SKInT and the fans, croissants and brackets of (Gonthier et al., 1992b): SKInT has identities I_m but lacks abstraction and application nodes and plugs.

Combinators and explicit substitution leads to a cleaner treatment of higher-order unification (Dougherty, 1993; Dowek et al., 1995). We believe that our calculi can be profitably used in a similar way.

Extending our approach to dependent types could also allow the incorporation of functional programming techniques (Peyton-Jones, 1986) in proof checkers and lead to simpler implementations.

8. Conclusions

In the introduction, we mentioned several disadvantages of Hilbert systems when compared with natural deduction systems. Let's see how they are addressed by sequent combinators:

• The size of sequent combinators does not grow much under the abstraction algorithm. The causes for growth are $[x]y$, which introduces a new term constructor, and the index for each of the combinator term constructors. As Curien notes (Curien, 1993), with indices in binary, abstraction increases the size from n to $n \log n$, and as we noted in Section 4.4, the actual implementation can use 32-bit integers to store indices without any problem.

• Our system behaves well with respect to meta-theoretic abstraction and substitution.

• Reduction and equality for λ-calculus are preserved, conservatively, by translation into sequent combinators, and there is a natural notion of standard reduction.

Furthermore, the system SKInT offers a confluent first-order calculus implementing λ-calculus reduction fully and faithfully and terminating in the typed case. We therefore believe we have demonstrated that Hilbert systems can be a viable alternative to λ-calculi.

Acknowledgments

Thanks to R. Burstall, A. Compagnoni, G. Dowek, E. Goubault, P. Hancock, M. Hyland, Z. Luo, J. McKinna and G. Plotkin for encouragement and useful comments. The first author was funded by an EPSRC fellowship, and his work reported here was in part carried out at the University of Cambridge Computer Laboratory.

This paper uses Paul Taylor's commutative diagram macro package (available at `ftp://ftp.dcs.qmw.ac.uk/pub/tex/contrib/pt/diagrams`).

References

Abadi, M., Cardelli, L., Curien, P.-L., and Lévy, J.-J. (1991). Explicit substitutions. *Journal of Functional Programming*, 1(4):375–416.

Barendregt, H. (1984). *The Lambda Calculus, Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Company, Amsterdam.

Bierman, G. and de Paiva, V. (1992). Intuitionistic necessity revisited. In *Logic at Work*, Amsterdam, the Netherlands.

Church, A. (1940). A formulation of the simple theory of types. *Journal of Symbolic Logic*, 5:56–68.

Church, A. (1956). *Introduction to Mathematical Logic*. Princeton University Press.

Curien, P.-L. (1993). *Categorical Combinators, Sequential Algorithms and Functional Programming*. Pitman, second edition.

Curry, H. B. and Feys, R. (1958). *Combinatory Logic*, volume 1. North Holland.

Dershowitz, N. (1987). Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116.

Dougherty, D. J. (1993). Higher-order unification via combinators. *Theoretical Computer Science*, 114(2):273–298.

Dowek, G., Hardin, T., and Kirchner, C. (1995). Higher-order unification via explicit substitutions. In *Proceedings of the 10th Annual IEEE Symposium on Logics in Computer Science (LICS'95)*, pages 366–374.

Gentzen, G. (1969). Investigations into logical deduction. In Szabo, M. E., editor, *The Collected Works of Gerhard Gentzen*. North Holland.

Girard, J.-Y., Lafont, Y., and Taylor, P. (1989). *Proofs and Types*. Cambridge University Press.

Goguen, H. (1995). Typed operational semantics. In *Proceedings of the 2nd International Conference on Typed Lambda-Calculi and Applications (TLCA'95)*, pages 186–200. Springer-Verlag LNCS 902.

Goguen, H. and Luo, Z. (1993). Inductive data types: Well-ordering types revisited. In Huet, G. and Plotkin, G., editors, *Logical Environments*, pages 198–218. Cambridge University Press.

Gonthier, G., Abadi, M., and Lévy, J.-J. (1992a). The geometry of optimal lambda reduction. In *Conference record of the 19th ACM Symposium on Principles of Programming Languages (POPL'92)*, pages 15–26. ACM Press.

Gonthier, G., Abadi, M., and Lévy, J.-J. (1992b). Linear logic without boxes. In *Proceedings of the 7th Annual IEEE Symposium on Logics in Computer Science (LICS'92)*, pages 223–234.

Goubault-Larrecq, J. (1996b). On computational interpretations of the modal logic S4 I. Cut elimination. Technical Report 1996-35, Universität Karlsruhe.

Goubault-Larrecq, J. (1996a). On computational interpretations of the modal logic S4 II. The $\lambda \mathtt{ev} Q$-calculus. Technical Report 1996-34, Universität Karlsruhe.

Goubault-Larrecq, J. (1997). On computational interpretations of the modal logic S4 IIIb. Confluence, termination of $\lambda \mathtt{ev} Q_H$. Research Report RR-3164, INRIA.

Goubault-Larrecq, J. (1998a). A few remarks on SKInT. Research Report RR-3475, INRIA. Submitted.

Goubault-Larrecq, J. (1998b). A proof of weak termination of typed $\lambda\sigma$-calculi. In *Proceedings of the TYPES'96 Workshop*, Aussois, France. Preliminary version as INRIA Research Report RR-3090, January 1997.

Goubault-Larrecq, J. and Mackie, I. (1997). *Proof Theory and Automated Deduction*, volume 6 of *Applied Logic Series*. Kluwer. ISBN 0-7923-4593-2.

Griffin, T. G. (1990). A formulas-as-types notion of control. In *Conference record of the 17th Annual ACM Symposium on Principles of Programming Languages (POPL'90)*, pages 47–58, San Francisco, California.

Hardin, T. (1989). Confluence results for the pure strong categorical logic CCL. Lambda-calculi as subsystems of CCL. *Theoretical Computer Science*, 65(3):291–342.

Hardin, T. and Lévy, J.-J. (1989). A confluent calculus of substitutions. In *France-Japan A.I. and Computer Science Symposium*.

Huet, G. (1977). Confluent reductions: Abstract properties and applications to term rewriting systems. In *18th Annual Symposium on Foundations of Computer Science (FOCS'77)*, pages 30–45. IEEE.

Hughes, J. (1982). Supercombinators, a new implementation method for applicative languages. *Conference Record of the 1982 ACM Symposium on Lisp and Functional Programming (LFP'82)*, pages 1–10.

Johnsson, T. (1984). Efficient compilation of lazy evaluation. *ACM Conference on Compiler Construction*, pages 58–69.

Kamareddine, F. and Ríos, A. (1997). Extending a λ-calculus with explicit substitution which preserves strong normalisation into a confluent calculus on open terms. *Journal of Functional Programming*, 7(4):395–420.

Kennaway, R. and Sleep, R. (1988). Director strings as combinators. *ACM Transactions on Programming Languages and Systems*, 10(4):602–626.

Lambek, J. and Scott, P. J. (1989). *Introduction to Higher Order Categorical Logic.* Cambridge University Press.

Lamping, J. (1990). An algorithm for optimal lambda-calculus reductions. In *Conference record of the 17th ACM Symposium on Principles of Programming Languages (POPL'90)*, pages 16–30. ACM Press.

Melliès, P.-A. (1995). Typed λ-calculi with explicit substitutions may not terminate. In *Proceedings of the 2nd International Conference on Typed Lambda-Calculi and Applications (TLCA'95)*, pages 328–334. Springer-Verlag LNCS 902.

Murthy, C. R. (1991). Classical proofs as programs: How, when and why. In *Proceedings of the First Annual Symposium on Constructivity in Computer Science.*

Peyton-Jones, S. (1986). *The Implementation of Functional Programming Languages.* Prentice-Hall.

Plotkin, G. D. (1975). Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1(2):125–159.

Turner, D. A. (1979). A new implementation technique for applicative languages. *Software–Practice and Experience*, 9:31–49.

Wand, M. and Friedman, D. P. (1986). The mystery of the tower revealed: A non-reflexive description of the reflexive tower. In *Conference Record of the 1982 ACM Symposium on Lisp and Functional Programming (LFP'86)*, pages 298–307.

Yokouchi, H. (1989). Church-Rosser theorem for a rewriting system on categorical combinators. *Theoretical Computer Science*, 65(3):271–290.

Appendix A. Completeness of SKIn

We first prove some auxiliary Lemmas:

Lemma A.1. If $P_i \rhd^* Q_i$ in the λ-calculus for every $0 \leq i < n$, then:

$$[\![M]\!](P_0, \ldots, P_{n-1}) \rhd^* [\![M]\!](Q_0, \ldots, Q_{n-1})$$

The same result holds if we replace $\rhd$ by $\rhd_\eta$.

Proof. By structural induction on M. We consider variables and K.

If M is a variable, then $[\![M]\!](P_0, \ldots, P_{n-1}) \equiv x(P_0, \ldots, P_{n-1}) \rhd^* x(Q_0, \ldots, Q_{n-1}) \equiv [\![M]\!](Q_0, \ldots, Q_{n-1})$.

If $M \equiv \mathrm{K}_m(N)$, then either $n > m$ and:

$$\begin{aligned}
&[\![M]\!](P_0, \ldots, P_{n-1}) \\
&\equiv [\![N]\!](P_0, \ldots, P_{m-1}, P_{m+1}, \ldots, P_{n-1}) \\
&\rhd^* [\![N]\!](Q_0, \ldots, Q_{m-1}, Q_{m+1}, \ldots, Q_{n-1}) \quad \text{(by induction hypothesis)} \\
&\equiv [\![M]\!](Q_0, \ldots, Q_{n-1})
\end{aligned}$$

or $n \leq m$ and:

$$\begin{aligned}
&[\![M]\!](P_0, \ldots, P_{n-1}) \\
&\equiv \lambda x_n \ldots \lambda x_m . [\![N]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1}) \\
&\rhd^* \lambda x_n \ldots \lambda x_m . [\![N]\!](Q_0, \ldots, Q_{n-1}, x_n, \ldots, x_{m-1}) \quad \text{(by induction hypothesis)} \\
&\equiv [\![M]\!](Q_0, \ldots, Q_{n-1})
\end{aligned}$$

$\triangleright_\eta$ follows similarly. □

Lemma A.2. For every SKIn-term M, for all $0 \le m \le n$, and for all λ-terms $P_0, \ldots, P_{n-1}$:

$$([\![M]\!](P_0, \ldots, P_{m-1}))(P_m, \ldots, P_{n-1}) \triangleright^* [\![M]\!](P_0, \ldots, P_{n-1})$$

in the λ-calculus.

Proof. It is enough to prove the Lemma when $m < n$, since the case $m = n$ is trivial. We then prove the claim by structural induction on M. We consider the cases for S.

If $M \equiv \mathrm{S}_k(N, P)$ and $k \le m < n$, then:

$$\begin{aligned}
&([\![M]\!](P_0, \ldots, P_{m-1}))(P_m, \ldots, P_{n-1}) \\
&\equiv ([\![N]\!](P_0, \ldots, P_{k-1}, [\![P]\!](P_0, \ldots, P_{k-1}), P_k, \ldots, P_{m-1}))(P_m, \ldots, P_{n-1}) \\
&\triangleright^* [\![N]\!](P_0, \ldots, P_{k-1}, [\![P]\!](P_0, \ldots, P_{k-1}), P_k, \ldots, P_{n-1}) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\text{(by induction hypothesis on } N) \\
&\equiv [\![M]\!](P_0, \ldots, P_{n-1})
\end{aligned}$$

If $M \equiv \mathrm{S}_k(N, P)$ and $m < k \le n$, then:

$$\begin{aligned}
&([\![M]\!](P_0, \ldots, P_{m-1}))(P_m, \ldots, P_{n-1}) \\
&\equiv (\lambda x_m \ldots \lambda x_{k-1}. \\
&\qquad [\![N]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}, [\![P]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}))) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad (P_m, \ldots, P_{n-1}) \\
&\triangleright^* ([\![N]\!](P_0, \ldots, P_{m-1}, P_m, \ldots, P_{k-1}, [\![P]\!](P_0, \ldots, P_{m-1}, P_m, \ldots, P_{k-1}))) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad (P_k, \ldots, P_{n-1}) \\
&\triangleright^* [\![N]\!](P_0, \ldots, P_{k-1}, [\![P]\!](P_0, \ldots, P_{k-1}), P_k, \ldots, P_{n-1}) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\text{(by induction hypothesis on } N) \\
&\equiv [\![M]\!](P_0, \ldots, P_{n-1})
\end{aligned}$$

If $M \equiv \mathrm{S}_k(N, P)$ and $m < n < k$, then:

$$\begin{aligned}
&([\![M]\!](P_0, \ldots, P_{m-1}))(P_m, \ldots, P_{n-1}) \\
&\equiv (\lambda x_m \ldots \lambda x_{k-1}. \\
&\qquad [\![N]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}, [\![P]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}))) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad (P_m, \ldots, P_{n-1}) \\
&\triangleright^* \lambda x_n \ldots \lambda x_{k-1}.[\![N]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1}, [\![P]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1})) \\
&\equiv [\![M]\!](P_0, \ldots, P_{n-1})
\end{aligned}$$

□

Lemma A.3. For every SKIn-term M, all pairwise distinct variables $x_m, \ldots, x_{n-1}$ not free in M, and all λ-terms $P_0, \ldots, P_{m-1}$, if $m \le n$, then:

$$\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{n-1}) \triangleright_\eta^* [\![M]\!](P_0, \ldots, P_{m-1})$$

Proof. By structural induction on M. Notice that the case $m = n$ is trivial, so that we can assume $m < n$. We consider S.

If $M \equiv \mathrm{S}_k(N,P)$ and $k \le m < n$, then:

$$
\begin{aligned}
&\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0,\ldots,P_{m-1},x_m,\ldots,x_{n-1})\\
&\equiv \lambda x_m \ldots \lambda x_{n-1}.[\![N]\!](P_0,\ldots,P_{k-1},[\![P]\!](P_0,\ldots,P_{k-1}),P_k,\ldots,P_{m-1},x_m,\ldots,x_{n-1})\\
&\triangleright_\eta^* [\![N]\!](P_0,\ldots,P_{k-1},[\![P]\!](P_0,\ldots,P_{k-1}),P_k,\ldots,P_{m-1}) \quad \text{(by induction hypothesis)}\\
&\equiv [\![M]\!](P_0,\ldots,P_{m-1})
\end{aligned}
$$

If $M \equiv \mathrm{S}_k(N,P)$ and $m < k \le n$, then:

$$
\begin{aligned}
&\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0,\ldots,P_{m-1},x_m,\ldots,x_{n-1})\\
&\equiv \lambda x_m \ldots \lambda x_{n-1}.[\![N]\!](P_0,\ldots,P_{k-1},[\![P]\!](P_0,\ldots,P_{k-1}),P_k,\ldots,P_{m-1},x_m,\ldots,x_{n-1})\\
&\triangleright_\eta^* \lambda x_m \ldots \lambda x_{k-1}.[\![N]\!](P_0,\ldots,P_{k-1},[\![P]\!](P_0,\ldots,P_{k-1}),P_k,\ldots,P_{m-1},x_m,\ldots,x_{k-1})\\
&\qquad\qquad\qquad\qquad \text{(obvious if } n=k\text{, otherwise by induction hypothesis)}\\
&\equiv [\![M]\!](P_0,\ldots,P_{m-1})
\end{aligned}
$$

If $M \equiv \mathrm{S}_k(N,P)$ and $m < n < k$, then:

$$
\begin{aligned}
&\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0,\ldots,P_{m-1},x_m,\ldots,x_{n-1})\\
&\equiv \lambda x_m \ldots \lambda x_{n-1}.\lambda x_n \ldots \lambda x_{k-1}.\\
&\qquad [\![M]\!](P_0,\ldots,P_{m-1},x_m,\ldots,x_{k-1},[\![N]\!](P_0,\ldots,P_{m-1},x_m,\ldots,x_{k-1}))\\
&\equiv [\![M]\!](P_0,\ldots,P_{m-1})
\end{aligned}
$$

□

Definition A.4. The *contexts*, i.e. terms with one hole [], are defined by the grammar:

$$\mathcal{C} ::= [] \mid \mathrm{S}_k(\mathcal{C},N) \mid \mathrm{S}_k(M,\mathcal{C}) \mid \mathrm{K}_k(\mathcal{C})$$

We write $\mathcal{C}[M]$ for the result of replacing the hole [] by M in $\mathcal{C}$.

Rewriting M to N by some rule means that there is a context $\mathcal{C}$ such that $M \equiv \mathcal{C}[L]$, $N \equiv \mathcal{C}[R]$, and $L \triangleright R$ is an instance of the given rule.

Proof. **(of Theorem 3.5)** We prove the result by structural induction on $\mathcal{C}$, and in the base case, by case analysis on the rule used. Let M, N, L, R, $\mathcal{C}$ be as above. We consider the cases where $\mathcal{C}$ is $\mathrm{S}_k(P,\mathcal{C}_1)$ or empty.

If $\mathcal{C} \equiv \mathrm{S}_k(P,\mathcal{C}_1)$, then either $n \ge k$ and:

$$
\begin{aligned}
&[\![M]\!](P_0,\ldots,P_{n-1})\\
&\equiv [\![P]\!](P_0,\ldots,P_{k-1},[\![\mathcal{C}_1[L]]\!](P_0,\ldots,P_{k-1}),P_k,\ldots,P_{n-1})\\
&\triangleright_\eta^* [\![P]\!](P_0,\ldots,P_{k-1},[\![\mathcal{C}_1[R]]\!](P_0,\ldots,P_{k-1}),P_k,\ldots,P_{n-1})\\
&\qquad\qquad \text{(by induction hypothesis and Lemma A.1)}\\
&\equiv [\![N]\!](P_0,\ldots,P_{n-1})
\end{aligned}
$$

or $n < k$ and:

$$
\begin{aligned}
&[\![M]\!](P_0,\ldots,P_{n-1})\\
&\equiv \lambda x_n \ldots \lambda x_{k-1}.[\![P]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{k-1},[\![\mathcal{C}_1[L]]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{k-1}))\\
&\triangleright_\eta^* \lambda x_n \ldots \lambda x_{k-1}.[\![P]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{k-1},[\![\mathcal{C}_1[R]]\!](P_0,\ldots,P_{n-1},x_n,\ldots,x_{k-1}))\\
&\qquad\qquad\qquad\qquad \text{(by induction hypothesis and Lemma A.1)}\\
&\equiv [\![N]\!](P_0,\ldots,P_{n-1})
\end{aligned}
$$

If $\mathcal{C}$ is the empty context, then we need to examine each rule in turn.

We consider rules (SI_k), (SK_k) and ($\mathrm{S}_k\mathrm{K}_p$), where $k < p$. Note that in all cases except in (SI_k), $[\![M]\!](P_0, \dots, P_{n-1})$ actually equals $[\![N]\!](P_0, \dots, P_{n-1})$.

- Rule (SI_k): $M \equiv \mathrm{S}_k(\mathrm{I}_k, N)$; either $n \geq k$ and:

$$\begin{aligned} [\![M]\!](P_0, \dots, P_{n-1}) &\equiv [\![\mathrm{I}_k]\!](P_0, \dots, P_{k-1}, [\![N]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}) \\ &\equiv ([\![N]\!](P_0, \dots, P_{k-1}))(P_k, \dots, P_{n-1}) \triangleright^* [\![N]\!](P_0, \dots, P_{n-1}) \end{aligned}$$

by Lemma A.2, or $n < k$ and:

$$\begin{aligned} &[\![M]\!](P_0, \dots, P_{n-1}) \\ &\equiv \lambda x_n \dots \lambda x_{k-1}.[\![\mathrm{I}_k]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}, [\![N]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1})) \\ &\equiv \lambda x_n \dots \lambda x_{k-1}.[\![N]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}) \triangleright_\eta^* [\![N]\!](P_0, \dots, P_{n-1}) \end{aligned}$$

by Lemma A.3, since $n \leq k$.

- Rule (SK_k): $M \equiv \mathrm{S}_k(\mathrm{K}_k(N), P)$; either $n \geq k$ and:

$$\begin{aligned} &[\![M]\!](P_0, \dots, P_{n-1}) \\ &\equiv [\![\mathrm{K}_k(N)]\!](P_0, \dots, P_{k-1}, [\![P]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}) \\ &\equiv [\![N]\!](P_0, \dots, P_{k-1}, P_k, \dots, P_{n-1}) && \text{(since } n+1 > k\text{)} \\ &\equiv [\![N]\!](P_0, \dots, P_{n-1}) \end{aligned}$$

or $n < k$ and:

$$\begin{aligned} &[\![M]\!](P_0, \dots, P_{n-1}) \\ &\equiv \lambda x_n \dots \lambda x_{k-1}.[\![\mathrm{K}_k(N)]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}, [\![P]\!](P_0, \dots, P_{k-1})) \\ &\equiv \lambda x_n \dots \lambda x_{k-1}.[\![N]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}) \\ &\equiv [\![N]\!](P_0, \dots, P_{n-1}) \end{aligned}$$

- Rule ($\mathrm{S}_k\mathrm{K}_p$), $k < p$: $M \equiv \mathrm{S}_k(\mathrm{K}_p(N_1), P)$, $N \equiv \mathrm{K}_{p-1}(\mathrm{S}_k(N_1, P))$; either $n \geq p$ and:

$$\begin{aligned} &[\![M]\!](P_0, \dots, P_{n-1}) \\ &\equiv [\![\mathrm{K}_p(N_1)]\!](P_0, \dots, P_{k-1}, [\![P]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}) \\ &\equiv [\![N_1]\!](P_0, \dots, P_{k-1}, [\![P]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{p-2}, P_p, \dots, P_{n-1}) \\ &\equiv [\![\mathrm{S}_k(N_1, P)]\!](P_0, \dots, P_{p-2}, P_p, \dots, P_{n-1}) \\ &\equiv [\![\mathrm{K}_{p-1}(\mathrm{S}_k(N_1, P))]\!](P_0, \dots, P_{n-1}) \equiv [\![N]\!](P_0, \dots, P_{n-1}) \end{aligned}$$

or $p > n \geq k$ and:

$$\begin{aligned} &[\![M]\!](P_0, \dots, P_{n-1}) \\ &\equiv [\![\mathrm{K}_p(N_1)]\!](P_0, \dots, P_{k-1}, [\![P]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}) \\ &\equiv \lambda x_{n+1} \dots \lambda x_p.[\![N_1]\!](P_0, \dots, P_{k-1}, [\![P]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}, x_{n+1}, \dots, x_{p-1}) \\ &\equiv \lambda x_n \dots \lambda x_{p-1}.[\![N_1]\!](P_0, \dots, P_{k-1}, [\![P]\!](P_0, \dots, P_{k-1}), P_k, \dots, P_{n-1}, x_n, \dots, x_{p-2}) \\ &\equiv \lambda x_n \dots \lambda x_{p-1}.[\![\mathrm{S}_k(N_1, P)]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{p-2}) \\ &\equiv [\![\mathrm{K}_{p-1}(\mathrm{S}_k(N_1, P))]\!](P_0, \dots, P_{n-1}) \equiv [\![N]\!](P_0, \dots, P_{n-1}) \end{aligned}$$

or $k > n$ and:

$$
\begin{aligned}
&[\![M]\!](P_0, \dots, P_{n-1}) \\
&\equiv \lambda x_n \dots \lambda x_{k-1}. \\
&\quad [\![\mathrm{K}_p(N_1)]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}, [\![P]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1})) \\
&\equiv \lambda x_n \dots \lambda x_{k-1}.\lambda x_{k+1} \dots \lambda x_p. \\
&\quad [\![N_1]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}, [\![P]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}), x_{k+1}, \dots, x_p) \\
&\equiv \lambda x_n \dots \lambda x_{k-1}.\lambda x_k \dots \lambda x_{p-1}. \\
&\quad [\![N_1]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}, [\![P]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{k-1}), x_k, \dots, x_{p-1}) \\
&\equiv \lambda x_n \dots \lambda x_{p-1}.[\![\mathrm{S}_k(N_1, P)]\!](P_0, \dots, P_{n-1}, x_n, \dots, x_{p-1}) \\
&\equiv [\![\mathrm{K}_{p-1}(\mathrm{S}_k(N_1, P))]\!](P_0, \dots, P_{n-1}) \equiv [\![N]\!](P_0, \dots, P_{n-1})
\end{aligned}
$$

□

Appendix B. Conservativity of SKIn

Lemma B.1. For every SKIn-term M, for every $n \geq 0$:

$$[\![[x]M]\!](x, Q_0, \dots, Q_{n-1}) \equiv [\![M]\!](Q_0, \dots, Q_{n-1})$$

Proof. By structural induction on M. We consider variables and S.
If $M \equiv x$, then:

$$
\begin{aligned}
[\![[x]M]\!](x, Q_0, \dots, Q_{n-1}) &\equiv [\![\mathrm{I}_0]\!](x, Q_0, \dots, Q_{n-1}) \\
&\equiv x(Q_0, \dots, Q_{n-1}) \equiv [\![x]\!](Q_0, \dots, Q_{n-1}) \\
&\equiv [\![M]\!](Q_0, \dots, Q_{n-1})
\end{aligned}
$$

If M is another variable y, then:

$$
\begin{aligned}
[\![[x]M]\!](x, Q_0, \dots, Q_{n-1}) &\equiv [\![\mathrm{K}_0(y)]\!](x, Q_0, \dots, Q_{n-1}) \\
&\equiv [\![y]\!](Q_0, \dots, Q_{n-1}) \equiv [\![M]\!](Q_0, \dots, Q_{n-1})
\end{aligned}
$$

If $M \equiv \mathrm{S}_m(M_1, M_2)$, then either $n \geq m$ and:

$$
\begin{aligned}
&[\![[x]M]\!](x, Q_0, \dots, Q_{n-1}) \\
&\equiv [\![\mathrm{S}_{m+1}([x]M_1, [x]M_2)]\!](x, Q_0, \dots, Q_{n-1}) \\
&\equiv [\![[x]M_1]\!](x, Q_0, \dots, Q_{m-1}, [\![[x]M_2]\!](x, Q_0, \dots, Q_{m-1}), Q_m \dots, Q_{n-1}) \\
&\equiv [\![M_1]\!](Q_0, \dots, Q_{m-1}, [\![M_2]\!](Q_0, \dots, Q_{m-1}), Q_m \dots, Q_{n-1}) \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(by induction hypothesis)} \\
&\equiv [\![\mathrm{S}_m(M_1, M_2)]\!](Q_0, \dots, Q_{n-1}) \\
&\equiv [\![M]\!](Q_0, \dots, Q_{n-1})
\end{aligned}
$$

or $n < m$ and:

$$\begin{aligned}
&[\![[x]M]\!](x, Q_0, \ldots, Q_{n-1})\\
&\equiv [\![\mathrm{S}_{m+1}([x]M_1, [x]M_2)]\!](x, Q_0, \ldots, Q_{n-1})\\
&\equiv \lambda x_{n+1} \ldots \lambda x_m.[\![[x]M_1]\!](x, Q_0, \ldots, Q_{n-1}, x_{n+1}, \ldots, x_m,\\
&\qquad\qquad [\![[x]M_2]\!](x, Q_0, \ldots, Q_{n-1}, x_{n+1}, \ldots, x_m))\\
&\equiv \lambda x_{n+1} \ldots \lambda x_m.\\
&\quad [\![M_1]\!](Q_0, \ldots, Q_{n-1}, x_{n+1}, \ldots, x_m, [\![M_2]\!](Q_0, \ldots, Q_{n-1}, x_{n+1}, \ldots, x_m))\\
&\qquad\qquad\qquad\qquad\text{(by induction hypothesis)}\\
&\equiv \lambda x_n \ldots \lambda x_{m-1}.\\
&\quad [\![M_1]\!](Q_0, \ldots, Q_{n-1}, x_n, \ldots, x_{m-1}, [\![M_2]\!](Q_0, \ldots, Q_{n-1}, x_n, \ldots, x_{m-1}))\\
&\equiv [\![\mathrm{S}_m(M_1, M_2)]\!](Q_0, \ldots, Q_{n-1})\\
&\equiv [\![M]\!](Q_0, \ldots, Q_{n-1})
\end{aligned}$$

□

Lemma B.2. For every SKIn-term M, $[\![[x]M]\!]() \equiv \lambda x.[\![M]\!]()$.

Proof. By structural induction on M. We consider variables and K.

If $M \equiv x$, then $[\![[x]M]\!]() \equiv [\![\mathrm{I}_0]\!]() \equiv \lambda x_0.x_0$, while $\lambda x \cdot [\![M]\!]() \equiv \lambda x.x$, so these terms are α-equivalent.

If M is another variable y, then $[\![[x]M]\!]() \equiv [\![\mathrm{K}_0(y)]\!]() \equiv \lambda x_0.[\![y]\!]() \equiv \lambda x_0.y$, while $\lambda x \cdot [\![M]\!]() \equiv \lambda x.y$, and these terms are α-equivalent.

If $M \equiv \mathrm{K}_m(M_1)$, then:

$$\begin{aligned}
&[\![[x]M]\!]()\\
&\equiv [\![\mathrm{K}_{m+1}([x]M_1)]\!]()\\
&\equiv \lambda x_0 \ldots \lambda x_{m+1}.[\![[x]M_1]\!](x_0, \ldots, x_m)\\
&\equiv \lambda x.\lambda x_0 \ldots \lambda x_m.[\![[x]M_1]\!](x, x_0, \ldots, x_{m-1}) && \text{(by α-renaming)}\\
&\equiv \lambda x.\lambda x_0 \ldots \lambda x_m.[\![M_1]\!](x_0, \ldots, x_{m-1}) && \text{(by Lemma B.1)}\\
&\equiv \lambda x.[\![\mathrm{K}_m(M_1)]\!]() \equiv \lambda x.[\![M]\!]()
\end{aligned}$$

□

Proof. **(of Lemma 3.22)** We show more generally that, for all λ-terms $Q_0, \ldots, Q_{n-1}$, $P(Q_0, \ldots, Q_{n-1}) \rhd^* [\![P^*]\!](Q_0, \ldots, Q_{n-1})$, by structural induction on P.

If P is a variable x, then:

$$[\![P^*]\!](Q_0, \ldots, Q_{n-1}) \equiv [\![x]\!](Q_0, \ldots, Q_{n-1}) \equiv x(Q_0, \ldots, Q_{n-1}) \equiv P(Q_0, \ldots, Q_{n-1})$$

If P is of the form $P_1(P_2)$, then $[\![P^*]\!](Q_0, \ldots, Q_{n-1}) \equiv [\![\mathrm{S}_0(P_1^*, P_2^*)]\!](Q_0, \ldots, Q_{n-1}) \equiv [\![P_1^*]\!]([\![P_2^*]\!](), Q_0, \ldots, Q_{n-1})$. But:

$$\begin{aligned}
& P_1(P_2)(Q_0, \ldots, Q_{n-1})\\
\equiv\ & P_1(P_2, Q_0, \ldots, Q_{n-1})\\
\rhd^*\ & P_1([\![P_2^*]\!](), Q_0, \ldots, Q_{n-1}) && \text{(by induction hypothesis on P_2)}\\
\rhd^*\ & [\![P_1^*]\!]([\![P_2^*]\!](), Q_0, \ldots, Q_{n-1}) && \text{(by induction hypothesis on P_1)}\\
\equiv\ & [\![P^*]\!](Q_0, \ldots, Q_{n-1})
\end{aligned}$$

If P is of the form $\lambda x.P_1$, then $[\![P^*]\!](Q_0,\ldots,Q_{n-1}) \equiv [\![[x]P_1^*]\!](Q_0,\ldots,Q_{n-1})$. If $n = 0$, then $[\![P^*]\!]() \equiv [\![[x]P_1^*]\!]() \equiv \lambda x.[\![P_1^*]\!]()$ (by Lemma B.2) and by induction $P_1 \triangleright^* [\![P_1^*]\!]()$, so $P \triangleright^* [\![P^*]\!]()$. If $n \neq 0$, then on the one hand $[\![P^*]\!](Q_0,\ldots,Q_{n-1}) \equiv [Q_0/x][\![[x]P_1^*]\!](x,Q_1,\ldots,Q_{n-1}) \equiv [Q_0/x][\![P_1^*]\!](Q_1,\ldots,Q_{n-1})$, but on the other hand $P(Q_0,Q_1,\ldots,Q_{n-1}) \triangleright ([Q_0/x]P_1)(Q_1,\ldots,Q_{n-1}) \equiv [Q_0/x]P_1(Q_1,\ldots,Q_{n-1})$ (by the variable naming conventions) which rewrites to $[Q_0/x][\![P_1^*]\!](Q_1,\ldots,Q_{n-1})$, that is, to $[\![P^*]\!](Q_0,\ldots,Q_{n-1})$ by induction. □

Now we prove a few properties about well-stagedness. The first is a subject reduction property:

Lemma B.3. For every well-staged SKIn-term M, if $M \triangleright N$, then N is well-staged and $lev(N) \geq lev(M)$.

Proof. Let's write $M \equiv \mathcal{C}[L]$ and $N \equiv \mathcal{C}[R]$, where $L \triangleright R$ is an instance of some rewrite rule in SKIn, and $\mathcal{C}$ is a context. We prove the claim by structural induction on $\mathcal{C}$. If $\mathcal{C} \equiv \mathrm{K}_m(\mathcal{C}_1)$ for some smaller context $\mathcal{C}_1$, then by induction hypothesis $\mathcal{C}_1[R]$ is well-staged and $lev(\mathcal{C}_1[R]) \geq lev(\mathcal{C}_1[L]) \geq m$ (since M is well-staged), so N is well-staged; moreover, $lev(N) = \max(m, lev(\mathcal{C}[R])) + 1 \geq \max(m, lev(\mathcal{C}[L])) + 1 = lev(M)$. Similarly when $\mathcal{C}$ is of the form $\mathrm{S}_m(M_1,\mathcal{C}_1)$ or when $\mathcal{C}$ is of the form $\mathrm{S}_m(\mathcal{C}_1,N_1)$.

Finally, we deal with the base case, namely when $\mathcal{C}$ is the empty context. We consider several cases:

Rule (SI_m): $M \equiv \mathrm{S}_m(\mathrm{I}_m,N)$, and since M is well-staged, so is its subterm N; since M is well-staged again, $lev(N) \geq m$. But $lev(M) = \max(m, m+1-1) = m$ (by definition) $\leq lev(N)$.

In the sequel, we assume that $0 \leq m < n$.

Rule ($\mathrm{S}_m\mathrm{K}_n$): $M \equiv \mathrm{S}_m(\mathrm{K}_n(M_1),P)$, $N \equiv \mathrm{K}_{n-1}(\mathrm{S}_m(M_1,P))$. Since M is well-staged, so are M_1 and P; moreover, (a) $lev(M_1) \geq n$ and (b) $lev(P) \geq m$. By (a) and (b), $\mathrm{S}_m(M_1,P)$ is well-staged. Moreover, $lev(\mathrm{S}_m(M_1,P)) = \max(m, lev(M_1) - 1) \geq n - 1$ by (a), so N is well-staged. Finally, $lev(M) = \max(m, \max(n, lev(M_1)) + 1 - 1) = \max(n, lev(M_1))$, while $lev(N) = \max(n-1, \max(m, lev(M_1) - 1)) + 1 = \max(n-1, lev(M_1) - 1) + 1 = \max(n, lev(M_1))$, so $lev(M) = lev(N)$.

□

Lemma B.4. Let M be a well-staged term of level n, then $[x]M$ is well-staged of level $n+1$.

Proof. By structural induction on M.

The base cases are straightforward. We consider the case where M is of the form $\mathrm{K}_m(N)$, then $lev(N) \geq m$ and by induction hypothesis $[x]N$ is well-staged of level $lev(N)+1$; in particular $lev([x]N) \geq m+1$, so $[x]M \equiv \mathrm{K}_{m+1}([x]N)$ is well-staged; and $lev([x]M) = \max(m+1, lev([x]N))+1 = \max(m+1, lev(N)+1)+1 = (\max(m, lev(N))+1)+1 = lev(M)+1$. □

Proof. **(of Lemma 3.19)** We show that M^* is well-staged by structural induction on M. If $M \equiv x$, then $x^* \equiv x$ is well-staged (and of level 0). If M is an application $M_1(M_2)$, then $M^* \equiv \mathrm{S}_0(M_1^*, M_2^*)$, where M_1^* and M_2^* are well-staged by induction hypothesis; moreover, $lev(M_1^*) \geq 0$ and $lev(M_2^*) \geq 0$ by definition of the levels, so M^* is well-staged.

And if M is an abstraction $\lambda x.M_1$, then $M^* \equiv [x]M_1^*$; by induction hypothesis, M_1^* is well-staged, and by Lemma B.4, $[x]M_1^*$ is well-staged, so M^* is well-staged. □

We now attack the proof of Theorem 3.20. We need to adapt some of the results of A.

Lemma B.5. For every well-staged SKIn-term M, all pairwise distinct variables x_m, ..., x_{n-1} not free in M, and all λ-terms P_0, ..., P_{m-1}, if $m \leq n \leq lev(M)$, then:

$$\lambda x_m \cdot \ldots \cdot \lambda x_{n-1} \cdot [\![M]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{n-1}) \rhd^* [\![M]\!](P_0, \ldots, P_{m-1})$$

Proof. This is as for Lemma A.3, except that the additional assumptions $n \leq lev(M)$ and M well-staged allow us to replace $\rhd_\eta$ by $\rhd$.

The proof is by structural induction on M. Again, the case $m = n$ is trivial, so that we can assume $m < n$. When M is a variable x, then $lev(M) = 0$, so this case is vacuous.

If $M \equiv \mathrm{K}_k(N)$ and $k < m < n$, then:

$$\begin{aligned}
&\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{n-1}) \\
&\equiv \lambda x_m \ldots \lambda x_{n-1}.[\![N]\!](P_0, \ldots, P_{k-1}, P_{k+1}, \ldots, P_{m-1}, x_m, \ldots, x_{n-1}) \\
&\rhd^* [\![N]\!](P_0, \ldots, P_{k-1}, P_{k+1}, \ldots, P_{m-1}) \quad \text{(by induction hypothesis)} \\
&\equiv [\![M]\!](P_0, \ldots, P_{m-1})
\end{aligned}$$

To show that the induction hypothesis was applicable, we need to show that $n - 1 \leq lev(N)$. By assumption, $n \leq lev(M)$, and by definition $lev(M) = \max(k, lev(N)) + 1$, so $n \leq \max(k, lev(N)) + 1$, therefore $n - 1 \leq \max(k, lev(N))$. Since we are in a case where $k < m < n$, in particular $k \leq n-2$, so $n-1 \leq \max(n-2, lev(N))$. It follows that $n-1 \leq n-2$ or $n-1 \leq lev(N)$, but the first case is impossible. So indeed $n-1 \leq lev(N)$.

If $M \equiv \mathrm{K}_k(N)$ and $m \leq k < n$, then:

$$\begin{aligned}
&\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{n-1}) \\
&\equiv \lambda x_m \ldots \lambda x_{n-1}.[\![N]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}, x_{k+1}, \ldots, x_{n-1}) \\
&\rhd^* \lambda x_m \ldots \lambda x_k.[\![N]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}) \\
&\qquad \text{(obvious if } k = n-1\text{, otherwise by induction hypothesis)} \\
&\equiv [\![M]\!](P_0, \ldots, P_{m-1})
\end{aligned}$$

Indeed, if $k \neq n-1$, then $k \leq n-2$, and the same argument as in the previous case (case $k < m < n$) shows that the induction hypothesis was applicable.

And if $M \equiv \mathrm{K}_k(N)$ and $m < n \leq k$, then:

$$\begin{aligned}
&\lambda x_m \ldots \lambda x_{n-1}.[\![M]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{n-1}) \\
&\equiv \lambda x_m \ldots \lambda x_{n-1}.\lambda x_n \ldots \lambda x_k.[\![N]\!](P_0, \ldots, P_{m-1}, x_m, \ldots, x_{k-1}) \\
&\equiv [\![M]\!](P_0, \ldots, P_{m-1})
\end{aligned}$$

□

Proof. **(of Theorem 3.20)** This is as for Theorem 3.5, with a few changes. We show that $M \rhd N$ and M well-staged implies $[\![M]\!](P_0, \ldots, P_{n-1}) \rhd^* [\![N]\!](P_0, \ldots, P_{n-1})$; that N is well-staged is by Lemma B.3. It will follow that $M \rhd N$ and M well-staged imply N well-staged and $[\![M]\!]() \rhd^* [\![N]\!]()$, trivially. That $M = N$ in SKIn, when M and N are well-staged, implies $[\![M]\!]() = [\![N]\!]()$ follows, in turn, from the fact that, assuming SKIn to be confluent, if $M = N$ then $M \rhd^* P$ and $N \rhd^* P$ for some term P, so $[\![M]\!]() \rhd^* [\![P]\!]()$ and $[\![N]\!]() \rhd^* [\![P]\!]()$.

Let therefore M be $\mathcal{C}[L]$, N be $\mathcal{C}[R]$, where $L \rhd R$ is some instance of a rule of SKIn. We show that $[\![M]\!](P_0, \ldots, P_{n-1}) \rhd^* [\![N]\!](P_0, \ldots, P_{n-1})$ by structural induction on $\mathcal{C}$, and in the base case, by analysis of the rule used. We only describe the differences with the proof of Theorem 3.5 (see A).

In the induction cases, i.e. when $\mathcal{C} \not\equiv []$, the argument is unchanged, except for replacing $\rhd_\eta$ by $\rhd$. So deal with the base case:

• Rule (SI_k): $M \equiv \mathrm{S}_k(\mathrm{I}_k, N)$, and $lev(M) = k$, $lev(N) \geq k$. If $n \geq k$, then the argument in as in the proof of Theorem 3.5. If $n < k$, however, we compute:

$$\begin{aligned}
&[\![M]\!](P_0, \ldots, P_{n-1}) \\
&\equiv \lambda x_n \ldots \lambda x_{k-1}.[\![\mathrm{I}_k]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1}, [\![N]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1})) \\
&\equiv \lambda x_n \ldots \lambda x_{k-1}.[\![N]\!](P_0, \ldots, P_{n-1}, x_n, \ldots, x_{k-1}) \\
&\rhd^* [\![N]\!](P_0, \ldots, P_{n-1})
\end{aligned}$$

by Lemma B.5, since $n \leq k \leq lev(N)$.

In the cases of the other rules, the argument remains unchanged, apart from the replacement of $\rhd_\eta$ by $\rhd$. □

Appendix C. Termination of ΣT

We show that ΣT terminates by proving that larger and larger groups of rules terminate.

C.1. *The* K *Group*

Let the K *group* denote the following group of SKInT reduction rules:

$$\begin{array}{rrcl}
(\mathrm{K}_m\mathrm{I}_n) & \mathrm{K}_m(\mathrm{I}_{n-1}) & \rhd & \mathrm{I}_n \\
(\mathrm{K}_m\mathrm{K}_n) & \mathrm{K}_m(\mathrm{K}_{n-1}(M)) & \rhd & \mathrm{K}_n(\mathrm{K}_m(M)) \\
(\mathrm{K}_m\mathrm{S}_{n+1}) & \mathrm{K}_m(\mathrm{S}_n(M, P)) & \rhd & \mathrm{S}_{n+1}(\mathrm{K}_m(M), \mathrm{K}_m(P))
\end{array}$$

where $0 \leq m < n$. In this section, we show that the K group terminates. To this end, we define a function $M \mapsto q_i(M)$ that predicts how the rules in the K group will change the indices of operators inside M when reducing $\mathrm{K}_i(M)$. The function $M \mapsto [\![M]\!]_q$ tries to predict all possible index changes due to rules in the K group. This is the basic idea, but the actual definition is heavily tweaked so as to make all proofs go through:

Definition C.1. Let q_i, $i \geq 0$, be the functions defined as follows:

$$\begin{array}{llll}
q_i(\mathrm{S}_j(M, N)) =_{\mathrm{df}} \begin{cases} \mathrm{S}_{j+1}(q_i(M), q_i(N)) & (i \leq j-1) \\ \mathrm{S}_j(q_{i+1}(M), N) & (j-1 < i) \end{cases} & & q_i(\mathrm{I}_j) =_{\mathrm{df}} \begin{cases} \mathrm{I}_{j+1} & (i \leq j) \\ \mathrm{I}_j & (j < i) \end{cases} \\
q_i(\mathrm{K}_j(M)) =_{\mathrm{df}} \begin{cases} \mathrm{K}_{j+1}(q_i(M)) & (i \leq j) \\ \mathrm{K}_j(q_i(M)) & (j < i) \end{cases} & & q_i(x) =_{\mathrm{df}} x
\end{array}$$

where $j \geq 0$.

$$
\begin{array}{rrcl}
[\![\mathrm{S}_m\mathrm{I}_n]\!]_q & \mathrm{S}_m(\mathrm{I}_n, P) & \triangleright & \mathrm{I}_{n-1} \\
[\![\mathrm{S}_m\mathrm{S}_n]\!]_q & \mathrm{S}_m(\mathrm{S}_n(M, N), P) & \triangleright & \mathrm{S}_{n-1}(\mathrm{S}_m(M, P), \mathrm{S}_m(N, P)) \\
[\![\mathrm{S}_m\mathrm{K}_n]\!]_q & \mathrm{S}_m(\mathrm{K}_n(M), P) & \triangleright & \mathrm{K}_{n-1}(\mathrm{S}_m(M, P)) \\
[\![\mathrm{K}_m\mathrm{I}_n]\!]_q & \mathrm{K}_m(\mathrm{I}_n) & \triangleright & \mathrm{I}_n \\
[\![\mathrm{K}_m\mathrm{S}_{n+1}]\!]_q & \mathrm{K}_m(\mathrm{S}_{n+1}(M, P)) & \triangleright & \mathrm{S}_{n+1}(\mathrm{K}_m(M), \mathrm{K}_m(P)) \\
[\![\mathrm{K}_m\mathrm{K}_n]\!]_q & \mathrm{K}_m(\mathrm{K}_n(M)) & \triangleright & \mathrm{K}_n(\mathrm{K}_m(M))
\end{array}
$$

Fig. 10. Translating rules by $[\![\_]\!]_q$ $(0 \leq m < n)$

Let $[\![\_]\!]_q$ be defined by:

$$
\begin{array}{rcl@{\quad}rcl}
[\![\mathrm{K}_m(M)]\!]_q & =_{\mathrm{df}} & \mathrm{K}_m(q_m([\![M]\!]_q)) & [\![\mathrm{S}_m(M, N)]\!]_q & =_{\mathrm{df}} & \mathrm{S}_m([\![M]\!]_q, [\![N]\!]_q) \\
[\![\mathrm{I}_m]\!]_q & =_{\mathrm{df}} & \mathrm{I}_m & [\![x]\!]_q & =_{\mathrm{df}} & x
\end{array}
$$

for every $m \geq 0$.

Lemma C.2. For every $0 \leq m < n$, $q_m \circ q_{n-1} = q_n \circ q_m$.

Proof. We prove that $(q_m \circ q_{n-1})(M) \equiv (q_n \circ q_m)(M)$ for every $0 \leq m < n$, by structural induction on M. We consider K.

If $M \equiv \mathrm{K}_j(N)$, then we have three cases:

— $j < m$: $(q_m \circ q_{n-1})(M) \equiv q_m(\mathrm{K}_j(q_{n-1}(N)))$ (because $j < n-1$) $\equiv \mathrm{K}_j((q_m \circ q_{n-1})(N))$ (because $j < m$) $\equiv \mathrm{K}_j((q_n \circ q_m)(N))$ (by induction hypothesis); and $(q_n \circ q_m)(M) \equiv q_n(\mathrm{K}_j(q_m(N))) \equiv \mathrm{K}_j((q_n \circ q_m)(N))$ (because $j < n$).

— $m \leq j < n-1$: $(q_m \circ q_{n-1})(M) \equiv q_m(\mathrm{K}_j(q_{n-1}(N))) \equiv \mathrm{K}_{j+1}((q_m \circ q_{n-1})(N)) \equiv \mathrm{K}_{j+1}((q_n \circ q_m)(N))$ (by induction hypothesis); and $(q_n \circ q_m)(M) \equiv q_n(\mathrm{K}_{j+1}(q_m(N))) \equiv \mathrm{K}_{j+1}((q_n \circ q_m)(N))$ (because $j+1 < n$).

— $n-1 \leq j$: $(q_m \circ q_{n-1})(M) \equiv q_m(\mathrm{K}_{j+1}(q_{n-1}(N))) \equiv \mathrm{K}_{j+2}((q_m \circ q_{n-1})(N))$ (because $m \leq j \leq j+1$) $\equiv \mathrm{K}_{j+2}((q_n \circ q_m)(N))$ (by induction hypothesis); and $(q_n \circ q_m)(M) \equiv q_n(\mathrm{K}_{j+1}(q_m(N))) \equiv \mathrm{K}_{j+2}((q_n \circ q_m)(N))$ (because $n \leq j+1$).

□

Recall that a context $\mathcal{C}$ is a term with a unique distinguished occurrence called the hole and written []. $\mathcal{C}[M]$ denotes the term obtained by replacing the hole by the term M. Recall that $M \triangleright N$ if and only if there is a rule $l \rightarrow r$ and a context $\mathcal{C}$ such that $M \equiv \mathcal{C}[l]$ and $N \equiv \mathcal{C}[r]$.

Let ΣT_0 be ΣT minus (SK_m), $m \geq 0$. In Lemma C.3, we show that applying the $[\![\_]\!]_q$ transformation to the rules in ΣT_0 yields the rules shown in Figure 10. Call this group of rules $\Sigma T_0'$.

Lemma C.3. For any rule $M \triangleright N$ in ΣT_0, $[\![M]\!]_q \triangleright [\![N]\!]_q$ is an instance of some rule of group $\Sigma T_0'$ (in Figure 10).

Proof. By case analysis on the rule. In the sequel, we assume $0 \leq m < n$.

The cases of rules $(\mathrm{S}_m\mathrm{I}_n)$ and $(\mathrm{S}_m\mathrm{S}_n)$ are trivial.

Rule ($\mathrm{S}_m\mathrm{K}_n$). The left-hand side translates to $\mathrm{S}_m(\mathrm{K}_n(q_n([\![M]\!]_q)), [\![P]\!]_q)$, while the right-hand side yields $\mathrm{K}_{n-1}(q_{n-1}(\mathrm{S}_m([\![M]\!]_q, [\![P]\!]_q))) \equiv \mathrm{K}_{n-1}(\mathrm{S}_m(q_n([\![M]\!]_q), [\![P]\!]_q))$ since $m-1 < n-1$.

Rule ($\mathrm{K}_m\mathrm{I}_n$). The left-hand side translates to $[\![\mathrm{K}_m(\mathrm{I}_{n-1})]\!]_q \equiv \mathrm{K}_m(q_m(\mathrm{I}_{n-1})) \equiv \mathrm{K}_m(\mathrm{I}_n)$ because $m \leq n-1$. And the right-hand side yields $[\![\mathrm{I}_n]\!]_q \equiv \mathrm{I}_n$.

Rule ($\mathrm{K}_m\mathrm{S}_{n+1}$). The left-hand side translates to:

$$[\![\mathrm{K}_m(\mathrm{S}_n(M,P))]\!]_q \equiv \mathrm{K}_m(q_m(\mathrm{S}_n([\![M]\!]_q, [\![P]\!]_q))) \equiv \mathrm{K}_m(\mathrm{S}_{n+1}(q_m([\![M]\!]_q), q_m([\![P]\!]_q)))$$

(because $m \leq n-1$). And the right-hand side translates to $[\![\mathrm{S}_{n+1}(\mathrm{K}_m(M), \mathrm{K}_m(P))]\!]_q \equiv \mathrm{S}_{n+1}(\mathrm{K}_m(q_m([\![M]\!]_q)), \mathrm{K}_m(q_m([\![P]\!]_q)))$.

Rule ($\mathrm{K}_m\mathrm{K}_n$). The left-hand side translates to:

$$\begin{array}{rcl}[\![\mathrm{K}_m(\mathrm{K}_{n-1}(M))]\!]_q & \equiv & \mathrm{K}_m(q_m(\mathrm{K}_{n-1}(q_{n-1}([\![M]\!]_q)))) \\ & \equiv & \mathrm{K}_m(\mathrm{K}_n((q_m \circ q_{n-1})([\![M]\!]_q))) \equiv \mathrm{K}_m(\mathrm{K}_n((q_n \circ q_m)([\![M]\!]_q)))\end{array}$$

by Lemma C.2. And the right-hand side yields:

$$[\![\mathrm{K}_n(\mathrm{K}_m(M))]\!]_q \equiv \mathrm{K}_n(q_n(\mathrm{K}_m(q_m([\![M]\!]_q)))) \equiv \mathrm{K}_n(\mathrm{K}_m((q_n \circ q_m)([\![M]\!]_q)))$$

□

Lemma C.4. We let the set of q-functions be the smallest set containing the identity function on SKInT-terms and stable by composition with any q_i, $i \geq 0$.

For every context $\mathcal{C}$, there is a context $[\![\mathcal{C}]\!]_q$ and a q-function $q_{\mathcal{C}}$ such that, for every term t, $[\![\mathcal{C}[t]]\!]_q \equiv [\![\mathcal{C}]\!]_q[q_{\mathcal{C}}(t)]$.

Proof. First observe that (*) for any context $\mathcal{C}$, for any q-function q, there is a context $\mathcal{C}'$ and a q-function q' such that $q(\mathcal{C}[t]) \equiv \mathcal{C}'[q'(t)]$ for any term t. Indeed, this is clear if q is the identity; when q is q_i for some i, then this is an easy structural induction on $\mathcal{C}$, using Definition C.1; and otherwise, this is an easy induction on the length n of a given presentation of q as composition of n q_i functions.

Then we prove the lemma by structural induction on $\mathcal{C}$. If $\mathcal{C} \equiv []$, we take $[\![\mathcal{C}]\!]_q =_{\mathrm{df}} []$ and $q_{\mathcal{C}}$ equal to the identity function. If $\mathcal{C} \equiv \mathrm{S}_m(M_1, \mathcal{C}_2)$ or $\mathcal{C} \equiv \mathrm{S}_m(\mathcal{C}_2, M_1)$, then $[\![\mathcal{C}]\!]_q =_{\mathrm{df}} \mathrm{S}_m(M_1, [\![\mathcal{C}_2]\!]_q)$ or $\mathrm{S}_m([\![\mathcal{C}_2]\!]_q, M_1)$ respectively, and $q_{\mathcal{C}} =_{\mathrm{df}} q_{\mathcal{C}_2}$.

If $\mathcal{C} \equiv \mathrm{K}_m(\mathcal{C}_1)$, then let $\mathcal{C}'$ be $[\![\mathcal{C}_1]\!]_q$, q' be $q_{\mathcal{C}_1}$ (using the induction hypothesis), so that $[\![\mathcal{C}[t]]\!]_q \equiv \mathrm{K}_m(q_m([\![\mathcal{C}_1[t]]\!]_q)) \equiv \mathrm{K}_m(q_m(\mathcal{C}'[q'(t)]))$. By remark (*), there is a context $\mathcal{C}''$ and a q-function q'' such that $q_m(\mathcal{C}'[q'(t)]) \equiv \mathcal{C}_2'[q''(t)]$, and we let $[\![\mathcal{C}]\!]_q =_{\mathrm{df}} \mathrm{K}_m(\mathcal{C}_2')$ and $q_{\mathcal{C}} =_{\mathrm{df}} q''$. □

Lemma C.5. For any rule $M \triangleright N$ in ΣT_0, for any context $\mathcal{C}$, $[\![\mathcal{C}[M]]\!]_q$ rewrites in one step to $[\![\mathcal{C}[N]]\!]_q$ by the rules of group $\Sigma T_0'$ respectively (see Figure 10).

Proof. First, if $M \triangleright N$ is any rule R in Figure 10, then $q_i(M) \triangleright q_i(N)$ is also an instance of some rule in Figure 10, which we shall call $q_i(R)$. For instance, $q_i([\![\mathrm{S}_m\mathrm{I}_n]\!]_q)$ splits into three cases: $[\![\mathrm{S}_{m+1}\mathrm{I}_{n+1}]\!]_q$ when $i \leq m-1$; $[\![\mathrm{S}_m\mathrm{I}_{n+1}]\!]_q$ when $m-1 < i \leq n-1$; and $[\![\mathrm{S}_m\mathrm{I}_n]\!]_q$ when $n-1 < i$.

It follows that for any q-function q, for any rule $M \triangleright N$ in Figure 10, $q(M) \triangleright q(N)$ is also an instance of some rule in Figure 10: this is by induction on the number of q_i's we compose to get q.

Let now $\mathcal{C}$ be a context. By Lemma C.4, $[\![\mathcal{C}[M]]\!]_q \equiv [\![\mathcal{C}]\!]_q[q_{\mathcal{C}}(M)]$ and $[\![\mathcal{C}[N]]\!]_q \equiv [\![\mathcal{C}]\!]_q[q_{\mathcal{C}}(N)]$. By the above, $q_{\mathcal{C}}(M) \triangleright q_{\mathcal{C}}(N)$ is an instance of some rule in Figure 10. Therefore $[\![\mathcal{C}[M]]\!]_q$ rewrites to $[\![\mathcal{C}[N]]\!]_q$ in one step. □

Lemma C.6. The set of function symbols occurring in any sequence of rewrite steps in the system $\Sigma T'_0$ of Figure 10 is finite.

Proof. For every term M, let $F_0(M)$ be the set of function symbols f_n (S_n, K_n or I_n) occurring in M. Let $F(M)$ be the set of all symbols f_m, where $f_n \in F_0(M)$ for some $n \geq m$. For any M, $F(M)$ is clearly finite. Check that, if M rewrites in one step to N, then $F(N) \subseteq F(M)$. In any derivation $M_0 \triangleright M_1 \triangleright \ldots \triangleright M_k \triangleright \ldots$, the set of function symbols that occur is therefore included in $\bigcup_{i \geq 0} F(M_i) = F(M_0)$, which is finite. □

We shall now use the *recursive path ordering* (Dershowitz, 1987), which we now define. Recall that a *quasi-ordering* $\succeq$ is a reflexive and transitive relation, that its associated equivalence relation $\approx$ is defined by $u \approx v$ if and only if $u \succeq v$ and $v \succeq u$, and that its strict part $\succ$ is defined by $u \succ v$ if $u \succeq v$ and $v \not\succeq u$. Consider now a set of first-order terms with a precedence (i.e., an ordering) on function symbols $\succeq$. Then it induces a recursive path ordering on terms $\succeq_{rpo}$, together with associated relations $\succ_{rpo}$ and $\approx_{rpo}$ as follows. Given two first-order terms $s =_{\mathrm{df}} f(s_1, \ldots, s_m)$ and $t =_{\mathrm{df}} g(t_1, \ldots, t_n)$, we have $s \succ_{rpo} t$ if and only if:

1 $s_i \succeq_{rpo} t$ for some i, $1 \leq i \leq m$;
2 or $f \succ g$ and $s \succ_{rpo} t_j$ for all j, $1 \leq j \leq n$,
3 or $f \approx g$ and $\{\!| s_1, \ldots, s_m |\!\} \succ_{rpo}^{mul} \{\!| t_1, \ldots, t_n |\!\}$.

Then, a rewrite system $\mathcal{R}$ over a set of first-order terms is terminating if and only if there exists a well-founded quasi-ordering $\succeq$ on the set of function symbols such that $t \succ_{rpo} u$ for every rule $t \to u$ in $\mathcal{R}$ (Dershowitz, 1987).

It follows:

Lemma C.7. Let $\succ$ be the precedence defined by $f_i \succ g_j$ if and only if $i < j$, for any $f, g \in \{\mathrm{S}, \mathrm{K}, \mathrm{I}\}$. Let $>_q$ denote the recursive path ordering $\succ_{rpo}$.

For any rule $M \triangleright N$ among $[\![(\mathrm{K}_m\mathrm{I}_n)]\!]_q$, $[\![(\mathrm{K}_m\mathrm{K}_n)]\!]_q$, and $[\![(\mathrm{K}_m\mathrm{S}_{n+1})]\!]_q$, $M >_q N$. In particular, any sequence of rewrite steps using only $(\mathrm{K}_m\mathrm{I}_n)$, $(\mathrm{K}_m\mathrm{K}_n)$, and $(\mathrm{K}_m\mathrm{S}_{n+1})$ terminates.

Proof. Notice that by our definition, $\mathrm{K}_0 > \mathrm{K}_1 > \mathrm{K}_2 > \ldots$, that is, the ordering on function symbols is the opposite of the natural ordering on integers. For any of the mentioned rules R, $[\![R]\!]_q$ is clearly decreasing for $>_q$. Fix some sequence $M_0 \triangleright M_1 \triangleright \ldots \triangleright M_k \triangleright \ldots$ using only these rules. In the corresponding sequence $[\![M_0]\!]_q \triangleright [\![M_1]\!]_q \triangleright \ldots \triangleright [\![M_k]\!]_q \triangleright \ldots$ (which is a valid rewrite sequence by Lemma C.5), the set of function symbols F is finite by Lemma C.6. On this finite set, $\succ$ is trivially well-founded. It follows that $>_q$ is well-founded on F, and that both reductions terminate. □

C.2. *The* S *Group*

We now deal with the next three rule schemes, which form the S *group*:

$$\begin{array}{lrcl}
(\mathrm{S}_m\mathrm{I}_n) & \mathrm{S}_m(\mathrm{I}_n, P) & \triangleright & \mathrm{I}_{n-1} \\
(\mathrm{S}_m\mathrm{K}_n) & \mathrm{S}_m(\mathrm{K}_n(M), P) & \triangleright & \mathrm{K}_{n-1}(\mathrm{S}_m(M, P)) \\
(\mathrm{S}_m\mathrm{S}_n) & \mathrm{S}_m(\mathrm{S}_n(M, N), P) & \triangleright & \mathrm{S}_{n-1}(\mathrm{S}_m(M, P), \mathrm{S}_m(N, P))
\end{array}$$

To ease our task, we first define a few auxiliary notations:

Definition C.8. An infinite sequence s over some alphabet A is any total function from the set of non-negative integers to A.

We write s_i the letter at position i in s, which is $s(i)$ by definition.

We denote by $s_{i..j}$ the finite sequence of all letters s_i, s_{i+1}, ... , s_j; if $i > j$, we take by convention $s_{i..j}$ to be the empty word ϵ. We denote by $s_{i..\infty}$ the infinite sequence of all letters s_i, s_{i+1}, ...

For any letter x, let x^ω be the infinite sequence consisting only of x. If s is a finite sequence and s' is an finite or infinite sequence, let $s \,.\, s'$ be the concatenation of s and s'. Concatenation is associative and has ϵ as unit element.

The idea is to use a modified form of the $M \mapsto [\![M]\!](s_0, \ldots, s_n)$ interpretation, mapping SKInT-terms to some well-founded domain instead of the set of λ-terms; this domain will be the set $\mathbb{N}$ of non-negative integers. Moreover, there will be two major modifications. First, instead of parameterizing the interpretation of M by an n-tuple of integers $s_0, \ldots,$ s_n, we shall use an infinite sequence of integers, ending in 0; this is equivalent, but will be easier to handle. Secondly, a few tweaks will be introduced in the form of auxiliary functions E_m, K_m, I_m.

Definition C.9. Let Γ be the set of all infinite sequences γ of non-negative integers containing only finitely many non-zero integers. Every such sequence can be written as the concatenation of some finite sequence $\gamma_{0..k}$ and of 0^ω.

For every $m \geq 0$, let E_m, K_m and I_m be functions from $\mathbb{N}$ to $\mathbb{N}$. Let finally δ be some fixed element of Γ.

We define the function $[\![\_]\!]_e\_$ from SKInT-terms and elements of Γ to non-negative integers as follows. To save a few parentheses, we write $[\![M]\!]_e s \,.\, \gamma$ instead of $[\![M]\!]_e(s \,.\, \gamma)$; $E_m[\![M]\!]_e\gamma$ instead of $E_m([\![M]\!]_e\gamma)$, and similarly with K_m; and parentheses are used to promote an integer n to a sequence (n) containing exactly the integer n.

$$\begin{array}{lcl}
[\![\mathrm{S}_m(M, N)]\!]_e\gamma & =_{\mathrm{df}} & [\![M]\!]_e\gamma_{0..m-1} \,.\, (E_m[\![N]\!]_e\gamma_{0..m-1} \,.\, \delta) \,.\, \gamma_{m..\infty} \\
[\![\mathrm{K}_m(M)]\!]_e\gamma & =_{\mathrm{df}} & [\![M]\!]_e\gamma_{0..m-1} \,.\, (K_m\gamma_m) \,.\, \gamma_{m+1..\infty} \\
[\![\mathrm{I}_m]\!]_e\gamma & =_{\mathrm{df}} & \sum_{i\geq 0, i\neq m} \gamma_i + I_m(\gamma_m)
\end{array}$$

Finally, we define $[\![M]\!]_{eq}\gamma$ as being $[\![[\![M]\!]_q]\!]_e\gamma$.

Say that a function f from $\mathbb{N}$ to $\mathbb{N}$ is *super-linear* if and only if $f(n) > n$ for every integer n.

Define the ordering $\geq$ on sequences pointwise, i.e. $\gamma \geq \gamma'$ if and only if $\gamma_i \geq \gamma'_i$ for every $i \geq 0$. Let $\gamma > \gamma'$ denote $\gamma \geq \gamma'$ and $\gamma \neq \gamma'$.

We say that a function f is *monotonic* if and only if $a > b$ implies $f(a) > f(b)$, where $>$ is defined on naturals or sequences appropriately.

We extend the ordering $>$ to functions pointwise, i.e. $f > g$ if and only if $f(a) > g(a)$ for every a in the common domain of f and g. Then, any family $(f_m)_{m\geq 0}$ of functions is said to be *increasing* if and only if, for all $0 \leq i < j$, $f_i < f_j$.

We shall assume the following properties in the sequel:

— (P1). For every $m \geq 0$, K_m, I_m are super-linear.
— (P2). For every $m \geq 0$, E_m, K_m, I_m are monotonic.
— (P3). $(E_m)_{m\geq 0}$, $(K_m)_{m\geq 0}$, $(I_m)_{m\geq 0}$ are increasing families of functions.

These properties are easy to satisfy. Take for instance $E_m(x) = K_m(x) = I_m(x) = x + m + 1$.

Lemma C.10. For every term M, for every $i \in \mathbb{N}$, and for every γ in Γ, $[\![M]\!]_e \gamma > \gamma_i$.

Proof. By structural induction on M, using only property (P1) (super-linearity).

If $M \equiv \mathrm{S}_m(N, P)$, then $[\![M]\!]_e \gamma = [\![N]\!]_e \gamma_{0..m-1} \cdot (E_m [\![P]\!]_e \gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..\infty}$. For every $i < m$, the claim follows by the induction hypothesis, applied to N and index i. For every $i \geq m$, it follows by the induction hypothesis applied to N and index $i + 1$.

If $M \equiv \mathrm{K}_m(N)$, then $[\![M]\!]_e \gamma = [\![N]\!]_e \gamma_{0..m-1} \cdot (K_m \gamma_m) \cdot \gamma_{m+1..\infty}$. For every $i \neq m$, the claim follows by induction hypothesis applied to N and index i. When $i = m$, $[\![M]\!]_e \gamma > K_m(\gamma_m)$ (by induction hypothesis) $> \gamma_m$ (by super-linearity of K_m).

If $M \equiv \mathrm{I}_m$, then $[\![M]\!]_e \gamma = \sum_{j\geq 0, j\neq m} \gamma_j + I_m(\gamma_m)$. For every $i \neq m$, notice that since I_m is super-linear, $I_m(\gamma_m) > \gamma_m \geq 0$, so $[\![M]\!]_e \gamma > \sum_{j\geq 0, j\neq m} \gamma_j \geq \gamma_i$. And when $i = m$, then $[\![M]\!]_e \gamma \geq I_m(\gamma_m) > \gamma_m$ by super-linearity of I_m. ☐

Lemma C.11. For every term M, for every γ in Γ, and for every i in $\mathbb{N}$, the function $k \mapsto [\![M]\!]_e \gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..\infty}$ is monotonic.

Proof. By structural induction on M, using property (P2). This is clear if M is I_m. In the sequel, we assume $k > n$.

If $M \equiv \mathrm{S}_m(N, P)$, then there are two cases, depending on whether $i \geq m$ or $i < m$. We consider the latter case.

$$\begin{aligned}
&[\![M]\!]_e \gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..\infty} \\
&= [\![N]\!]_e \gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot (E_m [\![P]\!]_e \gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&> [\![N]\!]_e \gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot (E_m [\![P]\!]_e \gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{by induction hypothesis on } N \\
&> [\![N]\!]_e \gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot (E_m [\![P]\!]_e \gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&\text{by induction hypothesis on } P \text{, monotonicity of } E_m \text{ and induction hypothesis on } N \\
&= [\![M]\!]_e \gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..\infty}
\end{aligned}$$

☐

Lemma C.12. For any γ in Γ, for any rule $L \triangleright R$ in Figure 10, $[\![L]\!]_e \gamma \geq [\![R]\!]_e \gamma$. Moreover, the inequality is strict except for rules among $(\mathrm{K}_m \mathrm{I}_n)$, $(\mathrm{K}_m \mathrm{K}_n)$, and $(\mathrm{K}_m \mathrm{S}_{n+1})$.

Proof. By case analysis on the rule, where we assume $0 \leq m < n$. For the schemes with S_m, be aware that the interpretation shifts the indices of the associated sequences. We consider a few cases.

— $[\![\mathrm{K}_m \mathrm{K}_n]\!]_q$:

$$\begin{aligned}
&[\![L]\!]_e\gamma = [\![\mathrm{K}_m(\mathrm{K}_n(M))]\!]_e\gamma \\
&= [\![\mathrm{K}_n(M)]\!]_e\gamma_{0..m-1} \cdot (K_m\gamma_m) \cdot \gamma_{m+1..\infty} \\
&= [\![M]\!]_e\gamma_{0..m-1} \cdot (K_m\gamma_m) \cdot \gamma_{m+1..n-1} \cdot (K_n\gamma_n) \cdot \gamma_{n+1..\infty} \\
&= [\![\mathrm{K}_m(M)]\!]_e\gamma_{0..n-1} \cdot (K_n\gamma_n) \cdot \gamma_{n+1..\infty} \\
&= [\![\mathrm{K}_n(\mathrm{K}_m(M))]\!]_e\gamma = [\![R]\!]_e\gamma
\end{aligned}$$

— $[\![\mathrm{S}_m \mathrm{I}_n]\!]_q$:

$$\begin{aligned}
&[\![L]\!]_e\gamma = [\![\mathrm{S}_m(\mathrm{I}_n, P)]\!]_e\gamma \\
&= [\![\mathrm{I}_n]\!]_e\gamma_{0..m-1} \cdot (E_m[\![P]\!]_e\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&= \textstyle\sum_{i\neq n-1} \gamma_i + E_m([\![P]\!]_e\gamma_{0..m-1} \cdot \delta) + I_n(\gamma_{n-1}) \\
&> \textstyle\sum_{i\neq n-1} \gamma_i + E_m([\![P]\!]_e\gamma_{0..m-1} \cdot \delta) + I_{n-1}(\gamma_{n-1}) \\
&\qquad \text{since } (I_i)_{i\geq 0} \text{ is increasing} \\
&\geq \textstyle\sum_{i\neq n-1} \gamma_i + I_{n-1}(\gamma_{n-1}) \\
&\qquad \text{since } E_m \text{ is non-negative} \\
&= [\![\mathrm{I}_{n-1}]\!]_e\gamma = [\![R]\!]_e\gamma
\end{aligned}$$

□

Lemma C.13. If $[\![M]\!]_e\gamma > [\![N]\!]_e\gamma$ (resp. $\geq$) for every γ in Γ, then for every context $\mathcal{C}$, for every γ in Γ, $[\![\mathcal{C}[M]]\!]_e\gamma > [\![\mathcal{C}[N]]\!]_e\gamma$ (resp. $\geq$).

Proof. We only treat the case of $>$, since the case of $\geq$ follows easily. The proof is by structural induction on $\mathcal{C}$. If $\mathcal{C} \equiv []$, this is clear. Otherwise, we have several cases.

If $\mathcal{C} \equiv \mathrm{S}_j(\mathcal{C}_1, P)$, then we have $[\![\mathcal{C}[M]]\!]_e\gamma = [\![\mathcal{C}_1[M]]\!]_e\gamma_{0..j-1} \cdot (E_j[\![P]\!]_e\gamma_{0..j-1} \cdot \delta) \cdot \gamma_{j..\infty} > [\![\mathcal{C}_1[N]]\!]_e\gamma_{0..j-1} \cdot (E_j[\![P]\!]_e\gamma_{0..j-1} \cdot \delta) \cdot \gamma_{j..\infty}$ (by induction hypothesis) $= [\![\mathcal{C}[N]]\!]_e\gamma$.

If $\mathcal{C} \equiv \mathrm{S}_j(P, \mathcal{C}_1)$, then $[\![\mathcal{C}[M]]\!]_e\gamma = [\![P]\!]_e\gamma_{0..j-1} \cdot (E_j[\![\mathcal{C}_1[M]]\!]_e\gamma_{0..j-1} \cdot \delta) \cdot \gamma_{j..\infty}$ and the claim follows by the induction hypothesis, monotonicity of E_j and Lemma C.11.

If $\mathcal{C} \equiv \mathrm{K}_j(\mathcal{C}_1)$, then the claim follows directly from the induction hypothesis. □

Lemma C.14. Let $>_e$ (resp. $\geq_e$) be defined by $M >_e N$ (resp. $\geq_e$) if and only if for every γ in Γ, $[\![M]\!]_e\gamma > [\![N]\!]_e\gamma$ (resp. $\geq$).

Let $>_{eq}$ be $(>_e, >_q)^{lex}$, i.e. the ordering defined by $M >_{eq} N$ if and only if $M >_e N$, or $M \geq_e N$ and $M >_q N$.

Then, whenever M rewrites to N by some rule in $\Sigma T'_0$, then $M >_{eq} N$.

Let $\succ_{eq}$ be defined by $M \succ_{eq} N$ if and only $[\![M]\!]_q >_{eq} [\![N]\!]_q$. If M rewrites to N by some rule in ΣT_0, then $M \succ_{eq} N$.

Proof. If M rewrites to N by some rule in $\Sigma T'_0$, then there exists a context $\mathcal{C}$ and a rule $L \triangleright R$ such that $M \equiv \mathcal{C}[L]$ and $N \equiv \mathcal{C}[R]$.

If this rule is among $(\mathrm{K}_m\mathrm{I}_n)$, $(\mathrm{K}_m\mathrm{K}_n)$, or $(\mathrm{K}_m\mathrm{S}_{n+1})$, then by Lemma C.12, $L \geq_e R$. By Lemma C.13, $M \geq_e N$. By Lemma C.7, $M >_q N$. So $M >_{eq} N$.

If the rule is among ($\mathrm{S}_m\mathrm{I}_n$), ($\mathrm{S}_m\mathrm{K}_n$), or ($\mathrm{S}_m\mathrm{S}_n$), then by Lemma C.12, $L >_e R$. By Lemma C.13, $M >_e N$, so $M >_{eq} N$.

Now by Lemmas C.3 and C.5, if M rewrites to N by some rule in ΣT_0, then $[\![M]\!]_q$ rewrites to $[\![N]\!]_q$ by some rule in $\Sigma T'_0$. So $[\![M]\!]_q >_{eq} [\![N]\!]_q$, and hence $M \succ_{eq} N$. $\square$

The $\succ_{eq}$ ordering is clearly well-founded for derivations (i.e., the intersection of $\succ_{eq}$ and the reduction pre-ordering $\triangleright^*$ is well-founded, see (Dershowitz, 1987)), so ΣT_0 terminates.

C.3. *The* (SK_m) *Rules*

We now apply another interpretation, which is very close to the $[\![\_]\!]_e$ interpretation, directly on SKInT-terms.

Definition C.15. Define the following interpretation mapping SKInT-terms and sequences in Γ to integers as follows:

$$\begin{array}{lll} [\![\mathrm{S}_m(M,N)]\!]'\gamma & =_{\mathrm{df}} & [\![M]\!]'\gamma_{0..m-1} \cdot (E[\![N]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\ [\![\mathrm{K}_m(M)]\!]'\gamma & =_{\mathrm{df}} & [\![M]\!]'\gamma_{0..m-1} \cdot (\gamma_m \oplus \gamma_{m+1}) \cdot \gamma_{m+2..\infty} \\ [\![\mathrm{I}_m]\!]'\gamma & =_{\mathrm{df}} & I_m(\sum \gamma) \end{array}$$

where δ is a fixed element of Γ, e.g. 0^ω, E is a unary function from $\mathbb{N}$ to $\mathbb{N}$, $\oplus$ is a binary function from $\mathbb{N} \times \mathbb{N}$ to $\mathbb{N}$, $(I_m)_{m\in\mathbb{N}}$ is a family of functions from $\mathbb{N}$ to $\mathbb{N}$ and $\sum$ is a function mapping elements of Γ to $\mathbb{N}$.

In the sequel, we shall assume that E, $\oplus$, I_m and $\sum$ obey the following axioms:

— (Q1). $\sum \gamma \geq \gamma_i$ for every $i \geq 0$.

— (Q2). $\oplus$ is super-linear in both its arguments, namely $m \oplus n > m$ and $m \oplus n > n$ for all $m, n \in \mathbb{N}$.

— (Q3). I_m is super-linear for every $m \geq 0$.

— (Q4). $\sum$ is monotonic, in that for every $k' > k$, for every $i \in \mathbb{N}$, $\sum(\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..\infty}) > \sum(\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..\infty})$.

— (Q5). $\oplus$ is monotonic in both arguments, namely $k' > k$ implies $k' \oplus n > k \oplus n$ and $m \oplus k' > m \oplus k$.

— (Q6). E is monotonic.

— (Q7). I_m is monotonic for every $m \geq 0$.

— (Q8). If $\gamma_i > 0$, then $I_m(\sum \gamma) \geq I_{m+1}(\sum \gamma_{0..i-1} \cdot \gamma_{i+1..\infty})$.

— (Q9). $I_{n-1}(\sum \gamma_{0..m-1} \cdot (\gamma_m \oplus \gamma_{m+1}) \cdot \gamma_{m+2..\infty}) \geq I_n(\sum \gamma)$ for every $0 \leq m < n$.

— (Q10). $(m \oplus n) \oplus p \geq m \oplus (n \oplus p)$ for all $m, n, p \in \mathbb{N}$.

For instance, we may take:

$$\begin{array}{ll} E(n) & = n \\ m \oplus n & = m + n + 1 \\ I_m(n) & = n + m + 1 \\ \sum \gamma & = \sum_{i \geq 0} \gamma_i \end{array}$$

Lemma C.16. For every term M, for every $i \in \mathbb{N}$, for every γ in Γ, $[\![M]\!]'\gamma > \gamma_i$.

Proof. By structural induction on M, as for Lemma C.10. If $M \equiv \mathrm{I}_m$, this is by (Q1) and (Q3). If M is of the form $\mathrm{S}_m(N, P)$, then this follows directly from the induction hypothesis. If M is of the form $\mathrm{K}_m N$, then $[\![M]\!]'\gamma > \gamma_i$ for every i except m and $m+1$, by induction hypothesis; also $[\![M]\!]'\gamma > \gamma_m \oplus \gamma_{m+1}$, which is $> \gamma_m$ and $> \gamma_{m+1}$ by (Q2) respectively. $\square$

Lemma C.17. For every term M, for every γ in Γ, for every i in $\mathbb{N}$, the function $k \mapsto [\![M]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..\infty}$ is monotonic.

Proof. By structural induction on M, as for Lemma C.11. When $M \equiv \mathrm{I}_m$, we use axioms (Q4) and (Q7); when $M \equiv \mathrm{K}_m(N)$, we apply the induction hypothesis, and axiom (Q5) if $i = m$ or $i = m+1$. When M is of the form $\mathrm{S}_m(N, P)$, we have two cases. If $i \geq m$, then the result follows by induction hypothesis. Otherwise, $i < m$ and whenever $k' > k$:

$$[\![P]\!]'\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot \delta > [\![P]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot \delta$$

by induction hypothesis on P, so:

$$E[\![P]\!]'\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot \delta > E[\![P]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot \delta$$

by axiom (Q6), so:

$$\begin{aligned}
&[\![M]\!]'\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..\infty} \\
&= [\![N]\!]'\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot (E[\![P]\!]'\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&> [\![N]\!]'\gamma_{0..i-1} \cdot (k') \cdot \gamma_{i+1..m-1} \cdot (E[\![P]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&\qquad\qquad \text{by the above and the induction hypothesis on } N \\
&> [\![N]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot (E[\![P]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&\qquad\qquad \text{by induction hypothesis on } N \\
&= [\![M]\!]'\gamma_{0..i-1} \cdot (k) \cdot \gamma_{i+1..\infty}
\end{aligned}$$

$\square$

Lemma C.18. For any γ in Γ, and any rule $L \triangleright R$ in ΣT, $[\![L]\!]'\gamma \geq [\![R]\!]'\gamma$. Moreover, the inequality is strict for rules (SK_m), $m \geq 0$.

Proof. Rule (SK_m) uses (Q2) and monotonicity, Rule $(\mathrm{K}_m\mathrm{I}_n)$ uses (Q9), and Rule $(\mathrm{K}_m\mathrm{K}_n)$ uses (Q10) and monotonicity if $n \leq m+1$. We consider two other cases:

Rule $(\mathrm{S}_m\mathrm{I}_n)$:

$$\begin{aligned}
&[\![\mathrm{S}_m(\mathrm{I}_n, P)]\!]'\gamma \\
&= [\![\mathrm{I}_n]\!]'\gamma_{0..m-1} \cdot (E[\![P]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..\infty} \\
&= I_n(\textstyle\sum \gamma_{0..m-1} \cdot (E[\![P]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..\infty}) \\
&\geq I_{n+1}(\textstyle\sum \gamma) && \text{by (Q8)} \\
&= [\![\mathrm{I}_{n+1}]\!]'\gamma
\end{aligned}$$

Indeed, by Lemma C.16, $[\![P]\!]'\gamma_{0..m-1} \cdot \delta > 0$, hence by (Q6) $E[\![P]\!]'\gamma_{0..m-1} \cdot \delta > 0$, so that (Q8) is indeed applicable.

Rule ($\mathrm{S}_m\mathrm{S}_n$):

$$
\begin{aligned}
&[\![\mathrm{S}_m(\mathrm{S}_n(M,N),P)]\!]'\gamma\\
&= [\![\mathrm{S}_n(M,N)]\!]'\gamma_{0..m-1} \cdot (E[\![P]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..\infty}\\
&= [\![M]\!]'\gamma_{0..m-1} \cdot (E[\![P]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..n-2}\\
&\quad \cdot (E[\![N]\!]'\gamma_{0..m-1} \cdot (E[\![P]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..n-2} \cdot \delta)\\
&\quad \cdot \gamma_{n-1..\infty}\\
&= [\![\mathrm{S}_m(M,P)]\!]'\gamma_{0..n-2}\\
&\quad \cdot (E[\![N]\!]'\gamma_{0..m-1} \cdot (E[\![P]\!]'\gamma_{0..m-1} \cdot \delta) \cdot \gamma_{m..n-2} \cdot \delta)\\
&\quad \cdot \gamma_{n-1..\infty}\\
&= [\![\mathrm{S}_m(M,P)]\!]'\gamma_{0..n-2} \cdot (E[\![\mathrm{S}_m(N,P)]\!]'\gamma_{0..n-2} \cdot \delta) \cdot \gamma_{n-1..\infty}\\
&= [\![\mathrm{S}_{n-1}(\mathrm{S}_m(M,P),\mathrm{S}_m(N,P))]\!]'\gamma
\end{aligned}
$$

□

Lemma C.19. Let $\succ_\oplus$ (resp. $\succeq_\oplus$) be defined by $M \succ_\oplus N$ (resp. $\succeq_\oplus$) if and only if for every γ in Γ, $[\![M]\!]'\gamma > [\![N]\!]'\gamma$ (resp. $\geq$).

Let $\succ_{\oplus eq}$ be $(\succ_\oplus, \succ_{eq})^{lex}$, i.e. the ordering defined by $M \succ_{\oplus eq} N$ if and only if $M \succ_\oplus N$, or $M \succeq_\oplus N$ and $M \succ_{eq} N$.

Then, whenever M rewrites to N by some rule in ΣT, then $M \succ_{\oplus eq} N$.

Proof. Since by Lemma C.17 the $[\![\_]\!]'$ interpretation passes to the context, it follows from Lemma C.18 that $M \succeq_\oplus N$ if M rewrites to N by some rule in ΣT_0 (and then by Lemma C.14 $M \succ_{eq} N$) and that $M \succ_\oplus N$ if M rewrites to N by (SK_m), for some $m \geq 0$. □

Since $\succ_{eq}$ is well-founded for derivations (i.e., the intersection of $\succ_{eq}$ and the reduction pre-ordering is well-founded, see (Dershowitz, 1987)), ΣT terminates.

C.4. *The η Rule*

We shall use the following theorem:

Theorem C.20 ((Dershowitz, 1987)). Let $\triangleright_1$ and $\triangleright_2$ be two rewrite relations. $\triangleright_1$ is said to *quasi-commute* over $\triangleright_2$ if and only if for every pair of rewrite steps of the form:

$$u \triangleright_1 v \triangleright_2 w$$

there is a sequence of rewrite steps from u to w of the form:

$$u \triangleright_2 v' \triangleright_{12}^* w$$

where $\triangleright_{12}^*$ denotes any finite number of $\triangleright_1$ and $\triangleright_2$ steps, i.e., $\triangleright_{12}^*$ is the reflexive-transitive closure of the union $\triangleright_{12}$ of $\triangleright_1$ and $\triangleright_2$.

Then $\triangleright_{12}$ terminates if and only if both $\triangleright_1$ and $\triangleright_2$ terminate.

Lemma C.21. Let η denote the rewrite relation generated by all rules ($\eta\mathrm{S}_m$), $m \geq 0$. Then η quasi-commutes over ΣT. More precisely, let $\triangleright_{(\eta)}$ denote one-step reduction by

η, and $\triangleright_{\Sigma T}$ denote one-step reduction by ΣT, then whenever:

$$M \triangleright_{(\eta)} N \triangleright_{\Sigma T} P$$

then:

$$M \triangleright^{+}_{\Sigma T} N' \triangleright^{*}_{(\eta)} P$$

Proof. Consider a given rewrite:

$$M \triangleright_{(\eta)} N \triangleright_{\Sigma T} P$$

M is of the form $\mathcal{C}[\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \mathrm{I}_m)]$, $N \equiv \mathcal{C}[Q]$ and also $N = \mathcal{C}'[L\sigma]$, $P = \mathcal{C}'[R\sigma]$ for some instance of a rule $L \triangleright R$ in ΣT by some substitution σ.

If the ΣT and η-redexes are side-by-side, namely, neither $\mathcal{C}$ nor $\mathcal{C}'$ is a sub-context of the other, then we have:

$$M \triangleright_{\Sigma T} N' \triangleright_{(\eta)} P$$

for some term N'.

If the η-contractum Q contains the ΣT-redex, namely if $\mathcal{C}$ is included in $\mathcal{C}'$ (or possibly equal to it), then again:

$$M \triangleright_{\Sigma T} N' \triangleright_{(\eta)} P$$

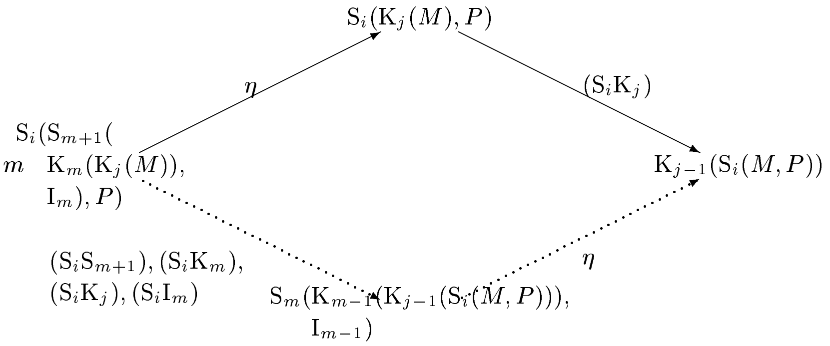
for some term N'. In fact, if $\mathcal{C}' \equiv \mathcal{C}[\mathcal{C}_1]$, then $N' =_{\mathrm{df}} \mathcal{C}[\mathrm{S}_{m+1}(\mathrm{K}_m(\mathcal{C}_1[R\sigma]), \mathrm{I}_m)]$.

If the ΣT-redex is above the η-redex, namely if $\mathcal{C}'$ is included in $\mathcal{C}$, i.e. $\mathcal{C} \equiv \mathcal{C}'[\mathcal{C}_1]$ for some context $\mathcal{C}_1$, then $N \equiv \mathcal{C}[Q] \equiv \mathcal{C}'[\mathcal{C}_1[Q]]$ and on the other hand $N \equiv \mathcal{C}'[L\sigma]$, so that $L\sigma \equiv \mathcal{C}_1[Q]$. We then have two cases. The first case is that the distinguished occurrence of Q is at or below some variable position in L, and then there is another substitution σ' such that $L\sigma' \equiv \mathcal{C}_1[\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \mathrm{I}_m)]$, so that:

$$M \triangleright_{\Sigma T} N' \triangleright^{*}_{(\eta)} P$$

where $N' =_{\mathrm{df}} \mathcal{C}'[R\sigma']$ and P is obtained by contracting the residuals of the occurrence of $\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \mathrm{I}_m)$ in N' by η. The second case is that $L\sigma \equiv \mathcal{C}_1[Q]$ but there is an overlap between the left-hand side L and the term Q. (This is a kind of critical pair, between the inverse relation η^{-1} and ΣT.) We have seven critical pairs to consider, each corresponding to a rule in ΣT. In the following, we assume $0 \leq i < j$. Each critical pair then decomposes into 2 or 3 sub-cases, where we consider one of them here.

case $i < m$



case $i \geq m$

$$\begin{tikzcd}
& \mathrm{S}_i(\mathrm{K}_j(M), P) \arrow[dr, "(\mathrm{S}_i\mathrm{K}_j)"] & \\
\mathrm{S}_i(\mathrm{S}_{m+1}(\mathrm{K}_m(\mathrm{K}_j(M)), \mathrm{I}_m), P) \arrow[ur, "\eta"] \arrow[dr, dotted, "{(\mathrm{K}_m\mathrm{K}_{j+1}), (\mathrm{S}_{m+1}\mathrm{K}_{j+1}), (\mathrm{S}_i\mathrm{K}_j)}"'] & & \mathrm{K}_{j-1}(\mathrm{S}_i(M, P)) \\
& \mathrm{K}_{j-1}(\mathrm{S}_i(\mathrm{S}_{m+1}(\mathrm{K}_m(M), \mathrm{I}_m), P)) \arrow[ur, dotted, "\eta"'] &
\end{tikzcd}$$

□

Notice that one consequence of the latter Lemma is postponement of η after ΣT. Write $\triangleright_{\Sigma T_\eta}$ the notion of reduction induced by ΣT_η, and $\triangleright_{\Sigma T}$ that induced by Σ, then:

Lemma C.22. If $M \triangleright^*_{\Sigma T_\eta} P$, then $M \triangleright^*_{\Sigma T} N \triangleright^*_{(\eta)} P$ for some term N.

Proof. Define a reduction relation $R \rightsquigarrow R'$ on reductions as follows: given a reduction R of the form:

$$M \triangleright^*_{\Sigma T} M_0 \triangleright^*_{(\eta)} M_1 \triangleright_{(\eta)} N_1 \triangleright_{\Sigma T} P_1 \triangleright^*_{\Sigma T_\eta} P$$

by Lemma C.21 there is a term N'_1 such that $M_1 \triangleright^+_{\Sigma T} N'_1 \triangleright^*_{(\eta)} P_1$, we produce the following reduction R':

$$M \triangleright^*_{\Sigma T} M_0 \triangleright^*_{(\eta)} M_1 \triangleright^+_{\Sigma} N'_1 \triangleright^*_{(\eta)} P_1 \triangleright^*_{\Sigma T_\eta} P$$

Since ΣT terminates (Lemma 4.11, proof in C), we may define $\nu(Q)$ for every term Q as the length of the longest ΣT-reduction starting from Q. Given a reduction R as above, let $w(R)$ be the couple (m, n), where m is $\nu(M_0)$ and n is the number of η steps from M_0 to N_1; if R is not a $\rightsquigarrow$-redex, let $w(R)$ be $(0, 0)$. Observe that when $R \rightsquigarrow R'$ as above, $w(R)$ is lexicographically strictly greater than $w(R')$, so $\rightsquigarrow$ terminates. We conclude by observing that $\rightsquigarrow$-normal SKInT$_\eta$-reductions are of the form $M \triangleright^*_{\Sigma T} N \triangleright^*_{(\eta)} P$. □

Proof. **(of Lemma 4.11)** ΣT terminates by Lemma C.19. ΣT_η terminates by Theorem C.20, using the facts that ΣT terminates and that η terminates (trivially, because η makes the size of terms decrease strictly). □

Appendix D. Strong Normalization of Typed SKInT

Lemma D.1. The typed $\lambda\oplus$-calculus has the subject reduction property, that is, whenever $\vdash s : \theta$ and $s \triangleright^* t$, then $\vdash t : \theta$.

Proof. This is standard for rule (β), and obvious for rules $(\oplus-)$, $(\oplus)$, (ϵ) and (ι). □

Lemma D.2. For every typed SKInT-term M, for all $\lambda\oplus$-terms $s_0, \ldots, s_{n-1}$ of the right types, $s_0, \ldots, s_{n-1}$ are proper subterms of $[\![M]\!]_\bullet(s_0, \ldots, s_{n-1})$. Moreover:

$$[\![M]\!]_\bullet(s_0, \ldots, s_{n-1}) \equiv [s_0/x_0, \ldots, s_{n-1}/x_{n-1}][\![M]\!]_\bullet(x_0, \ldots, x_{n-1})$$

for all n distinct variables $x_0, \ldots, x_{n-1}$.

Proof. Easy induction on the definition of $[\![M]\!]_\bullet(s_0, \ldots, s_{n-1})$. The only difficulty lies in checking that $s_0, \ldots, s_{n-1}$ indeed occur as subterms of $[\![M]\!]_\bullet(s_0, \ldots, s_{n-1})$: it is precisely the purpose of terms like $s \oplus t$ to represent t while keeping s around. □

It follows:

Lemma D.3. If $s_i \rhd^* s'_i$ (resp. $s_i \rhd^+ s'_i$) for some i, $0 \leq i \leq n-1$, then:

$$[\![M]\!]_\bullet(s_0, \ldots, s_{i-1}, s_i, s_{i+1}, \ldots, s_{n-1}) \rhd^* [\![M]\!]_\bullet(s_0, \ldots, s_{i-1}, s'_i, s_{i+1}, \ldots, s_{n-1})$$

(resp. $\rhd^+$).

Another monotonicity property is the following. We assume that typed SKInT-terms are given in a fixed typing context Γ. Then all typed SKInT-terms have a unique type.

Lemma D.4. Let $\succ_\lambda$ be defined by $M \succ_\lambda N$ if and only M and N have the same type, and for all $s_0, \ldots, s_{n-1}$ of the right types, $[\![M]\!]_\bullet(s_0, \ldots, s_{n-1}) \rhd^+ [\![N]\!]_\bullet(s_0, \ldots, s_{n-1})$ in the typed $\lambda\oplus$-calculus.

For every context $\mathcal{C}$ such that $\mathcal{C}[M]$ is typable, if $M \succ_\lambda N$, then $\mathcal{C}[M] \succ_\lambda \mathcal{C}[N]$.

Proof. An easy induction on the context $\mathcal{C}$. □

We can now proceed to examine how each rule in SKInT$_\eta$ translates by the $[\![\_]\!]_\bullet$ translation.

We say that a rule $L \to R$ is *decreasing* if and only if, for all $s_0, \ldots, s_{n-1}$ of the right types, $[\![L]\!]_\bullet(s_0, \ldots, s_{n-1}) \rhd^+ [\![R]\!]_\bullet(s_0, \ldots, s_{n-1})$. We say that it is *non-increasing* if $[\![L]\!]_\bullet(s_0, \ldots, s_{n-1}) \rhd^* [\![R]\!]_\bullet(s_0, \ldots, s_{n-1})$ for all $s_0, \ldots, s_{n-1}$ of the right types.

Lemma D.5. The following table summarizes the relationship between left- and right-hand sides of the reductions of SKInT:

(SI_m)	:	$\succ_\lambda$	$-$
(SK_m)	:	$\succ_\lambda$	$-$
$(\mathrm{S}_m\mathrm{I}_p)$	:	$\succ_\lambda$	$\succ_{eq}$
$(\mathrm{S}_m\mathrm{K}_p)$	:	$\succeq_\lambda$	$\succ_{eq}$
$(\mathrm{S}_m\mathrm{S}_p)$	:	$\succeq_\lambda$	$\succ_{eq}$
$(\mathrm{K}_m\mathrm{I}_p)$	:	$\succeq_\lambda$	$\succ_{eq}$
$(\mathrm{K}_m\mathrm{K}_p)$	:	$\succeq_\lambda$	$\succ_{eq}$
$(\mathrm{K}_m\mathrm{S}_p)$	:	$\succeq_\lambda$	$\succ_{eq}$
$(\eta\mathrm{S}_m)$	:	$\succ_\lambda$	$\succ_{eq}$

Proof. Each rule follows by analysis. Rules (SK_m), $(\mathrm{K}_m\mathrm{K}_p)$ and $(\eta\mathrm{S}_m)$ follow by Lemma D.3, Rule $(\mathrm{S}_m\mathrm{I}_p)$ follows because $\oplus$ is right-associative and the sequence $s_m \oplus \ldots \oplus s_{p-1}$ is not empty, and Rule $(\mathrm{K}_m\mathrm{I}_p)$ follows because $\oplus$ is right-associative.

We consider two cases:

— Rule $(\mathrm{S}_m\mathrm{K}_p)$ is non-increasing for every $0 \leq m < p$.

$$\begin{aligned}
&[\![\mathrm{S}_m(\mathrm{K}_p(M), P)]\!]_\bullet(s_0, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{K}_p(M)]\!]_\bullet(s_0, \ldots, s_{m-1}, \\
&\qquad \epsilon(\lambda z_m \ldots \lambda z_{q-1}.[\![P]\!]_\bullet(s_0, \ldots, s_{m-1}, z_m, \ldots, z_{q-1})), \\
&\qquad s_m, \ldots, s_{n-1}) \\
&\equiv [\![M]\!]_\bullet(s_0, \ldots, s_{m-1}, \\
&\qquad \epsilon(\lambda z_m \ldots \lambda z_{q-1}.[\![P]\!]_\bullet(s_0, \ldots, s_{m-1}, z_m, \ldots, z_{q-1})), \\
&\qquad s_m, \ldots, s_{p-2}, s_{p-1} \oplus s_p, s_{p+1}, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{S}_m(M, P)]\!]_\bullet(s_0, \ldots, s_{p-2}, s_{p-1} \oplus s_p, s_{p+1}, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{K}_{p-1}(\mathrm{S}_m(M, P))]\!]_\bullet(s_0, \ldots, s_{n-1})
\end{aligned}$$

— Rule $(\mathrm{K}_m\mathrm{S}_{p+1})$ is non-increasing for every $0 \leq m < p$.

$$\begin{aligned}
&[\![\mathrm{K}_m(\mathrm{S}_p(M, N))P]\!]_\bullet(s_0, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{S}_p(M, N)]\!]_\bullet(s_0, \ldots, s_{m-1}, s_m \oplus s_{m+1}, s_{m+2}, \ldots, s_{n-1}) \\
&\equiv [\![M]\!]_\bullet(s_0, \ldots, s_{m-1}, s_m \oplus s_{m+1}, s_{m+2}, \ldots, s_p, \\
&\qquad \epsilon(\lambda z_p \ldots \lambda z_{q-1}.[\![N]\!]_\bullet(s_0, \ldots, s_{m-1}, s_m \oplus s_{m+1}, s_{m+2}, \ldots, s_p, z_p, \ldots, z_{q-1})), \\
&\qquad s_{p+1}, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{K}_m(M)]\!]_\bullet(s_0, \ldots, s_p, \\
&\qquad \epsilon(\lambda z_p \ldots \lambda z_{q-1}.[\![N]\!]_\bullet(s_0, \ldots, s_{m-1}, s_m \oplus s_{m+1}, s_{m+2}, \ldots, s_p, z_p, \ldots, z_{q-1})), \\
&\qquad s_{p+1}, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{K}_m(M)]\!]_\bullet(s_0, \ldots, s_p, \\
&\qquad \epsilon(\lambda z_p \ldots \lambda z_{q-1}.[\![\mathrm{K}_m(N)]\!]_\bullet(s_0, \ldots, s_p, z_p, \ldots, z_{q-1})), \\
&\qquad s_{p+1}, \ldots, s_{n-1}) \\
&\equiv [\![\mathrm{S}_{p+1}(\mathrm{K}_m(M), \mathrm{K}_m(N))]\!]_\bullet(s_0, \ldots, s_{n-1})
\end{aligned}$$

□

Therefore, the typed SKInT and SKInT$_\eta$-calculi terminate (are strongly normalizing) if $\succ_\lambda$ is well-founded, and in particular if the typed $\lambda\oplus$-calculus is strongly normalizing:

Lemma D.6. The typed $\lambda\oplus$-calculus is strongly normalizing.

Proof. This follows the standard approach of reducibility. Say that a $\lambda\oplus$-term is *neutral* if and only if it is not a λ-abstraction. A term t is A*-reducible* if and only if either A is a base type and t is strongly normalizing, or $A = B \Rightarrow C$ and for every B-reducible term u, $t(u)$ is C-reducible. For every strongly normalizing term t, let $\nu(t)$ be the length of the longest reduction starting from t. Let the size $|t|$ of t be defined as: $|x| =_{\mathrm{df}} 1$, $|s(t)| =_{\mathrm{df}} |s| + |t| + 1$, $|\lambda x_A \cdot t| =_{\mathrm{df}} |\epsilon(t)| =_{\mathrm{df}} |\iota(t)| =_{\mathrm{df}} 1 + |t|$, $|s \oplus t| =_{\mathrm{df}} |s| + |t| + 1$.

We define the usual properties for candidates of reducibility, which are shown to hold by simultaneous structural induction on the type A:

— (CR1). If s is A-reducible, then s is strongly normalizing.
— (CR2). If s is A-reducible and $s \triangleright t$, then t is A-reducible.
— (CR3). If s is neutral, and for every t such that $s \triangleright t$, t is A-reducible, then s is, too.

We now have: (i) every variable x is A-reducible, for every type A. This is by (CR3), since x is neutral and normal.

Then: (ii) if s is $(B \Rightarrow C)$-reducible and t is B-reducible, then $s(t)$ is C-reducible: this is by definition.

(iii): if s is strongly normalizing and t is B-reducible, then $s \oplus t$ is B-reducible. Indeed, we show this by induction on $(\nu(s), |s|, \nu(t))$ ordered lexicographically ($\nu(t)$ is well-defined by (CR1)). Consider any one-step reduct u of $s \oplus t$. If this is by $(\oplus-)$ at the top, $u \equiv t$ is B-reducible. If this is by $(\oplus)$ at the top, then s must be of the form $s_1 \oplus s_2$, and $u \equiv s_1 \oplus (s_2 \oplus t)$; since $\nu(s) \geq \nu(s_2)$ and $|s| > |s_2|$, the induction hypothesis is applicable, so $s_2 \oplus t$ is B-reducible; since $\nu(s) \geq \nu(s_1)$ and $|s| > |s_1|$, the induction hypothesis again applies, so that $s_1 \oplus (s_2 \oplus t)$ is B-reducible. If the reduction took place inside s, then $s \triangleright s'$ and $u \equiv s' \oplus t$: since $\nu(s) > \nu(s')$, the induction hypothesis applies and shows that u is B-reducible. Finally, if the reduction took place in t, then $t \triangleright t'$, $\nu(t) > \nu(t')$ and by induction u is therefore B-reducible again. In any case, every one-step reduct of $s \oplus t$ is B-reducible; since $s \oplus t$ is neutral, by (CR3) it is B-reducible.

(iv): if s is B-reducible, then $\epsilon(s)$ is B-reducible. We show this by induction on $\nu(s)$ (which is well-defined by (CR1)). Consider any one-step reduct u of $\epsilon(s)$: either $u \equiv s$ (by (ϵ)) or $u \equiv \epsilon(s')$ with $s \triangleright s'$; in the first case, u is B-reducible by assumption, in the second case this is by induction. Since $\epsilon(s)$ is neutral, it is B-reducible by (CR3). Similarly, we show that $\iota(s)$ is B-reducible as soon as s is.

(v): if $[t/x]s$ is C-reducible for every B-reducible term t, then $\lambda x_B \cdot s$ is $(B \Rightarrow C)$-reducible. Indeed, let t be any B-reducible term t. We show that $(\lambda x_B \cdot s)(t)$ is C-reducible by induction on $(\nu(s), |s|, \nu(t))$ ordered lexicographically. Indeed, $(\lambda x_B \cdot s)(t)$ may reduce either to $[t/x]s$, which is C-reducible by assumption, or to $(\lambda x_B \cdot s')(t)$ with $s \triangleright s'$, or to $(\lambda x_B \cdot s)(t')$ with $t \triangleright t'$, or to $s'(t)$ provided that $s \equiv s'(x)$ with x not free in s', the last three being C-reducible by induction hypothesis. Since $(\lambda x_B \cdot s)(t)$ is neutral, by (CR3) it is C-reducible. Since t is arbitrary, by definition $\lambda x_B \cdot s$ is $(B \Rightarrow C)$-reducible.

We now show that for any $k \in \mathbb{N}$, for any reducible terms $t_1, \ldots, t_k$, if $[t_1/x_1, \ldots, t_k/x_k]t$ is typable of type A, then it is A-reducible. If $t \in \{x_1, \ldots, x_k\}$, this is obvious. If t is any other variable, this is by (i). If t is an application, this is by (ii). If t is of the form $t_1 \oplus t_2$, with t_1 B-reducible for some type B and t_2 A-reducible, then by (CR1) t_1 is strongly normalizing, and by (iii) t is A-reducible. When t is of the form $\epsilon(t_1)$ or $\iota(t_1)$, this is by (iv). If t is an abstraction $\lambda x_B \cdot s$, with $[u/x, t_1/x_1, \ldots, t_k/x_k]s$ C-reducible for every B-reducible term u by induction, then by (v) $[t_1/x_1, \ldots, t_k/x_k](\lambda x_B \cdot s)$ is $(B \Rightarrow C)$-reducible.

For $k = 0$, it follows that every typed $\lambda\oplus$-term of type A is A-reducible, hence strongly normalizing. □

It follows that the typed SKInT and SKInT$_\eta$-calculi are strongly normalizing. For SKInT$_\eta$, notice that an alternate proof would be to use Theorem C.20 and the following Lemma:

Lemma D.7. η quasi-commutes over SKInT. More precisely, let $\triangleright_{(\eta)}$ denote one-step reduction by η, then whenever: $M \triangleright_{(\eta)} N \triangleright P$ then: $M \triangleright^+ N' \triangleright^*_{(\eta)} P$.

Proof. This is as for Lemma C.21, but with one additional case. □

Appendix E. Standardization

Induction on terms viewed as trees of spines is defined formally as well-founded induction on the strict ordering $\succ$, defined by $S[u_1, \ldots, u_n] \succ u_i$ for any i, $1 \leq i \leq n$.

For convenience, we shall also consider *partial spines* P, which are sequences of operators S_m or K_m, $m \geq 0$. A spine is just the concatenation $P\mathrm{I}_m$ or Px for some partial spine P. In general, we define $P(M)[M_1, \ldots, M_n]$ as follows (letting ϵ denote the empty partial spine):

$$\begin{array}{rcl} \epsilon(M)[] & =_{\mathrm{df}} & M \\ (\mathrm{S}_m P')(M)[M_1, \ldots, M_n] & =_{\mathrm{df}} & \mathrm{S}_m(P'(M)[M_1, \ldots, M_{n-1}], M_n) \\ (\mathrm{K}_m P')(M)[M_1, \ldots, M_n] & =_{\mathrm{df}} & \mathrm{K}_m(P'(M)[M_1, \ldots, M_n]) \end{array}$$

Lemma E.1. If $N \triangleright^{std*} M_1 \triangleright M_2$, then $N \triangleright^{std*} M_2$.

Proof. By induction on M_1, ordered by $\succ$.

If the reduction from M_1 to M_2 is not a spine reduction, then $M_1 \equiv S[M_{11}, \ldots, M_{1n}]$, $M_2 \equiv S[M_{21}, \ldots, M_{2n}]$, where for some i $(1 \leq i \leq n)$, $M_{1i} \triangleright M_{2i}$, and for all other j, $M_{1j} \equiv M_{2j}$. Now by definition the standard reduction from N to M_1 can be decomposed as the spine reduction $N \triangleright^{s*} M_0 \equiv S[M_{01}, \ldots, M_{0n}]$ plus standard reductions $M_{0j} \triangleright^{std*} M_{1j}$, $1 \leq j \leq n$. Since $M_1 \succ M_{1i}$, and $M_{0i} \triangleright^{std*} M_{1i} \triangleright M_{2i}$, by induction hypothesis there is a standard reduction $M_{0i} \triangleright^{std*} M_{2i}$. For all $j \neq i$, we already have standard reductions $M_{0j} \triangleright^{std*} M_{1j} \equiv M_{2j}$. Together with the spine reduction $N \triangleright^{s*} M_0 \equiv S[M_{01}, \ldots, M_{0n}]$, this builds a standard reduction $N \triangleright^{std*} M_2$, as claimed. (This is the only place in the proof where we need to use the induction hypothesis.)

Examine the cases where the reduction from M_1 to M_2 is a spine reduction. Consider the contracted spine redex in M_1, and do a case analysis on the rule used. We decompose the standard reduction from N to M_1 as the spine reduction $N \triangleright^{s*} M_0 \equiv S[M_{01}, \ldots, M_{0n}]$ plus standard reductions $M_{0j} \triangleright^{std*} M_{1j}$, $1 \leq j \leq n$, with $M_1 \equiv S[M_{11}, \ldots, M_{1n}]$. We consider several cases.

— Rule (SI_m): then the spine S is of the form $P\mathrm{S}_m\mathrm{I}_m$, and $n \geq 1$; letting $M_{11} \equiv S''[Q_1, \ldots, Q_p]$, then $M_2 \equiv (PS'')[Q_1, \ldots, Q_p, M_{12}, \ldots, M_{1n}]$, where PS'' denotes the concatenation of the partial spine P and the spine S''. Now because of the form of M_{11}, and because $M_{01} \triangleright^{std*} M_{11}$ by assumption, we can decompose the latter standard reduction as a spine reduction $M_{01} \triangleright^{s*} S''[P_1, \ldots, P_p]$ plus standard reductions $P_k \triangleright^{std*} Q_k$, $1 \leq k \leq p$.

  We then build a spine reduction:

$$\begin{array}{rl} N \triangleright^{s*} M_0 \equiv & (P\mathrm{S}_m\mathrm{I}_m)[M_{01}, \ldots, M_{0n}] \\ \triangleright^{s} & P(M_{01})[M_{02}, \ldots, M_{0n}] \\ \triangleright^{s*} & P(S''[P_1, \ldots, P_p])[M_{02}, \ldots, M_{0n}] \\ \equiv & (PS'')[P_1, \ldots, P_p, M_{02}, \ldots, M_{0n}] \end{array}$$

  by definition. By assumption $P_k \triangleright^{std*} Q_k$ $(1 \leq k \leq p)$ and $M_{0j} \triangleright^{std*} M_{1j}$ $(2 \leq j \leq n)$. This defines a standard reduction $N \triangleright^{std*} (PS'')[Q_1, \ldots, Q_p, M_{12}, \ldots, M_{1n}] \equiv M_2$.

— Rule $(\mathrm{S}_m\mathrm{S}_n)$: then S is of the form $P\mathrm{S}_m\mathrm{S}_nS''$, with $0 \leq m < n$, for some partial spine

P and some spine S'' (of arity, say, n_1), and:

$$M_2 \equiv (P\mathrm{S}_{n-1}\mathrm{S}_m S'')[\begin{array}{l} M_{11}, \ldots, M_{1n_1}, M_{1(n_1+2)}, \\ \mathrm{S}_m(M_{1(n_1+1)}, M_{1(n_1+2)}), \\ M_{1(n_1+3)}, \ldots, M_{1n}] \end{array}$$

Now build a spine reduction:

$$\begin{array}{rl} N \rhd^{s*} & M_0 \equiv (P\mathrm{S}_m\mathrm{S}_n S'')[M_{01}, \ldots, M_{0n}] \\ \rhd^{s} & (P\mathrm{S}_{n-1}\mathrm{S}_m S'')[\begin{array}{l} M_{01}, \ldots, M_{0n_1}, M_{0(n_1+2)}, \\ \mathrm{S}_m(M_{0(n_1+1)}, M_{0(n_1+2)}), \\ M_{0(n_1+3)}, \ldots, M_{0n}] \end{array} \end{array}$$

By assumption, we also have standard reductions from M_{0j} to M_{1j} for $1 \leq j \leq n_1$, for $j = n_1 + 2$, and for $n_1 + 3 \leq j \leq n$. So to get a standard reduction from N to M_2, it just remains to show that $\mathrm{S}_m(M_{0(n_1+1)}, M_{0(n_1+2)}) \rhd^{std*} \mathrm{S}_m(M_{1(n_1+1)}, M_{1(n_1+2)})$. Since $M_{0(n_1+1)} \rhd^{std*} M_{1(n_1+1)}$, we can decompose this standard reduction into a spine reduction $M_{0(n_1+1)} \rhd^{s*} S'[P_1, \ldots, P_p]$, followed by standard reductions $P_k \rhd^{std*} Q_k$, $1 \leq k \leq p$, such that $M_{1(n_1+1)} \equiv S'[Q_1, \ldots, Q_p]$. Therefore $\mathrm{S}_m(M_{0(n_1+1)}, M_{0(n_1+2)})$ $\rhd^{s*}$ $(\mathrm{S}_m S')[P_1, \ldots, P_p, M_{0(n_1+2)}]$; and since $P_k \rhd^{std*} Q_k$ $(1 \leq k \leq p)$ and $M_{0(n_1+2)}$ $\rhd^{std*}$ $M_{1(n_1+2)}$, it follows that $\mathrm{S}_m(M_{0(n_1+1)}, M_{0(n_1+2)}) \rhd^{std*} \mathrm{S}_m(M_{1(n_1+1)}, M_{1(n_1+2)})$. This finishes to prove the claim in this case.

— Rule $(\mathrm{K}_m\mathrm{K}_n)$: then S is of the form $P\mathrm{K}_m\mathrm{K}_{n-1}S''$, with $0 \leq m < n$, P a partial spine, and S'' a spine; and $M_2 \equiv (P\mathrm{K}_n\mathrm{K}_m S'')[M_{11}, \ldots, M_{1n}]$. We then build the spine reduction $N \rhd^{s*} M_0 \equiv (P\mathrm{K}_m\mathrm{K}_{n-1}S'')[M_{01}, \ldots, M_{0n}] \rhd^{s} (P\mathrm{K}_n\mathrm{K}_m S'')[M_{01}, \ldots, M_{0n}]$, followed by the standard reductions $M_{0j} \rhd^{std*} M_{1j}$, $1 \leq j \leq n$, yielding a standard reduction $N \rhd^{std*} (P\mathrm{K}_n\mathrm{K}_m S'')[M_{11}, \ldots, M_{1n}] \equiv M_2$.

□

Appendix F. Weak Normalization of Σ and SKIn

Let $\rhd_I$ denote one-step reduction by the I group.

Lemma F.1. If M is SKT-normal, and $M \rhd_I N$, then N is SKT-normal.

Proof. Consider each rule in turn.

Rule $(\mathrm{S}_m\mathrm{I}_n)$: let M be $\mathcal{C}[\mathrm{S}_m(\mathrm{I}_n, P)]$, and N be its contractum $\mathcal{C}[\mathrm{I}_{n-1}]$. A straightforward structural induction on the context $\mathcal{C}$ shows that N is SKT-normal.

Rule $(\mathrm{K}_m\mathrm{I}_n)$: let M be $\mathcal{C}[\mathrm{K}_m(\mathrm{I}_{n-1})]$, and N be its contractum $\mathcal{C}[\mathrm{I}_n]$. Another straightforward structural induction on $\mathcal{C}$ shows that N is SKT-normal.

Rule $(\mathrm{K}_m\mathrm{S}'_{m+1})$, finally.

Assume $M \equiv \mathcal{C}[\mathrm{K}_m(\mathrm{S}_m(Q, R))]$, $N \equiv \mathcal{C}[\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \downarrow(\mathrm{K}_m(R)))]$, where $\mathcal{C}$ is the context under which contraction occurs. We prove the claim by structural induction on $\mathcal{C}$.

If $\mathcal{C}$ is empty, then we prove that: (1) if $\mathrm{K}_m(\mathrm{S}_m(Q, R))$ is SKT-normal, then the term $\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \downarrow(\mathrm{K}_m(R)))$ is also SKT-normal. Notice that if the latter contains a SKT-redex, then it must be $\mathrm{K}_m(Q)$, so (1) follows from the claim that: (2) $\mathrm{K}_m(Q)$ is not an SKT-redex. Now we reason by cases on Q. If Q is of the form I_j, then $\mathrm{K}_m(Q)$ is not an SKT-redex. If Q is of the form $\mathrm{S}_j(Q_1, Q_2)$, then by assumption $\mathrm{S}_m(Q, R)$ is SKT-normal, so in particular $j \leq m$ (otherwise rule $(\mathrm{S}_m\mathrm{S}_j)$ would apply); then $\mathrm{K}_m(Q) \equiv \mathrm{K}_m(\mathrm{S}_j(Q_1, Q_2))$ is not an SKT-redex. And if Q is of the form $\mathrm{K}_j(Q_1)$, then by assumption $\mathrm{S}_m(Q, R)$ is SKT-normal, so in particular $j < m$ (otherwise rule $(\mathrm{S}_m\mathrm{K}_j)$, if $j > m$, or rule (SK_j) if $j = m$ would apply); then $\mathrm{K}_m(Q) \equiv \mathrm{K}_m(\mathrm{K}_j(Q_1))$ is not an SKT-redex. This proves (2), hence (1).

If $\mathcal{C}$ is of the form $\mathrm{S}_j([], M_2)$, then since $M \equiv \mathrm{S}_j(\mathrm{K}_m(\mathrm{S}_m(Q, R)), M_2)$ is SKT-normal, $j > m$; otherwise, if $j < m$, then rule $(\mathrm{S}_j\mathrm{K}_m)$ would apply, and if $j = m$, then rule (SK_m) would apply. Now $N \equiv \mathrm{S}_j(\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \downarrow(\mathrm{K}_m(R))), M_2)$. By induction hypothesis, $\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \downarrow(\mathrm{K}_m(R)))$ is SKT-normal, so the only possible SKT-redex in N is N itself. However, since $j > m$, it follows that $j \geq m + 1$, hence N is not an SKT-redex. This proves the claim in this case.

If $\mathcal{C}$ is of the form $\mathrm{K}_j([])$, then since $M \equiv \mathrm{K}_j(\mathrm{K}_m(\mathrm{S}_m(Q, R)))$ is SKT-normal, we must have $j > m$; otherwise rule $(\mathrm{K}_j\mathrm{K}_{m+1})$ would apply. Then $N \equiv \mathrm{K}_j(\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \downarrow(\mathrm{K}_m(R))))$. By induction hypothesis, $\mathrm{S}_{m+1}(\mathrm{K}_m(Q), \downarrow(\mathrm{K}_m(R)))$ is SKT-normal, so the only possible SKT-redex in N is N itself. However, $j > m$, so that $j \geq m + 1$, and therefore N cannot be an SKT-redex (although, if $j = m + 1$, it is a $(\mathrm{K}_j\mathrm{S}_{j+1})$-redex).

In all other cases, the only SKT-redex in N must again be N itself, by induction hypothesis. But if N is an SKT-redex, then M would be an SKT-redex as well (by the same rule); this is ruled out by assumption. □

We now give a algorithm to compute $\downarrow(\mathrm{K}_m(N))$, when N is SKT-normal:

Lemma F.2. For each $m \geq 0$, let Q_m be the function defined as follows:

$$\begin{array}{rcl} Q_m(x) & =_{\mathrm{df}} & \mathrm{K}_m(x) \\ Q_m(\mathrm{I}_j) & =_{\mathrm{df}} & \mathrm{K}_m(\mathrm{I}_j) \\ Q_m(\mathrm{K}_j(N_1)) & =_{\mathrm{df}} & \begin{cases} \mathrm{K}_{j+1}(Q_m(N_1)) & (m \leq j) \\ \mathrm{K}_m(\mathrm{K}_j(N_1)) & (m > j) \end{cases} \\ Q_m(\mathrm{S}_j(N_1, N_2)) & =_{\mathrm{df}} & \begin{cases} \mathrm{S}_{j+1}(Q_m(N_1), Q_m(N_2)) & (m < j) \\ \mathrm{K}_m(\mathrm{S}_j(N_1, N_2)) & (m \geq j) \end{cases} \end{array}$$

Whenever N is SKT-normal, $Q_m(N) =_{\mathrm{df}} \downarrow(\mathrm{K}_m(N))$.

Proof. Clearly, $\mathrm{K}_m(N)$ SKT-reduces to $Q_m(N)$. Then, we only have to show that $Q_m(N)$ is SKT-normal; then $Q_m(N)$ and $SKT(\mathrm{K}_m(N))$ will be SKT-normal forms of the same term—namely $\mathrm{K}_m(N)$—and therefore will be equal, by confluence of SKT.

We show that: (1) $Q_m(N)$ is SKT-normal whenever N is, by structural induction on N. Simultaneously, we shall show that: (2) $i(Q_m(N)) \leq \max(m, i(N)) + 1$, where the i function is defined by:

$$\begin{array}{rcl} i(x) & =_{\mathrm{df}} & 0 \\ i(\mathrm{I}_j) & =_{\mathrm{df}} & 0 \\ i(\mathrm{K}_j(N_1)) & =_{\mathrm{df}} & j + 1 \\ i(\mathrm{S}_j(N_1, N_2)) & =_{\mathrm{df}} & j \end{array}$$

(1) and (2) are clear when N is of the form x (a variable) or I_j.

We consider the case when N is of the form $\mathrm{K}_j(N_1)$. We observe that: (3) $j \geq i(N_1)$ (because N is SKT-normal, and in particular not an SKT-redex; we invite the reader to check all cases). We then have two sub-cases:

— If: (4) $m \leq j$, then $Q_m(N) \equiv \mathrm{K}_{j+1}(Q_m(N_1))$, where by induction hypothesis (1), $Q_m(N_1)$ is SKT-normal, so that the only possible redex in $Q_m(N)$ is $Q_m(N)$ itself. However, by induction hypothesis (2), $i(Q_m(N_1)) \leq \max(m, i(N_1))+1 \leq \max(m, j)+1$ (by (3)) $\leq j+1$ (by (4)), so $Q_m(N) \equiv \mathrm{K}_{j+1}(Q_m(N_1))$ cannot be an SKT-redex (we let the reader check that this is implied by the fact that $i(Q_m(N_1)) \leq j+1$), so $Q_m(N)$ is SKT-normal. This shows (1).
  Then, $i(Q_m(N)) = j+2 = i(N)+1$ (by definition of $i(N)$) $\leq \max(m, i(N))+1$, so (2) holds.

— If: (5) $m > j$, then $Q_m(N) \equiv \mathrm{K}_m(\mathrm{K}_j(N_1)) \equiv \mathrm{K}_m(N)$. By assumption, N is SKT-normal, so that the only possible redex in $Q_m(N)$ is $Q_m(N)$ itself. But this is not the case, by (5). So (1) holds.
  Furthermore, $i(Q_m(N)) = m+1 \leq \max(m, i(N))+1$, so (2) holds.

□

Lemma F.3. Define the following measure on terms (basically, its size, not counting K_m nodes):

$$\begin{array}{rcl} \kappa(x) & =_{\mathrm{df}} & 1 \\ \kappa(\mathrm{I}_m) & =_{\mathrm{df}} & 1 \\ \kappa(\mathrm{K}_m(M)) & =_{\mathrm{df}} & \kappa(M) \\ \kappa(\mathrm{S}_m(M,N)) & =_{\mathrm{df}} & \kappa(M)+\kappa(N)+1 \end{array}$$

Then $\kappa(Q_m(N)) = \kappa(N)$.

Proof. Straightforward structural induction on N. □

Lemma F.4. Define the following weight function:

$$\begin{array}{rcl} W(x) & =_{\mathrm{df}} & 1 \\ W(\mathrm{I}_m) & =_{\mathrm{df}} & 1 \\ W(\mathrm{K}_m(M)) & =_{\mathrm{df}} & \kappa(M)+W(M) \\ W(\mathrm{S}_m(M,N)) & =_{\mathrm{df}} & W(M)+W(N) \end{array}$$

Then $W(Q_m(M)) \leq W(\mathrm{K}_m(M))$.

Proof. By structural induction on M.

If $M \equiv \mathrm{K}_j(N_1)$ with $m \leq j$, then $W(Q_m(M)) = W(\mathrm{K}_{j+1}(Q_m(N_1))) = \kappa(Q_m(N_1)) + W(Q_m(N_1)) = \kappa(N_1) + W(Q_m(N_1))$ (by Lemma F.3) $\leq \kappa(N_1) + W(\mathrm{K}_m(N_1))$ (by induction hypothesis) $= 2\kappa(N_1) + W(N_1)$, while $W(\mathrm{K}_m(M)) = \kappa(M) + W(M) = \kappa(N_1) + W(\mathrm{K}_j(N_1)) = 2\kappa(N_1) + W(N_1)$.

If $M \equiv \mathrm{S}_j(N_1, N_2)$ with $m < j$, then $W(Q_m(M)) = W(\mathrm{S}_{j+1}(Q_m(N_1), Q_m(N_2))) = W(Q_m(N_1)) + W(Q_m(N_2)) \leq W(\mathrm{K}_m(N_1)) + W(\mathrm{K}_m(N_2))$ (by induction hypothesis) $= \kappa(N_1) + \kappa(N_2) + W(N_1) + W(N_2)$, while $W(\mathrm{K}_m(M)) = \kappa(M) + W(M) = \kappa(N_1) + \kappa(N_2) + 1 + W(N_1) + W(N_2)$, so $W(Q_m(M)) < W(\mathrm{K}_m(M))$.

In all other cases, $Q_m(M) \equiv \mathrm{K}_m(M)$, so $W(Q_m(M)) = W(\mathrm{K}_m(M))$. □

Lemma F.5. The rewrite relation $\triangleright_I$ terminates on SKT-normal terms.

Proof. First, we claim that if Q is an I-redex and R is its contractum, then $W(Q) > W(R)$ and $\kappa(Q) \geq \kappa(R)$. Indeed, consider each rule in turn:

Rule $(\mathrm{S}_m\mathrm{I}_n)$: then $Q \equiv \mathrm{S}_m(\mathrm{I}_n, P)$, $R \equiv \mathrm{I}_{n-1}$. So $W(Q) = 1 + W(P) > 1 = W(R)$; indeed, an easy structural induction on P shows that $W(P) > 0$ for every term P. Also, $\kappa(Q) = 2 + \kappa(P) \geq 1 = \kappa(R)$.

Rule $(\mathrm{K}_m\mathrm{I}_n)$: then $Q \equiv \mathrm{K}_m(\mathrm{I}_{n-1})$, $R \equiv \mathrm{I}_n$, $W(Q) = \kappa(\mathrm{I}_{n-1}) + W(\mathrm{I}_{n-1}) = 2 > 1 = W(R)$. And $\kappa(Q) = 1 = \kappa(R)$.

Rule $(\mathrm{K}_m\mathrm{S}'_{m+1})$: Q is of the form $\mathrm{K}_m(\mathrm{S}_m(M, N))$, with $R \equiv \mathrm{S}_{m+1}(\mathrm{K}_m(M), \downarrow(\mathrm{K}_m(N)))$. So $W(Q) = \kappa(\mathrm{S}_m(M, N)) + W(\mathrm{S}_m(M, N)) = \kappa(M) + \kappa(N) + 1 + W(M) + W(N)$, while $W(R) = W(\mathrm{K}_m(M)) + W(\downarrow(\mathrm{K}_m(N))) = \kappa(M) + W(M) + W(\downarrow(\mathrm{K}_m(N))) = \kappa(M) + W(M) + W(Q_m(N))$ (by Lemma F.2, since N is SKT-normal) $\leq \kappa(M) + W(M) + W(\mathrm{K}_m(N))$ (by Lemma F.4) $= \kappa(M) + \kappa(N) + W(M) + W(N)$, which is therefore strictly less than $W(Q)$. Moreover, $\kappa(Q) = \kappa(M) + \kappa(N) + 1$, while $\kappa(R) = \kappa(M) + \kappa(Q_m(N)) + 1 = \kappa(M) + \kappa(N) + 1 = \kappa(Q)$ (by Lemma F.3).

Now if $Q \triangleright_I R$, a straightforward structural induction on the context under which the contraction occurs shows that again $W(Q) > W(R)$ and $\kappa(Q) \geq \kappa(R)$. Since $>$ is well-founded, this entails that $\triangleright_I$ terminates. $\square$

Let's call a Σ-reduction SKT*-eager* if and only if it is of the form:

$$M_0 \triangleright M_1 \triangleright M_2 \triangleright \ldots \triangleright M_i \triangleright \ldots$$

where, if the reduction from M_i to M_{i+1} is by some rule not in SKT, then M_i is SKT-normal—that is, all SKT-redexes are contracted prioritarily.

Proof. **(of Lemma 5.2)** We show that any SKT-eager Σ-reduction terminates; and in fact that these reductions are all of the form $M \triangleright^*_{SKT} P \triangleright^*_I Q$.

Consider a maximal SKT-eager reduction (see above). Let $i_0, i_1, \ldots, i_j, \ldots$ be the sequence of indices i (in increasing order) such that M_{i_j}, $j \geq 0$, are the SKT-normal terms in the sequence. This sequence may be finite, infinite, or even empty.

Since SKT terminates (as a subsystem of ΣT), there is an i such that M_i is SKT-normal, so i_0 exists (i.e., the sequence is non-empty). Moreover, we claim that: (1) $M_{i_0} \triangleright_I M_{i_1} \triangleright_I \ldots \triangleright_I M_{i_j} \triangleright_I \ldots$. More formally, we claim that $M_{i_j} \triangleright_I M_{i_{j+1}}$ for every $j \geq 0$. Indeed, consider the first rewrite step from M_{i_j} to $M_{i_{j+1}}$: if it is by $(\mathrm{S}_m\mathrm{I}_n)$ or by $(\mathrm{K}_m\mathrm{I}_n)$, then M_{i_j+1} is SKT-normal by Lemma F.1, so $i_{j+1} = i_j + 1$ (since $M_{i_{j+1}}$ is the next SKT-normal term after M_{i_j} in the sequence by definition), so (1) holds in this case. And if the first rewrite step from M_{i_j} is by rule $(\mathrm{S}_m\mathrm{K}_{m+1})$, then let N be the term obtained from M_{i_j} by applying rule $(\mathrm{S}_m\mathrm{K}'_{m+1})$ at the same position: by definition, (2) $M_{i_j} \triangleright_I N$; moreover, we can compute N by first rewriting by $(\mathrm{S}_m\mathrm{K}_{m+1})$ (getting M_{i_j+1}, clearly), and then taking the SKT-normal form (by Lemma F.1), so $N \equiv \downarrow(M_{i_j+1})$; but by definition $M_{i_{j+1}}$ is also an SKT-normal form of M_{i_j+1}, so by confluence $M_{i_{j+1}} \equiv N$, and therefore by (2) $M_{i_j} \triangleright_I M_{i_{j+1}}$.

By Lemma F.5, the sequence (1) is finite, and stops at M_{i_j} for some j. It follows that the initial maximal SKT-eager sequence stops at index i_j, and is therefore finite. This proves the claim. □

Lemma F.6. For every $M \in T_k$, $m \geq 0$, for all λ-variables $x_0, \ldots, x_{n-1}$, the variable x_i is free in $[\![M]\!]_\circ(x_0, \ldots, x_{n-1})$ whenever $k \leq i < n$.

Proof. By structural induction on M. We consider the case where $M \equiv \mathrm{K}_m(N)$, with $k > m$ and $M \in T_{k-1}$, then:

1if $n > m$, then:

$$[\![M]\!]_\circ(x_0, \ldots, x_{n-1}) \equiv [\![M]\!]_\circ(x_0, \ldots, x_{m-1}, x_{m+1}, \ldots, x_{n-1})$$

and for every i such that $k \leq i < n$, in particular $i \geq m+1$, and by induction hypothesis x_i is free in the right-hand side, hence in the left-hand side;

2if $n \leq m$, then $k > n$, so there is no i such that $k \leq i < n$.

□

Lemma F.7. For every $M \in T_k$, if $P_i \rhd^* Q_i$ in the λ-calculus for every $0 \leq i < n$, and $P_i \rhd^+ Q_i$ for some $i \geq k$, then:

$$[\![M]\!]_\circ(P_0, \ldots, P_{n-1}) \rhd^+ [\![M]\!]_\circ(Q_0, \ldots, Q_{n-1})$$

Similarly if we replace $\rhd$ by $\rhd_\eta$.

Proof. Check that $[\![M]\!]_\circ(P_0, \ldots, P_{n-1}) \rhd^* [\![M]\!]_\circ(Q_0, \ldots, Q_{n-1})$, by replaying the proof of Lemma A.1. By Lemma F.6, and the easily proved fact that:

$$[\![M]\!]_\circ(P_0, \ldots, P_{n-1}) \equiv [P_0/x_0, \ldots, P_{n-1}/x_{n-1}][\![M]\!]_\circ(x_0, \ldots, x_{n-1})$$

P_i occurs at some position in $[\![M]\!]_\circ(P_0, \ldots, P_{n-1})$, namely all the positions where x_i occurs free in $[\![M]\!]_\circ(x_0, \ldots, x_{n-1})$. It follows that the reduction takes at least one step. □

Lemma F.8. Every syntactically safe context is safe.

Proof. That the empty context [] is safe is clear. We consider the case where $\mathcal{C} \equiv \mathrm{S}_m(\mathcal{C}_1, P)$ with $\mathcal{C}_1$ syntactically safe, hence safe by induction, then $M \succ_\circ N$ implies $\mathcal{C}_1[M] \succ_\circ \mathcal{C}_1[N]$. We then have two cases. If $n \geq m$, then:

$$\begin{aligned}
&[\![\mathcal{C}[M]]\!]_\circ(P_0, \ldots, P_{n-1}) \\
&\equiv [\![\mathcal{C}_1[M]]\!]_\circ(P_0, \ldots, P_{m-1}, [\![P]\!]_\circ(P_0, \ldots, P_{m-1}), P_m, \ldots, P_{n-1}) \\
&\rhd_\eta^+ [\![\mathcal{C}_1[N]]\!]_\circ(P_0, \ldots, P_{m-1}, [\![P]\!]_\circ(P_0, \ldots, P_{m-1}), P_m, \ldots, P_{n-1}) \quad \text{since } \mathcal{C}_1[M] \succ_\circ \mathcal{C}_1[N] \\
&\equiv [\![\mathcal{C}[N]]\!]_\circ(P_0, \ldots, P_{n-1})
\end{aligned}$$

If $n < m$, then:

$$\begin{aligned}
&[\![\mathcal{C}[M]]\!]_\circ(P_0, \ldots, P_{n-1}) \\
&\equiv \lambda x_n \ldots \lambda x_{m-1}. \\
&\qquad [\![\mathcal{C}_1[M]]\!]_\circ(P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1}, [\![P]\!]_\circ(P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1})) \\
&\triangleright_\eta^+ \lambda x_n \ldots \lambda x_{m-1}. \\
&\qquad [\![\mathcal{C}_1[N]]\!]_\circ(P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1}, [\![P]\!]_\circ(P_0, \ldots, P_{n-1}, x_n, \ldots, x_{m-1})) \\
&\hfill \text{since } \mathcal{C}_1[M] \succ_\circ \mathcal{C}_1[N] \\
&\equiv [\![\mathcal{C}[N]]\!]_\circ(P_0, \ldots, P_{n-1})
\end{aligned}$$

In any case, $\mathcal{C}[M] \succ_\circ \mathcal{C}[N]$.

□

Lemma F.9. If $\mathcal{C}[M]$ is Σ-normal, then $\mathcal{C}$ is syntactically safe.

Proof. Let S_m denote the set of SKIn-terms M such that $\mathrm{S}_m(M, N)$ is Σ-normal for every Σ-normal term N, and K_m denote the set of SKIn-terms M such that $\mathrm{K}_m(M)$ is Σ-normal. By examination of the rules in Σ, we have:

— $\mathrm{S}_k(M, N) \in S_m$ if and only if $m \geq k$ and $M \in S_k$ and N is Σ-normal;
— $\mathrm{K}_k(M) \in S_m$ if and only if $m > k$ and $M \in K_k$;
— $\mathrm{I}_k \in S_m$ if and only if $m \geq k$;
— $x \in S_m$, for every variable x;
— $\mathrm{S}_k(M, N) \in K_m$ if and only if $m > k$ and $M \in S_k$ and N is Σ-normal;
— $\mathrm{K}_k(M) \in K_m$ if and only if $m > k$ and $M \in K_k$;
— $\mathrm{I}_k \in K_m$ if and only if $m > k$;
— $x \in K_m$, for every variable x;

A quick check shows that $P \in K_m$ implies $P \in S_m$ and that $P \in S_m$ implies $P \in K_{m+1}$: both results are by case analysis on P. So: (*i*) $K_m \subseteq S_m$ for every $m \geq 0$, and: (*ii*) $S_m \subseteq K_{m+1}$, $m \geq 0$. It follows that K_m and S_m form increasing sequences of sets, in particular: (*iii*) $S_m \subseteq S_p$ for every $m \leq p$. Moreover, by examination of the rules of Σ: (*iv*) $M \in S_m$ if and only if $\mathrm{S}_m(M, N)$ is Σ-normal for some N. Indeed, this hinges on the fact that $\mathrm{S}_m(M, N)$ is a Σ-redex if and only if M is of the form I_p, $\mathrm{S}_p(M_1, M_2)$, or $\mathrm{K}_p(M_1)$, $p > m$, and this condition is independent of N.

We claim that: (*v*) if $M \in S_m$, then $M \in T_m$. This is by structural induction on M. If M is of the form $\mathrm{S}_k(M_1, M_2)$, then by assumption $m \geq k$ and $M_1 \in S_k$; by (*iii*), $M_1 \in S_{m+1}$, so by induction hypothesis $M_1 \in T_{m+1}$, therefore $M \in T_m$. If M is of the form $\mathrm{K}_k(M_1)$, then by assumption $m > k$ and $M_1 \in K_k$; by (*i*), $M_1 \in S_k$, so by (*iii*), $M_1 \in S_{m-1}$, therefore by induction hypothesis $M_1 \in T_{m-1}$; it follows that $M \in T_m$. If $M \equiv \mathrm{I}_k$, then $m > k$, so $M \in T_m$. If M is a variable, then $M \in T_m$ by definition.

We now prove the Lemma by structural induction on $\mathcal{C}$. The only interesting case is when $\mathcal{C}$ is of the form $\mathrm{S}_m(P, \mathcal{C}_1)$, where $\mathcal{C}_1[M]$ is Σ-normal, so $\mathcal{C}_1$ is syntactically safe by induction. Since $\mathrm{S}_m(P, \mathcal{C}_1[M])$ is Σ-normal, by (*iv*) $P \in S_m$, so by (*v*) $P \in T_m$. It follows that $\mathcal{C} \equiv \mathrm{S}_m(P, \mathcal{C}_1)$ is syntactically safe. □

Appendix G. Confluence of SKIn

Recall that SKT is ΣT minus the rules involving I_m, and that I is the set of rules:

$$\begin{array}{rrcl}
(\mathrm{K}_m\mathrm{I}_n) & \mathrm{K}_m(\mathrm{I}_{n-1}) & \triangleright & \mathrm{I}_n \\
(\mathrm{S}_m\mathrm{I}_n) & \mathrm{S}_m(\mathrm{I}_n, N) & \triangleright & \mathrm{I}_{n-1} \\
(\mathrm{K}_m\mathrm{S}'_{m+1}) & \mathrm{K}_m(\mathrm{S}_m(M,N)) & \triangleright & \mathrm{S}_{m+1}(\mathrm{K}_m(M), \downarrow(\mathrm{K}_m(N)))
\end{array}$$

where $0 \leq m < n$, and $\downarrow(P)$ denotes the unique SKT-normal form of P.

We first need a way of computing the SKT-normal form of a given term. We build on the definition of $N \mapsto Q_m(N)$ (Lemma F.2), and define the following:

Lemma G.1. For each $m \geq 0$, let E_m be the function defined as follows:

$$\begin{array}{rcl}
E_m(x,P) & =_{\mathrm{df}} & \mathrm{S}_m(x,P) \\
E_m(\mathrm{I}_j,P) & =_{\mathrm{df}} & \mathrm{S}_m(\mathrm{I}_j,P) \\
E_m(\mathrm{K}_j(N_1),P) & =_{\mathrm{df}} & \left\{ \begin{array}{ll} \mathrm{K}_{j-1}(E_m(N_1,P)) & (m<j) \\ N_1 & (m=j) \\ \mathrm{S}_m(\mathrm{K}_j(N_1),P) & (m>j) \end{array} \right. \\
E_m(\mathrm{S}_j(N_1,N_2),P) & =_{\mathrm{df}} & \left\{ \begin{array}{ll} \mathrm{S}_{j-1}(E_m(N_1,P),E_m(N_2,P)) & (m<j) \\ \mathrm{S}_m(\mathrm{S}_j(N_1,N_2),P) & (m \geq j) \end{array} \right.
\end{array}$$

Whenever N and P are SKT-normal, $E_m(N,P) \equiv \downarrow(\mathrm{S}_m(N,P))$.

Proof. Let $\triangleright_{SKT}$ be one-step reduction by rules in SKT. Clearly, $\mathrm{S}_m(N,P) \triangleright^*_{SKT} E_m(N,P)$, so we just have to prove that $E_m(N,P)$ is SKT-normal. In fact, we show by structural induction on N that: (1) $E_m(N,P)$ is SKT-normal whenever N and P are, and: (2) $i(E_m(N,P)) \leq \max(m, i(N)-1)$. Recall that i is defined by $i(x) =_{\mathrm{df}} 0$, $i(\mathrm{I}_j) =_{\mathrm{df}} 0$, $i(\mathrm{K}_j(N_1)) =_{\mathrm{df}} j+1$, $i(\mathrm{S}_j(N_1,N_2)) =_{\mathrm{df}} j$ (Lemma F.2), that $\mathrm{K}_j(N)$ is SKT-normal if and only if N is and $j \geq i(N)$, and that $\mathrm{S}_j(N,P)$ is SKT-normal if and only if N and P are, and $j \geq i(N)$; these are facts which we have already used in Lemma F.2.

If N is a variable, of the form I_j, or $\mathrm{S}_j(\ldots,)$ with $m \geq j$, or $\mathrm{K}_j(\ldots)$ with $m > j$, then $E_m(N,P) = \mathrm{S}_m(N,P)$ is clearly SKT-normal (showing (1)), and $i(E_m(N,P)) = m \leq \max(m, i(N)-1)$ (showing (2)).

If $N \equiv \mathrm{K}_j(N_1)$ with $m = j$, then $E_m(N,P) \equiv N_1$ is clearly SKT-normal, hence (1) holds. Moreover, since N is SKT-normal by assumption, $j \geq i(N_1)$, so $i(E_m(N,P)) = i(N_1) \leq j = m \leq \max(m, i(N)-1)$.

If $N \equiv \mathrm{K}_j(N_1)$ with $m < j$, then $E_m(N,P) \equiv \mathrm{K}_{j-1}(E_m(N_1,P))$. By induction hypothesis, $E_m(N_1,P)$ is SKT-normal and $i(E_m(N_1,P)) \leq \max(m, i(N_1)-1) \leq \max(m, j-1)$ (since $j \geq i(N_1)$, by SKT-normality of N) $= j-1$ (since $m < j$). In particular, $E_m(N,P) = \mathrm{K}_{j-1}(E_m(N_1,P))$ is SKT-normal; this shows (1). Moreover, $i(E_m(N,P)) = j = \max(m,j) = \max(m, i(N)-1)$, showing (2).

If $N \equiv \mathrm{S}_j(N_1,N_2)$ with $m < j$, then $E_m(N,P) \equiv \mathrm{S}_{j-1}(E_m(N_1,P), E_m(N_2,P))$. By induction hypothesis, $E_m(N_1,P)$ is SKT-normal, $E_m(N_2,P)$ is SKT-normal, and $i(E_m(N_1,P)) \leq \max(m, i(N_1)-1) \leq \max(m, j-1)$ (since $j \geq i(N_1)$, by SKT-normality of N) $= j-1$ (since $m < j$). In particular, $E_m(N,P) = \mathrm{S}_{j-1}(E_m(N_1,P), E_m(N_2,P))$ is SKT-normal; this shows (1). And $i(E_m(N,P)) = j-1 = \max(m, j-1) = \max(m, i(N)-1)$, thus showing (2). □

Lemma F.2 and Lemma G.1 together give us the following way to compute the SKT-normal form of a term:

Corollary G.2. For all terms M and N:

$$\begin{array}{rcl} \downarrow(x) & \equiv & x \\ \downarrow(\mathrm{I}_m) & \equiv & \mathrm{I}_m \\ \downarrow(\mathrm{K}_m(M)) & \equiv & Q_m(\downarrow(M)) \\ \downarrow(\mathrm{S}_m(M,N)) & \equiv & E_m(\downarrow(M),\downarrow(N)) \end{array}$$

Now call I_0 the set of all rules $(\mathrm{K}_m\mathrm{S}'_{m+1})$, $m \geq 0$. Let $\triangleright_{I_0}$ denote one-step reduction by I_0, and $\triangleright^+_{I_0}$ be its transitive closure.

Lemma G.3. If M is SKT-normal and $M \triangleright_{I_0} N$, then $Q_m(M) \triangleright^+_{I_0} Q_m(N)$.

Proof. By structural induction on the context under which the reduction occurs in M.

If $M \equiv \mathrm{S}_j(M_1, M_2)$, $N \equiv \mathrm{S}_j(N_1, M_2)$, with $M_1 \triangleright_{I_0} N_1$, and $m < j$, then $Q_m(M) \equiv \mathrm{S}_{j+1}(Q_m(M_1), Q_m(M_2)) \triangleright^+_{I_0} \mathrm{S}_{j+1}(Q_m(N_1), Q_m(M_2))$ (by induction hypothesis), and this is just $Q_m(N)$.

If $M \equiv \mathrm{S}_j(M_1, M_2)$, $N \equiv \mathrm{S}_j(M_1, N_2)$, with $M_2 \triangleright_{I_0} N_2$, and $m < j$, then $Q_m(M) \equiv \mathrm{S}_{j+1}(Q_m(M_1), Q_m(M_2)) \triangleright^+_{I_0} \mathrm{S}_{j+1}(Q_m(M_1), Q_m(N_2))$ (by induction hypothesis), and this is exactly $Q_m(N)$.

If $M \equiv \mathrm{K}_j(M_1)$, $N \equiv \mathrm{K}_j(N_1)$, with $M_1 \triangleright_{I_0} N_1$, and $m \leq j$, then $Q_m(M) \equiv \mathrm{K}_{j+1}(Q_m(M_1)) \triangleright^+_{I_0} \mathrm{K}_{j+1}(Q_m(N_1))$ (by induction hypothesis) $\equiv Q_m(N)$.

In all other cases where the enclosing context is not empty, then $Q_m(M) \equiv \mathrm{K}_m(M) \triangleright_{I_0} \mathrm{K}_m(N) \equiv Q_m(N)$.

Finally, when M is itself the reduced I_0-redex, then $M \equiv \mathrm{K}_j(\mathrm{S}_j(M_1, M_2))$, and $N \equiv \mathrm{S}_{j+1}(\mathrm{K}_j(M_1), Q_j(M_2))$ for some $j \geq 0$. There are three cases, corresponding to when $m < j$, $m = j$ and $m > j$. We consider the first:

$$\begin{array}{ll} Q_m(M) \equiv \mathrm{K}_{j+1}(\mathrm{S}_{j+1}(Q_m(M_1), Q_m(M_2))) & \\ \triangleright_{I_0} \mathrm{S}_{j+2}(\mathrm{K}_{j+1}(Q_m(M_1)), \downarrow(\mathrm{K}_{j+1}(Q_m(M_2)))) & \\ \equiv \mathrm{S}_{j+2}(\mathrm{K}_{j+1}(Q_m(M_1)), \downarrow(\mathrm{K}_{j+1}(\downarrow(\mathrm{K}_m(M_2))))) & \text{(by Lemma F.2)} \\ \equiv \mathrm{S}_{j+2}(\mathrm{K}_{j+1}(Q_m(M_1)), \downarrow(()\mathrm{K}_{j+1}(\mathrm{K}_m(M_2)))) & \\ \equiv \mathrm{S}_{j+2}(\mathrm{K}_{j+1}(Q_m(M_1)), \downarrow(()\mathrm{K}_m(\mathrm{K}_j(M_2)))) & \text{(by rule } (\mathrm{K}_m\mathrm{K}_{j+1})) \\ \equiv \mathrm{S}_{j+2}(\mathrm{K}_{j+1}(Q_m(M_1)), Q_m(Q_j(M_2))) & \text{(by Corollary G.2)} \\ \equiv Q_m(\mathrm{S}_{j+1}(\mathrm{K}_j(M_1), Q_j(M_2))) \equiv Q_m(N) & \end{array}$$

□

Lemma G.4. If M and P are SKT-normal and $M \triangleright_{I_0} N$, then $E_m(M, P) \triangleright^*_{I_0} E_m(N, P)$.

Proof. By structural induction on the context under which the reduction occurs in M.

If $M \equiv \mathrm{S}_j(M_1, M_2)$, $N \equiv \mathrm{S}_j(N_1, M_2)$, with $M_1 \triangleright_{I_0} N_1$, and $m < j$, then $E_m(M, P) \equiv \mathrm{S}_{j-1}(E_m(M_1, P), E_m(M_2, P)) \triangleright^*_{I_0} \mathrm{S}_{j-1}(E_m(N_1, P), E_m(M_2, P))$ (by induction hypothesis) $\equiv E_m(N, P)$.

If $M \equiv \mathrm{S}_j(M_1, M_2)$, $N \equiv \mathrm{S}_j(M_1, N_2)$, with $M_2 \triangleright_{I_0} N_2$, and $m < j$, then $E_m(M, P) \equiv \mathrm{S}_{j-1}(E_m(M_1, P), E_m(M_2, P)) \triangleright^*_{I_0} \mathrm{S}_{j-1}(E_m(M_1, P), E_m(N_2, P))$ (by induction hypothesis) $\equiv E_m(N, P)$.

If $M \equiv \mathrm{K}_j(M_1)$, $N \equiv \mathrm{K}_j(N_1)$, with $M_1 \triangleright_{I_0} N_1$, and $m < j$, then $E_m(M,P) \equiv \mathrm{K}_{j-1}(E_m(M_1,P)) \triangleright^*_{I_0} \mathrm{K}_{j-1}(E_m(N_1,P))$ (by induction hypothesis) $\equiv E_m(N,P)$.

If $M \equiv \mathrm{K}_j(M_1)$, $N \equiv \mathrm{K}_j(N_1)$, with $M_1 \triangleright_{I_0} N_1$, and $m = j$, then $E_m(M,P) \equiv M_1 \triangleright_{I_0} N_1 \equiv E_m(N,P)$.

In all other cases where the enclosing context is not empty, $E_m(M,P) \equiv \mathrm{S}_m(M,P) \triangleright_{I_0} \mathrm{S}_m(N,P) \equiv E_m(N,P)$.

Finally, when M is itself the reduced I_0-redex, then $M \equiv \mathrm{K}_j(\mathrm{S}_j(M_1,M_2))$, and $N \equiv \mathrm{S}_{j+1}(\mathrm{K}_j(M_1), Q_j(M_2))$ for some $j \geq 0$. There are three cases:

— If $m < j$, then:

$$\begin{array}{ll}
E_m(M,P) \equiv \mathrm{K}_{j-1}(\mathrm{S}_{j-1}(E_m(M_1,P), E_m(M_2,P))) & \\
\triangleright_{I_0} \mathrm{S}_j(\mathrm{K}_{j-1}(E_m(M_1,P)), {\downarrow}(\mathrm{K}_{j-1}(E_m(M_2,P)))) & \\
\equiv \mathrm{S}_j(\mathrm{K}_{j-1}(E_m(M_1,P)), {\downarrow}(\mathrm{K}_{j-1}({\downarrow}(\mathrm{S}_m(M_2,P))))) & \text{(by Lemma G.1)} \\
\equiv \mathrm{S}_j(\mathrm{K}_{j-1}(E_m(M_1,P)), {\downarrow}(\mathrm{K}_{j-1}(\mathrm{S}_m(M_2,P)))) & \\
\equiv \mathrm{S}_j(\mathrm{K}_{j-1}(E_m(M_1,P)), {\downarrow}(\mathrm{S}_m(\mathrm{K}_j(M_2),P))) & \text{(by rule } (\mathrm{S}_m\mathrm{K}_j)\text{)} \\
\equiv \mathrm{S}_j(\mathrm{K}_{j-1}(E_m(M_1,P)), E_m(Q_j(M_2),P)) & \text{(by Corollary G.2)} \\
\equiv E_m(\mathrm{S}_{j+1}(\mathrm{K}_j(M_1), \mathrm{K}_j(M_2)), P) \equiv E_m(N,P) &
\end{array}$$

— If $m = j$, then $E_m(M,P) \equiv \mathrm{S}_m(M_1,M_2)$, and $E_m(N,P) \equiv \mathrm{S}_m(M_1, E_m(Q_m(M_2),P)) \equiv \mathrm{S}_m(M_1,M_2)$. Indeed, $E_m(Q_m(M_2),P) \equiv E_m({\downarrow}(\mathrm{K}_m(M_2)),P)$ (since M_2 is SKT-normal) $\equiv {\downarrow}(\mathrm{S}_m({\downarrow}(\mathrm{K}_m(M_2)),P))$ (since P is SKT-normal) $\equiv {\downarrow}(\mathrm{S}_m(\mathrm{K}_m(M_2),P)) \equiv {\downarrow}(M_2)$ (by rule (SK_m)) $\equiv M_2$ (since M_2 is SKT-normal).

— If $m > j$, then $E_m(M,P) \equiv \mathrm{S}_m(M,P) \triangleright_{I_0} \mathrm{S}_m(N,P) \equiv \mathrm{S}_m(\mathrm{S}_{j+1}(\mathrm{K}_j(M_1), \mathrm{K}_j(M_2)), P) \equiv E_m(\mathrm{S}_{j+1}(\mathrm{K}_j(M_1), \mathrm{K}_j(M_2)))$ (since $m \geq j+1$) $\equiv E_m(N,P)$.

□

Lemma G.5. If $N \triangleright_{I_0} P$, then $E_m(M,N) \triangleright^*_{I_0} E_m(M,P)$.

Proof. Straightforward induction on the structure of M. □

Lemma G.6. If $M \triangleright N$ by rule $(\mathrm{K}_j\mathrm{S}_{j+1})$ for some $j \geq 0$, then ${\downarrow}(M) \triangleright^*_{I_0} {\downarrow}(N)$.

Proof. By structural induction on the context under which reduction takes place inside M.

If $M \equiv \mathrm{S}_m(M_1,M_2)$, $N \equiv \mathrm{S}_m(N_1,M_2)$ with $M_1 \triangleright N_1$ by rule $(\mathrm{K}_j\mathrm{S}_{j+1})$, then ${\downarrow}(M) \equiv E_m({\downarrow}(M_1), {\downarrow}(M_2))$. By induction hypothesis, ${\downarrow}(M_1) \triangleright^*_{I_0} {\downarrow}(N_1)$. By Lemma G.4, it follows that $E_m({\downarrow}(M_1), {\downarrow}(M_2)) \triangleright^*_{I_0} E_m({\downarrow}(N_1), {\downarrow}(M_2))$, i.e., ${\downarrow}(M) \triangleright^*_{I_0} {\downarrow}(N)$.

If $M \equiv \mathrm{S}_m(M_1,M_2)$, $N \equiv \mathrm{S}_m(M_1,N_2)$ with $M_2 \triangleright N_2$ by rule $(\mathrm{K}_j\mathrm{S}_{j+1})$, then ${\downarrow}(M) \equiv E_m({\downarrow}(M_1), {\downarrow}(M_2))$. By induction hypothesis, ${\downarrow}(M_2) \triangleright^*_{I_0} {\downarrow}(N_2)$. By Lemma G.5, it follows that $E_m({\downarrow}(M_1), {\downarrow}(M_2)) \triangleright^*_{I_0} E_m({\downarrow}(M_1), {\downarrow}(N_2))$, i.e., ${\downarrow}(M) \triangleright^*_{I_0} {\downarrow}(N)$.

If $M \equiv \mathrm{K}_m(M_1)$, $N \equiv \mathrm{K}_m(N_1)$ with $M_1 \triangleright N_1$ by rule $(\mathrm{K}_j\mathrm{S}_{j+1})$, then ${\downarrow}(M) \equiv Q_m({\downarrow}(M_1))$. By induction hypothesis, ${\downarrow}(M_1) \triangleright^*_{I_0} {\downarrow}(N_1)$. By Lemma G.3, it follows that $Q_m({\downarrow}(M_1)) \triangleright^*_{I_0} Q_m({\downarrow}(N_1))$, i.e., ${\downarrow}(M) \triangleright^*_{I_0} {\downarrow}(N)$. □

Corollary G.7. Let SK denote the group of rules SKT plus $(\mathrm{K}_m\mathrm{S}_{m+1})$, $m \geq 0$, (alternatively, all rules in Σ except $(\mathrm{K}_m\mathrm{I}_n)$ and $(\mathrm{S}_m\mathrm{I}_n)$, $0 \leq m < n$).

Then SK is confluent.

Proof. By Hardin's interpretation method (Hardin, 1989): whenever M rewrites to M_1 and to M_2 in any finite number of SK-steps, it follows by Lemma G.6 and a straightforward induction on the length of reductions that $\downarrow(M) \triangleright^*_{I_0} \downarrow(M_1)$ and $\downarrow(M) \triangleright^*_{I_0} \downarrow(M_2)$. But I_0 is trivially confluent, since it is a left-linear rewrite system without critical pairs (Huet, 1977), so $\downarrow(M_1)$ and $\downarrow(M_2)$ have a common I_0-reduct N. But N is now a common SK-reduct of M_1 and M_2, since I_0-reductions can be simulated in SK. □

Observe, by the way, that SK does not terminate, even in the typed case—although reducing SKT-redexes first, then I_0-redexes, always terminates by Lemma 5.2.

Lemma G.8. Σ is confluent.

Proof. By Hardin's interpretation method again. Let I_1 be the group of rules $(\mathrm{S}_m\mathrm{I}_n)$ and $(\mathrm{K}_m\mathrm{I}_n)$, $0 \leq m < n$. S is clearly terminating (the size of terms decreases) and confluent (no critical pair). The remaining rules of Σ are just those of SK.

A quick check show that:

$$(i) \qquad \begin{tikzcd} M \arrow[r, "I_1"] \arrow[d, "SK"'] & M_1 \arrow[d, dotted, "I_1^* SK^="] \\ M_2 \arrow[r, dotted, "I_1^*"'] & N \end{tikzcd}$$

where $I_1^* SK^=$ denotes any number of I_1 steps followed by zero or one application of a rule in SK, and I_1^* is the reflexive-transitive closure of I_1. Indeed, there are five critical pairs between I_1 and SK, and they all join by just using I_1^* (not SK); that we need at most one application of a rule in SK follows from the fact that I_1 is right-linear. There are five critical pairs, of which the following is an example, with $0 \leq m < n < p$:

$$(2) \qquad \begin{tikzcd}[column sep=small] & \mathrm{K}_m(\mathrm{S}_{n-1}(\mathrm{I}_{p-1}, P)) \arrow[dl, "(\mathrm{K}_m\mathrm{S}_n)"'] \arrow[dr, "(\mathrm{S}_{n-1}\mathrm{I}_{p-1})"] & \\ \mathrm{S}_n(\mathrm{K}_m(\mathrm{I}_{p-1}), P) \arrow[dr, dotted, "{(\mathrm{K}_m\mathrm{I}_p), (\mathrm{S}_n\mathrm{I}_p)}"'] & & \mathrm{K}_m(\mathrm{I}_{p-2}) \arrow[dl, dotted, "(\mathrm{K}_m\mathrm{I}_{p-1})"] \\ & \mathrm{I}_{p-1} & \end{tikzcd}$$

Since I_1 terminates (trivial), we may define $\nu_1(M)$ as the length of the longest I_1-

reduction starting from M. By induction on $\nu_1(M)$, we get:

$$(ii)\qquad \begin{tikzcd} M \arrow[r, "I_1^*"] \arrow[d, "SK"'] & M_1 \arrow[d, dotted, "I_1^* SK^="] \\ M_2 \arrow[r, dotted, "I_1^*"'] & N \end{tikzcd}$$

Indeed, this is obvious if the length of the top I_1^*-reduction is zero, and otherwise this follows by the following pasting:

$$\begin{tikzcd} M \arrow[r, "I_1"] \arrow[dd, "SK"'] \arrow[ddr, phantom, "(\text{by } (i))"] & M_1' \arrow[r, "I_1^*"] \arrow[d, dotted, "I_1^*"'] \arrow[dr, phantom, "(\text{by confluence of } I_1)"] & M_1 \arrow[d, dotted, "I_1^*"] \\ & N_1 \arrow[r, dotted, "I_1^*"'] \arrow[d, dotted, "SK^="'] \arrow[dr, phantom, "(\text{by induction hypothesis})"] & N_2 \arrow[d, dotted, "I_1^* SK^="] \\ M_2 \arrow[r, dotted, "I_1^*"'] & M_2' \arrow[r, dotted, "I_1^*"'] & N \end{tikzcd}$$

The induction hypothesis applies, since $\nu_1(N_1) \leq \nu_1(M_1') < \nu_1(M)$.

Let $\triangleright_{SK}$ be one-step reduction by rules in the SK group. Whenever $M \triangleright_{SK} N$, then the I_1-normal form $I_1(M)$ of M reduces to that, $I_1(N)$, of N in at most one application of a rule in SK, by (ii). A straightforward induction on the lengths of reductions then shows that if $M \triangleright_\Sigma^* M_1$ and $M \triangleright_\Sigma^* M_2$, then $I_1(M) \triangleright_{SK}^* I_1(M_1)$ and $I_1(M) \triangleright_{SK}^* I_1(M_2)$. By Corollary G.7, $I_1(M_1)$ and $I_1(M_2)$ must then have a common SK-reduct, which is then also a common Σ-reduct of M_1 and M_2. $\square$

Lemma G.9. SKIn is confluent.

Proof. Same proof as for Lemma G.8, replacing I_1 by I_2, which is I_1 plus (SI_m), $m \geq 0$. The argument is the same, we just have two additional critical pairs. $\square$

Appendix H. The L Translation

Lemma H.1. If z and z' are distinct variables, and z is not free in N, then:

(i) $[P/z]L_{z'}(N) \equiv L_{z'}(N)$;

(ii) $[P/z]L(N) \equiv L(N)$.

Proof. Observe that (ii) follows from (i). We prove (i) by structural induction on N. If N is a variable x, then $x \not\equiv z$ by assumption, so $[P/z]L_{z'}(N) \equiv [P/z](x(z')) \equiv x([P/z]z') \equiv x(z')$ (since $z \not\equiv z'$) $\equiv L_{z'}(N)$.

If N is an application $N_1(N_2)$, then $[P/z]L_{z'}(N) \equiv ([P/z]L_{z'}(N_1))([P/z]L(N_2)) \equiv L_{z'}(N_1)(L(N_2))$ (by induction hypothesis) $\equiv L_{z'}(N)$.

If N is an abstraction $\lambda x.N_1$, then $[P/z]L_{z'}(N) \equiv \lambda x.[P/z]L_{z'}(N_1) \equiv \lambda x.L_{z'}(N_1)$ (by induction hypothesis) $\equiv L_{z'}(N)$. □

Lemma H.2. For every variable z not free in N, for every term P:

$$[P/z]L_z(N) \equiv L_P(N)$$

Proof. By structural induction on N.

If N is a variable x, then $[P/z]L_z(x) \equiv [P/z](x(z)) \equiv x(P)$ since z is not free in x, and the latter is just $L_P(N)$.

If N is an application $N_1(N_2)$, then $[P/z]L_z(N_1(N_2)) \equiv [P/z](L_z(N_1)(L(N_2))) \equiv ([P/z]L_z(N_1))([P/z]L(N_2)) \equiv L_P(N_1)([P/z]L(N_2))$ (by induction hypothesis) $\equiv L_P(N_1)$ $(L(N_2))$ (by Lemma H.1 (ii)) $\equiv L_P(N)$.

And if N is an abstraction $\lambda x.N_1$, then $[P/z]L_z(N) \equiv \lambda x.[P/z]L_z(N_1) \equiv \lambda x.L_P(N_1)$ (by induction hypothesis) $\equiv L_P(N)$. □

Lemma H.3. For all λ-terms M_1, M_2:

(i) $[L(M_2)/x]L_P(M_1) \rhd^* L_P([M_2/x]M_1))$ in the λ_V-calculus, for every P having no variable in common with M_1;

(ii) $[L(M_2)/x]L(M_1) \rhd^* L([M_2/x]M_1)$ in the λ_V-calculus.

Proof. Again, (ii) follows from (i). We prove (i) by structural induction on M_1.

If $M_1 \equiv x$, then $[L(M_2)/x]L_P(M_1) \equiv [L(M_2)/x](x(P)) \equiv L(M_2)(P)$ (since P has no variable in common with M_1, in particular x is not free in P) $\equiv (\lambda z.L_z(M_2))(P) \rhd L_P(M_2)$ (by Lemma H.2) $\equiv L_P([M_2/x]M_1)$.

If M_1 is another variable y, then $[L(M_2)/x]L_P(M_1) \equiv [L(M_2)/x](y(P)) \equiv y(P)$ (since P has no variable in common with M_1, hence y is not free in P) $\equiv L_P(y) \equiv L_P([M_2/x]M_1)$.

If M_1 is an application $M_3(M_4)$, then:

$$\begin{array}{lcll}
[L(M_2)/x]L_P(M_1) & \equiv & [L(M_2)/x](L_P(M_3)(L(M_4))) & \\
 & \equiv & ([L(M_2)/x]L_P(M_3))([L(M_2)/x]L(M_4)) & \\
 & \rhd^* & L_P([M_2/x]M_3)(L([M_2/x]M_4)) & \text{(by induction hypothesis)} \\
 & \equiv & L_P(([M_2/x]M_3)([M_2/x]M_4)) & \\
 & \equiv & L_P([M_2/x]M_1) &
\end{array}$$

And if M_1 is a λ-abstraction $\lambda y.M_3$, then:

$$\begin{array}{lcll}
[L(M_2)/x]L_P(M_1) & \equiv & [L(M_2)/x](\lambda y.L_P(M_3)) & \\
 & \equiv & \lambda y.[L(M_2)/x]L_P(M_3) & \\
 & \rhd^* & \lambda y.L_P([M_2/x]M_3) & \text{(by induction hypothesis)} \\
 & \equiv & L_P([M_2/x]M_1) &
\end{array}$$

□

Lemma H.4. If $M \rhd N$ in the λ-calculus, then:

(i) $L_P(M) \rhd^+ L_P(N)$ in the λ_V-calculus, for every P having no variables in common with M;
(ii) $L(M) \rhd^+ L(N)$ in the λ_V-calculus.

Proof. Again, (ii) follows from (i), so we prove (i) by structural induction on the context under which the contraction occurs.

Let's examine the base case, where M is itself the contracted redex. So M is a β-redex $(\lambda x.M_1)(M_2)$, then $N \equiv [M_2/x]M_1$, and $L_P(M) \equiv (\lambda x.L_P(M_1))(L(M_2))$ rewrites in one (β_V) step (since $L(M_2) \equiv \lambda z \cdot L_z(M_2)$ is a value) to $[L(M_2)/x]L_P(M_1)$, which rewrites in zero or more (β_V) steps to $L_P([M_2/x]M_1)$ by Lemma H.3 (i), i.e. to $L_P(N)$.

In the induction case, we have three sub-cases.

If (Case 1) M is of the form $M_1(M_2)$, with $M_1 \rhd N_1$ and $N \equiv N_1(M_2)$, then $L_P(M) \equiv L_P(M_1)(L(M_2)) \rhd^+ L_P(N_1)(L(M_2))$ (by induction hypothesis) $\equiv L_P(N)$.

If (Case 2) M is of the form $M_1(M_2)$ with $M_2 \rhd N_2$ and $N \equiv M_1(N_2)$, then the argument is similar, except that apply the induction hypothesis to show that $L(M_2) \rhd^+ L(N_2)$.

Finally, if (Case 3) M is of the form $\lambda x.M_1$, $M_1 \rhd N_1$ and $N \equiv \lambda x.N_1$, then $L_P(M) \equiv \lambda x.L_P(M_1) \rhd^+ \lambda x.L_P(N_1)$ (by induction hypothesis) $\equiv L_P(N)$. □

Proof. **(of Theorem 6.3)** We first claim that, if $M \rhd N$ in the λ-calculus, then $L^*(M) \rhd^* L^*(N)$ in SKInT. Indeed, $L(M) \rhd^+ L(N)$ in λ_V, by Lemma H.4 (ii), so $L^*(M) \rhd^* L^*(N)$ by Lemma 6.1 (ii). It follows that $M \rhd^* N$ also implies $L^*(M) \rhd^* L^*(N)$, by induction on the length of reductions. It is then immediate that $M = N$ implies $L^*(M) = L^*(N)$. □

We have a subject reduction property for the types T_z and V:

Lemma H.5. If $s \in T_z$ and $s \rhd^*_\eta t$ in the λ_η-calculus, then $t \in T_z$. If $u \in V$ and $u \rhd^*_\eta v$ in the λ_η-calculus, then $v \in V$.

Proof. By induction on the length of reductions, then on the context under which contraction occurs. We first establish a few auxiliary claims.

$(i)$for all $x \in X$, $s \in T_z$, $u \in V$, $v \in V$, $[v/x]s$ is in T_z, and $[v/x]u$ is in V; this is by mutual induction on the structure of s and u, noticing that we replace $x \in X \subseteq V$ by a term in V.
$(ii)$for all $s \in T_z$, $u \in V$, z is the only variable in Z free in s, and no variable in Z is free in u. This is straightforward.
$(iii)$for all $z, z' \in Z$, $s \in T_{z'}$, $[z/z']s \in T_z$. This is an easy structural induction on s, using (ii) to show that $[z/z']u \equiv u$ for every immediate subterm $u \in V$ of s.

We now claim that, whenever $s \in T_z$ is a λ_η-redex, and t is its reduct, then $t \in T_z$. There are several cases. If $s \equiv (\lambda z'.s')(z)$ with $s' \in T_{z'}$, then $t \equiv [z/z']s'$ (by (β)) is in T_z by (iii). If $s \equiv (\lambda x.s')(v)$ with $s' \in T_z$ and $v \in V$, then $t \equiv [v/x]s'$ (by β), therefore $t \in T_z$ by (i). If s is an η-redex, then s is of the form $\lambda x.t(x)$ with $t \in T_z$, so $t \in T_z$, obviously.

Similarly, we claim that, whenever $u \in V$ is a λ_η-redex, and v is its reduct, then $v \in V$. There is only one case: u must be an η-redex, $u \equiv \lambda z.v(z)$ with $v \in V$, so $v \in V$, trivially.

It follows that whenever $s \in T_z$ and $s \rhd_\eta t$, then $t \in T_z$, and whenever $u \in V$ and $u \rhd_\eta v$, then $v \in V$. This is by induction on the depth of the contracted redex. We have just dealt with the base case, and the induction case is trivial.

The Lemma then follows by induction on the length of the reduction. □

Lemma H.6. If $s \in T_z$ and $s \rhd^* t$ in the λ-calculus, then $L_z^{-1}(s) \rhd^* L_z^{-1}(t)$ in the λ-calculus. If $u \in V$ and $u \rhd^* v$ in the λ-calculus, then $L^{-1}(u) \rhd^* L^{-1}(v)$ in the λ-calculus.

Proof. We first claim that: (i) for all $z, z' \in Z$, for all $s \in T_{z'}$, $L_{z'}^{-1}(s) \equiv L_z^{-1}([z/z']s)$. This is an easy structural induction on s, using the fact that z' is not free in any term in V (Lemma H.5, claim (ii)).

We then claim that: (ii) $[L^{-1}(v)/x]L_z^{-1}(t) \equiv L_z^{-1}([v/x]t)$, and $[L^{-1}(v)/x]L^{-1}(u) \equiv L^{-1}([v/x]u)$, for all $t \in T_z$, $x \in X$, and $u, v \in V$. This is by mutual structural induction on t and u. If t is of the form $u(z)$ with $u \in V$, then $[L^{-1}(v)/x]L_z^{-1}(t) \equiv [L^{-1}(v)/x]L^{-1}(u) \equiv L^{-1}([v/x]u)$ (by induction) $\equiv L_z^{-1}(([v/x]u)(z)) \equiv L_z^{-1}(([v/x]u(z))) \equiv L_z^{-1}([v/x]t)$ (since $x \not\equiv z$). If t is of the form $s(u)$ with $s \in T_z$, $u \in V$, then:

$$\begin{array}{rcll} [L^{-1}(v)/x]L_z^{-1}(t) & \equiv & [L^{-1}(v)/x](L_z^{-1}(s)(L^{-1}(u))) & \\ & \equiv & L_z^{-1}([v/x]s)(L^{-1}([v/x]u)) & \text{(by induction)} \\ & \equiv & L_z^{-1}([v/x]t) & \end{array}$$

If t is of the form $\lambda y.s$ with $s \in T_z$, then $[L^{-1}(v)/x]L_z^{-1}(t) \equiv [L^{-1}(v)/x]\lambda y.L_z^{-1}(s) \equiv \lambda y.L_z^{-1}([v/x]s)$ (by induction) $\equiv L_z^{-1}(\lambda y.[v/x]s) \equiv L_z^{-1}(t)$. On the other hand, if $u \in V$ is of the form $\lambda z.t$ with $t \in T_z$, then $[L^{-1}(v)/x]L^{-1}(u) \equiv [L^{-1}(v)/x]L_z^{-1}(t) \equiv L_z^{-1}([v/x]t)$ (by induction) $\equiv L^{-1}(\lambda z.([v/x]t)) \equiv L^{-1}(\lambda z.[v/x]t)$. If u is a variable y in X, different from x, then $[L^{-1}(v)/x]L^{-1}(u) \equiv y \equiv L^{-1}(y) \equiv L^{-1}([v/x]u)$. Finally, if $u \equiv x$, then $[L^{-1}(v)/x]L^{-1}(u) \equiv L^{-1}(v) \equiv L^{-1}([v/x]u)$.

If $s \in T_z$ is a β-redex, then there are two cases. If $s \equiv (\lambda z'.s')(z)$ with $s' \in T_{z'}$, then $L_z^{-1}(s) \equiv L^{-1}(\lambda z'.s') \equiv L_{z'}^{-1}(s') \equiv L_z^{-1}([z/z']s')$ by (i). If $s \equiv \lambda x.t(v)$ with $t \in T_z$ and $v \in V$, then $L_z^{-1}(s) \equiv (\lambda x.L_z^{-1}(t))(L^{-1}(v)) \rhd [L^{-1}(v)/x]L_z^{-1}(t) \equiv L_z^{-1}([v/x]t)$ by (ii).

On the other hand, no term in V can be a β-redex. The Lemma then follows by a straightforward induction on reduction lengths, and when this length is 1, by induction on the depth at which the contracted redex occurs. We have just dealt with the base case, and the induction case is trivial. □