Automath type inclusion in Barendregt's Cube

Fairouz Kamareddine[1], J.B. Wells[1] and Daniel Ventura[2]

[1] School of Maths and Computer Sc., Heriot-Watt Univ., Edinburgh, UK
[2] Instituto de Informática, Universidade Federal de Goiás, Goiânia GO, Brazil.

Abstract. The introduction of a general definition of function was key to Frege's formalisation of logic. Self-application of functions was at the heart of Russell's paradox. Russell introduced type theory to control the application of functions and avoid the paradox. Since, different type systems have been introduced, each allowing different functional power. Most of these systems use the two binders λ and Π to distinguish between functions and types, and allow β-reduction but not Π-reduction. That is, $(\pi_{x:A}.B)C \to B[x := C]$ is only allowed when π is λ but not when it is Π. Since Π-reduction is not allowed, these systems cannot allow unreduced typing. Hence types do not have the same instantiation right as functions. In particular, when b has type B, the type of $(\lambda_{x:A}.b)C$ is immediately given as $B[x := C]$ instead of keeping it as $(\Pi_{x:A}.B)C$ to be Π-reduced to $B[x := C]$ later. Extensions of modern type systems with both β- and Π-reduction and unreduced typing have appeared in [11, 12, **?**] and lead naturally to unifying the λ and Π abstractions [9, 10]. The Automath system combined the unification of binders λ and Π with β- and Π-reduction together with a type inclusion rule that allows the different expressions that define the same term to share the same type. In this paper we extend the cube of 8 influential type systems [3] with the Automath notion of type inclusion [5] and study its properties.

1 Introduction

Different type systems exist, each allowing different functional power. The λ-calculus is a higher-order rewriting system which allows the elegant incorporation of functions and types, explains the notion of computability and is at the heart of programming languages (e.g., Haskell and ML) and formalisations of mathematics (e.g., Automath and Coq). Typed versions of the λ-calculus provide a vehicle where logics, types and rewriting converge. Heyting [7], Kolmogorov [13] and Curry and Feys [6] (improved by Howard [8]) observed the "**propositions as types**" or "**proofs as terms**" (PAT) correspondence. In PAT, logical operators are embedded in the types of λ-terms rather than in the propositions and λ-terms are viewed as proofs of the propositions represented by their types. Advantages of PAT include the ability to manipulate proofs, easier support for independent proof checking, the possibility of the extraction of computer programs from proofs, and the ability to prove properties of the logic via the termination of the rewriting system. And so, typed λ-calculi have been the subject of extensive studies in the second half of the 20th century. For example:

– Both mathematics and programming languages make heavy use of the so-called let expressions/abbreviations where a large expression is given a name which can be replaced by the whole expression (we say in this case that the definition/abbreviation is unfolded) when the need to do so arises.
– Some type systems (e.g., AUTOMATH and the system of [12] have Π-reduction $(\Pi x{:}A.B)N \to_\Pi B[x{:=}N]$ and unreduced typing:

$$\frac{\Gamma \vdash M : \Pi x{:}A.B \qquad \Gamma \vdash N : A}{\Gamma \vdash MN : (\Pi x{:}A.B)N}$$

[12] showed that Π-reduction and unreduced typing lead to the loss of Subject Reduction (SR) which can be restored by adding abbreviations (cf. [11]). Note that the abbreviation/definition system of AUTOMATH itself is not "smart enough" for restoring SR: take the same counterexample as in [12].
– Some versions of the λ-calculus (e.g., in Automath and in the Barendregt cube with unified binders [10]) used the same binder for both λ and Π abstraction. In particular, Automath used $[x : A]B$ for both $\lambda x : A.B$ and $\Pi x : A.B$. Consequences of unifying λ and Π are:
  - A term can have many distinct types [10]. E.g., in λP of [3], we have:

    $$\alpha : * \vdash_\beta (\lambda x{:}\alpha.\alpha) : (\Pi x{:}\alpha.*) \qquad \text{and} \qquad \alpha : * \vdash_\beta (\Pi x{:}\alpha.\alpha) : *$$

    which, when we give up the difference between λ and Π, result in:

    $$I)\ \alpha : * \vdash_\beta [x{:}\alpha]\alpha : [x{:}\alpha]\, * \qquad \text{and} \qquad II)\ \alpha : * \vdash_\beta [x{:}\alpha]\alpha : *$$

    Indeed, both equations I) and II) hold in AUT-QE.
  - More generally, in AUT-QE we have the dervived rule:

    $$\frac{\Gamma \vdash_\beta [x_1{:}A_1]\cdots[x_n{:}A_n]B : [x_1{:}A_1]\cdots[x_n{:}A_n]*}{\Gamma \vdash_\beta [x_1{:}A_1]\cdots[x_n{:}A_n]B : [x_1{:}A_1]\cdots[x_m{:}A_m]*} \qquad 0 \le m \le n \qquad (1)$$

  This derived rule (1) has the following equivalent derived rule in λP (and hence in the higher systmes like $\lambda P\omega$):

  $$\frac{\Gamma \vdash_\beta \lambda x_1{:}A_1.\cdots\lambda x_n{:}A_n.B : \Pi x_1{:}A_1.\cdots\Pi x_n{:}A_n.* \qquad 0 \le m \le n}{\Gamma \vdash_\beta \lambda x_1{:}A_1.\cdots\lambda x_m{:}A_m.\Pi x_{m+1}{:}A_{m+1}.\cdots\Pi x_n{:}A_n.B : \Pi x_1{:}A_1.\cdots\Pi x_m{:}A_m.*}$$

  However, AUT-QE goes further and generalises (1) to a rule of *type inclusion*:

  $$\frac{\Gamma \vdash_\beta M : [x_1{:}A_1]\cdots[x_n{:}A_n]*}{\Gamma \vdash_\beta M : [x_1{:}A_1]\cdots[x_m{:}A_m]*} \qquad 0 \le m \le n \qquad (Q)$$

  Such type inclusion guarantees that two equal definitions will share (at least) one type and appears in higher order AUTOMATH systems like AUT-QE.

Remark 1 *Rule (Q) may be motivated by looking at the definition system of* AUTOMATH *where I) allows us to introduce a definition $\zeta(\alpha) := [x{:}\alpha]\alpha : [x{:}\alpha]*$ and II) enables us to define $\xi(\alpha) := [x{:}\alpha]\alpha : *$. Now $\zeta(\alpha)$ and $\xi(\alpha)$ are defining exactly the same term (and are therefore called "definitionally equal"), but without Rule (Q) they wouldn't share the same type (whilst $[x{:}\alpha]\alpha$ has both the type of $\zeta(\alpha)$ and the type of $\xi(\alpha)$). By generalizing (1) to (Q) we get that $\zeta(\alpha)$ also has type $*$, so $\zeta(\alpha)$ and $\xi(\alpha)$ share (at least one) type.*

The behaviour of (variants of) Rule (Q) has never been studied in modern type systems. This paper fills these gaps and gives the first extensive account of modern type systems with/without Π-reduction, unreduced typing and type inclusion. We chose to use as basis for these extensions, a flexible and general framework: Barendregt's β-cube. In the β-cube of [3], eight well-known type systems are given in a uniform way. The weakest system is Church's simply typed λ-calculus $\lambda\!\!\to$, and the strongest system is the Calculus of Constructions $\lambda P\omega$. The second order λ-calculus figures on the β-cube between $\lambda\!\!\to$ and $\lambda P\omega$. The paper is divided as follows:

- Section 2 introduces a number of cubes, establishes necessary properties, and shows that in the cube with type inclusion, 4 systems get merged into two due to type inclusion.
- Section 3 establishes the generation lemma that is crucial for type checking in all the cubes. Then, correctness of types and subject reduction (safety) as well as preservation of types under reduction are studied for all the cubes. Strong normalisation, typability of subterms and unicity of types are laid out to be studied for each cube separately in the later sections.
- In Section 4 we relate the various cubes showing exactly which includes which and whether these inclusions are strict. We then study strong normalisation, typability of subterms and unicity of types in these cubes.
- We conclude in Section 5 and add an appendix containing missing proofs.

2 Notions of reduction and typing

We define the set of terms $\mathcal{T}$ by: $\mathcal{T} ::= * \mid \Box \mid \mathcal{V} \mid \pi_{\mathcal{V}:\mathcal{T}}.\mathcal{T} \mid \mathcal{T}\mathcal{T}$ where $\pi \in \{\lambda, \Pi\}$. We let s, s', s_1, etc. range over the sorts $\{*, \Box\}$. We assume that $\{*, \Box\} \cap \mathcal{V} = \emptyset$. We take $\mathcal{V}$ to be a set of variables over which, x, y, z, x_1, etc. range. We let A, B, M, N, a, b, etc. sometimes also indexed by Arabic numerals such as A_1, A_2 range over terms. We use $\textsc{fv}(A)$ to denote the free variables of A, and $A[x := B]$ to denote the substitution of all the free occurrences of x in A by B. We assume familiarity with the notion of *compatibility.* As usual, we take terms to be equivalent up to variable renaming and let $\equiv$ denote syntactic equality. We also assume the Barendregt convention (BC) where names of bound variables are always chosen so that they differ from free ones in a term and where different abstraction operators bind different variables. For example, we write $(\pi_{y:A}.y)x$ instead of $(\pi_{x:A}.x)x$ and $\pi_{x:A}.\pi_{y:B}.C$ instead of $\pi_{x:A}.\pi_{x:B}.C$. (BC) will also be assumed for contexts and typings (for each of the calculi presented) so that for example, if $\Gamma \vdash \pi_{x:A}.B : C$ then x will not occur in Γ. We define subterms in the usual way. For $\Lambda \in \{\lambda, \Pi\}$, we write $\Lambda_{x_m:A_m} \ldots \Lambda_{x_n:A_n}.A$ as $\Lambda^{i:m..n}_{x_i:A_i}.A$.

Definition 2 [Reductions]

- Let β-reduction $\to_\beta$ be the compatible closure of $(\lambda_{x:A}.B)C \to_\beta B[x := C]$.
- Let Π-reduction $\to_\Pi$ be the compatible closure of $(\Pi_{x:A}.B)C \to_\Pi B[x := C]$.
- We define the union of reduction relations as usual. E.g., $\to_{\beta\Pi} = \to_\beta \cup \to_\Pi$.
- Let $r \in \{\beta, \Pi, \beta\Pi\}$. We define r-redexes in the usual way. Moreover:
  - $\twoheadrightarrow_r$ is the reflexive transitive closure of $\to_r$ and $=_r$ is the equivalence closure of $\to_r$. We write $\overset{+}{\twoheadrightarrow}_r$ to denote one or more steps of r-reduction.

- If $A \to_r B$ (resp. $A \twoheadrightarrow_r B$), we also write $B\ {}_r\!\leftarrow A$ (resp. $B\ {}_r\!\twoheadleftarrow A$).
- We say that A is strongly normalising with respect to $\to_r$ (we use the notation $\mathrm{SN}_{\to_r}(A)$) if there are no infinite $\to_r$-reductions starting at A.
- We say that A is in r-normal form if there is no B such that $A \to_r B$.
- We use $\mathrm{nf}_r(A)$ to refer to the r-normal form of A if it exists.

In order to investigate the connection between the various type systems, it is useful to change Π-redexes into λ-redexes and to contract Π-redexes:

Definition 3 [Changing Π-redexes, $\leq$, $\leq_r$]

- For $A \in \mathcal{T}$, we define $[A]_\Pi \in \mathcal{T}$ and $\widetilde{A} \in \mathcal{T}$ as follows:
  - $[A]_\Pi$ is A where all Π-redexes are contracted.
  - $\widetilde{A}$ is A where every Π-redex $(\Pi_{x:-}.-)$ is changed into a λ-redex $(\lambda_{x:-}.-)$.
-
  - Let $\leq$ be the smallest reflexive and transitive relation on terms such that $\Lambda^{i:1..n}_{x_i:A_i}.* \leq \Lambda^{i:1..m}_{x_i:A_i}.*$ for all $m \leq n$.
  - Let $r \in \{\beta, \beta\Pi\}$. For terms A, B we define $A \leq_r B$ by: There are terms $A' =_r A$ and $B' =_r B$ such that $A' \leq B'$.

Theorem 4 (Church-Rosser for $\to_r$ where $r \in \{\beta, \beta\Pi\}$). *Let $r \in \{\beta, \beta\Pi\}$. If $B_1\ {}_r\!\twoheadleftarrow A \twoheadrightarrow_r B_2$ then there is a C such that $B_1 \twoheadrightarrow_r C\ {}_r\!\twoheadleftarrow B_2$.*

Proof. For the β-case see [3]. For the $\beta\Pi$-case see [12]. ⊠

Corollary 5

1. If $A \leq_r B$ and $B \leq_r C$ then $A \leq_r C$.
2. If $\Pi_{x:A}.B_1 \leq_r \Pi_{x:A}.B_2$ then $B_1 \leq_r B_2$.

Proof. 1. Determine $A' =_r A$ and $B' =_r B$ such that $A' \leq B'$, and determine $C' =_r C$ and $B'' =_r B$ such that $B'' \leq C'$. Note that we can write: $A' \equiv \Lambda^{i:1..n}_{x_i:A_i}.*$; $B' \equiv \Lambda^{i:1..m}_{x_i:A_i}.*$; $B'' \equiv \Lambda^{i:1..p}_{x_i:B_i}.*$ and $C' \equiv \Lambda^{i:1..q}_{x_i:B_i}.*$ for some $m \leq n$, $q \leq p$. As $B' =_r B''$, they have a common r-reduct by the Church Rosser Theorem 4. Note that this reduct must be of the form $\Lambda^{i:1..m}_{x_i:C_i}.*$ for some $C_i =_r A_i =_r B_i$, and that $m = p$. Define $A'' \equiv \Lambda^{i:1..m}_{x_i:C_i}.\Lambda^{j:m+1..n}_{x_j:A_j}*$ and $C'' \equiv \Lambda^{i:1..q}_{x_i:C_i}.*$. Since $A'' \leq C''$ (as $q \leq p = m \leq n$), $A'' =_r A' =_r A$ and $C'' =_r C' =_r C$, so we have $A \leq_r C$.
2. Determine $P =_r \Pi_{x:A}.B_1$ and $Q =_r \Pi_{x:A}.B_2$ where $P \leq Q$. For some $m \leq n$, $P \equiv \Lambda^{i:1..n}_{x_i:A_i}.*$ and $Q \equiv \Lambda^{i:1..m}_{x_i:A_i}.*$. Since $B_1 =_r \Lambda^{i:2..n}_{x_i:A_i}.* \leq \Lambda^{i:2..m}_{x_i:A_i}.* =_r B_2$ we get $B_1 \leq_r B_2$. ⊠

Definition 6 [$\bot$, Declarations, contexts, $\subseteq$, $\subseteq'$]

1. There are two forms of declarations over which $d, d', d_1, \ldots$ range.
2. A *variable declaration* (v-dec) d is of the form $x : A$. We define $\texttt{var}(d) = x$, $\texttt{type}(d) = A$ and $\mathrm{FV}(d) = \mathrm{FV}(A)$.
3. An *abbreviation declaration* (a-dec) d is of the form $x = B : A$ and abbreviates B of type A to be x. We define $\texttt{var}(d) = x$, $\texttt{type}(d) = A$, $\texttt{ab}(d) = B$ and $\mathrm{FV}(d) = \mathrm{FV}(A) \cup \mathrm{FV}(M)$.
4. A *context* Γ is a (possibly empty) concatenation of declarations $d_1, d_2, \cdots, d_n$ such that if $i \neq j$, then $\texttt{var}(d_i) \not\equiv \texttt{var}(d_j)$. Let $\mathrm{DOM}\,(\Gamma) = \{\texttt{var}(d) \mid d \in \Gamma\}$, Γ-$\texttt{decl} = \{d \in \Gamma \mid d$ is a v-dec$\}$ and Γ-$\texttt{abb} = \{d \in \Gamma \mid d$ is an a-dec $\}$. Let $\Gamma, \Delta, \Gamma', \Gamma_1, \Gamma_2, \ldots$ range over contexts and denote the *empty context* by $\langle\rangle$.

5. We define substitutions on contexts by: $\langle\rangle[x := A] \equiv \langle\rangle$,
   $(\Gamma, y : B)[x := A] \equiv \Gamma[x := A], y : B[x := A]$,
   $(\Gamma, y = B : C)[x := A] \equiv \Gamma[x := A], y = B[x := A] : C[x := A]$.
6. If d is the a-dec $x = E : F$, we write Γ_d for $\Gamma[x := E]$ and A_d for $A[x := E]$.
7. We define $\subseteq$ (resp. $\subseteq'$) between contexts as the least reflexive transitive relation satisfying $\Gamma, \Delta \subseteq \Gamma, d, \Delta$ (resp. $\Gamma, \Delta \subseteq' \Gamma, d, \Delta$ and $\Gamma, x : A, \Delta \subseteq' \Gamma, x = B : A, \Delta$).
8. We extend Definition 3 to contexts as follows: $[\langle\rangle]_\Pi \equiv \langle\rangle$
   $[\Gamma, x : A]_\Pi \equiv [\Gamma]_\Pi, x : [A]_\Pi$ $\quad [\Gamma, x = B : A]_\Pi \equiv [\Gamma]_\Pi, x = [B]_\Pi : [A]_\Pi$
   $\widetilde{\langle\rangle} \equiv \langle\rangle$ $\quad \widetilde{\Gamma, x : A} \equiv \widetilde{\Gamma}, x : \widetilde{A}$ $\quad \widetilde{\Gamma, x = B : A} \equiv \widetilde{\Gamma}, x = \widetilde{B} : \widetilde{A}$.

All systems of the β-cube have the same typing rules but are distinguished from one another by the set $\boldsymbol{R}$ of pairs of sorts (s_1, s_2) allowed in the *type-formation* or *Π-formation* rule, (Π) given in $BT(\lambda, \Pi)$ of Figure 4. Each system of the β-cube has its set $\boldsymbol{R}$ such that $(*, *) \in \boldsymbol{R} \subseteq \{(*, *), (*, \Box), (\Box, *), (\Box, \Box)\}$ and hence there are only eight possible different systems of the β-cube (see Figure 2). The dependencies between these systems is depicted in Figure 1. A Π-type can only be formed in a specific system of the β-cube if rule (Π) of Figure 4 is satisfied for some (s_1, s_2) in the set $\boldsymbol{R}$ of that system. The type system $\lambda\boldsymbol{R}$ describes how judgements $\Gamma \vdash^{\boldsymbol{R}} A : B$ (or $\Gamma \vdash A : B$, if it is clear which $\boldsymbol{R}$ is used) can be derived. Rule (Π) provides a factorisation of the expressive power into three features: *polymorphism, type constructors*, and *dependent types*:

- $(*, *)$ is the basic rule that forms types. All the β-cube systems have this rule.
- $(\Box, *)$ takes care of polymorphism. $\lambda 2$ is the weakest system with $(\Box, *)$.
- $(\Box, \Box)$ takes care of type constructors. $\lambda\underline{\omega}$ is the weakest system with $(\Box, \Box)$.
- $(*, \Box)$ takes care of term dependent types. λP is the weakest system with $(*, \Box)$.

$\lambda\to$	$(*, *)$			
$\lambda 2$	$(*, *)$	$(\Box, *)$		
λP	$(*, *)$		$(*, \Box)$	
λP2	$(*, *)$	$(\Box, *)$	$(*, \Box)$	
$\lambda\underline{\omega}$	$(*, *)$			$(\Box, \Box)$
$\lambda\omega$	$(*, *)$	$(\Box, *)$		$(\Box, \Box)$
λP$\underline{\omega}$	$(*, *)$		$(*, \Box)$	$(\Box, \Box)$
$\lambda P\omega$	$(*, *)$	$(\Box, *)$	$(*, \Box)$	$(\Box, \Box)$

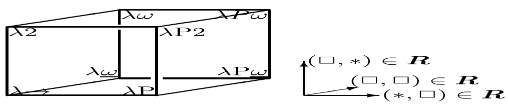

Fig. 1. Barendregt's β-cube

Cubes	Rules	References
$\beta, \to_\beta$	BT(λ, Π) + conv$_\beta$ + app$_\Pi$	[3]
$\pi_i, \to_{\beta\Pi}$	BT(λ, Π) + conv$_{\beta\Pi}$ + i-app$_\Pi$	[12]
$\beta_a, \to_\beta$	BT(λ, Π) + conv$_\beta$ + app$_\Pi$ + BA + let$_\lambda$	[11]
$\pi_{ai}, \to_{\beta\Pi}$	BT(λ, Π) + conv$_{\beta\Pi}$ + i-app$_\Pi$ + BA + let$_\lambda$ + let$_\Pi$	[11]
$\pi, \to_{\beta\Pi}$	BT(λ, Π) + conv$_{\beta\Pi}$ + app$_\Pi$	This paper
$\pi_a, \to_{\beta\Pi}$	BT(λ, Π) + conv$_{\beta\Pi}$ + app$_\Pi$ + BA + let$_\lambda$ + let$_\Pi$	This paper
$\beta_Q, \to_\beta$	BT(λ, Π) + conv$_\beta$ + app$_\Pi$ + Q$_\beta$	This paper
	$\vdash_\pi = \vdash_\beta \subset \vdash_r \subset \vdash_{\pi_{ai}} = \vdash_{\pi_a}$ for $r \in \{\beta_a, \pi_i\}$ (Lemma 16) $\vdash_{\beta_a}$ and $\vdash_{\pi_i}$ are unrelated (Lemma 16) $\vdash_{\beta_{Q\omega}} = \vdash_{\beta_{Q\underline{\omega}}}$ (Lemma 14) $\vdash_{\beta_{QP_\omega}} = \vdash_{\beta_{QP\underline{\omega}}}$ (Lemma 14)	

Fig. 2. Systems studied in this paper

Cubes	lemmas hold	lemmas restricted
β	15..21	
π_i	15 and 19	16→23, 17→23+25, 18→23+25, 20→26, 21→23
β_a	15..19 and 21	20→27
π_{ai}	15..19 and 21	20→27
π	15..21	
π_a	15..19	20→27
β_Q	15..16	

Fig. 3. Properties of various cubes

The next definition sets out the basic notions needed for our type systems.

Definition 7 [Statements, judgements] Let Γ be a context, A, B, C be terms. Let $\vdash$ be one of the typing relations of this paper.

1. $A : B$ is called a *statement*. A and B are its *subject* and *predicate* respectively.
2. $\Gamma \vdash A : B$ is a *judgement* which states that A has type B in context Γ. $\Gamma \vdash A : B : C$ denotes $\Gamma \vdash A : B \wedge \Gamma \vdash B : C$.
3. Γ is $\vdash$-*legal* (or simply *legal*) if $\exists A_1, B_1$ terms such that $\Gamma \vdash A_1 : B_1$.
4. A is a $\Gamma^{\vdash}$-term (or simply Γ-*term*) if $\exists B_1$ such that $[\Gamma \vdash A : B_1 \vee \Gamma \vdash B_1 : A]$.
5. A is $\vdash$-*legal* (or simply *legal*) if $\exists \Gamma_1 [A$ is a $\Gamma_1^{\vdash}$-term$]$.
6. Let r be a reduction relation. We define $\Gamma \Vdash B =_r B'$ as the smallest equivalence relation closed under A and B where: A. If $B =_r B'$ then $\Gamma \Vdash B =_r B'$. B. If $x = D : C \in \Gamma$ and B' arises from B by substituting one particular free occurrence of x in B by D then $\Gamma \Vdash B =_r B'$.
   Note that if Γ does not have a-decs, then $\Gamma \Vdash B =_r B'$ becomes $B =_r B'$.
7. We define $\Gamma \vdash d$ by: $\bullet \Gamma \vdash \mathtt{var}(d) : \mathtt{type}(d)$.
   $\bullet$ And, if d is a-dec then $\Gamma \vdash \mathtt{ab}(d) : \mathtt{type}(d)$ and $\Gamma \Vdash \mathtt{var}(d) =_r \mathtt{ab}(d)$.
8. We define $\Gamma \vdash \Delta$ by: $\Gamma \vdash d$ for every $d \in \Delta$.

In this paper we study extended versions of the β-cube. The extensions considered are summarized in Figure 2 which shows for each cube, its reduction relation and its typing rules. For example, the β-cube uses β-reduction and the BT(λ, Π) rules of Figure 4 with conv$_\beta$ of Figure 7 and app$_\Pi$ of Figure 8.

(axiom)	$\langle\rangle \vdash * : \square$	
(start)	$\dfrac{\Gamma \vdash A : s}{\Gamma, x{:}A \vdash x : A}$	$x \notin \mathrm{DOM}\,(\Gamma)$
(weak)	$\dfrac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s}{\Gamma, x{:}C \vdash A : B}$	$x \notin \mathrm{DOM}\,(\Gamma)$
(Π)	$\dfrac{\Gamma \vdash A : s_1 \qquad \Gamma, x{:}A \vdash B : s_2}{\Gamma \vdash \Pi_{x:A}.B : s_2}$	$(s_1, s_2) \in \boldsymbol{R}$
(λ)	$\dfrac{\Gamma, x{:}A \vdash b : B \qquad \Gamma \vdash \Pi_{x:A}.B : s}{\Gamma \vdash \lambda_{x:A}.b : \Pi_{x:A}.B}$	

Fig. 4. Basic typing $BT(\lambda, \Pi)$

Definition 8 We define a number of cubes, all of which have $\mathcal{T}$ as the set of terms, contexts as in Definition 6.4 and use the BT(λ, Π) rules of Figure 4. For each c-cube we define, we write $\vdash_c$ to denote type derivation in the c-cube.

– **The β- and β_Q-cubes** have contexts that are free of a-decs, use β-reduction $\to_\beta$, and the rules conv$_\beta$ of Figure 7 and app$_\Pi$ of Figure 8. In addition, the β_Q-cube uses the Q$_\beta$ rule of Figure 10.

(start-a)	$\dfrac{\Gamma \vdash A : s \qquad \Gamma \vdash B : A}{\Gamma, x = B{:}A \vdash x : A}$	$x \notin \textsc{dom}\,(\Gamma)$
(weak-a)	$\dfrac{\Gamma \vdash A : B \qquad \Gamma \vdash C : s \qquad \Gamma \vdash D : C}{\Gamma, x = D{:}C \vdash A : B}$	$x \notin \textsc{dom}\,(\Gamma)$

Fig. 5. Basic abbreviation rules BA

$$(\mathrm{let}_{\setminus}) \quad \frac{\Gamma, x = B{:}A \vdash C : D}{\Gamma \vdash (\setminus_{x:A}.C)B : D[x := B]}$$

Fig. 6. $(\mathrm{let}_{\setminus})$ where $\setminus = \lambda$ or $\setminus = \Pi$

- **The π_i-cube** has contexts that are free of a-decs, uses $\beta\Pi$-reduction $\to_{\beta\Pi}$, and the rules $\mathrm{conv}_{\beta\Pi}$ of Figure 7 and i-app_Π of Figure 9.
- **The β_a-cube** uses β-reduction $\to_\beta$, and the BA rules of Figure 5, conv_β of Figure 7, app_Π of Figure 8 and let_λ of Figure 6.
- **The π_{ai}-cube** uses $\beta\Pi$-reduction $\to_{\beta\Pi}$, and the $\mathrm{BT}(\lambda, \Pi)$ rules of Figure 4 with the BA rules of Figure 5, $\mathrm{conv}_{\beta\Pi}$ of Figure 7, i-app_Π of Figure 9 and let_λ and let_Π of Figure 6.
- **The π-cube** has contexts that are free of a-decs, uses $\beta\Pi$-reduction $\to_{\beta\Pi}$, and the rules $\mathrm{conv}_{\beta\Pi}$ of Figure 7 and app_Π of Figure 8. In addition, the π_Q-cube uses the Q_β rule of Figure 10.
- **The π_a-cube** uses $\beta\Pi$-reduction $\to_{\beta\Pi}$, and the BA rules of Figure 5, $\mathrm{conv}_{\beta\Pi}$ of Figure 7, app_Π of Figure 8 and let_λ and let_Π of Figure 6.

In what follows we establish basic properties for the cubes listed above. Unless spcifically mentioned, these properties hold for all the cubes.

Lemma 9 (Free Variable Lemma for $\vdash$ and $\to_r$) *Let Γ be $\vdash$-legal.*
1. *If d and d' are different elements in Γ, then $\mathtt{var}(d) \not\equiv \mathtt{var}(d')$.*
2. *If $\Gamma \vdash B : C$ then $\textsc{fv}(B), \textsc{fv}(C) \subseteq \textsc{dom}\,(\Gamma)$.*
3. *If $\Gamma = \Gamma_1, d, \Gamma_2$ then $\textsc{fv}(d) \subseteq \textsc{dom}\,(\Gamma_1)$.*

Proof. We prove 1, 2 and 3 by induction on the derivation of $\Gamma \vdash_c B : C$. ⊠

Lemma 10 (Start/Context Lemma for $\vdash$ and $\to_r$)
1. *If Γ is $\vdash$-legal then $\Gamma \vdash * : \Box$ and for all $d \in \Gamma$, $\Gamma \vdash d$.*
2. *On the derivation tree to $\Gamma_1, d, \Gamma_2 \vdash A : B$ we have*
   - *$\Gamma_1 \vdash \mathtt{type}(d) : s$ for some sort s and $\Gamma_1, d \vdash \mathtt{var}(d) : \mathtt{type}(d)$.*
   - *If d is a-dec then $\Gamma_1 \vdash \mathtt{ab}(d) : \mathtt{type}(d)$ and $\Gamma_1, d \Vdash \mathtt{var}(d) =_r \mathtt{ab}(d)$.*

Proof. 1. Show by induction on $\Gamma \vdash_c B : C$ that if $\Gamma = \langle\rangle$ then $\Gamma \vdash * : \Box$ and if $\Gamma = \Gamma', d$ then both $\Gamma' \vdash * : \Box$ and $\Gamma \vdash * : \Box$. 2. By induction on $\Gamma \vdash_c B : C$. ⊠

Lemma 11 (Transitivity Lemma for $\vdash$ and $\to_r$) *Let Γ, Δ be $\vdash$-legal contexts such that $\Gamma \vdash \Delta$. The following hold:*
1. *If $\Delta \Vdash A =_r B$ then $\Gamma \Vdash A =_r B$.*
2. *If $\Delta \vdash A : B$ then $\Gamma \vdash A : B$.*

$$(\mathrm{conv}_r) \quad \frac{\Gamma \vdash A : B \qquad \Gamma \vdash B' : s \qquad \Gamma \Vdash B =_r B'}{\Gamma \vdash A : B'}$$

Fig. 7. (conv_r) where $r = \beta$ or $r = \beta\Pi$

$$(\text{app}_\backslash) \quad \frac{\Gamma \vdash F : \Pi_{x:A}.B \qquad \Gamma \vdash a : A}{\Gamma \vdash Fa : B[x{:=}a]}$$

Fig. 8. (app_Π)

$$(\text{i-app}_\Pi) \quad \frac{\Gamma \vdash F : \Pi_{x:A}.B \qquad \Gamma \vdash a : A}{\Gamma \vdash Fa : (\Pi_{x:A}.B)a}$$

Fig. 9. (i-app_Π)

Proof. By induction on the derivation $\Delta \vdash A : B$. We do the let case. Assume $\Delta \vdash (\backslash_{x:A}.C)B : D[x := B]$ comes from $\Delta, x = B{:}A \vdash C : D$ where $x \notin \text{DOM}(\Delta)$ (else rename x). By start lemma on the derivation tree to $\Delta, x = B{:}A \vdash C : D$ we have $\Delta \vdash B : A$ and $\Delta \vdash A : s$. Hence by IH, $\Gamma \vdash B : A$ and $\Gamma \vdash A : s$. Hence, by (start-a), $\Gamma, x = B{:}A \vdash x : A$ and $\Gamma, x = B{:}A$ is legal. Furthermore, by start lemma, $\Gamma, x = B{:}A \Vdash x =_r B$. Hence, $\Gamma, x = B{:}A \vdash \Delta, x = B{:}A$. By IH, $\Gamma, x = B{:}A \vdash C : D$ and by $\text{let}_\backslash$,$\Gamma \vdash (\backslash_{x:A}.C)B : D[x := B]$. ⊠

Lemma 12 (Thinning Lemma for $\vdash$ and $\to_r$)

1. *If Γ and Δ are $\vdash$-legal, $\Gamma \subseteq' \Delta$, and $\Gamma \Vdash A =_r B$ then $\Delta \Vdash A =_r B$.*
2. *If Γ and Δ are $\vdash$-legal, $\Gamma \subseteq' \Delta$, and $\Gamma \vdash A : B$ then $\Delta \vdash A : B$.*

Lemma 13 (Substitution Lemma for $\vdash$ and $\to_r$)

1. *If $\Gamma, d, \Delta \Vdash B =_r C$, d is a-dec, and B, C are $\Gamma, d, \Delta^{\vdash}$-legal then $\Gamma, \Delta_d \Vdash B_d =_r C_d$.*
2. *If B is Γ, d-legal and d is a-dec then $\Gamma, d \Vdash B =_r B_d$.*
3. *If $\Gamma, d, \Delta \vdash B : C$ and d is a-dec then $\Gamma, \Delta_d \vdash B_d : C_d$.*
4. *If $\Gamma, d, \Delta \vdash B : C$ and $\Gamma \vdash A : \texttt{type}(d)$ then $\Gamma, \Delta[\texttt{var}(d) := A] \vdash B[\texttt{var}(d) := A] : C[\texttt{var}(d) := A]$.*

Proof. 1. By induction on the derivation $\Gamma, d, \Delta \Vdash B =_r C$.
2. By induction on the derivation $\Gamma, d, \Delta \vdash A : B$ we show that $\Gamma, d, \Delta \Vdash A =_r A_d$ and $\Gamma, d, \Delta \Vdash B =_r B_d$.
3. and 4. By induction on the derivation $\Gamma, d, \Delta \vdash B : C$. ⊠

Lemma 14

1. *If $\Gamma \vdash A : B$ then $\Box$ does not occur in A, Γ, and if $\Box$ occurs in B then $B \equiv \Box$.*
2. *If $\Gamma \vdash A : B$ then $A \neq_r \Box$ and if $B =_r \Box$ then $B \equiv \Box$.*
3. *In all cubes that don't use $\text{let}_\backslash$, $\Gamma \not\vdash AB : \Box$.*
4. *If $\text{let}_\backslash$ is permissible then we can have $\Gamma \vdash AB : \Box$.*
5. *Let $(\Lambda, r) \in \{(\Pi, \beta), (\Pi, \beta\Pi)\}$. If $\Gamma \vdash A : \Box$ then $A =_r \Lambda^{i:1..l}_{x_i:A_i}.*$ where $l \geq 0$ and $\Gamma \vdash \Lambda^{i:1..l}_{x_i:A_i}.* : \Box$.*
6. *If $\Gamma \vdash \pi^1_{x_1:A_1}\pi^2_{x_2:A_2}\ldots\pi^l_{x_l:A_l}.* : A$ where $\pi \in \{\lambda, \Pi\}$ and $l \geq 0$ then $\pi^i = \Pi$ for all $1 \leq i \leq l$ and $A =_\beta \Box$ (hence $A \equiv \Box$).*

$$(\text{Q}_\beta) \quad \frac{\Gamma \vdash \lambda^{i:1..k}_{x_i:A_i}.A : \Pi^{i:1..n}_{x_i:A_i}.*}{\Gamma \vdash \lambda^{i:1..m}_{x_i:A_i}.\Pi^{i:m+1..k}_{x_i:A_i}A : \Pi^{i:1..m}_{x_i:A_i}.*} \quad 0 \leq m \leq n,\ A \not\equiv \lambda_{x:B}.C$$

Fig. 10. (Q_β)

7. *If $\Gamma \vdash \Pi^{i:1..l}_{x_i:A_i}.* : \Box$ then $\Gamma \vdash \Pi^{i:1..p}_{x_i:A_i}.* : \Box$, $\Gamma, x_1 : A_1, x_2 : A_2, \ldots, x_p : A_p \vdash \Pi^{i:p+1..l}_{x_i:A_i}.* : \Box$ and $\Gamma, x_1 : A_1, x_2 : A_2, \ldots, x_{p-1} : A_{p-1} \vdash A_p : s_p$ for some sort s_p where $(s_p, \Box) \in \boldsymbol{R}$ and $1 \leq p \leq l$.*
8. *If $\Gamma \vdash \lambda_{x:A}.B : C$ then $C \neq_r s$.*
9. *If $\Gamma \vdash A : \Box$ then for $A_1, A_2, \ldots A_l$ where $l \geq 0$, $\Gamma \vdash \Pi^{i:1..l}_{x_i:A_i}.* : \Box$ and*
   - *If $let_\backslash$ is not permissible, then $A \equiv \Pi^{i:1..l}_{x_i:A_i}.*$.*
   - *If let_Π is not permissible, then $A =_\beta \Pi^{i:1..l}_{x_i:A_i}.*$.*
   - *If let_Π is permissible, then $A =_{\beta\Pi} \Pi^{i:1..l}_{x_i:A_i}.*$.*
10. *Rule Q_β and rule $(s, \Box)$ for $s \in \{*, \Box\}$ imply rule $(s, *)$.*
    This means that the type systems $\lambda_Q\underline{\omega}$ and $\lambda_Q\omega$ are equal, and that $\lambda_Q P\underline{\omega}$ and $\lambda_Q P\omega$ are equal as well.

3 Desirable properties

In this section we study the desirable properties of our cubes. Note that these are generalised versions of those of the standard β-cube because they type more terms. Unless otherwise stated, $\vdash$ ranges over $\vdash_c$ for any of $c \in \{\pi_i, \beta_a, \pi, \pi_a, \beta_Q\}$.

Lemma 15 (Generation Lemma for $\vdash$ and $\to_r$)

1. *If $\Gamma \vdash s : C$ then $s \equiv *$ and $C \equiv \Box$.*
2. *If $\Gamma \vdash x : C$ then for some d in Γ, $x \equiv \mathtt{var}(d)$, $\Gamma \vdash C : s$ and $\Gamma \vdash \mathtt{type}(d) : s$ for some sort s. For all systems that exclude rule (Q), $\Gamma \Vdash \mathtt{type}(d) =_r C$. In β_Q, $\mathtt{type}(d) \leq_\beta C$.*
3. *If $\Gamma \vdash \Pi_{x:A}.B : C$ then there is $(s_1, s_2) \in \boldsymbol{R}$ such that $\Gamma \vdash A : s_1$, $\Gamma, x{:}A \vdash B : s_2$, and if $C \not\equiv s_2$ then $\Gamma \vdash C : s$ for some sort s. For all systems that exclude rule (Q), $\Gamma \Vdash C =_r s_2$. In β_Q, $C =_\beta s_2$.*
4. *If $\Gamma \vdash \lambda_{x:A}.b : C$ then there are s and B where $\Gamma \vdash \Pi_{x:A}.B : s$, $\Gamma, x{:}A \vdash b : B$, and if $C \not\equiv \Pi_{x:A}.B$ then $\Gamma \vdash C : s'$ for some sort s'. For all systems that exclude rule (Q), $\Gamma \Vdash \Pi_{x:A}.B =_r C$. In β_Q, $\Pi x{:}A.B \leq_\beta C$.*
5. (a) *If abbreviations are not included then: If $\Gamma \vdash Fa : C$ then $\exists A, B$ with $\Gamma \vdash F : \Pi_{x:A}.B$, $\Gamma \vdash a : A$ and if $C \not\equiv T$ then $\Gamma \vdash C : s$ for some s, where:*
      - *$T \equiv B[x{:=}a]$ if unreduced typing i-app is not used;*
      - *$T \equiv (\Pi_{x:A}.B)a$ otherwise.*

      For all systems that exclude rule (Q), $\Gamma \Vdash T =_r C$. In β_Q, $T \leq_\beta C$.
   (b) *If abbreviations are included then for all systems that exclude rule (Q):*
      i. *If $\Gamma \vdash Fa : C$ and $F \not\equiv \pi_{y:D}.E$ then there are A, B such that $\Gamma \vdash F : \Pi_{x:A}.B$, $\Gamma \vdash a : A$ and $\Gamma \Vdash C =_r T$ and if $C \not\equiv T$ then $\Gamma \vdash C : s$ for some s, where $T \equiv B[x{:=}a]$ if unreduced typing is not used, and $T \equiv (\Pi_{x:A}.B)a$ otherwise.*
      ii. *If $\Gamma \vdash (\pi_{y:D}.E)a : C$ then $\Gamma, y = a : D \vdash E : C$.*

Lemma 16 (Correctness of types for $\vdash$ and $\to_r$) *In all systems except $\vdash_{\pi_i}$: If $\Gamma \vdash A : B$ then ($B \equiv \Box$ or $\Gamma \vdash B : s$ for some sort s).*

Proof. By induction on the derivation $\Gamma \vdash A : B$ using the substitution lemma. We only do the Q_β rule. If $\Gamma \vdash \lambda^{i:1..m}_{x_i:A_i}.\Pi^{i:m+1..k}_{x_i:A_i}A : \Pi^{i:1..m}_{x_i:A_i}.*$ comes from $\Gamma \vdash \lambda^{i:1..k}_{x_i:A_i}.A : \Pi^{i:1..n}_{x_i:A_i}.*$ then since $\Pi^{i:1..n}_{x_i:A_i}.* \not\equiv \Box$, by IH, $\Gamma \vdash \Pi^{i:1..n}_{x_i:A_i}.* : s$ for some sort s. By lemma 14.6 and 14.7, we have $\Gamma \vdash \Pi^{i:1..n}_{x_i:A_i}.* : \Box$. For a counterexample and a weaker form of this lemma for $\vdash_{\pi_i}$, see Section 4.1. $\boxtimes$

Lemma 17 (Subject Reduction for $\vdash$ and $\to_r$) *Let $r \in \{\beta, \beta\Pi\}$. In all systems except $\vdash_{\pi_i}$: If $\Gamma \vdash A : B$ and $A \twoheadrightarrow_r A'$ then $\Gamma' \vdash A : B$.*

Proof. First, we prove by simultaneous induction the following:
1. If $\Gamma \vdash A : B$ and $A \to_r A'$ then $\Gamma \vdash A' : B$.
2. If $\Gamma \vdash A : B$ and $\Gamma \to_r \Gamma'$ then $\Gamma \vdash A' : B$.

Then, we prove the lemma by induction on the derivation $A \twoheadrightarrow_r A'$. For a counterexample and a weaker form of this lemma for $\vdash_{\pi_i}$, see Section 4.1. $\boxtimes$

Lemma 18 (Reduction preserves types for $\vdash$ and $\to_r$) *Let $r \in \{\beta, \beta\Pi\}$. In all systems except $\vdash_{\pi_i}$: If $\Gamma \vdash A : B$ and $B \twoheadrightarrow_r B'$ then $\Gamma \vdash A : B'$.*

Proof. Standard using subject reduction and corrrectness of types. First, note that $B =_r B'$. By correctness of types, either $B \equiv \Box$ (hence $B' \equiv \Box$ and we are done) or $\Gamma \vdash B : s$ for some sort s in which case $\Gamma \vdash B' : s$ by subject reduction and hence by conv_r, $\Gamma \vdash A : B'$. Again, for $\vdash_{\pi_i}$, see Section 4.1. $\boxtimes$

The next 3 lemmas will be studied for each cube in the relevant sections.

Lemma 19 (Strong Normalisation for $\vdash$ and $\to_r$)
If A is $\vdash$-legal then $SN_{\to_r}(A)$.

Lemma 20 (Typability of subterms for $\vdash$ and $\to_r$)
If A is $\vdash$-legal and B is a subterm of A, then B is $\vdash$-legal.

Lemma 21 (Unicity of Types for $\vdash$ and $\to_r$)
1. *If $\Gamma \vdash A : B_1$ and $\Gamma \vdash A : B_2$, then $\Gamma \Vdash B_1 =_r B_2$.*
2. *If $\Gamma \vdash A_1 : B_1$ and $\Gamma \vdash A_2 : B_2$ and $\Gamma \Vdash A_1 =_r A_2$, then $\Gamma \Vdash B_1 =_r B_2$.*
3. *If $\Gamma \vdash B_1 : s$, $\Gamma \Vdash B_1 =_r B_2$ and $\Gamma \vdash A : B_2$ then $\Gamma \vdash B_2 : s$.*

4 Connecting the various extensions of the cube

In this section we will connect the various extensions of the cube and we will complete the properties of $\vdash_c$ where $c \in \{\pi_i, \beta_a, \pi_{ai}, \pi, \pi_a, Q_\beta\}$.

Lemma 22
1. *Let $c \in \{\beta, \pi_i, \beta_a, \pi\}$. Then: $\Gamma \not\vdash_c (\Pi_{x:A}.B)a : C$ and if $\Gamma \vdash_\beta A : B$ then Γ, A and B are all free of Π-redexes.*
2. *Terms of the form $(\Pi_{x:A}.B)a$ can be $\vdash_{\pi_i}$-legal, but, $\Gamma \not\vdash_{\pi_i} (\Pi_{x:A}.B)a : C$.*
3. *If $\Gamma \vdash_{\pi_i} A : B$ then Γ and A are free of Π-redexes and B is the only possible Π-redex in B.*
4. *Let $c \in \{\pi_{ai}, \pi_a\}$. $(\Pi_{x:A}.B)a$ can be $\vdash_c$-typable and we can have $\Gamma \vdash_c AB : \Box$.*
5. *We can have $\Gamma \vdash_{\beta_a} (\lambda_{x:A}.B)a : \Box$.*
6. *Let $c \in \{\pi_{ai}, \pi_a\}$. If $\Gamma \Vdash A =_\beta B$ then $\Gamma \Vdash A =_{\beta\Pi} B$. Moreover, If $\Gamma \vdash_c A : B$ then any of Γ, A and B may contain Π-redexes.*
7. *Let $c \in \{\beta, \pi_i, \beta_a, \beta_{ai}\}$. If $\Pi_{x:A}.B$ is $\vdash_c$-legal then $\Gamma \vdash_c \Pi_{x:A}.B : s$.*

8. *a) If* $\Gamma \vdash_\beta A : B$ *then* $\Gamma \vdash_{\pi_i} A : B$. *b) If* $\Gamma \vdash_{\pi_i} A : B$ *then* $\Gamma \vdash_\beta A : [B]_\Pi$.
   c) If $\Gamma \vdash_{\pi_i} A : B$ *and* B *is free of* Π*-redexes then* $\Gamma \vdash_\beta A : B$.
   d) $\vdash_\beta \subset \vdash_{\pi_i}$
9. *a) If* $\Gamma \vdash_\beta A : B$ *then* $\Gamma \vdash_{\beta_a} A : B$.
   b) If $\Gamma \vdash_{\beta_a} A : B$ *then* $\Gamma \vdash_{\pi_{ai}} A : B$.
   c) If $\Gamma \vdash_{\pi_i} A : B$ *then* $\Gamma \vdash_{\pi_{ai}} A : B$ *but the opposite does not hold.*
   d) If $\Gamma \vdash_{\pi_{ai}} A : B$ *then* $\widetilde{\Gamma} \vdash_{\beta_a} \widetilde{A} : \widetilde{B}$.
10. *It does not hold that* $\Gamma \vdash_{\beta_a} A : B$ *for* Γ *free of a-decs implies* $\Gamma \vdash_\beta A : B$.
11. $\vdash_\beta \subset \vdash_{\beta_a} \subset \vdash_{\pi_{ai}}$.
12. *a) If* $\Gamma \vdash_{\beta_a} A : B$ *then* $\Gamma \vdash_{\pi_a} A : B$.
    b) If $\Gamma \vdash_{\pi_a} A : B$ *then* $\Gamma \vdash_{\pi_{ai}} A : B$.
    c) It is possible that $\Gamma \vdash_{\pi_a} A : B$ *but* $\Gamma \not\vdash_{\beta_a} A : B$. *Hence* $\vdash_{\beta_a} \subset \vdash_{\pi_a}$.
13. *Let* $\Gamma \vdash_\pi A : B$ *and* $R \in \{\to, \twoheadrightarrow\}$. *If* $AR_{\beta\Pi}A'$ *then* $AR_\beta A'$.
14. $\Gamma \vdash_\beta A : B$ *if and only if* $\Gamma \vdash_\pi A : B$.
15. *Assume* $\mathtt{var}(d) \notin \mathrm{FV}(A) \cup \mathrm{FV}(B) \cup \mathrm{FV}(\Delta)$. *Then:*
    - *If* $\Gamma, d, \Delta \vdash_{\pi_a} A : B$ *then* $\Gamma, \Delta \vdash_{\pi_a} A : B$.
    - *If* $\Gamma, d, \Delta \Vdash A =_{\beta\Pi} B$ *then* $\Gamma, \Delta \Vdash A =_{\beta\Pi} B$.
16. *a.* $\Gamma \vdash_{\pi_a} A : B$ *if and only if* $\Gamma \vdash_{\pi_{ai}} A : B$.
    b. $\vdash_\pi = \vdash_\beta \subset \vdash_r \subset \vdash_{\pi_{ai}} = \vdash_{\pi_a}$ *for* $r \in \{\beta_a, \pi_i\}$.
    c. $\vdash_{\beta_a}$ *and* $\vdash_{\pi_i}$ *are unrelated.*

4.1 The π_i-cube: Π-reduction and unreduced typing

[12] provided the π_i-cube which extends the β-cube with both Π-reduction and unreduced typing. In addition to the success of Automath in using these notions, there are many arguments as to why such notions are useful; the reader is refered to [11, 12, **?**]. Here, we complete the results for the π_i-cube. [12] showed that Lemmas 15 and 19 as well as the following hold for the π_i-cube:

Lemma 23 (See [12])

1. *A restricted correctness of types Lemma 16: If* $\Gamma \vdash_{\pi_i} A : B$ *and* B *is not a* Π*-redex then* ($B \equiv \Box$ *or* $\Gamma \vdash_{\pi_i} B : s$ *for some sort* s).
2. *A weak subject reduction Lemma 17: If* $\Gamma \vdash_{\pi_i} A : B$ *and* $A \twoheadrightarrow_{\beta\Pi} A'$ *then* $\Gamma \vdash_{\pi_i} A' : [B]_\Pi$.
3. *A weak reduction preserves types Lemma 18: If* $\Gamma \vdash_{\pi_i} A : B$ *and* $B \twoheadrightarrow_{\beta\Pi} B'$ *then* $\Gamma \vdash_{\pi_i} A : [B']_\Pi$.
4. *An almost unicity of Types Lemma 21 where clause 3 is restricted to* β*: If* $\Gamma \vdash_{\pi_i} B_1 : s$, $B_1 =_\beta B_2$ *and* $\Gamma \vdash_{\pi_i} A : B_2$ *then* $\Gamma \vdash_{\pi_i} B_2 : s$.

Items 1, 3 and 8 of Lemma 22 can be understood to imply that the π_i-cube is an almost trivial extension of the β-cube. If $\Gamma \vdash_{\pi_i} A : B$ then $\Gamma \vdash_\beta A : [B]_\Pi$ but whereas B can be a Π-redex, $[B]_\Pi$ cannot. Since by item 2 of Lemma 22, $\Gamma \not\vdash_{\pi_i} (\Pi_{x:A}.B)a : C$, the new legal terms $(\Pi_{x:A}.B)a$ cannot have type s. Hence, since also $(\Pi_{x:A}.B)a \not\equiv \Box$, we lose correctness of types and hence subject reduction:

Example 24 *Let* $\Gamma = z : *, x : z$, $A \equiv (\lambda_{y:z}.y)x$ *and* $B \equiv (\Pi_{y:z}.z)x$. *We have* $\Gamma \vdash_{\pi_i} A : B$, $B \not\equiv \Box$ *and by Lemma 22,* $\Gamma \not\vdash_{\pi_i} B : s$. *Hence we lose correctness of types. Also,* $A \to_{\beta\Pi} x$ *but* $\Gamma \not\vdash_{\pi_i} x : B$ *and we lose subject reduction.*

In addition to *weak* correctness of types/subject reduction (cf. Lemma 23):

Lemma 25 (Restricted Subject reduction/reduction preserves types)

1. *If* $\Gamma \vdash_{\pi_i} A : B$*,* B *is not a* Π*-redex and* $A \twoheadrightarrow_{\beta\Pi} A'$ *then* $\Gamma \vdash_{\pi_i} A' : B$*.*
2. *If* $\Gamma \vdash_{\pi_i} A : B$*,* B *is not a* Π*-redex and* $B \twoheadrightarrow_{\beta\Pi} B'$ *then* $\Gamma \vdash_{\pi_i} A : B'$*.*

Proof. 1. By Lemma 22.8 c), since B is not a Π-redex, $\Gamma \vdash_\beta A : B$. Hence by subject reduction for the cube, $\Gamma \vdash_\beta A' : B$. Hence, by Lemma 22.8 a), $\Gamma \vdash_{\pi_i} A' : B$. For 2., use Lemma 22.8. ⊠

Finally, we complete the results of [12] by addressing Lemma 20.

Lemma 26 (Restricted typability of subterms for $\vdash_{\pi_i}$ and $\rightarrow_{\beta\Pi}$) *If* $\Gamma \vdash_{\pi_i} A : B$ *then every subterm of* A *and every proper subterm of* B *is* $\vdash_{\pi_i}$*-legal.*

Proof. By induction on the derivation $\Gamma \vdash_{\pi_i} A : B$ using Lemma 22.7. ⊠

4.2 Completing the β_a- and π_{ai}-cubes: abbreviations without/with Π-reduction and unreduced typing

In order to obtain full (rather than weak) correctness of types and subject reduction, [11] proposed the π_{ai}-cube which has in addition to Π-reduction and unreduced typing, the so-called *definitions* or *abbreviations*. If k occurs in a text f (such a text can be a single expression or a list of expressions, e.g. a book), it is sometimes practical to introduce an abbreviation for k, for several reasons.

Of course, for $c \in \{\beta_a, \pi_{ai}\}$, the c-cube is a non trivial extension of the β-cube. [11] showed that Lemma 19 holds for the β_a- and π_{ai}-cubes. Here we study typability of subterms Lemma 20, and unicity of types Lemma 21. Before doing so, let us see explain how the problem of Example 24 disappears in the π_{ai}-cube:

- First, the example is no longer a counterexample for correctness of types:
  By (weak-a) $z : *, x : z, y = x : z \vdash_{\pi_{ai}} z : *$.
  Hence by (let_Π) $z : *, x : z \vdash_{\pi_{ai}} (\Pi_{y:z}.z)x : *[y := x] \equiv *$.
- Second, the example is no longer a counterexample for subject reduction:
  Since $z : *, x : z \vdash_{\pi_{ai}} x : z$, and $z : *, x : z \vdash_{\pi_{ai}} (\Pi_{y:z}.z)x : *$ and
  $z : *, x : z \Vdash z =_{\beta\Pi} (\Pi_{y:z}.z)x$, we use ($\mathrm{conv}_{\beta\Pi}$) to get:
  $z : *, x : z \vdash_{\pi_{ai}} x : (\Pi_{y:z}.z)x$.

As for typability of subterms Lemma 20, it only holds in a restricted form in all the cubes that have abbreviations. For this we need the *bachelor* notion: Let $\backslash \in \{\lambda, \Pi\}$; we say that $\backslash_{x:D}$ is bachelor in B if there are no E, F such that $(\backslash_{x:D}.E)F$ is a subterm of B.

Lemma 27 (Restricted typability of subterms for $\vdash$ and $\rightarrow_r$) *If* A *is* $\vdash$*-legal and* B *is a subterm of* A *such that every bachelor* $\lambda_{x:D}$ *in* B *is also bachelor in* A*, then* B *is* $\vdash$*-legal.*

The next example (adapted from [4]), shows why typability of subterms fails in the β_a- and π_{ai}-cubes when the bachelor condition is dropped.

Example 28 *Let $c \in \{\beta_a, \pi_{ai}\}$ and let $\overline{\beta_a} = \beta$ and $\overline{\pi_{ai}} = \beta\Pi$. We have the following derivation (we miss out obvious steps):*

1. $\alpha : *, \beta = \alpha : *, y : \beta \vdash_c y : \beta$
2. $\alpha : *, \beta = \alpha : *, y : \beta \vdash_c y : \alpha$ *by 1, conv*$_{\overline{c}}$
3. $\alpha : *, \beta = \alpha : *, y : \beta, z = y : \alpha \vdash_c z : \alpha$ *by 2, start-a*
4. $\alpha : *, \beta = \alpha : *, y : \beta \vdash_c (\lambda_{z:\alpha}.z)y : \alpha$ *by 3, let*$_\lambda$
5. $\alpha : *, \beta = \alpha : *, y : \beta \vdash_c (\lambda_{z:\alpha}.z)y : \beta$ *by 4, conv*$_{\overline{c}}$
6. $\alpha : *, \beta = \alpha : * \vdash_c \lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y : \Pi_{y:\beta}.\beta$ *by 5,* λ
7. $\alpha : * \vdash_c (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ *by 6, let*$_\lambda$

However, $\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y$ is not $\vdash_c$-legal. To show this, assume, it is $\vdash_c$-legal. Hence, by correctness of types and Lemma 22, there is Γ, A such that $\Gamma \vdash_c \lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y : A$. Then, by four applications of the generation lemma, there is α', s such that $\Gamma' \Vdash \alpha =_{\overline{c}} \alpha'$ and $\Gamma' \vdash \alpha' : s$ where $\Gamma' = \beta : *, y : \beta, z = y : \alpha$. Now it is easy to show that $\Gamma' \Vdash \alpha =_{\overline{c}} \beta$ and $\Gamma' \not\Vdash \alpha =_{\overline{c}} \beta$, contradiction.

The appendix shows the unicity of types Lemma 21 for the β_a- and π_{ai}-cubes.

4.3 The π-cube

Lemmas 15..16 and 20 hold for the π-cube and have the same proofs as the β-cube. As for subject reduction Lemma 17 and strong normalisation Lemma 19:

Proof (Subject Reduction for $\vdash_\pi$ and $\to_{\beta\Pi}$). Similar to the β-cube as by Lemma 22, in the (app) case, it is not possible that F be of the form $\Pi_{y:C}.D$ in $\Gamma \vdash_\pi Fa : B[x := a]$. Or, use the isomorphism with the β-cube given in lemma 22. ⊠

Proof (Strong Normalisation for $\vdash_\pi$ and $\to_{\beta\Pi}$). By correctness of types, we only need to show that if $\Gamma \vdash_\pi A : B$ then $\mathrm{SN}_{\to_{\beta\Pi}}(A)$. By Lemma 22, $\Gamma \vdash_\beta A : B$ and by Lemma 19 $\mathrm{SN}_{\to_\beta}(A)$. If there is an infinite path $A \to_{\beta\Pi} A_1 \to_{\beta\Pi} A_2 \ldots$ then by Lemma 22, there is an infinite path $A \to_\beta A_1 \to_\beta A_2 \ldots$. Absurd. ⊠

Finally, Unicity of types lemma 21 holds for the π-cube and can be easily established using the isomorphism with the β-cube given in lemma 22.

4.4 The π_a-cube: allowing Π-reduction and abbreviations

Since $\vdash_{\pi_a}$ and $\vdash_{\pi_{ai}}$ are the same relation and the π_a- and π_{ai}-cubes have the same terms, contexts and reduction relation, we have that in the π_a-cube the remaining subject reduction, reduction preserves types, strong normalisation and typability of subterms have the same status as in the π_{ai}-cube. They all hold except for typability of subterms which is restricted as in Lemma 27.

4.5 The Q-cube

De Bruijn's system AUT-QE had the rule $\dfrac{\Gamma \vdash A : \Pi^{i:1..n}_{x_i:A_i}.*}{\Gamma \vdash A : \Pi^{i:1..m}_{x_i:A_i}.*} 0 \le m \le n$. However, in AUT-QE, Π and λ are identified. This is not the case in the β-Cube which motivated us to formulate the rule as in Q_β. We will call the type systems that result from adding Q_β to $\lambda \to$, $\lambda 2$, $\lambda\mathrm{P}$, etc.: $\lambda_{\mathrm{Q}\to}$, $\lambda_{\mathrm{Q}2}$, λ_{QP}, etc..

One might worry that by this rule we can show unexpected things. E.g., if $m = n = 0$ and $k = 1$ we may think that we could show $\Gamma \vdash \lambda_{x_1:A_1}.A : *$ and $\Gamma \vdash \Pi_{x_1:A_1}.A : *$. This is not the case because by lemma 22, $\Gamma \nvdash \lambda_{x:A}.B : s$.

Unicity of types lemma 21 fails for the β_Q-cube. Take: $A : *, x : \Pi_{y:A}.* \vdash x : \Pi_{y:A}.*$ and hence by Q_β, $A : *, x : \Pi_{y:A}.* \vdash x : *$. We have shown that Unicity of Types is not provable in any system with the strength of at least $\lambda_{\mathrm{Q}}P$.

5 Conclusion

De Bruijn introduced the type inclusion rule to allow the well typed behaviour of definitions. Since Automath, numerous systems have studied notions of subtyping (e.g., [9, 1, 14]). However, there is still no study of modern type systems with de Bruijn's type inclusion. This paper bridges the gap and studies the systems of the Barendregt cube with type inclusions showing that 4 systems turn into two systems and that unicity of types fails.

References

1. David Aspinall and Adriana Compagnoni. Subtyping dependent types. *Theoretical Computer Science*, 266: 273-309. 2001.
2. H.P. Barendregt. *The Lambda Calculus: its Syntax and Semantics*. Studies in Logic and the Foundations of Mathematics 103. North-Holland. 1984.
3. H.P. Barendregt. Lambda calculi with types. In *Handbook of Logic in Computer Science, Volume 2*, S. Abramsky, Dov M. Gabbay, and T.S.E. Maibaum, editors, pages 117–309. Oxford University Press, 1992.
4. R. Bloo, F. Kamareddine, and R. P. Nederpelt. The Barendregt Cube with Definitions and Generalised Reduction. *Information and Computation* 126:123–143, 1996.
5. N.G. de Bruijn. The mathematical language AUTOMATH, its usage and some of its extensions. In M. Laudet, D. Lacombe, and M. Schuetzenberger, editors, *Symposium on Automatic Demonstration*, pages 29–61, IRIA, Versailles, 1968. Springer Verlag, Berlin, 1970. Lecture Notes in Mathematics **125**.
6. H. B. Curry and R. Feys. *Combinatory Logic I.* Studies in Logic and the Foundations of Mathematics. North-Holland, Amsterdam, 1958.
7. A. Heyting. *Mathematische Grundlagenforschung. Intuitionismus. Beweistheorie.* Ergebnisse der Mathematik und ihrer Grenzgebiete. Springer-Verlag, Berlin, 1934.
8. W.A. Howard. The formulas-as-types notion of construction. In Hindley and Seldin 1980, pages 479–490, 1980.
9. DeLesley Hutchins. Pure Subtype Systems. *Proceedings of the 37th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages.* 2010.
10. F. Kamareddine. Typed lambda calculi with unified binders. *Journal of Functional Programming*, Volume 15, no. (5), pages 771-796, September 2005, ISSN: 0956-7968. Cambridge University Press.
11. F. Kamareddine, R. Bloo, and R. Nederpelt. On π-conversion in the λ-cube and the combination with abbreviations. *Annals of Pure and Applied Logic*, 97:27–45, 1999.
12. F. Kamareddine and R.P. Nederpelt. Canonical typing and Π-conversion in the Barendregt Cube. *Journal of Functional Programming*, 6(2):245–267, 1996.
13. A. N. Kolmogorov. Zur Deutung der Intuitionistischen Logik. *Mathematisches Zeitschrift*, 35:58–65, 1932.
14. Jan Zwanenburg. Pure Type Systems with Subtyping. TLCA 1991, LNCS 381:396.

A Proofs

This appendix gives the proofs for lemmas 12, 14, 15, 22, 27, as well as the unicity of types Lemma 21 for the β_a- and π_{ai}-cubes.

Proof (Thinning Lemma 12).

1. First show by induction on $\Gamma \Vdash A =_r B$ that if Γ and Δ are $\vdash$-legal then:
   - If $\Gamma \equiv \Gamma_1, \Gamma_2 \subseteq' \Gamma_1, d, \Gamma_2 \equiv \Delta$ and $\Gamma \Vdash A =_r B$ then $\Delta \Vdash A =_r B$.
   - If $\Gamma \equiv \Gamma_1, x : A, \Gamma_2 \subseteq' \Gamma_1, x = B : A, \Gamma_2 \equiv \Delta$ and $\Gamma \Vdash A =_r B$ then $\Delta \Vdash A =_r B$.

   Then, show the statement by induction on $\Gamma \subseteq' \Delta$.
2. First show by induction on $\Gamma \vdash A : B$ that if Γ and Δ are $\vdash$-legal then:
   - If $\Gamma \equiv \Gamma_1, \Gamma_2 \subseteq' \Gamma_1, d, \Gamma_2 \equiv \Delta$ and $\Gamma \vdash A : B$ then $\Delta \vdash A : B$.
   - If $\Gamma \equiv \Gamma_1, x : A, \Gamma_2 \subseteq' \Gamma_1, x = B : A, \Gamma_2 \equiv \Delta$ and $\Gamma \vdash A : B$ then $\Delta \vdash A : B$.

   Then, show the statement by induction on $\Gamma \subseteq' \Delta$. ⊠

Proof (Lemma 14).

1. By induction on the derivation $\Gamma \vdash A : B$.
2. This is a corollary of 1. above.
3. By induction on the derivation $\Gamma \vdash AB : \Box$.
4. Since $y : *, x = y : * \vdash * : \Box$, then $y : * \vdash (\lambda_{x:*}.*)y : y$.
5. By induction on the derivation $\Gamma \vdash A : \Box$ using Start/Context Lemma 10 to show that the start and start-a rules do not apply, 1. above to show that conv$_r$ and app$_\backslash$ do not apply, and Substitution Lemma 13.
6. By induction on the derivation $\Gamma \vdash \pi^1_{x_1:A_1}\pi^2_{x_2:A_2}\dots\pi^l_{x_l:A_l}.* : A$ using 1. above. Then, use 2. to deduce $A \equiv \Box$.
7. By induction on the derivation $\Gamma \vdash \Pi^{i:1..l}_{x_i:A_i}.* : \Box$. Conv$_r$ and Q$_\beta$ don't apply.
8. By induction on the derivation $\Gamma \vdash \lambda_{x:A}.B : C$.
9. By induction on the derivation $\Gamma \vdash A : \Box$.
10. Assume $\Gamma \vdash A : s$ and $\Gamma, x{:}A \vdash B : *$. Then:

(1)	$\Gamma, x{:}A \vdash *{:}\Box$	(by the Start Lemma)
(2)	$\Gamma \vdash (\Pi_{x:A}.*) : \Box$	($(s,\Box)$ on (1))
(3)	$\Gamma \vdash (\lambda_{x:A}.B) : (\Pi_{x:A}.*)$	((λ) on (2))
(4)	$\Gamma \vdash (\Pi_{x:A}.B) : *$	(Rule (Q) on (3)) ⊠

Proof (Generation Lemma 15). 1. By induction on the derivation $\Gamma \vdash s : C$. The Q-rule does not apply.

2. By induction on the derivation $\Gamma \vdash x : C$. We only do the Q-rule. Assume $\Gamma \vdash x : *$ comes from $\Gamma \vdash x : \Pi^{i:1..n}_{x_i:A_i}.*$. By IH, there is d in Γ such that $x \equiv \mathtt{var}(d)$, $\Gamma \vdash \Pi^{i:1..n}_{x_i:A_i}.* : s$, $\Gamma \vdash \mathtt{type}(d) : s$ for some sort s and $\mathtt{type}(d) \leq_\beta \Pi^{i:1..n}_{x_i:A_i}.*$. Since $\Pi^{i:1..n}_{x_i:A_i}.* \leq_\beta *$, by Lemma 1, $\mathtt{type}(d) \leq_\beta *$. By lemma 14, $s \equiv \Box$ and $\Gamma \vdash * : \Box$.

3., 4., and 5.: By induction on the generation $\Gamma \vdash M : C$. We only do the new cases: the Q-rule and the difficult case of $\vdash_{\pi_a}$. First the Q-rule in $\vdash_{\beta_Q}$.

Assume $M \equiv \lambda^{i:1..m}_{x_i:A_i}.\Pi^{i:m+1..k}_{x_i:A_i}.M'$, M' is not of the form $\lambda_{x:N_1}.N_2$, $C \equiv \Pi^{i:1..m}_{x_i:A_i}.*$, $m \leq n$ and $\Gamma \vdash M : C$ because $\Gamma \vdash \lambda^{i:1..k}_{x_i:A_i}.M' : \Pi^{i:1..n}_{x_i:A_i}.*$. Write $U \equiv \lambda^{i:1..k}_{x_i:A_i}.M'$ and $W \equiv \Pi^{i:1..n}_{x_i:A_i}.*$.

i. $M \equiv \Pi_{x_1:A_1}.B$. Then $m = 0$, $k > 0$, and we used Rule (Q) to derive:

$$\frac{\Gamma \vdash \lambda^{i:1..k}_{x_i:A_i}.M' : \Pi^{i:1..n}_{x_i:A_i}.*}{\Gamma \vdash \Pi^{i:1..k}_{x_i:A_i}.M' : *}$$

By IH, there are s, B such that $\Gamma, x_1{:}A_1 \vdash \lambda^{i:2..k}_{x_i:A_i}.M' : B$, $\Gamma \vdash \Pi_{x_1:A_1}.B : s$, and $\Pi_{x_1:A_1}.B \leq_\beta \Pi^{i:1..n}_{x_i:A_i}.*$ and if $\Pi_{x_1:A_1}.B \not\equiv \Pi^{i:1..n}_{x_i:A_i}.*$ then $\Gamma \vdash \Pi^{i:1..n}_{x_i:A_i}.* : \Box$ (note Lemma 14). By Lemma 14, $\Gamma, x_1 : A_1 \vdash \Pi^{i:2..n}_{x_i:A_i}.* : \Box$ and there is s_1 such that $\Gamma \vdash A_1 : s_1$ and $(s_1, \Box) \in \boldsymbol{R}$, hence also, $(s_1, *) \in \boldsymbol{R}$. By Corollary 5.2, $B \leq_\beta \Pi^{i:2..n}_{x_i:A_i}.*$. Determine $B' =_\beta B$ where $B' \leq \Pi^{i:2..n}_{x_i:A_i}.*$, say $B' \equiv \Pi^{i:2..\ell}_{x_i:A_i}.*$ where $l \geq n$ and $\Gamma, x_1 : A_1 \vdash \Pi^{i:2..\ell}_{x_i:A_i}.* : \Box$. By conversion, $\Gamma, x_1{:}A_1 \vdash \lambda^{i:2..k}_{x_i:A_i}.M' : \Pi^{i:2..\ell}_{x_i:A_i}.*$, and as M' is not of the form $\lambda_{x:N_1}.N_2$, we can use (Q) and obtain $\Gamma, x_1{:}A_1 \vdash \Pi^{i:2..k}_{x_i:A_i}.M' : *$. Since $\Gamma \vdash A_1 : s_1$ and $(s_1, *) \in \boldsymbol{R}$ we are done.

ii. $M \equiv \lambda_{x_1:A_1}.b$. Then $k > 0$ and $b \equiv \lambda^{i:2..m}_{x_i:A_i}.\Pi^{i:m+1..k}_{x_i:A_i}.M'$. By the induction hypothesis there are s, B such that $\Gamma \vdash \Pi_{x_1:A_1}.B : s$, $\Gamma, x_1{:}A_1 \vdash \lambda^{i:2..k}_{x_i:A_i}.M' : B$ and $\Pi_{x_1:A_1}.B \leq_\beta \Pi^{i:1..n}_{x_i:A_i}.*$ and if $\Pi^{i:1..n}_{x_i:A_i}.* \not\equiv \Pi_{x_1:A_1}.B$ then $\Gamma \vdash \Pi^{i:1..n}_{x_i:A_i}.* : \Box$ (note Lemma 14). Note that $\Pi_{x_1:A_1}.B \leq_\beta \Pi^{i:1..n}_{x_i:A_i}.* \leq_\beta \Pi^{i:1..m}_{x_i:A_i}.*$, so by Lemma 1, $\Pi_{x_1:A_1}.B \leq_\beta \Pi^{i:1..m}_{x_i:A_i}.*$. Determine $B' =_\beta \Pi_{x_1:A_1}.B$ such that $B' \leq \Pi^{i:1..n}_{x_i:A_i}.*$ and $\Gamma \vdash B' : \Box$. We can write $B' \equiv \Pi^{i:1..\ell}_{x_i:A_i}.*$ for an ℓ such that $m \leq n \leq \ell$. Distinguish two cases:
   - $k \leq m$. Then $M \equiv \lambda^{i:1..k}_{x_i:A_i}.M'$, $b \equiv \lambda^{i:2..k}_{x_i:A_i}.M'$ and hence $\Gamma, x_1{:}A_1 \vdash b : B$.
   - $k > m$. Then $M \equiv \lambda^{i:1..m}_{x_i:A_i}.\Pi^{i:m+1..k}_{x_i:A_i}.M'$. By conversion, $\Gamma, x_1{:}A_1 \vdash \lambda^{i:2..k}_{x_i:A_i}.M' : B'$, and as M' is not of the form $\lambda_{x:N_1}.N_2$, and $m \leq n \leq \ell$, we get by (Q) that $\Gamma, x_1{:}A_1 \vdash \lambda^{i:2..m}_{x_i:A_i}.\Pi^{i:m+1..k}_{x_i:A_i}.M' : \Pi^{i:2..m}_{x_i:A_i}.*$.

iii. $M \equiv AB$. Then $k = m = 0$, so $U \equiv AB$. By induction there are x, P, Q such that $\Gamma \vdash A : \Pi_{x:P}.Q$, $\Gamma \vdash B : P$ and $Q[x{:=}B] \leq_\beta W$. Notice that $W \leq_\beta C \equiv *$, so by Lemma 1, $B \leq_\beta C$.

Next we do the case 5(b)ii. of $\vdash_{\pi_a}$. By induction on the derivation rules we first prove that if $\Gamma \vdash (\pi_{y:D}.E)a : C$ then one of the following holds:

- $\Gamma, y = a : D \vdash E : H$ and $\Gamma \Vdash H[y := a] =_{\beta\Pi} C$ and if $H[y := a] \not\equiv C$ then $\Gamma \vdash C : s$ for some s.
- $\Gamma \vdash a : F$, $\Gamma \vdash \lambda_{y:D}.E : \Pi_{z:F}.G$, $\Gamma \Vdash C =_{\beta\Pi} G[z := a]$ and if $G[z := a] \not\equiv C$ then $\Gamma \vdash C : s$ for some s.

If the first case holds, then by substitution and thinning, $\Gamma, y = a : D \Vdash H[y := a] =_{\beta\Pi} H$ and $\Gamma, y = a : D \Vdash H[y := a] =_{\beta\Pi} C$. Hence, $\Gamma, y = a : D \Vdash H =_{\beta\Pi} C$ and we use $\mathrm{conv}_{\beta\Pi}$ to get $\Gamma, y = a : D \vdash E : C$.
In the second case, by generation case 3. on $\Gamma \vdash \lambda_{y:D}.E : \Pi_{z:F}.G$ we get $\Gamma, y : D \vdash E : L$, $\Gamma \Vdash \Pi_{y:D}.L =_{\beta\Pi} \Pi_{z:F}.G$ and if $\Pi_{y:D}.L \not\equiv \Pi_{z:F}.G$ then $\Gamma \vdash \Pi_{z:F}.G : s'$ for some s'. Hence $y = z$ and $\Gamma \Vdash D =_{\beta\Pi} F$ and $\Gamma \Vdash L =_{\beta\Pi} G$. Now, using generation case 4. we prove that $\Gamma, y = a : D \vdash E : L$. Since $\Gamma \Vdash C =_{\beta\Pi} G[y := a]$ we get $\Gamma, y = a : D \Vdash C =_{\beta\Pi} G$. Since $\Gamma \Vdash L =_{\beta\Pi} G$ we get $\Gamma, y = a : D \Vdash L =_{\beta\Pi} G$. Hence, $\Gamma, y = a : D \Vdash L =_{\beta\Pi} C$. We show that

$\Gamma, y = a : D \vdash C : s''$ for some sort s''. Hence using $\Gamma, y = a : D \vdash E : L$ and $\mathrm{conv}_{\beta\Pi}$, we get $\Gamma, y = a : D \vdash E : C$. $\boxtimes$

Proof (Connecting cubes Lemma 22).

1. If $\Gamma \vdash_c (\Pi_{x:A}.B)a : C$, then by Lemma 15, $\exists A', B'$ such that $\Gamma \vdash_c \Pi_{x:A}.B : \Pi_{y:A'}.B'$. Again by Lemma 15, $\Gamma \Vdash \Pi_{y:A'}.B' =_r s_2$ for sort s_2, contradicting Church Rosser.
   As for the second statement, first show by induction on the derivation $\Gamma, x : C, \Delta \vdash_c A : B$ that if both A and a are free of Π-redexes, $\Gamma, x : C, \Delta \vdash_c A : B$ and $\Gamma \vdash_c a : C$, then $A[x := a]$ is free of Π-redexes. Then show the statement by induction on $\Gamma \vdash_c A : B$.
2. Take for example $z : *, x : z \vdash_{\pi_i} (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$ and hence terms of the form $(\Pi_{x:A}.B)a$ can be $\vdash_{\pi_i}$-legal. It is the new legal terms that led to the loss of correctness of types of the π_i-cube and hence of subject reduction because they can not be typable.
3. By induction on $\Gamma \vdash_{\pi_i} A : B$.
4. $z : *, x : z \vdash_c (\Pi_{y:z}.z)x : *$ and $z : * \vdash_c (\lambda_{y:*}.*)z : \Box$ provide examples.
5. $y : * \vdash_{\beta_a} (\lambda_{x:*}.*)y : \Box$.
6. Note that $=_\beta \subseteq =_{\beta\Pi}$.
   Also, note that $z : *, x : z \vdash_c (\Pi_{y:z}.z)x : *$ and $z : *, x : z \vdash_c x : (\Pi_{y:z}.z)x$.
   Note also that $z : *, x : z, y = (\Pi_{y:z}.z)x : * \vdash_c y : *$.
7. By correctness (resp. restricted correctness) of types, it is enough to show that if $\Gamma \vdash_c \Pi_{x:A}.B : C$ then $\Gamma \vdash_c \Pi_{x:A}.B : s$. We do this by induction on the derivation $\Gamma \vdash_c \Pi_{x:A}.B : C$.
8. a) By induction on the derivation $\Gamma \vdash_\beta A : B$ using the substitution lemma for the π_i-cube and 7 above. b) By induction on the derivation $\Gamma \vdash_{\pi_i} A : B$.
   c) By b) $\Gamma \vdash_\beta A : [B]_\Pi$. Since B is free of Π-redexes, $B = [B]_\Pi$ and $\Gamma \vdash_\beta A : B$.
   d) Using a), it is enough to find Γ, A, B such that $\Gamma \vdash_{\pi i} A : B$ but $\Gamma \not\vdash_\beta A : B$. We know that $z : *, x : z \vdash_{\pi_i} (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$ but by 3 above, $z : *, x : z \not\vdash_\beta (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$.
9. a) holds since the rules of $\vdash_\beta$ are a subset of the rules of $\vdash_{\beta_a}$.
   b) is by induction on $\Gamma \vdash_{\beta_a} A : B$.
   c) holds because the rules of $\vdash_{\pi_i}$ are a subset of the rules of $\vdash_{\pi_{ai}}$. As for strict inclusion, note that $\alpha : * \vdash_{\pi_{ai}} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ but $\alpha : * \not\vdash_{\pi_i} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ since we don't have $y : \alpha$.
   d) by induction on $\Gamma \vdash_{\pi_{ai}} A : B$. We only do the i-app rule. Let $\Gamma \vdash_{\pi_{ai}} Fa : (\Pi_{x:A}.B)a$ come from $\Gamma \vdash_{\pi_{ai}} F : \Pi_{x:A}.B$ and $\Gamma \vdash_{\pi_{ai}} a : A$. By IH, $\widetilde{\Gamma} \vdash_{\beta_a} \widetilde{F} : \widetilde{\Pi_{x:A}.B} \equiv \Pi_{x:\widetilde{A}}.\widetilde{B}$ and $\widetilde{\Gamma} \vdash_{\beta_a} \widetilde{a} : \widetilde{A}$. Hence by app, $\widetilde{\Gamma} \vdash_{\beta_a} \widetilde{F}\widetilde{a} : \widetilde{B}[x := \widetilde{A}]$. Since $\Pi_{x:\widetilde{A}}.\widetilde{B}$ is $\widetilde{\Gamma}^{\vdash_{\beta_a}}$-term, by correctness of types, $\exists s$ such that $\widetilde{\Gamma} \vdash_{\beta_a} \Pi_{x:\widetilde{A}}.\widetilde{B} : s$. Hence by generation, $\widetilde{\Gamma}, x : \widetilde{A} \vdash_{\beta_a} \widetilde{B} : s$. Hence by thinning, $\widetilde{\Gamma}, x = \widetilde{a} : \widetilde{A} \vdash_{\beta_a} \widetilde{B} : s$. By let_λ, $\widetilde{\Gamma} \vdash_{\beta_a} (\lambda_{x:\widetilde{A}}.\widetilde{B})\widetilde{a} : s$. By $\mathrm{conv}_{\beta\Pi}$, $\widetilde{\Gamma} \vdash_{\beta_a} \widetilde{F}\widetilde{a} : (\lambda_{x:\widetilde{A}}.\widetilde{B})\widetilde{a}$. If $\widetilde{F}$ was a Π-term, then by generation, $\widetilde{\Gamma} \Vdash \Pi_{x:\widetilde{A}}.\widetilde{B} =_\beta s_2$ for some s_2 absurd. Hence, $\widetilde{F}\widetilde{a} \equiv \widetilde{Fa}$.

10. $\alpha : * \vdash_{\beta_a} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ (see Example 28). However, $\alpha : * \not\vdash_{\beta} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ since we don't have $y : \alpha$.
Another way to prove this is to assume $\alpha : * \vdash_{\beta} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$. Hence, by correctness of types, $\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y$ is $(\alpha : *)^{\vdash_\beta}$-term and by 9 a) above it is $(\alpha : *)^{\vdash_{\beta_a}}$-legal, contradicting Example 28.
11. For $\vdash_\beta \subset \vdash_{\beta_a}$, use 9.a) and 10. above. For $\vdash_{\beta_a} \subset \vdash_{\pi_{ai}}$, use 9.b) above and this example: $z : *, x : z \vdash_{\pi_{ai}} (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$ but by 1 above, $z : *, x : z \not\vdash_{\beta_a} (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$.
12. a) By induction on the derivation $\Gamma \vdash_{\beta_a} A : B$ using 6 above.
b) By induction on the derivation $\Gamma \vdash_{\pi_a} A : B$. we only do the (app) case. Assume $\Gamma \vdash_{\pi_a} Fa : B[x := a]$ comes from $\Gamma \vdash_{\pi_a} F : \Pi_{x:A}.B$ and $\Gamma \vdash_{\pi_a} a : A$. By IH, $\Gamma \vdash_{\pi_{ai}} F : \Pi_{x:A}.B$ and $\Gamma \vdash_{\pi_{ai}} a : A$ and hence $\Gamma \vdash_{\pi_{ai}} Fa : (\Pi_{x:A}.B)a$ by (i-app). By correctness of types, $\Gamma \vdash_{\pi_{ai}} \Pi_{x:A}.B : s$ for some s and hence by generation, $\Gamma, x : A \vdash_{\pi_{ai}} B : s'$. Since $\Gamma \vdash_{\pi_{ai}} a : A$ then by substitution lemma, $\Gamma \vdash_{\pi_{ai}} B[x := a] : s'$. Now, since $\Gamma \Vdash B[x := a] =_{\beta\Pi} (\Pi_{x:A}.B)a$ we use $(\mathrm{conv}_{\beta\Pi})$ to get $\Gamma \vdash_{\pi_a} Fa : B[x := a]$.
c) Note that $z : *, x : z \vdash_{\pi_a} (\Pi_{y:z}.z)x : *$ but by 4 above, if $\Gamma \vdash_{\beta_a} A : B$ then all of Γ, A and B are free of Π-redexes.
13. a) By 1 above, A is free of Π-redexes.
b) By induction on $A \twoheadrightarrow_{\beta\Pi} A'$. Assume $A \twoheadrightarrow^n_{\beta\Pi} A'' \to_{\beta\Pi} A'$. By subject reduction, $\Gamma \vdash_\pi A'' : B$ and hence by IH, $A \twoheadrightarrow^n_\beta A''$ and $A'' \to_\beta A'$. Hence, $A \twoheadrightarrow_\beta A'$.
14. One direction is trivial because every $\vdash_\beta$-rule is also a $\vdash_\pi$-rule (for (conv_r), note that $=_\beta \subseteq =_{\beta\Pi}$). For the other direction, use induction on $\Gamma \vdash_\pi A : B$. We only show the (conv_r) case. Let $\Gamma \vdash_\pi A : B$ come from $\Gamma \vdash_\pi A : B'$, $\Gamma \vdash_\pi B' : s$ and $B =_{\beta\Pi} B'$. By Church-Rosser, $\exists B''$ such that $B' \twoheadrightarrow^n_{\beta\Pi} B'' \twoheadleftarrow_{\beta\Pi} B$. By Correctness of types, $B \equiv \Box$ or $\exists s'$ such that $\Gamma \vdash_\pi B : s'$. If $B \equiv \Box$ then $B'' \equiv \Box$ and $B' \twoheadrightarrow^n_{\beta\Pi} \Box$, hence by subject reduction and $\Gamma \vdash_\pi B' : s$ we get $\Gamma \vdash_\pi \Box : s$ contradicting 1 above. Hence $\Gamma \vdash_\pi B : s'$ and by 13 above, $B \twoheadrightarrow_\beta B''$. Also, by 13, $B' \twoheadrightarrow_\beta B''$. Hence, $B =_\beta B'$. Hence, by IH and (conv_r), $\Gamma \vdash_\beta A : B$.
15. This is a corollary of item 12 above.
16. a. One direction holds by 12 above. The other direction is by induction on $\Gamma \vdash_{\pi_{ai}} A : B$. Since every $\vdash_{\pi_{ai}}$-rule (except the (i-app) rule) is also a rule of $\vdash_{\pi_a}$, we only deal with the (i-app) case. Assume $\Gamma \vdash_{\pi_{ai}} Fa : (\Pi_{x:A}.B)a$ comes from $\Gamma \vdash_{\pi_{ai}} F : \Pi_{x:A}.B$ and $\Gamma \vdash_{\pi_{ai}} a : A$. By IH, $\Gamma \vdash_{\pi_a} F : \Pi_{x:A}.B$ and $\Gamma \vdash_{\pi_a} a : A$ and hence by (app), $\Gamma \vdash_{\pi_a} Fa : B[x := a]$. Since $\Gamma \Vdash (\Pi_{x:A}.B)a =_{\beta\Pi} B[x := a]$, to derive $\Gamma \vdash_{\pi_a} Fa : (\Pi_{x:A}.B)a$, it is enough to show that $\Gamma \vdash_{\pi_a} (\Pi_{x:A}.B)a : s$ for some s. Since $\Gamma \vdash_{\pi_a} F : \Pi_{x:A}.B$, by correctness of types, $\Gamma \vdash_{\pi_a} \Pi_{x:A}.B : s$ and by generation, $\Gamma, x : A \vdash_{\pi_a} B : s'$ and $\Gamma \vdash_{\pi_a} A : s''$. It is easy to show that $\Gamma, x = a : A$ is legal. Hence, since $\Gamma, x : A \subseteq' \Gamma, x = a : A$, we can use thinning to get $\Gamma, x = a : A \vdash_{\pi_a} B : s'$. And so, by (let), $\Gamma \vdash_{\pi_a} (\Pi_{x:A}.B)a : s'$.
b. $\vdash_\pi = \vdash_\beta$ by 14 above. $\vdash_\beta \subset \vdash_{\beta_a} \subset \vdash_{\pi_{ai}}$ by 9 above. $\vdash_{\pi_{ai}} = \vdash_{\pi_a}$ by a. above. $\vdash_\beta \subset \vdash_{\pi_i}$ by 8 above. $\vdash_{\pi_i} \subset \vdash_{\pi_{ai}}$ by 9 above.
c. $z : *, x : z \vdash_{\pi_i} (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$ but $z : *, x : z \not\vdash_{\beta_a} (\lambda_{y:z}.y)x : (\Pi_{y:z}.z)x$

by 1 above.
Also, $\alpha : * \vdash_{\beta_a} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ but
$\alpha : * \not\vdash_{\pi_i} (\lambda_{\beta:*}.\lambda_{y:\beta}.(\lambda_{z:\alpha}.z)y)\alpha : \Pi_{y:\alpha}.\alpha$ since we don't have $y : \alpha$. ⊠

Proof (Restricted typability of subterms Lemma 27 for $\vdash_{\beta_a} +\!\!\rightarrow_{be}$ and $\vdash_{\pi_{ai}} +\!\!\rightarrow_{\beta\Pi}$). We will prove that:

1. If A is $\vdash$-legal and B is a subterm of A such that every bachelor $\lambda_{x:D}$ in B is also bachelor in A, then B is $\vdash$-legal.
2. If A is $\vdash_{\pi_{ai}}$-legal and B is a subterm of A such that every bachelor $\pi_{x:D}$ in B is also bachelor in A, then B is $\vdash_{\pi_{ai}}$-legal.

Let $c \in \{\beta_a, \pi_{ai}\}$. If $\Gamma \vdash_c C : A$, then by correctness of types, $A \equiv \Box$ (and there is nothing to prove) or $\Gamma \vdash_c A : s$. Hence, it is enough to prove the lemma for $\Gamma \vdash_c A : C$. For 1, we prove this by induction on the derivation that if $\Gamma \vdash_{\beta_a} A : C$ and B is a subterm of A resp. Γ such that every bachelor $\lambda_{x:D}$ in B is also bachelor in A resp. Γ, then B is $\vdash_{\beta_a}$-legal. For 2, we prove this by induction on the derivation that if $\Gamma \vdash_{\pi_{ai}} A : C$ and B is a subterm of A resp. Γ such that every bachelor $\pi_{x:D}$ in B is also bachelor in A resp. Γ, then B is $\vdash_{\pi_{ai}}$-legal. ⊠

Proof (Unicity of Types for $\vdash_{\beta_a} +\!\!\rightarrow_{\beta}$ and for $\vdash_{\pi_{ai}} +\!\!\rightarrow_{\beta\Pi}$).

1. By induction on the structure of A using the generation lemma.
2. First, show by Church-Rosser and subject reduction using 1 that:
   If $\Gamma \vdash_c A_1 : B_1$ and $\Gamma \vdash_c A_2 : B_2$ and $A_1 =_{\overline{c}} A_2$, then $\Gamma \Vdash B_1 =_{\overline{c}} B_2$. (*)
   Then, define
   – $[A]_{\langle\rangle} \equiv A$, $[A]_{\Gamma,x:C} \equiv [A]_\Gamma$ and $[A]_{\Gamma,x=B:C} \equiv [A[x := B]]_\Gamma$.
   – $[x : A]_\Gamma$ as $x : [A]_\Gamma$ and $[x = B : A]_\Gamma$ as $x = [B]_\Gamma : [A]_\Gamma$.
   – Γ^0 as Γ and Γ^n as Γ where n elements $d_1, \ldots, d_n$ of Γ have been replaced by $[d_1]_\Gamma, \ldots, [d_n]_\Gamma$.
   Note that $[A]_{\Gamma,\Gamma'} \equiv [[A]_{\Gamma'}]_\Gamma$, $\Gamma \Vdash A =_{\overline{c}} [A]_\Gamma$, and if $\Gamma \Vdash A_1 =_{\overline{c}} A_2$ then $[A_1]_\Gamma =_{\overline{c}} [A_2]_\Gamma$.
   Now prove by induction on $\Gamma \vdash_c A : B$ that:
   If $\Gamma \vdash_c A : B$ then $\Gamma^n \vdash_c [A]_\Gamma : [B]_\Gamma$ and $\Gamma^n \vdash_c A : B$ for $n \leq$ the number of elements in Γ.
   Finally, assume $\Gamma \vdash_c A_1 : B_1$ and $\Gamma \vdash_c A_2 : B_2$ and $\Gamma \Vdash A_1 =_{\overline{c}} A_2$. Then, $\Gamma \vdash_c [A_1]_\Gamma : [B_1]_\Gamma$, $\Gamma \vdash_c [A_2]_\Gamma : [B_2]_\Gamma$ and $[A_1]_\Gamma =_{\overline{c}} [A_2]_\Gamma$. Hence, by (*), $\Gamma \Vdash [B_1]_\Gamma =_{\overline{c}} [B_2]_\Gamma$. But, $\Gamma \Vdash B_1 =_{\overline{c}} [B_1]_\Gamma$ and $\Gamma \Vdash B_2 =_{\overline{c}} [B_2]_\Gamma$. Hence, $\Gamma \Vdash_c B_1 =_{\overline{c}} B_2$.
3. As $\Gamma \vdash_c A : B_2$, by correctness of types $B_2 \equiv \Box$ or $\Gamma \vdash_c B_2 : s'$ for some s'.
   – If $\Gamma \vdash_c B_2 : s'$ then by 2 above, $\Gamma \Vdash s =_{\overline{c}} s'$. By the proof of 2 above, $s \equiv [s]_\Gamma =_{\overline{c}} [s']_\Gamma \equiv s'$. Hence, $s \equiv s'$ and so, $\Gamma \vdash_c B_2 : s$.
   – If $B_2 \equiv \Box$, we prove that if $\Gamma \Vdash A =_{\overline{c}} \Box$ then $\Gamma \not\vdash_c A : B$. If $\Gamma \Vdash A =_{\overline{c}} \Box$ and $\Gamma \vdash_c A : B$ then by the proof of 2 above, $[A]_\Gamma =_{\overline{c}} [\Box]_\Gamma$ and $\Gamma^n \vdash_c [A]_\Gamma : [B]_\Gamma$ for $n \leq$ the number of elements in Γ. Hence $[A]_\Gamma \twoheadrightarrow_{\overline{c}} \Box$ and by SR, $\Gamma^n \vdash_c \Box : [B]_\Gamma$ contradicting Lemma 22. ⊠