Mechanized Subject Expansion in Uniform Intersection Types for Perpetual Reductions

Andrej Dudenhefner
TU Dortmund University, Germany

Daniele Pautasso
University of Turin, Italy

Abstract

We provide a new, purely syntactical proof of strong normalization for the simply typed λ-calculus. The result relies on a novel proof of the equivalence between typability in the simple type system and typability in the uniform intersection type system (a restriction of the non-idempotent intersection type system). For formal verification, the equivalence is mechanized using the Coq proof assistant.

In the present work, strong normalization of a given simply typed term M is shown in four steps. First, M is reduced to a normal form N via a suitable reduction strategy with a decreasing measure. Second, a uniform intersection type for the normal form N is inferred. Third, a uniform intersection type for M is constructed iteratively via subject expansion. Fourth, strong normalization of M is shown by induction on the size of the type derivation.

A supplementary contribution is a family of perpetual reduction strategies, i.e. strategies which preserve infinite reduction paths. This family allows for subject expansion in the intersection type systems of interest, and contains a reduction strategy with a decreasing measure in the simple type system. A notable member of this family is Barendregt's F_∞ reduction strategy.

2012 ACM Subject Classification Theory of computation → Type theory

Keywords and phrases lambda-calculus, simple types, intersection types, strong normalization, mechanization, perpetual reductions

Digital Object Identifier 10.4230/LIPIcs.FSCD.2024.8

Supplementary Material
Software (Source Code): `https://github.com/tudo-seal/uniform-intersection`
archived at `swh:1:dir:724ed6c1181e635ae0d7992cbe344798d0e7457b`

Acknowledgements The authors are grateful to Simona Ronchi Della Rocca for many insightful discussions, and to the anonymous referees for their careful reading and suggestions.

1 Introduction

Simple Types and Strong Normalization. Strong normalization (SN) of the simply typed λ-calculus (STLC) is arguably one of the cornerstones of Type Theory. Many proofs of this fundamental property have been proposed during the past decades; some are achieved by semantical means, while others adopt syntactic (inductive) approaches. On the semantical side, many SN results for typed calculi, including Gödel's system T, system F, and STLC, are obtained using *reducibility models*: these are essentially variations of the classical method due to Tait [28], or of the subsequent refinements based on Girard's reducibility candidates [15, 29, 13]. Due to their general nature, such methods often do not take into account specificities of the problem at hand, which could entail more direct and instructive SN proofs.

An alternative line of work [20, 19, 25, 26, 32, 1] focuses instead on a fine-grained analysis of combinatorial properties of term rewriting and type assignment systems. Such syntactical approaches do not always scale to more expressive calculi, but they provide insights on the

reduction process, allowing for *decreasing measures* for typed terms. The proposal at hand is a further step in this direction: our goal is to provide an accessible inductive proof of SN for STLC, capitalizing on results about intersection type systems.

Uniform Intersection Types. Pioneered by Coppo and Dezani in the late '70s [5, 6, 7], *intersection type systems* can assign to terms more than one type: writing $M : A \cap B$ intuitively means that the term M is assigned an *intersection* of the types A and B. The intersection connective $\cap$ can be understood as a notation for a *set* of types; if the idempotency of $\cap$ is dropped, i.e. $A \cap A \neq A$, intersection becomes a notation for a *multiset* of types, yielding the so called *non-idempotent* intersection types (also known as *quantitative* types). As their name suggests, non-idempotent type systems have been extensively used to study quantitative properties of programs, such as the number of reduction steps needed to reach a normal form [14, 10, 4]. It is well known that intersection type systems characterize various forms of termination, i.e. a term can be assigned an intersection type if and only if it is (strongly) normalizing [24, 8]; consequently, the type inference problem for such systems is inherently undecidable. An exception to this general rule is obtained by considering *uniform* intersection types, a particular restriction of non-idempotent intersection types; indeed, it is possible to design a *decidable* system assigning a quantitative type to all and only the simply typable terms [23]. The correspondence between the simple and uniform systems was originally established via a type inference algorithm for the uniform case: however, even if the procedure shares many similarities with the classical unification algorithm, the proof of termination in the quantitative setting relies precisely on strong normalization of simply typed terms. This makes such an approach not viable for the purposes of the present work.

Related work and contributions of this paper. Even when restricted to work adopting a syntax-oriented viewpoint, the literature about SN of STLC is quite extensive. We cannot hope to provide a detailed account here: we just point out some of the most influential contributions exploring ways to infer strong normalization of one notion of reduction from weak normalization of a finer notion of reduction, possibly after performing a translation into a suitable calculus. Some seminal ideas can be tracked back to Nederpelt [20] and Klop [19]; their techniques can be directly applied to STLC, and have been (more or less implicitly) the starting point for a number of remarkable investigations [25, 26, 32]. One of the most delicate aspects in relating weak and strong normalization is the treatment of term-erasing reductions (see [16] for a study of the connections between different proof techniques and translations into the λI-calculus, where no erasure can occur): Nederpelt and Klop's idea is to retain the subterm that would have been discarded; another solution is to delay the erasing steps as much as possible. In this sense, of great inspiration has been the research carried out by Kfoury and Wells [18], from which we borrow the notion of γ-reduction (see also [17] for a brief comparison with other methods, most notably one by de Groote [11]). Lastly, we mention recent work [1], which suggests that the search for decreasing measures for STLC is far from over.

Our contributions to this line of research are three-fold. First, the present work shows that proving SN of STLC boils down to proving that all simply typable terms are uniformly typable. This correspondence, whose novel proof is of some interest in itself, greatly simplifies the arguments commonly used when going from weak to strong normalization. In particular, we do not need to show that normalization w.r.t. the newly introduced notions of reduction implies β-strong normalization: this result comes for free thanks to the properties of quantitative type systems.

The second contribution is the identification of a family of perpetual reduction strategies, that is, strategies that diverge whenever possible [31]. We do so by carefully designing reductions for which it is easy to obtain inductive proofs of subject expansion in intersection type systems. Remarkably, this family contains Barendregt's F_∞ strategy, which was used to show that all strongly normalizing terms can be typed by a *rigid intersection* (i.e. not enjoying associativity, commutativity nor idempotency) type system without nullary intersection [22]; our approach may also be understood as an extension of such a proof method.

Third, all new results we present are mechanically verified: indeed, being fully constructive, our technique is well-suited to be formalized in a proof assistant. Experience tells that the syntactical study of term rewriting and quantitative type systems, when done exclusively by hand, is particularly error-prone; we believe that, by getting rid of this eventuality, the mechanization constitutes a valuable addition.

Paper organization. The present work is structured as follows:
Section 2: Preliminaries on the λ-calculus, definition of the $I\gamma K'$-reduction.
Section 3: Measure-based weak $I\gamma K'$-normalization of simply typed terms (Theorem 15).
Section 4: Uniform intersection type system (Definition 21), uniform typability of β-normal forms (Lemma 24), subject expansion for the $I\gamma K'$-reduction (Lemma 27, Lemma 30, and Lemma 31), and consequently, strong normalization of simply typed terms (Theorem 34).
Section 5: Family of perpetual reduction strategies (Corollary 40) via generalization of subject expansion properties to the non-idempotent intersection type system (Theorem 39).
Section 6: Overview over the mechanization in the Coq proof assistant.
Section 7: Concluding remarks.

2 Preliminaries on Calculus and Reductions

Let us fix the basic notation for the remainder of the paper (following standard literature [2]). *Terms* of the λ-*calculus* are generated by the following grammar:

$$M, N \ ::= \ x \ \mid \ \lambda x.M \ \mid \ MN$$

where x ranges over a countable set of term variables. $\mathsf{FV}(M)$ denotes the set of free variables of the term M.

The β-*reduction*, denoted $\to_\beta$, is the contextual closure of the rule

$$(\lambda x.M)N \mapsto_\beta M[N/x]$$

where $M[N/x]$ denotes the capture-free substitution of x by N in M. A term of shape $(\lambda x.M)N$ is called a β-redex. Such β-redexes are partitioned into I-redexes and K-redexes, depending on whether the variable x occurs free in M or not; this distinction is of central importance in the subsequent sections. Formally, the reductions $\to_I$ and $\to_K$ are, respectively, the contextual closure of the rules:

$$(\lambda x.M)N \mapsto_I M[N/x] \text{ if } x \in \mathsf{FV}(M) \qquad (\lambda x.M)N \mapsto_K M \text{ if } x \notin \mathsf{FV}(M)$$

Given a binary relation $\to_r$, we denote its reflexive, transitive closure with $\to_r^*$. Given two binary relations $\to_{r_1}$ and $\to_{r_2}$, we write $\to_{r_1 r_2}$ for $\to_{r_1} \cup \to_{r_2}$; clearly $\to_\beta = \to_{IK} = \to_I \cup \to_K$. A term is in r-*normal form* when it does not contain any r-redex; it is (weakly) r-*normalizing* if it can be reduced to a term in r-normal form by a r-reduction sequence; it is *strongly* r-*normalizing* if every r-reduction sequence starting from it eventually stops.

We introduce the following reduction $\to_{K'}$ (Definition 1) as a refinement of the K-reduction. The reduction $\to_{K'}$ can only contract K-redexes in specific positions, motivated by two goals. First, K'-expansion should preserve strong normalization, allowing for subject expansion in suitable intersection type systems. Second, a term in IK'-normal form should be β-normal, otherwise IK'-reduction would get stuck on β-reducible terms.

▶ **Definition 1** ($\to_{K'}$).
1. *If N is in β-normal form and $x \notin \mathsf{FV}(M)$, then $(\lambda x.M)N \to_{K'} M$.*
2. *If $M \to_{K'} N$, then $\lambda x.M \to_{K'} \lambda x.N$.*
3. *If $N_1 \to_{K'} N_2$, then $xM_1 \dots M_n N_1 \to_{K'} xM_1 \dots M_n N_2$, where $n \geq 0$.*
4. *If $N_1 \to_{K'} N_2$ and $x \notin \mathsf{FV}(M)$, then $(\lambda x.M)N_1 \to_{K'} (\lambda x.M)N_2$.*
5. *If $M_1 M_2 \to_{K'} M_3$, then $(M_1 M_2)N \to_{K'} M_3 N$.*

The following Example 2, Remark 3, and Remark 4 give insight into the relationship between $\to_{K'}$ and perpetual reduction strategies.

▶ **Example 2.** Consider the term $\omega = \lambda z.zz$ and the term $M = (\lambda x.(\lambda y.x)(xx))\omega$, which is not strongly β-normalizing because of the sequence $M \to_\beta (\lambda y.\omega)(\omega\omega) \to_\beta (\lambda y.\omega)(\omega\omega) \to_\beta \dots$ The term M K-reduces to $(\lambda x.x)\omega$ for which the infinite β-reduction sequence is lost. However, the term M cannot be K'-reduced.

▶ Remark 3. By proving subject expansion for K'-reduction in suitable intersection type systems, we obtain that any IK'-reduction strategy is perpetual, thanks to the characterization of strongly normalizing terms via typability (see for example [2, Theorem 17.2.15] for the idempotent case, and [4, Corollary 8.4] for the non-idempotent one).

▶ Remark 4. The IK'-reduction admits the F_∞ perpetual reduction strategy [31, Definition 3.21].

The following Lemma 5 shows that $\to_{IK'}$ cannot get stuck on $\to_\beta$-reducible terms.

▶ **Lemma 5.** *If $M \to_\beta N$ then there exists N' such that $M \to_{IK'} N'$.*

Proof. If M contains at least one I-redex, the result is immediate. Otherwise, M can contain K-redexes only, and we proceed by induction on M. Case $M = x$ vacuously holds, while case $M = \lambda x.P$ follows by inductive hypothesis and point (2) of Definition 1. Lastly, consider the case $M = PQ$ is an application.

- If $P = x$ then by inductive hypothesis and point (3) of Definition 1 we can reduce Q.
- If $P = \lambda x.S$ then PQ is a K-redex and we distinguish two subcases:
  - if Q is in normal form, by point (1) of Definition 1 we can reduce PQ;
  - if Q is not in normal form, by inductive hypothesis and point (4) of Definition 1 we can reduce Q.
- If $P = S_1 S_2$ we distinguish two subcases:
  - if P is in normal form, then Q is not in normal form; by inductive hypothesis and point (3) of Definition 1 we can reduce Q;
  - if P is not in normal form, by inductive hypothesis and point (5) of Definition 1 we can reduce P. ◀

The last essential ingredient of the present work is the commutation rule

$$(\lambda x.\lambda y.M)N \mapsto_\gamma \lambda y.(\lambda x.M)N,$$

whose contextual closure we denote $\to_\gamma$ [18, Definition 3.1].

A γ-reduction step can be understood as the combination of one β-reduction step and one β-expansion step, rearranging the structure of a term, without altering its "meaning". The idea is that γ-reduction can be used to postpone K-redexes, possibly exposing I-redexes, as illustrated by the following Example 6.

▶ **Example 6.** Consider the term $M = (\lambda y.\lambda x.x)wz$. Using the IK'-reduction, the term M is reduced to a normal form as follows: $M \to_{K'} (\lambda x.x)z \to_I z$. Using γ-reduction, the K'-redex can be postponed: $M \to_\gamma (\lambda x.(\lambda y.x)w)z \to_I (\lambda y.z)w \to_{K'} z$.

3 Simple Types and a Decreasing Measure

In this section we show that simply typed terms are $I\gamma K'$-normalizing. Similarly to the approach by Kfoury and Wells [18], we proceed in two steps. First, given a simply typed term we construct an $I\gamma$-normal form using a decreasing measure for rightmost $I\gamma$-redex contraction. The decreasing measure is different from the one by Kfoury and Wells [18, Lemma 4.3], and does not require a specific interleaving of I-reductions and γ-reductions (cf. $\star$-reduction [18, Definition 3.8]). Second, we iteratively contract K'-redexes in order to construct a β-normal form.

Let us briefly recollect the simple type assignment system.

▶ **Definition 7** (Simple Types). *The set $\mathcal{T}_{\mathcal{S}}$ of* simple types *is defined by the grammar $\sigma, \tau ::= a \mid \sigma \to \tau$, where a ranges over a countable set of type variables.*

A *type environment* is a finite, functional set of pairs $x : \sigma$, where x is a term variable and σ a simple type; environments are ranged over by $\Gamma, \Delta, \Phi, \Psi$. If $x : \sigma \in \Gamma$, then $\Gamma(x) = \sigma$; the domain of an environment Γ is $\mathsf{dom}(\Gamma) = \{x \mid x : A \in \Gamma\}$; Γ and Δ *agree*, written $\Gamma \smile \Delta$, if $\Gamma(x) = \Delta(x)$ for all $x \in \mathsf{dom}(\Gamma) \cap \mathsf{dom}(\Delta)$. The writing Γ, Δ is short for $\Gamma \cup \Delta$ in case $\mathsf{dom}(\Gamma) \cap \mathsf{dom}(\Delta) = \emptyset$; the writing $\Gamma, x : \sigma$ is a special case of such notation when $x \notin \mathsf{dom}(\Gamma)$.

▶ **Definition 8** (Simple Type Assignment System). *The simple type assignment system $\mathcal{S}$ derives judgments of shape $\Gamma \vdash M : \sigma$, where Γ is an environment, M is a term, and σ is a simple type. The rules of $\mathcal{S}$ are as follows:*

$$\frac{}{\Gamma, x : \sigma \vdash x : \sigma}\ (\mathsf{var}) \qquad \frac{\Gamma, x : \sigma \vdash M : \tau}{\Gamma \vdash \lambda x.M : \sigma \to \tau}\ (\to_{\mathsf{I}})$$

$$\frac{\Gamma \vdash M : \sigma \to \tau \quad \Delta \vdash N : \sigma \quad \Gamma \smile \Delta}{\Gamma \cup \Delta \vdash MN : \tau}\ (\to_{\mathsf{E}})$$

Type derivations are ranged over by Π, Σ, Θ. We often write $\Gamma \vdash M : \sigma$ as a shorthand for the existence of a derivation proving $\Gamma \vdash M : \sigma$, and when we want to name a particular derivation with such conclusion we write $\Pi \triangleright \Gamma \vdash M : \sigma$. Additionally, each subterm N of M is associated with exactly one judgment $\Delta \vdash N : \tau$ in Π, and we may annotate the assigned simple type τ onto N, by writing N^τ.

▶ **Lemma 9.** *If $\Gamma \vdash M : \sigma$ and $M \to_{I\gamma K'} N$, then $\Gamma \vdash N : \sigma$.*

Proof. For $\to_{IK'} \subseteq \to_\beta$ the property follows by subject reduction ([2, Proposition 1.2.6]). For $\to_\gamma$ the property follows from the generation lemma ([2, Proposition 1.2.3]). ◀

The *measure* $\mathsf{meas}(M)$ of a simply typed term M is a multiset of pairs (m, n) of natural numbers, where m carries information about the size of involved types, and n is the depth of the redex w.r.t. abstraction. Multisets are written in square brackets, and multiset union is denoted $\uplus$, taking multiplicities into account.

Each pair in the multiset $\mathsf{meas}(M)$ is associated with one $I\gamma$-redex in M, and is called *rank* of the redex. If a subterm $(\lambda x.\lambda y.M)^{\sigma_1\to\sigma_2\to\tau}N$ is both a γ-redex and an I-redex, then its rank is that of the I-redex. Correspondingly, we prefer I-redexes in case of ambiguity for the rightmost redex.

▶ **Definition 10** (Measure of a simply typed λ-term)**.**

- *The* size *of a simple type is defined as:*

$$\mathsf{size}(a) = 1 \qquad \mathsf{size}(\sigma\to\tau) = 1 + \mathsf{size}(\sigma) + \mathsf{size}(\tau)$$

- *The* measure *of a simply typed term with corresponding simple type annotations is:*

$$\begin{array}{lcll}
\mathsf{meas}(x) & = & [\,] & \\
\mathsf{meas}(\lambda x.M) & = & [(k, n+1) \mid (k,n) \in \mathsf{meas}(M)] & \\
\mathsf{meas}((\lambda x.M)^{\sigma\to\tau}N) & = & \mathsf{meas}(\lambda x.M) \uplus \mathsf{meas}(N) \uplus & \\
 & & [(\mathsf{size}(\sigma\to\tau), 0)] & \text{if } x \in \mathsf{FV}(M) \\
\mathsf{meas}((\lambda x.\lambda y.M)^{\sigma_1\to\sigma_2\to\tau}N) & = & \mathsf{meas}(\lambda x.\lambda y.M) \uplus \mathsf{meas}(N) \uplus & \\
 & & [(1 + \mathsf{size}(\sigma_2\to\tau), 0)] & \text{if } x \notin \mathsf{FV}(M) \\
\mathsf{meas}(MN) & = & \mathsf{meas}(M) \uplus \mathsf{meas}(N) & \text{otherwise}
\end{array}$$

Pairs are ordered lexicographically and multisets are ordered by the multiset ordering [12]. The following Lemma 11 shows that rightmost $I\gamma$-redex contraction is measure-decreasing.

▶ **Lemma 11.** *If M, N are simply typed terms such that $M \to_{I\gamma} N$ by contracting the rightmost (in the textual presentation) redex, then $\mathsf{meas}(N) < \mathsf{meas}(M)$ by the multiset order.*

Proof. Since we reduce the rightmost $I\gamma$-redex, no $I\gamma$-redex is duplicated. Therefore, it suffices to show that each new $I\gamma$-redex created by the reduction is of smaller rank than that of the contracted redex.

First, we consider cases in which the contraction of an I-redex (as the rightmost $I\gamma$-redex) may create new redexes [21].

- Case $(\lambda x.M)^{\sigma\to\tau}N \to_I M[N/x]$ and there is a subterm xP of M.
  - In case $N = \lambda y.Q$ and $y \in \mathsf{FV}(Q)$ we have that $\sigma = \sigma_1 \to \sigma_2$ for some σ_1, σ_2 and $\mathsf{size}(\sigma_1\to\sigma_2) < \mathsf{size}(\sigma\to\tau)$. Therefore, the created I-redex $(\lambda y.Q)^{\sigma_1\to\sigma_2}P$ is of smaller rank (regardless of its depth in M).
  - In case $N = \lambda y.\lambda z.Q$ and $y \notin \mathsf{FV}(Q)$ we have that $\sigma = \sigma_1 \to \sigma_2 \to \sigma_3$ for some $\sigma_1, \sigma_2, \sigma_3$ and $1 + \mathsf{size}(\sigma_2\to\sigma_3) < \mathsf{size}(\sigma\to\tau)$. Therefore, the created γ-redex $(\lambda y.\lambda z.Q)^{\sigma_1\to\sigma_2\to\sigma_3}P$ is of smaller rank.
- Case $(\lambda x.x)^{(\sigma\to\tau)\to(\sigma\to\tau)}NP \to_I NP$.
  - In case $N = \lambda y.Q$ and $y \in \mathsf{FV}(Q)$ the created I-redex $(\lambda y.Q)^{\sigma\to\tau}P$ is of smaller rank.
  - In case $N = \lambda y.\lambda z.Q$ and $y \notin \mathsf{FV}(Q)$ the created γ-redex $(\lambda y.\lambda z.Q)^{\sigma\to\tau}P$ is of smaller rank.
- Case $(\lambda x.\lambda y.M)^{\sigma_1\to\sigma_2\to\tau}NP \to_I (\lambda y.M[N/x])^{\sigma_2\to\tau}P$ and $y \in \mathsf{FV}(M)$.
  Since $\mathsf{size}(\sigma_2\to\tau) < \mathsf{size}(\sigma_1\to\sigma_2\to\tau)$, the created I-redex is of smaller rank.
- Case $(\lambda x.\lambda y.\lambda z.M)^{\sigma_1\to\sigma_2\to\sigma_3\to\tau}NP \to_I (\lambda y.\lambda z.M[N/x])^{\sigma_2\to\sigma_3\to\tau}P$ and $y \notin \mathsf{FV}(M)$.
  Since $1 + \mathsf{size}(\sigma_3\to\tau) < \mathsf{size}(\sigma_1\to\sigma_2\to\sigma_3\to\tau)$, the created γ-redex is of smaller rank.

Second, we consider cases in which the contraction of a γ-redex (as the rightmost $I\gamma$-redex) may create new redexes. The contracted γ-redex cannot be an (otherwise preferred) I-redex.

- Case $(\lambda x.\lambda y.\lambda z.M)^{\sigma_1\to\sigma_2\to\sigma_3\to\tau}N \to_\gamma \lambda y.(\lambda x.\lambda z.M)^{\sigma_1\to\sigma_3\to\tau}N$ such that $x \notin \mathsf{FV}(M)$.
  Since $1 + \mathsf{size}(\sigma_3\to\tau) < 1 + \mathsf{size}(\sigma_2\to\sigma_3\to\tau)$, the created γ-redex is of smaller rank.

- Case $(\lambda x.\lambda y.M)^{\sigma_1\to\sigma_2\to\tau}NP \to_\gamma (\lambda y.(\lambda x.M)N)^{\sigma_2\to\tau}P$ such that $x \notin \mathsf{FV}(M)$, $y \in \mathsf{FV}(M)$. Since $\mathsf{size}(\sigma_2 \to \tau) < 1 + \mathsf{size}(\sigma_2 \to \tau)$, the created I-redex is of smaller rank. The subterm $(\lambda x.M)N$ is not an I-redex because $x \notin \mathsf{FV}(M)$; the case in which $(\lambda x.M)N$ is a new γ-redex is already treated above.
- Case $(\lambda x.(\lambda y.\lambda z.M)^{\sigma_2\to\sigma_3\to\tau}N)P \to_\gamma (\lambda x.\lambda z.(\lambda y.M)N)^{\sigma_1\to\sigma_3\to\tau}P$ such that $y \notin \mathsf{FV}(M)$, $x \notin \mathsf{FV}(M) \cup \mathsf{FV}(N)$. The contracted γ-redex is of rank $(1+\mathsf{size}(\sigma_3 \to \tau), n+1)$ for some n, and the created γ-redex is of smaller rank $(1 + \mathsf{size}(\sigma_3 \to \tau), n)$. ◀

▶ Remark 12. The last case in the above proof of Lemma 11 explains the definition of the rank of a γ-redex. In particular, the rank of a γ-redex *does not* depend on the type of the first abstracted variable, and it *does* depend on the depth of the γ-redex w.r.t. abstraction.

The following Lemma 13 is a weaker variant of the strong normalization property of $I\gamma$-normal forms [18, Lemma 3.10].

▶ **Lemma 13.** *If M, N are terms such that $M \to_{K'} N$ by contracting the rightmost K'-redex and M is $I\gamma$-normal, then N is $I\gamma$-normal and N has fewer β-redexes than M.*

Proof. Consider the K'-redex $(\lambda x.P)Q$ in M. The term P cannot be an abstraction, otherwise $(\lambda x.P)Q$ would have been a γ-redex, contradicting the assumption that M is in $I\gamma$-normal form. Therefore, contracting $(\lambda x.P)Q$ to P introduces neither β-redexes nor γ-redexes. ◀

The combination of Lemma 11 and Lemma 13 provides a normalization strategy (Theorem 15) for simply typed terms, which is illustrated in the following Example 14.

▶ **Example 14.** Consider the annotated term $M = (\lambda y.\lambda x.x)^{b\to a\to a}wz$ from Example 6, which can be a assigned the type a in the type environment $\{z : a, w : b\}$. The term M with measure $\mathsf{meas}(M) = [(4, 0)]$ can be reduced to a normal form as follows:

$$(\lambda y.\lambda x.x)^{b\to a\to a}wz \quad \to_\gamma \quad (\lambda x.(\lambda y.x)^{b\to a}w)^{a\to a}z \quad \to_I \quad (\lambda y.z)^{b\to a}w \quad \to_{K'} \quad z$$

The initial γ-reduction postpones the top-level K-redex, exposes an I-redex, and decreases the measure to $\mathsf{meas}((\lambda x.(\lambda y.x)^{b\to a}w)^{a\to a}z) = [(3, 0)]$. The subsequent I-reduction leaves only K-redexes and the measure decreases to $\mathsf{meas}((\lambda y.z)^{b\to a}w) = [\,]$. Finally, the term is normalized using the K'-reduction.

▶ **Theorem 15.** *Given a simply typed term M, there exists a β-normal form N such that $M \to^*_{I\gamma K'} N$.*

Proof. By induction on $\mathsf{meas}(M)$, repeatedly contracting the rightmost $I\gamma$-redex we obtain a $I\gamma$-normal form P by Lemma 11. By induction on the number of K-redexes in P, repeatedly contracting the rightmost K'-redex (we cannot get stuck by Lemma 5) we obtain a β-normal form by Lemma 13. ◀

▶ Remark 16. Of course, we can show Theorem 15 using a strong normalization argument for the simply typed λ-calculus together with Lemma 5. However, it is methodologically intriguing to utilize measure-based weak normalization in pursuit of typability in a non-idempotent intersection type system (which constitutes a strong normalization proof).

▶ Remark 17. The advantage of the additional γ-reduction is apparent in the design of a type-based decreasing measure for IK'-normalization. Consider the term $(\lambda x.uxx)^{\sigma_1\to\tau_1}(vM)$ where $M = (\lambda y.\lambda z.N)^{\sigma_2\to\sigma_2\to\tau_2}ww$ such that $y \notin \mathsf{FV}(N)$ and $z \in \mathsf{FV}(N)$. Since $x \in \mathsf{FV}(uxx)$ we cannot contract the K-redex occurring in M using K'-reduction. However, contracting the I-redex $(\lambda x.uxx)^{\sigma_1\to\tau_1}(vM) \to_I u(vM)(vM)$ duplicates M. Contracting each redex

copy in M results in two new I-redexes $(\lambda z.N)^{\sigma_2 \to \tau_2} w$. In sum, contracting an I-redex with the associated type $\sigma_1 \to \tau_1$ results in two copies of an I-redex with the associated (arbitrary large) type $\sigma_2 \to \tau_2$. For reference, the F_∞ [31, Definition 3.21] perpetual reduction strategy also involves the described duplication. In comparison, by γ-reduction $(\lambda y.\lambda z.N)ww \to_\gamma (\lambda z.(\lambda y.N)w)w$ the K-redex $(\lambda y.N)w$ is delayed and the I-redex is exposed for contraction (without duplication).

4 Uniform Intersection Types

As previously mentioned, uniform intersection types are a restriction of non-idempotent intersection types based on the notion of uniform multiset. From now on we will frequently use indexed types, where indexes are natural numbers: the symbols I, J will denote sets of indexes. For the sake of simplicity we adopt the same notation for types and uniform types (resp. multisets and uniform multisets), as the intended meaning can be easily inferred from the context.

▶ **Definition 18.**

- *Non-idempotent intersection types ($\mathcal{T}_\mathcal{I}$) are inductively defined by the grammar:*

$$\begin{array}{llll} \textsc{Intersection Types} & A, B, C & ::= & a \mid \mu \to A \\ \textsc{Multisets} & \mu, \nu & ::= & [A_1, \dots, A_n] \quad (n \geq 1) \end{array}$$

- *Equivalence relation $\sim$ on intersection types (uniformity):*

$$a \sim a \text{ for all type variables } a$$
$$\mu \to A \sim \nu \to B \text{ iff } \mu \sim \nu \text{ and } A \sim B$$
$$[A_i]_{i \in I} \sim [B_j]_{j \in J} \text{ iff } A_i \sim B_j \text{ for all } i \in I, j \in J$$

- *Uniform intersection types ($\mathcal{T}_\mathcal{U}$) are inductively defined by the grammar:*

$$\begin{array}{llll} \textsc{Unif. Int. Types} & A, B, C & ::= & a \mid \mu \to A \\ \textsc{Unif. Multisets} & \mu, \nu & ::= & [A_1, \dots, A_n] \quad \forall i, j \in \{1, \dots, n\}. A_i \sim A_j \end{array}$$

Remark that in both grammars the empty multiset is *not* allowed. The types $A = [[a, a, a] \to b, [a] \to b] \to c$ and $B = [[a, a] \to b] \to c$ are uniform, whereas $[a, [a] \to b] \to c$ is not; moreover, observe that $A \sim B$. The intuition is that uniform types are the quantitative version of simple types; two uniform types are equivalent if they correspond to the same underlying simple type. Given a uniform type, the underlying simple type can be easily recovered by means of a translation that "forgets" non-idempotency.

▶ **Definition 19.** *The* collapse translation $\mathsf{c} : \mathcal{T}_\mathcal{I} \to \mathcal{T}_\mathcal{S}$ *is a partial function recursively defined as:*

$$\begin{array}{lcll} \mathsf{c}(a) & = & a & \\ \mathsf{c}([A_1, \dots, A_n] \to B) & = & \sigma \to \mathsf{c}(B) & \text{if } \mathsf{c}(A_1) = \dots = \mathsf{c}(A_n) = \sigma \end{array}$$

▶ **Lemma 20.**

- *The collapse translation $\mathsf{c}(\cdot)$ is a total function on $\mathcal{T}_\mathcal{U}$.*
- *If $A, B \in \mathcal{T}_\mathcal{U}$, then $A \sim B$ if and only if $\mathsf{c}(A) = \mathsf{c}(B)$.*

Proof. The two points are proved at the same time, by mutual induction on the structure of uniform types and the definition of collapse translation. ◀

We are now ready to introduce the uniform intersection type assignment system $\mathcal{U}$. In this setting, we say that a type environment Γ is *uniform* if it associates each term variable to a *uniform multiset*. If Γ and Δ are two such environments, $\Gamma \sim \Delta$ means that $\Gamma(x) \sim \Delta(x)$ for all $x \in \mathsf{dom}(\Gamma) \cap \mathsf{dom}(\Delta)$. The union of environments is defined as $(\Gamma_0 \uplus \Gamma_1)(x) = \Gamma_0(x) \uplus \Gamma_1(x)$ if $x \in \mathsf{dom}(\Gamma_0) \cap \mathsf{dom}(\Gamma_1)$, while $(\Gamma_0 \uplus \Gamma_1)(x) = \Gamma_i(x)$ if $x \in \mathsf{dom}(\Gamma_i)$ and $x \notin \mathsf{dom}(\Gamma_{1-i})$.

▶ **Definition 21.** *The uniform intersection type assignment system $\mathcal{U}$, assigning types in $\mathcal{T}_{\mathcal{U}} \subset \mathcal{T}_{\mathcal{I}}$ to terms, consists of the following rules:*

$$\frac{A \in \mu \quad \Gamma \text{ and } \mu \text{ uniform}}{\Gamma, x : \mu \vdash_{\mathtt{u}} x : A}\ (\mathsf{var}) \qquad \frac{\Gamma, x : \mu \vdash_{\mathtt{u}} M : A}{\Gamma \vdash_{\mathtt{u}} \lambda x.M : \mu \to A}\ (\to_{\mathsf{I}})$$

$$\frac{\Gamma_0 \vdash_{\mathtt{u}} M : [A_1, \dots, A_n] \to B \quad (\Gamma_i \vdash_{\mathtt{u}} N : A_i)_{1 \le i \le n} \quad \forall i, j \in \{0, \dots, n\}.\Gamma_i \sim \Gamma_j}{\biguplus_{i=0}^{n} \Gamma_i \vdash_{\mathtt{u}} MN : B}\ (\to_{\mathsf{E}})$$

The full non-idempotent intersection type assignment system, which we call system $\mathcal{I}$, is easily obtained from system $\mathcal{U}$ by removing the uniformity constraint on multisets. It is clear that each derivation in system $\mathcal{U}$ is also a valid derivation in system $\mathcal{I}$; we use the symbol $\vdash_{\mathtt{i}}$ to explicitly distinguish judgments in system $\mathcal{I}$.

The collapse translation is naturally extended to uniform multisets and uniform type environments, so that the translation of a derivation in system $\mathcal{U}$ is a derivation in system $\mathcal{S}$.

▶ **Definition 22.** *The collapse translation $\mathsf{c}(\Pi)$ of a derivation $\Pi \rhd \Gamma \vdash_{\mathtt{u}} M : A$ is the simple type derivation inductively defined as follows:*

- *If Π ends with a (var) rule, i.e. $\Pi \rhd \Gamma \vdash_{\mathtt{u}} x : A$, then $\mathsf{c}(\Pi) \rhd \mathsf{c}(\Gamma) \vdash M : \mathsf{c}(A)$.*
- *If Π ends with a $(\to_{\mathsf{I}})$ rule, i.e. has shape:*

$$\frac{\Pi_0 \rhd \Gamma, x : \mu \vdash_{\mathtt{u}} N : B}{\Pi \rhd \Gamma \vdash_{\mathtt{u}} \lambda x.N : A = \mu \to B} \quad \text{then} \quad \frac{\mathsf{c}(\Pi_0) \rhd \mathsf{c}(\Gamma), x : \mathsf{c}(\mu) \vdash N : \mathsf{c}(B)}{\mathsf{c}(\Pi) \rhd \mathsf{c}(\Gamma) \vdash \lambda x.N : \mathsf{c}(A) = \mathsf{c}(\mu) \to \mathsf{c}(B)}$$

- *If Π ends with a $(\to_{\mathsf{E}})$ rule, i.e. has shape:*

$$\frac{\Pi_0 \rhd \Gamma_0 \vdash_{\mathtt{u}} P : [B_1, \dots, B_n] \to A \qquad (\Pi_i \rhd \Gamma_i \vdash_{\mathtt{u}} Q : B_i)_{1 \le i \le n}}{\Pi \rhd \Gamma \vdash_{\mathtt{u}} PQ : A}$$

then, letting $\mathsf{c}(\Pi_i) \rhd \mathsf{c}(\Gamma_i) \vdash_{\mathtt{u}} Q : \mathsf{c}(B_i)$ and recalling that $\mathsf{c}(B_i) = \tau$ for all $1 \le i \le n$:

$$\frac{\mathsf{c}(\Pi_0) \rhd \mathsf{c}(\Gamma_0) \vdash P : \tau \to \mathsf{c}(A) \qquad (\bigcup_{i=1}^{n} \mathsf{c}(\Gamma_i)) \vdash Q : \tau}{\mathsf{c}(\Pi) \rhd \mathsf{c}(\Gamma) \vdash PQ : \mathsf{c}(A)} \qquad \text{where } \mathsf{c}(\Gamma) = \bigcup_{i=0}^{n} \mathsf{c}(\Gamma_i)$$

▶ **Lemma 23** ([23, Theorem 34])**.** *$\Pi \rhd \Gamma \vdash_{\mathtt{u}} M : A$ implies $\mathsf{c}(\Pi) \rhd \mathsf{c}(\Gamma) \vdash M : \mathsf{c}(A)$.*

Showing the converse, namely that all simply typable terms can also be assigned a uniform intersection type, is not as easy. In what follows we provide an alternative proof of this claim, adopting a dual approach w.r.t. previous work [23]: instead of reasoning about term reduction, we reason about term expansion. Additionally, strong normalization is a consequence in our case, and not a prerequisite [23, Theorem 25].

4.1 Uniform Typability of Normal Forms

As a first step, we show that system $\mathcal{U}$ can assign a uniform type to all simply typable terms in β-normal form.

▶ **Lemma 24.** *If $\Sigma \triangleright \Gamma \vdash M : \sigma$ and M is in β-normal form, then there exists $\Pi \triangleright \Gamma' \vdash_{\mathtt{u}} M : A$ such that $\mathsf{c}(\Pi) = \Sigma$.*

Proof. By induction on the term M. Recall that β-normal forms are defined by the grammar: $M, N ::= \lambda x.M \mid xM_1 \dots M_n$ where $n \geq 0$.

Case $M = x$ is immediate, and case $M = \lambda x.N$ follows by inductive hypothesis. Lastly, consider the case $M = xM_1 \dots M_n$ such that $n \geq 1$ and we have $\Sigma \triangleright \bigcup_{i=0}^{n} \Gamma_i \vdash xM_1 \dots M_n : \sigma$. Then there exist $\Sigma_0 \triangleright \Gamma_0 \vdash x : \tau_1 \to \dots \to \tau_n \to \sigma$ and $\Sigma_i \triangleright \Gamma_i \vdash M_i : \tau_i$ $(1 \leq i \leq n)$ such that $\Gamma_i \smile \Gamma_j$ for all $i, j \in \{0, \dots, n\}$. By inductive hypothesis there are $\Pi_i \triangleright \Gamma'_i \vdash_{\mathtt{u}} M_i : B_i$ such that $\mathsf{c}(\Pi_i) = \Sigma_i$, and consequently $\mathsf{c}(\Gamma'_i) = \Gamma_i$ and $\mathsf{c}(B_i) = \tau_i$ $(1 \leq i \leq n)$. Moreover, it is easy to build $\Pi_0 \triangleright \Gamma'_0 \vdash_{\mathtt{u}} x : [B_1] \to \dots \to [B_n] \to A$ such that $\mathsf{c}(\Gamma'_0) = \Gamma_0$ and $\mathsf{c}(A) = \sigma$, thus satisfying $\mathsf{c}(\Pi_0) = \Sigma_0$. Remark that

$$\mathsf{c}([B_1] \to \dots \to [B_n] \to A) = \mathsf{c}(B_1) \to \dots \to \mathsf{c}(B_n) \to \mathsf{c}(A) = \tau_1 \to \dots \to \tau_n \to \sigma$$

Since $\mathsf{c}(\Gamma'_i) = \Gamma_i$ and $\Gamma_i \smile \Gamma_j$, we know that $\Gamma'_i(y) \sim \Gamma'_j(y)$ for all $y \in \mathsf{dom}(\Gamma'_i) \cap \mathsf{dom}(\Gamma'_j)$ $(i, j \in \{0, \dots, n\})$. Therefore, we can use the various Π_i $(0 \leq i \leq n)$ to build a derivation $\Pi \triangleright \biguplus_{i=0}^{n} \Gamma'_i \vdash_{\mathtt{u}} xM_1 \dots M_n : A$ such that $\mathsf{c}(\Pi) = \Sigma$. ◀

4.2 Typability-preserving Expansions

Now that we know that simply typable β-normal forms are uniformly typable, the crucial step is showing that (simply typed) subject expansion w.r.t. $\to_I$, $\to_\gamma$ and $\to_{K'}$ preserves typability in system $\mathcal{U}$. The fact that I-expansion preserves typability in system $\mathcal{I}$ is folklore; here we need to specialize the result to the particular case of system $\mathcal{U}$.

▶ **Notation 25.** *Given $\Pi \triangleright \Gamma \vdash_{\mathtt{u}} M : A$ and $\Delta \sim \Gamma$, we write $\Pi^{(\Delta)} \triangleright \Gamma \uplus \Delta \vdash_{\mathtt{u}} M : A$ for the derivation obtained from Π by weakening.*

▶ **Notation 26.** *Let $\Sigma \triangleright \Gamma \vdash M : \sigma$. If $M \to_\beta N$, we write $\Sigma \rightsquigarrow \Sigma' \triangleright \Gamma \vdash N : \sigma$ meaning that Σ' is obtained from Σ by mimicking the β-reduction on the simple type derivation.*[1]

▶ **Lemma 27.** *Let $\Pi_N \triangleright \Gamma \vdash_{\mathtt{u}} N : A$. If $M \to_I N$ and there is $\Sigma \triangleright \Phi \vdash M : \sigma$ such that $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)}) \triangleright \Phi \vdash N : \sigma$ for some $\Delta \sim \Gamma$, then $\Gamma \vdash_{\mathtt{u}} M : A$.*

Proof. By induction on the reduction context. Remark that $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)}) \triangleright \mathsf{c}(\Gamma \uplus \Delta) = \Phi \vdash N : \mathsf{c}(A) = \sigma$ for some $\Delta \sim \Gamma$ implies $\mathsf{c}(\Gamma) \subseteq \Phi$. For the base case, let $M = (\lambda x.P)Q \to_I P[Q/x] = N$ and, wlog, assume $x \notin \mathsf{FV}(Q)$. We show how to build $\Pi_M \triangleright \Gamma \vdash_{\mathtt{u}} M : A$ starting from Π_N. The derivation Π_N contains a finite number of subderivations with subject Q: let them be $\Theta_i \triangleright \Gamma_i \vdash_{\mathtt{u}} Q : B_i$ $(i \in I = \{1, \dots, n\})$, and let $\Gamma = \Gamma_0 \uplus_{i \in I} \Gamma_i$. Since $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$, it must be the case that $\mathsf{c}(B_i) = \mathsf{c}(B_j)$ for all $i, j \in I$. By Lemma 20 this implies $B_i \sim B_j$ for all $i, j \in I$, hence the multiset $[B_i]_{i \in I}$ is uniform. Substituting in Π_N each subderivation Θ_i with an axiom $x : [B_i] \vdash_{\mathtt{u}} x : B_i$ yields a derivation with conclusion $\Gamma_0, x : [B_i]_{i \in I} \vdash_{\mathtt{u}} P : A$. Therefore we can build:

$$\dfrac{\dfrac{\Gamma_0, x : [B_i]_{i \in I} \vdash_{\mathtt{u}} P : A}{\Gamma_0 \vdash_{\mathtt{u}} \lambda x.P : [B_i]_{i \in I} \to A}\,(\to_{\mathtt{I}}) \qquad (\Gamma_i \vdash_{\mathtt{u}} Q : B_i)_{i \in I}}{\Pi_M \triangleright \Gamma \vdash_{\mathtt{u}} (\lambda x.P)Q : A}\,(\to_{\mathtt{E}})$$

For the inductive step, consider the contexts in which a reduction may take place:

[1] That is: given a subderivation typing $(\lambda x.P)Q$, substitute the axioms typing x in the subderivation for P by the subderivation typing Q, so to obtain a subderivation with subject $P[Q/x]$.

- Case $M = \lambda x.P \to_I \lambda x.Q = N$. Letting $\sigma = \tau_1 \to \tau_2$ and $A = \mu \to B$, the derivations Σ and Π_N have shape:

$$\frac{\Sigma_0 \rhd \Phi, x : \tau_1 \vdash P : \tau_2}{\Sigma \rhd \Phi \vdash \lambda x.P : \tau_1 \to \tau_2}\,(\to_{\mathsf{I}}) \qquad \frac{\Pi_0 \rhd \Gamma, x : \mu \vdash_{\mathtt{u}} Q : B}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} \lambda x.Q : \mu \to B}\,(\to_{\mathsf{I}})$$

  $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ means that $\Sigma_0 \rightsquigarrow \mathsf{c}(\Pi_0^{(\Delta)})$. Therefore by inductive hypothesis there exists $\Gamma, x : \mu \vdash_{\mathtt{u}} P : B$, from which one obtains $\Pi_M \rhd \Gamma \vdash_{\mathtt{u}} \lambda x.P : \mu \to B$.
- Case $M = PQ$ is an application. Let Σ be:

$$\frac{\Sigma_1 \rhd \Phi_1 \vdash P : \tau \to \sigma \qquad \Sigma_2 \rhd \Phi_2 \vdash Q : \tau}{\Sigma \rhd \Phi \vdash PQ : \sigma}\,(\to_{\mathsf{E}})$$

  First, consider the case $M = PQ \to_I PR = N$. Letting $I = \{1, \dots, n\}$, the derivation Π_N has shape:

$$\frac{\Pi_0 \rhd \Gamma_0 \vdash_{\mathtt{u}} P : [B_i]_{i \in I} \to A \qquad (\Pi_i \rhd \Gamma_i \vdash_{\mathtt{u}} R : B_i)_{i \in I}}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} PR : A}\,(\to_{\mathsf{E}})$$

  $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ means that, for each $i \in I$, $\Sigma_2 \rightsquigarrow \mathsf{c}(\Pi_i^{(\Delta_i)})$ for some $\Delta_i \sim \Gamma_i$. Therefore by inductive hypothesis there exist $\Gamma_i \vdash_{\mathtt{u}} Q : B_i$ $(i \in I)$, and we conclude.
  Now consider the case $M = PQ \to_{K'} RQ = N$. The derivation Π_N has shape:

$$\frac{\Pi_0 \rhd \Gamma_0 \vdash_{\mathtt{u}} R : [B_i]_{i \in I} \to A \qquad (\Pi_i \rhd \Gamma_i \vdash_{\mathtt{u}} Q : B_i)_{i \in I}}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} RQ : A}\,(\to_{\mathsf{E}})$$

  Similarly to the previous scenario, $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ implies $\Sigma_1 \rightsquigarrow \mathsf{c}(\Pi_0^{(\Delta_0)})$ for some $\Delta_0 \sim \Gamma_0$. Therefore by inductive hypothesis there is $\Gamma_0 \vdash_{\mathtt{u}} P : [B_i]_{i \in I} \to A$, and we conclude. ◀

The following Example 28 illustrates subject expansion w.r.t. the I-reduction.

▶ **Example 28.** Consider $M = (\lambda x.yxx)P \to_I yPP = N$. Assuming N is uniformly typable, let Π_N be the uniform type derivation:

$$\dfrac{\dfrac{\Gamma_0 \vdash_{\mathtt{u}} y : [B_1] \to [B_2] \to A \qquad \Gamma_1 \vdash_{\mathtt{u}} P : B_1}{\Gamma_0 \uplus \Gamma_1 \vdash_{\mathtt{u}} yP : [B_2] \to A} \qquad \Gamma_2 \vdash_{\mathtt{u}} P : B_2}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} yPP : A}$$

Its collapse translation is the simple type derivation $\mathsf{c}(\Pi_N)$:

$$\dfrac{\dfrac{\mathsf{c}(\Gamma_0) \vdash y : \mathsf{c}(B_1) \to \mathsf{c}(B_2) \to \mathsf{c}(A) \qquad \mathsf{c}(\Gamma_1) \vdash P : \mathsf{c}(B_1)}{\mathsf{c}(\Gamma_0 \uplus \Gamma_1) \vdash yP : \mathsf{c}(B_2) \to \mathsf{c}(A)} \qquad \mathsf{c}(\Gamma_2) \vdash P : \mathsf{c}(B_2)}{\mathsf{c}(\Pi_N) \rhd \mathsf{c}(\Gamma) \vdash yPP : \mathsf{c}(A)}$$

Saying that there is $\Sigma \rhd \Phi \vdash M : \sigma$ such that $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N)$, implies that Σ has shape:

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathsf{c}(\Gamma_0) \vdash y : \tau \to \tau \to \mathsf{c}(A) \qquad \mathsf{c}(\Gamma_1), x : \tau \vdash x : \tau}{\mathsf{c}(\Gamma_0 \uplus \Gamma_1), x : \tau \vdash yx : \tau \to \mathsf{c}(A)} \qquad \mathsf{c}(\Gamma_2), x : \tau \vdash x : \tau}{\mathsf{c}(\Gamma), x : \tau \vdash yxx : \mathsf{c}(A)}}{\mathsf{c}(\Gamma) \vdash \lambda x.yxx : \tau \to \mathsf{c}(A)} \qquad \mathsf{c}(\Gamma_1) \cap \mathsf{c}(\Gamma_2) \vdash P : \tau}{\Sigma \rhd \mathsf{c}(\Gamma) \vdash (\lambda x.yxx)P : \mathsf{c}(A)}$$

where $\mathsf{c}(\Gamma) = \Phi$, $\mathsf{c}(A) = \sigma$, and $\mathsf{c}(B_1) = \mathsf{c}(B_2) = \tau$. In turn, this entails $B_1 \sim B_2$; therefore the multiset $[B_1, B_2]$ is uniform, and we can build the uniform derivation Π_M as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_0 \vdash_{\mathtt{u}} y : [B_1] \to [B_2] \to A \qquad x : [B_1] \vdash_{\mathtt{u}} x : B_1}{\Gamma_0, x : [B_1] \vdash_{\mathtt{u}} yx : [B_2] \to A} \qquad x : [B_2] \vdash_{\mathtt{u}} x : B_2}{\Gamma_0, x : [B_1, B_2] \vdash_{\mathtt{u}} yxx : A}}{\Gamma_0 \vdash_{\mathtt{u}} \lambda x.yxx : [B_1, B_2] \to A} \qquad \Gamma_1 \vdash_{\mathtt{u}} P : B_1 \qquad \Gamma_2 \vdash_{\mathtt{u}} P : B_2}{\Pi_M \rhd \Gamma \vdash_{\mathtt{u}} (\lambda x.yxx)P : A}$$

It might not be obvious why one needs the condition $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ in the statement of Lemma 27 (and, similarly, in the later Lemma 31). Given $\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} N : A$ and a term M such that $M \to_I N$, the reader may wonder if a weaker hypothesis, e.g. only requiring $\Sigma \rhd \mathsf{c}(\Gamma \uplus \Delta) \vdash M : \mathsf{c}(A)$, would suffice to prove $\Pi_M \rhd \Gamma \vdash_{\mathtt{u}} M : A$. The following Example 29 shows that such a formulation would not work in the inductive case, specifically when N is an application: in order to use the inductive hypothesis, one must relate the structures of Σ and Π_N.

▶ **Example 29.** Let $\Delta = \emptyset$ and consider a closed, simply typable term P such that $P \to_I Q$ (for instance, $P = (II)I \to_I II = Q$ where $I = \lambda x.x$). For $N = (\lambda z.y)Q$ let Π_N be the uniform derivation:

$$\dfrac{\dfrac{z : [B], y : [A] \vdash_{\mathtt{u}} y : A}{y : [A] \vdash_{\mathtt{u}} \lambda z.y : [B] \to A} \qquad \Pi_Q \rhd \vdash_{\mathtt{u}} Q : B}{\Pi_N \rhd y : [A] \vdash_{\mathtt{u}} (\lambda z.y)Q : A}$$

Assume there is $\Sigma \rhd y : \mathsf{c}(A) \vdash (\lambda z.y)P : \mathsf{c}(A)$. We would like to exploit the information about Σ to build a derivation $\Pi_M \rhd y : [A] \vdash_{\mathtt{u}} (\lambda z.y)P : A$; however, there is no guarantee that there exists a simple derivation $\Sigma_P \rhd \vdash P : \mathsf{c}(B)$, so we cannot use the inductive hypothesis to get $\Pi_P \rhd \vdash_{\mathtt{u}} P : B$. On the other hand, if we know $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N)$, we can deduce that Σ has shape:

$$\dfrac{\dfrac{z : \mathsf{c}(B), y : \mathsf{c}(A) \vdash y : \mathsf{c}(A)}{y : \mathsf{c}(A) \vdash \lambda z.y : \mathsf{c}(B) \to \mathsf{c}(A)} \qquad \Sigma_P \rhd \vdash P : \mathsf{c}(B)}{\Sigma \rhd y : \mathsf{c}(A) \vdash (\lambda z.y)P : \mathsf{c}(A)}$$

where $\Sigma_P \rightsquigarrow \mathsf{c}(\Pi_Q)$. Thus, by inductive hypothesis there exists $\Pi_P \rhd \vdash_{\mathtt{u}} P : B$, from which it is possible to build the desired Π_M.

Proving that typability in $\mathcal{U}$ is preserved by γ-expansion is straightforward. Notice that, as opposed to Lemma 27, the term M is not explicitly required to be simply typable.

▶ **Lemma 30.** *If $\Gamma \vdash_{\mathtt{u}} N : A$ and $M \to_\gamma N$, then $\Gamma \vdash_{\mathtt{u}} M : A$.*

Proof. The proof proceeds by induction on the reduction context. For the base case, let $M = (\lambda x.\lambda y.P)Q \to_\gamma \lambda y.(\lambda x.P)Q = N$; observe that by α-conversion we can freely assume $y \notin \mathsf{FV}(Q)$. Thus, the derivation $\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} N : A$ has shape:

$$\dfrac{\dfrac{\dfrac{\Delta, x : [B_i]_{i\in I}, y : [C_j]_{j\in J} \vdash_{\mathtt{u}} P : D}{\Delta, y : [C_j]_{J\in j} \vdash_{\mathtt{u}} \lambda x.P : [B_i]_{i\in I} \to D}\,(\to_{\mathsf{I}}) \qquad (\Delta_i \vdash_{\mathtt{u}} Q : B_i)_{i\in I}}{\Gamma, y : [C_j]_{J\in j} \vdash_{\mathtt{u}} (\lambda x.P)Q : D}\,(\to_{\mathsf{E}})}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} \lambda y.(\lambda x.P)Q : [C_j]_{j\in J} \to D}\,(\to_{\mathsf{I}})$$

By rearranging the derivation rules we can easily build $\Pi_M \rhd \Gamma \vdash_{\mathtt{u}} M : A$:

$$\dfrac{\dfrac{\dfrac{\Delta, x : [B_i]_{i\in I}, y : [C_j]_{j\in J} \vdash_{\mathtt{u}} P : D}{\Delta, x : [B_i]_{i\in I} \vdash_{\mathtt{u}} \lambda y.P : [C_j]_{j\in J} \to D}\,(\to_{\mathsf{I}})}{\Delta \vdash_{\mathtt{u}} \lambda x.\lambda y.P : [B_i]_{i\in I} \to [C_j]_{j\in J} \to D}\,(\to_{\mathsf{I}}) \qquad (\Delta_i \vdash_{\mathtt{u}} Q : B_i)_{i\in I}}{\Pi_M \rhd \Gamma \vdash_{\mathtt{u}} (\lambda x.\lambda y.P)Q : [C_j]_{j\in J} \to D}\,(\to_{\mathsf{E}})$$

The inductive cases immediately follow using the inductive hypothesis. ◀

Before moving on to K'-expansion, we briefly discuss why dealing with arbitrary K-expansion would be quite problematic, even in the unrestricted system $\mathcal{I}$. Assume two derivations $\Theta \triangleright \Gamma \vdash_{\mathtt{i}} M : A$ and $\Delta \vdash_{\mathtt{i}} N : B$, where $x \notin \mathsf{FV}(M)$. It is straightforward to build $\Theta' \triangleright \Gamma \uplus \Delta \vdash_{\mathtt{i}} (\lambda x.M)N : A$, thus reversing the K-reduction step $(\lambda x.M)N \to_K M$. However, notice that in general $\Gamma \uplus \Delta$ contains bigger multisets than the ones originally found in Γ; therefore, if Θ is a subderivation of a larger derivation Π, simply replacing Θ by Θ' may not result in a correct derivation. In order to be consistent with the enlarged multisets, it may be necessary to *globally* update the structure of Π: this means introducing new subderivations and/or replicating existing ones (along with their type environments), which in turn may lead to further inconsistencies.

On the other hand, restricting the focus to K'-expansions has the great advantage of keeping the required modifications *local*, thus allowing for an elegant inductive reasoning. The proof of the following Lemma 31 clearly illustrates this point.

▶ **Lemma 31.** *Let $\Pi_N \triangleright \Gamma \vdash_{\mathtt{u}} N : A$. If $M \to_{K'} N$ and there is $\Sigma \triangleright \Phi \vdash M : \sigma$ such that $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)}) \triangleright \Phi \vdash N : \sigma$ for some $\Delta \sim \Gamma$, then:*

- *if M is not an abstraction, then $\Gamma' \vdash_{\mathtt{u}} M : A$ for some Γ' such that $\mathsf{c}(\Gamma') \subseteq \Phi$;*
- *if M is an abstraction, then $\Gamma' \vdash_{\mathtt{u}} M : A'$ for some Γ' and A' such that $\mathsf{c}(\Gamma') \subseteq \Phi$ and $A' \sim A$.*

Proof. By induction on the reduction context. Remark that $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)}) \triangleright \mathsf{c}(\Gamma \uplus \Delta) = \Phi \vdash N : \mathsf{c}(A) = \sigma$ for some $\Delta \sim \Gamma$ implies $\mathsf{c}(\Gamma) \subseteq \Phi$. We show how to build a derivation Π_M with the desired properties starting from Π_N. For the base case, consider $M = (\lambda x.N)P \to_{K'} N$, where P is in β-normal form. The derivation Σ has shape:

$$\dfrac{\dfrac{\Phi_1, x : \tau \vdash N : \sigma}{\Sigma_1 \triangleright \Phi_1 \vdash \lambda x.N : \tau \to \sigma}\,(\to_{\mathtt{I}}) \qquad \Sigma_2 \triangleright \Phi_2 \vdash P : \tau}{\Sigma \triangleright \Phi \vdash (\lambda x.N)P : \sigma}\,(\to_{\mathtt{E}})$$

By Lemma 24 there exists $\Pi_2 \triangleright \Gamma_2 \vdash_{\mathtt{u}} P : B$ such that $\mathsf{c}(\Pi_2) = \Sigma_2$, which entails $\mathsf{c}(\Gamma_2) = \Phi_2 \subseteq \Phi$ and $\mathsf{c}(B) = \tau$. Note that $\mathsf{c}(\Gamma) \subseteq \Phi$ guarantees $\Gamma \sim \Gamma_2$. Starting from Π_N, it is easy to exploit weakening in the axioms and obtain a derivation with conclusion $\Gamma, x : [B] \vdash_{\mathtt{u}} N : A$. Hence we can build:

$$\dfrac{\dfrac{\Gamma, x : [B] \vdash_{\mathtt{u}} N : A}{\Gamma \vdash_{\mathtt{u}} \lambda x.N : [B] \to A}\,(\to_{\mathtt{I}}) \qquad \Gamma_2 \vdash_{\mathtt{u}} P : B}{\Pi_M \triangleright \Gamma' = \Gamma \uplus \Gamma_2 \vdash_{\mathtt{u}} (\lambda x.N)P : A}\,(\to_{\mathtt{E}})$$

satisfying the requirements. Indeed, $\mathsf{c}(\Gamma) \subseteq \Phi$ and $\mathsf{c}(\Gamma_2) = \Phi_2 \subseteq \Phi$ imply $\mathsf{c}(\Gamma') \subseteq \Phi$.

For the inductive step, consider the reduction contexts in which a K'-reduction may take place:

- Case $M = \lambda x.P \to_{K'} \lambda x.Q = N$. Letting $\sigma = \tau_1 \to \tau_2$ and $A = \mu \to B$, the derivations Σ and Π_N have shape:

$$\dfrac{\Sigma_0 \triangleright \Phi, x : \tau_1 \vdash P : \tau_2}{\Sigma \triangleright \Phi \vdash \lambda x.P : \tau_1 \to \tau_2}\,(\to_{\mathtt{I}}) \qquad \dfrac{\Pi_0 \triangleright \Gamma, x : \mu \vdash_{\mathtt{u}} Q : B}{\Pi_N \triangleright \Gamma \vdash_{\mathtt{u}} \lambda x.Q : \mu \to B}\,(\to_{\mathtt{I}})$$

  $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ means that $\Sigma_0 \rightsquigarrow \mathsf{c}(\Pi_0^{(\Delta)})$. Therefore, if P is an abstraction, by inductive hypothesis there are Γ', μ' and B' such that $\mathsf{c}(\Gamma', x : \mu') \subseteq (\Phi, x : \tau_1)$, $B' \sim B$ and $\Gamma', x : \mu' \vdash_{\mathtt{u}} P : B'$. From this we obtain $\Pi_M \triangleright \Gamma' \vdash_{\mathtt{u}} \lambda x.P : \mu' \to B'$ satisfying the requirements. If P is not an abstraction, the reasoning is similar.

- Case $M = xM_1 \dots M_m P \to_{K'} xM_1 \dots M_m Q = N$, where $m \geq 0$. The derivation Σ has shape:

$$\frac{\Sigma_1 \rhd \Phi_1 \vdash xM_1 \dots M_m : \tau \to \sigma \qquad \Sigma_2 \rhd \Phi_2 \vdash P : \tau}{\Sigma \rhd \Phi \vdash xM_1 \dots M_m P : \sigma}\ (\to_{\mathsf{E}})$$

  Letting $I = \{1, \dots, n\}$, the derivation Π_N has shape:

$$\dfrac{\dfrac{\Psi \vdash_{\mathtt{u}} x : \mu_1 \to \ldots \to \mu_m \to [B_i]_{i\in I} \to A \qquad (\Psi_C \vdash_{\mathtt{u}} M_1 : C)_{C\in\mu_1}}{\vdots}\ (\to_{\mathsf{E}})}{\dfrac{\Pi_0 \rhd \Gamma_0 \vdash_{\mathtt{u}} xM_1 \dots M_m : [B_i]_{i\in I} \to A \qquad (\Pi_i \rhd \Gamma_i \vdash_{\mathtt{u}} Q : B_i)_{i\in I}}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} xM_1 \dots M_m Q : A}\ (\to_{\mathsf{E}})}$$

  $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ means that, for each $i \in I$, $\Sigma_2 \rightsquigarrow \mathsf{c}(\Pi_i^{(\Delta_i)})$ for some $\Delta_i \sim \Gamma_i$. Therefore, if P is an abstraction, by inductive hypothesis there are $\Pi'_i \rhd \Gamma'_i \vdash_{\mathtt{u}} P : B'_i$ such that $\mathsf{c}(\Gamma'_i) \subseteq \Phi_2 \subseteq \Phi$ and $B'_i \sim B_i$ $(i \in I)$. Note that $\mathsf{c}(\Gamma'_i) \subseteq \Phi$ for all $i \in I$ guarantees $\Gamma'_i \sim \Gamma'_j$ and $\Gamma_0 \sim \Gamma'_i$ for all $i, j \in I$. Hence to obtain Π_M it suffices to replace Π_i by Π'_i, and change the type of the axiom introducing x into $\mu_1 \to \ldots \to \mu_m \to [B'_i]_{i\in I} \to A$. In case P is not an abstraction, the reasoning is similar.
- Case $M = (\lambda x.P)Q \to_{K'} (\lambda x.P)R = N$, where $x \notin \mathsf{FV}(P)$. The derivation Σ has shape:

$$\frac{\Sigma_1 \rhd \Phi_1 \vdash \lambda x.P : \tau \to \sigma \qquad \Sigma_2 \rhd \Phi_2 \vdash Q : \tau}{\Sigma \rhd \Phi \vdash (\lambda x.P)Q : \sigma}\ (\to_{\mathsf{E}})$$

  Letting $I = \{1, \dots, n\}$, the derivation Π_N has shape:

$$\frac{\Pi_0 \rhd \Gamma_0 \vdash_{\mathtt{u}} \lambda x.P : [B_i]_{i\in I} \to A \qquad (\Pi_i \rhd \Gamma_i \vdash_{\mathtt{u}} R : B_i)_{i\in I}}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} (\lambda x.P)R : A}\ (\to_{\mathsf{E}})$$

  $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ means that, for each $i \in I$, $\Sigma_2 \rightsquigarrow \mathsf{c}(\Pi_i^{(\Delta_i)})$ for some $\Delta_i \sim \Gamma_i$. Therefore, if Q is an abstraction, by inductive hypothesis there exists $\Pi'_i \rhd \Gamma'_i \vdash_{\mathtt{u}} Q : B'_i$ such that $\mathsf{c}(\Gamma'_i) \subseteq \Phi_2 \subseteq \Phi$ and $B'_i \sim B_i$ $(i \in I)$. As in the previous case, from $\mathsf{c}(\Gamma'_i) \subseteq \Phi$ $(i \in I)$ we deduce $\Gamma'_i \sim \Gamma'_j$ and $\Gamma_0 \sim \Gamma'_i$ $(i, j \in I)$. Thus, in order to build Π_M, it suffices to replace Π_i by Π'_i and change the multiset associated to the dummy variable x, so that it matches the new types B'_i. The case in which Q is not an abstraction is similar.
- Case $M = (PQ)R \to_{K'} SR = N$. The derivation Σ has shape:

$$\frac{\Sigma_1 \rhd \Phi_1 \vdash PQ : \tau \to \sigma \qquad \Sigma_2 \rhd \Phi_2 \vdash R : \tau}{\Sigma \rhd \Phi \vdash (PQ)R : \sigma}\ (\to_{\mathsf{E}})$$

  Letting $I = \{1, \dots, n\}$, the derivation Π_N has shape:

$$\frac{\Pi_0 \rhd \Gamma_0 \vdash_{\mathtt{u}} S : [B_i]_{i\in I} \to A \qquad (\Pi_i \rhd \Gamma_i \vdash_{\mathtt{u}} R : B_i)_{i\in I}}{\Pi_N \rhd \Gamma \vdash_{\mathtt{u}} SR : A}\ (\to_{\mathsf{E}})$$

  $\Sigma \rightsquigarrow \mathsf{c}(\Pi_N^{(\Delta)})$ implies $\Sigma_1 \rightsquigarrow \mathsf{c}(\Pi_0^{(\Delta_0)})$ for some $\Delta_0 \sim \Gamma_0$. Therefore, as PQ is not an abstraction, by inductive hypothesis there is $\Pi'_0 \rhd \Gamma'_0 \vdash_{\mathtt{u}} PQ : [B_i]_{i\in I} \to A$ such that $\mathsf{c}(\Gamma'_0) \subseteq \Phi_1 \subseteq \Phi$. Since $\mathsf{c}(\Gamma_i) \subseteq \Phi$, it holds that $\Gamma'_0 \sim \Gamma_i$ for all $i \in I$. The derivation Π_M is obtained by replacing Π_0 by Π'_0. ◀

Finally, we have all prerequisites in order to prove that if a term is typable by $\mathcal{S}$, then it is also typable by $\mathcal{U}$.

▶ **Theorem 32.** $\Sigma \triangleright \Gamma \vdash M : \sigma$ *implies there is* $\Pi \triangleright \Gamma' \vdash_{\mathtt{u}} M : A$ *such that* $\mathsf{c}(\Pi) = \Sigma$.

Proof. Consider a $I\gamma K'$-reduction sequence s from M to a β-normal form, which exists by Theorem 15. We reason by induction on the length n of the sequence s. If $n = 0$, i.e. M is in β-normal form, the result immediately follows from Lemma 24. For the inductive part of the proof, we rely on Lemma 27 for I-reduction steps, on Lemma 30 for γ-reduction steps, and on Lemma 31 for K'-reduction steps. ◀

4.3 From Uniform Typability to Strong Normalization

It is notoriously easy to show that all terms which are typable in system $\mathcal{I}$ (and, consequently, all terms which are typable in system $\mathcal{U}$) are strongly β-normalizing. Intuitively, this is because non-idempotent intersection type systems internalize the reduction process: if M is a term containing a subterm N, a quantitative derivation for M requires (at least) as many subderivations for N as there are copies of N that can be produced during any β-reduction sequence from M to its normal form. Since all the required copies are already there to begin with, mimicking β-reduction on the quantitative derivation necessarily decreases its total size: indeed, I-reduction steps simply rearrange the derivation structure, replacing axioms by subderivations, while K-reduction steps erase subderivations. Formally, writing $\mathsf{size}(\Pi)$ for the number of rules in a derivation Π, one has that:

▶ **Theorem 33** (Weighted Subject Reduction [3, Theorem 4.2])**.** *If* $\Pi_M \triangleright \Gamma \vdash_{\mathtt{i}} M : A$ *and* $M \to_\beta N$*, then there exists* $\Pi_N \triangleright \Gamma \vdash_{\mathtt{i}} N : A$ *such that* $\mathsf{size}(\Pi_N) < \mathsf{size}(\Pi_M)$.

Finally, we can state the following.

▶ **Theorem 34.** $\Gamma \vdash M : \sigma$ *implies* M *is strongly* β*-normalizing.*

Proof. Immediate consequence of Theorem 32 and Theorem 33. ◀

5 A Family of Perpetual Reductions

This brief section shows that $I\gamma K'$-expansion also holds in the general system $\mathcal{I}$. Actually, the proof is simpler, because there are no requirements concerning simple typability. Such a result, together with the fact that all $\mathcal{I}$-typable terms are strongly normalizing, allows us to identify a family of perpetual reduction strategies.

We begin by pointing out that Lemma 24 and Lemma 27 are the system $\mathcal{U}$ counterparts of the following well-known properties of system $\mathcal{I}$:

▶ **Lemma 35.** *If* M *is in* β*-normal form, then* $\Gamma \vdash_{\mathtt{i}} M : A$.

Proof. Essentially as in Lemma 24. ◀

▶ **Lemma 36** ([4, Theorem 4.3])**.** *If* $\Gamma \vdash_{\mathtt{i}} N : A$ *and* $M \to_I N$*, then* $\Gamma \vdash_{\mathtt{i}} M : A$.

Similar considerations can be made for the other typability-preserving expansions.

▶ **Lemma 37.** *If* $\Gamma \vdash_{\mathtt{i}} N : A$ *and* $M \to_\gamma N$*, then* $\Gamma \vdash_{\mathtt{i}} M : A$.

Proof. By observing that the proof of Lemma 30 never mentions uniformity. ◀

One needs to be careful with the statement of subject expansion w.r.t. the K'-reduction (the counterpart of Lemma 31): the case in which the contracted term is an abstraction does not preserve the assigned type.

▶ **Lemma 38.** *If* $\Gamma \vdash_{\mathtt{i}} N : A$ *and* $M \to_{K'} N$*, then:*
- *if* M *is not an abstraction, then* $\Gamma' \vdash_{\mathtt{i}} M : A$ *for some* Γ'*;*
- *if* M *is an abstraction, then* $\Gamma' \vdash_{\mathtt{i}} M : A'$ *for some* Γ' *and* A'*.*

Proof. By induction on the reduction context. The proof is analogous to that of Lemma 31, without the conditions on simple typability. The base case, namely $M = (\lambda x.N)P \to_{K'} N$ where P is in β-normal form, relies on Lemma 35. ◀

We now have all the ingredients to state the following:

▶ **Theorem 39.** *If* M *is* $I\gamma K'$*-normalizing, then it is strongly* β*-normalizing.*

Proof. Lemmas 35, 36, 37 and 38 guarantee that there exists $\Gamma \vdash_{\mathtt{i}} M : A$. Then the result immediately follows from Theorem 33. ◀

▶ **Corollary 40.** *Any* $I\gamma K'$*-reduction strategy is perpetual.*

6 Mechanization

This section provides an overview over the mechanization[2] of uniform typability of simply typed terms (Theorem 32) using the Coq proof assistant [30]. The mechanization is axiom-free and spans approximately 2000 lines of code, consisting of the following four parts:
- `stlc.v` and `stlc_facts.v` contain definitions and facts (such as subject reduction and substitution lemmas) for the simple type system.
- `stlc_nf.v` proves that simply typed terms are $I\gamma K'$-normalizing (Theorem 15).
- `nitlc.v` and `nitlc_facts.v` contain definitions and facts (such as weakening) for the uniform intersection type system.
- `nitlc_typ.v` proves the equivalence between simple type typability and uniform intersection type typability (Lemma 23 and Theorem 32).

Simple types and annotated λ-terms are mechanized in `stlc.v` as `sty` and `tm` respectively. Variable binding is addressed via the unscoped de Bruijn approach [9], with infrastructure partially generated by `Autosubst 2` [27].

```
Inductive sty : Type :=
  | satom (x : nat)   (* type variable *)
  | sarr (s t : sty). (* function type *)

Inductive tm : Type :=
  | var (n : nat)           (* term variable *)
  | app (M N : tm)          (* application *)
  | lam (t : sty) (M : tm). (* type-annotated abstraction *)
```

In congruence with Section 3, the annotation `t` in the abstraction constructor `lam` is the simple type assigned to the whole term (not just the bound variable).

The proposition `stlc Gamma M t` mechanizes that the term `M` is assigned the simple type `t` in the simple type environment `Gamma`. $I\gamma K'$-normalization (Theorem 15) of simply typed terms is mechanized in `stlc_nf.v` as follows.

```
Theorem stlc_nf M Gamma t : stlc Gamma M t -> exists N, steps M N /\ nf N.
```

In the above, `steps M N` mechanizes $\mathtt{M} \to^*_{I\gamma K'} \mathtt{N}$, and `nf N` mechanizes that the term `N` is β-normal.

[2] `https://github.com/tudo-seal/uniform-intersection`

Non-idempotent intersection types are mechanized in `nitlc.v` as `nity`.

```
Inductive nity : Type :=
  | niatom (x : nat)                    (* type variable *)
  | niarr (u : list nity) (A : nity).  (* function type *)
```

The proposition `nitlc Gamma M A` mechanizes that the term `M` is assigned the uniform intersection type `A` in the uniform intersection type environment `Gamma`. Simple type typability of terms which can be assigned a uniform intersection type (Lemma 23) is mechanized in `nitlc_typ.v` as follows.

```
Theorem nitlc_stlc Gamma0 M Gamma A :
  nitlc Gamma M A ->
  env_ssim Gamma0 Gamma ->
  allfv (fun x => nth_error Gamma0 x <> None) M ->
  exists t, stlc Gamma0 M t /\ ssim t A.
```

In the above, the pointwise collapse of the uniform intersection type environment `Gamma` to the simple type environment `Gamma0` is mechanized by `env_ssim Gamma0 Gamma` together with `allfv (fun x => nth_error Gamma0 x <> None) M`. The proposition `ssim t A` mechanizes that the non-idempotent type `A` is uniform and collapses to the existentially quantified simple type `t` (cf. Definition 19).

Finally, uniform typability of simply typed terms (Theorem 32) is mechanized in `nitlc_typ.v` as follows.

```
Theorem nitlc_type_inference M Gamma0 t : stlc Gamma0 M t ->
  exists Gamma A,
    nitlc Gamma M A /\
    Forall2 (fun s u => u <> [] /\ Forall (ssim s) u) Gamma0 Gamma /\
    ssim t A.
```

In the above, if a term `M` is assigned a simple type `t` in the simple type environment `Gamma0`, then there exists a uniform intersection type environment `Gamma` and a uniform intersection type `A` such that the following conditions hold:

- The term `M` is assigned the type `A` in the environment `Gamma`.
- Each multiset `u` in `Gamma` is nonempty and collapses to the corresponding simple type `s` in `Gamma0`.
- The type `A` collapses to the simple type `t`.

The proof structure of the above `Theorem nitlc_type_inference` relies on the mechanization of typability preserving expansion, namely `Theorem stepI_expansion` (Lemma 27), `Theorem stepG_expansion` (Lemma 30), and `Theorem stepK_expansion` (Lemma 31).

There are three interdependent aspects of the proof of `Theorem nitlc_type_inference` which highlight the utility of the Coq proof assistant.

- The definition of the K'-reduction (Definition 1) allows for an expansion lemma.
- The inductive hypothesis for the expansion lemma (cf. Lemma 31) is chosen carefully.
- The particular inductive proof involves extensive, nested case analyses for the chosen definition and inductive hypothesis.

In all three aspects the development of the proof was guided by the proof assistant: the technical details listed by the tool motivated the particular definition of the K'-reduction. In fact, the proof was developed via interaction with the mechanized statement prior to being transcribed into a traditional written format.

7 Conclusion

By providing an alternative proof that all simply typable terms can be assigned a quantitative type by system $\mathcal{U}$ (Theorem 32), we are able to easily infer strong normalization of STLC (Theorem 34). The presented typability proof, fully formalized in Coq, is constructive: this means that an actual type inference algorithm for system $\mathcal{U}$ can be extracted from it. Such an algorithm is conceptually dual to the one proposed by [23]; indeed, the technique presented in the work at hand focuses on term expansion rather than on term reduction.

Inductive proofs of subject expansion in both systems $\mathcal{U}$ and $\mathcal{I}$ (most notably Lemma 31 and Lemma 38) are achieved by means of the introduced $I\gamma K'$-reduction, for which we show that simply typed terms are normalizing (Theorem 15). In addition, our perspective on SN leads to the discovery of an interesting family of perpetual reduction strategies (Corollary 40).

The present work also highlights the role of Coq as a proof *assistant*: its contribution was crucial in the design of the $I\gamma K'$-reduction and the mechanical verification of technical details of the aforementioned results.

References

1 Pablo Barenbaum and Cristian Sottile. Two decreasing measures for simply typed λ-terms. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction, FSCD 2023, July 3-6, 2023, Rome, Italy*, volume 260 of *LIPIcs*, pages 11:1–11:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPICS.FSCD.2023.11.

2 Hendrik Pieter Barendregt, Wil Dekkers, and Richard Statman. *Lambda Calculus with Types*. Perspectives in logic. Cambridge University Press, 2013. URL: http://www.cambridge.org/de/academic/subjects/mathematics/logic-categories-and-sets/lambda-calculus-types.

3 Antonio Bucciarelli, Delia Kesner, and Simona Ronchi Della Rocca. Inhabitation for non-idempotent intersection types. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:7)2018.

4 Antonio Bucciarelli, Delia Kesner, and Daniel Ventura. Non-idempotent intersection types for the lambda-calculus. *Log. J. IGPL*, 25(4):431–464, 2017. doi:10.1093/JIGPAL/JZX018.

5 Mario Coppo and Mariangiola Dezani-Ciancaglini. A new type assignment for λ-terms. *Arch. Math. Log.*, 19(1):139–156, 1978. doi:10.1007/BF02011875.

6 Mario Coppo and Mariangiola Dezani-Ciancaglini. An extension of the basic functionality theory for the lambda-calculus. *Notre Dame Journal of Formal Logic*, 21(4):685–693, 1980. doi:10.1305/ndjfl/1093883253.

7 Mario Coppo, Mariangiola Dezani-Ciancaglini, and Betti Venneri. Functional characters of solvable terms. *Mathematical Logic Quarterly*, 27(2-6):45–58, 1981. doi:10.1002/malq.19810270205.

8 René David. Normalization without reducibility. *Ann. Pure Appl. Log.*, 107(1-3):121–130, 2001. doi:10.1016/S0168-0072(00)00030-0.

9 Nicolaas Govert De Bruijn. Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation, with application to the Church-Rosser theorem. In *Indagationes Mathematicae (Proceedings)*, volume 75, pages 381–392. North-Holland, 1972.

10 Daniel de Carvalho. Execution time of λ-terms via denotational semantics and intersection types. *Math. Struct. Comput. Sci.*, 28(7):1169–1203, 2018. doi:10.1017/S0960129516000396.

11 Philippe de Groote. The conservation theorem revisited. In Marc Bezem and Jan Friso Groote, editors, *Typed Lambda Calculi and Applications, International Conference on Typed Lambda Calculi and Applications, TLCA '93, Utrecht, The Netherlands, March 16-18, 1993, Proceedings*, volume 664 of *Lecture Notes in Computer Science*, pages 163–178. Springer, 1993. doi:10.1007/BFB0037105.

12 Nachum Dershowitz and Zohar Manna. Proving termination with multiset orderings. *Commun. ACM*, 22(8):465–476, 1979. doi:10.1145/359138.359142.

13 Jean H. Gallier. *On Girard's "Candidats de Reductibilité"*. University of Pennsylvania, 1989. URL: https://api.semanticscholar.org/CorpusID:14688391.

14 Philippa Gardner. Discovering needed reductions using type theory. In Masami Hagiya and John C. Mitchell, editors, *Theoretical Aspects of Computer Software, International Conference TACS '94, Sendai, Japan, April 19-22, 1994, Proceedings*, volume 789 of *Lecture Notes in Computer Science*, pages 555–574. Springer, 1994. doi:10.1007/3-540-57887-0_115.

15 Jean-Yves Girard. Une extension de l'interpretation de Gödel a l'analyse, et son application a l'elimination des coupures dans l'analyse et la theorie des types. In J.E. Fenstad, editor, *Proceedings of the Second Scandinavian Logic Symposium*, volume 63 of *Studies in Logic and the Foundations of Mathematics*, pages 63–92. Elsevier, 1971. doi:10.1016/S0049-237X(08)70843-7.

16 Inge Li Gørtz, Signe Reuss, and Morten Heine Sørensen. Strong normalization from weak normalization by translation into the lambda-I-calculus. *High. Order Symb. Comput.*, 16(3):253–285, 2003. doi:10.1023/A:1025693307470.

17 Assaf J. Kfoury and Joe B. Wells. Addendum to "New notions of reduction and non-semantic proofs of strong beta-normalization in typed lambda calculi", 1995. URL: https://open.bu.edu/handle/2144/1568.

18 Assaf J. Kfoury and Joe B. Wells. New notions of reduction and non-semantic proofs of beta-strong normalization in typed lambda-calculi. In *Proceedings, 10th Annual IEEE Symposium on Logic in Computer Science, San Diego, California, USA, June 26-29, 1995*, pages 311–321. IEEE Computer Society, 1995. doi:10.1109/LICS.1995.523266.

19 Jan Willem Klop. *Combinatory reduction systems*. PhD thesis, Univ. Utrecht, 1980.

20 Robert Pieter Nederpelt Lazarom. *Strong normalization in a typed lambda calculus with lambda structured types*. PhD thesis, TU Eindhoven, 1973.

21 Jean-Jacques Lévy. *Réductions correctes et optimales dans le lambda-calcul*. PhD thesis, Université de Paris 7, 1978.

22 Peter Møller Neergaard. Theoretical pearls: A bargain for intersection types: a simple strong normalization proof. *J. Funct. Program.*, 15(5):669–677, 2005. doi:10.1017/S0956796805005587.

23 Daniele Pautasso and Simona Ronchi Della Rocca. A quantitative version of simple types. In Marco Gaboardi and Femke van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023)*, volume 260 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 29:1–29:21, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.FSCD.2023.29.

24 Garrel Pottinger. A type assignment for the strongly normalizable λ-terms. *To HB Curry: essays on combinatory logic, lambda calculus and formalism*, pages 561–577, 1980.

25 Helmut Schwichtenberg. An upper bound for reduction sequences in the typed λ-calculus. *Arch. Math. Log.*, 30(5-6):405–408, 1991. doi:10.1007/BF01621476.

26 Morten Heine Sørensen. Strong normalization from weak normalization in typed lambda-calculi. *Inf. Comput.*, 133(1):35–71, 1997. doi:10.1006/INCO.1996.2622.

27 Kathrin Stark, Steven Schäfer, and Jonas Kaiser. Autosubst 2: reasoning with multi-sorted de Bruijn terms and vector substitutions. In Assia Mahboubi and Magnus O. Myreen, editors, *Proceedings of the 8th ACM SIGPLAN International Conference on Certified Programs and Proofs, CPP 2019, Cascais, Portugal, January 14-15, 2019*, pages 166–180. ACM, 2019. doi:10.1145/3293880.3294101.

28 William W. Tait. Intensional interpretations of functionals of finite type I. *Journal of Symbolic Logic*, 32(2):198–212, 1967. doi:10.2307/2271658.

29 William W. Tait. A realizability interpretation of the theory of species. In Rohit Parikh, editor, *Logic Colloquium*, pages 240–251, Berlin, Heidelberg, 1975. Springer Berlin Heidelberg.

30 The Coq Development Team. The Coq proof assistant, July 2023. doi:10.5281/zenodo.8161141.

31 Femke van Raamsdonk, Paula Severi, Morten Heine Sørensen, and Hongwei Xi. Perpetual reductions in lambda-calculus. *Inf. Comput.*, 149(2):173–225, 1999. doi:10.1006/INCO.1998.2750.

32 Hongwei Xi. Weak and strong beta normalisations in typed lambda-calculi. In Philippe de Groote, editor, *Typed Lambda Calculi and Applications, Third International Conference on Typed Lambda Calculi and Applications, TLCA '97, Nancy, France, April 2-4, 1997, Proceedings*, volume 1210 of *Lecture Notes in Computer Science*, pages 390–404. Springer, 1997. doi:10.1007/3-540-62688-3_48.