Chapter 5

Type Assignment

The lambda calculus as treated so far is usually referred to as a *type-free* theory. This is so, because every expression (considered as a function) may be applied to every other expression (considered as an argument). For example, the identity function $\mathbf{I} \equiv \lambda x.x$ may be applied to any argument x to give as result that same x. In particular $\mathbf{I}$ may be applied to itself.

There are also typed versions of the lambda calculus. These are introduced essentially in Curry (1934) (for the so called Combinatory Logic, a variant of the lambda calculus) and in Church (1940). Types are usually objects of a syntactic nature and may be assigned to lambda terms. If M is such a term and a type A is assigned to M, then we say 'M *has type* A' or 'M *in* A'; the denotation used for this is $M : A$. For example in some typed systems one has $\mathbf{I} : (A \to A)$, that is, the identity $\mathbf{I}$ may get as type $A \to A$. This means that if x being an argument of $\mathbf{I}$ is of type A, then also the value $\mathbf{I}x$ is of type A. In general, $A \to B$ is the type of functions from A to B.

Although the analogy is not perfect, the type assigned to a term may be compared to the dimension of a physical entity. These dimensions prevent us from wrong operations like adding 3 volt to 2 ampère. In a similar way types assigned to lambda terms provide a partial specification of the algorithms that are represented and are useful for showing partial correctness.

Types may also be used to improve the efficiency of compilation of terms representing functional algorithms. If for example it is known (by looking at types) that a subexpression of a term (representing a funtional program) is purely arithmetical, then fast evaluation is possible. This is because the expression then can be executed by the ALU of the machine and not in the slower way in which symbolic expressions are evaluated in general.

The two original papers of Curry and Church introducing typed versions of the lambda calculus give rise to two different families of systems. In the typed lambda calculi *à la* Curry terms are those of the type-free theory. Each term has a set of possible types. This set may be empty, be a singleton or consist of several (possibly infinitely many) elements. In the systems *à la* Church the terms are annotated versions of the type-free terms. Each term has (up to an equivalence relation) a unique type that is usually derivable from the way the term is annotated.

The Curry and Church approaches to typed lambda calculus correspond to

two paradigms in programming. In the first of these a program may be written without typing at all. Then a compiler should check whether a type can be assigned to the program. This will be the case if the program is correct. A well-known example of such a language is *ML*, see Milner (1984). The style of typing is called *implicit typing*. The other paradigm in programming is called *explicit typing* and corresponds to the Church version of typed lambda calculi. Here a program should be written together with its type. For these languages type-checking is usually easier, since no types have to be constructed. Examples of such languages are *Algol 68* and *Pascal*. Some authors designate the Curry systems as 'lambda calculi *with type assignment*' and the Church systems as 'systems of *typed* lambda calculus'.

Within each of the two paradigms there are several versions of typed lambda calculus. In many important systems, especially those *à la* Church, it is the case that terms that do have a type always possess a normal form. By the unsolvability of the halting problem this implies that not all computable functions can be represented by a typed term, see Barendregt (1990), Theorem 4.2.15. This is not so bad as it sounds, because in order to find such computable functions that cannot be represented, one has to stand on one's head. For example in $\lambda 2$, the second order typed lambda calculus, only those partial recursive functions cannot be represented that happen to be total, but not provably so in mathematical analysis (second order arithmetic).

Considering terms and types as programs and their specifications is not the only possibility. A type A can also be viewed as a proposition and a term M in A as a proof of this proposition. This so called propositions-as-types interpretation is independently due to de Bruijn (1970) and Howard (1980) (both papers were conceived in 1968). Hints in this direction were given in Curry and Feys (1958) and in Läuchli (1970). Several systems of proof checking are based on this interpretation of propositions-as-types and of proofs-as-terms. See e.g. de Bruijn (1980) for a survey of the so called AUTOMATH proof checking system. Normalization of terms corresponds in the formulas-as-types interpretation to normalisation of proofs in the sense of Prawitz (1965). Normal proofs often give useful proof theoretic information, see e.g. Schwichtenberg (1977).

In this section a typed lambda calculus will be introduced in the style of Curry. For more information, see Barendregt (1992).

The system $\lambda\!\!\rightarrow$-Curry

Originally the implicit typing paradigm was introduced in Curry (1934) for the theory of combinators. In Curry and Feys (1958) and Curry et al. (1972) the theory was modified in a natural way to the lambda calculus assigning elements of a given set $\mathbb{T}$ of types to type free lambda terms. For this reason these calculi *à la* Curry are sometimes called *systems of type assignment*. If the type $\sigma \in \mathbb{T}$ is assigned to the term $M \in \Lambda$ one writes $\vdash M : \sigma$, sometimes with a subscript under $\vdash$ to denote the particular system. Usually a set of assumptions Γ is needed to derive a type assignment and one writes $\Gamma \vdash M : \sigma$ (pronounce this as 'Γ yields M in σ'). A particular Curry type assignment system depends on two parameters, the set $\mathbb{T}$ and the rules of type assignment. As an example we

now introduce the system $\lambda\!\!\rightarrow$-Curry.

5.1. DEFINITION. The set of *types* of $\lambda\!\!\rightarrow$, notation Type($\lambda\!\!\rightarrow$), is inductively defined as follows. We write $\mathbb{T} = \text{Type}(\lambda\!\!\rightarrow)$. Let $\mathbb{V} = \{\alpha, \alpha', \ldots\}$ be a set of *type variables*. It will be convenient to allow *type constants* for basic types such as Nat, Bool. Let $\mathbb{B}$ be such a collection. Then

$$\begin{array}{rcl} \alpha \in \mathbb{V} & \Rightarrow & \alpha \in \mathbb{T}, \\ \mathsf{B} \in \mathbb{B} & \Rightarrow & \mathsf{B} \in \mathbb{T}, \\ \sigma, \tau \in \mathbb{T} & \Rightarrow & (\sigma\!\rightarrow\!\tau) \in \mathbb{T} \quad \text{(function space types)}. \end{array}$$

For such definitions it is convenient to use the following abstract syntax to form $\mathbb{T}$.

$$\mathbb{T} = \mathbb{V} \mid \mathbb{B} \mid \mathbb{T}\!\rightarrow\!\mathbb{T}$$

with

$$\mathbb{V} = \alpha \mid \mathbb{V}' \qquad \text{(type variables)}.$$

NOTATION. (i) If $\sigma_1, \ldots, \sigma_n \in \mathbb{T}$ then

$$\sigma_1\!\rightarrow\!\sigma_2\!\rightarrow\cdots\rightarrow\!\sigma_n$$

stands for

$$(\sigma_1\!\rightarrow\!(\sigma_2\!\rightarrow\cdots\rightarrow\!(\sigma_{n-1}\!\rightarrow\!\sigma_n)\cdot\cdot));$$

that is, we use association to the right.

(ii) $\alpha, \beta, \gamma, \ldots$ denote arbitrary type variables.

5.2. DEFINITION. (i) A *statement* is of the form $M : \sigma$ with $M \in \Lambda$ and $\sigma \in \mathbb{T}$. This statement is pronounced as 'M in σ'. The type σ is the *predicate* and the term M is the *subject* of the statement.

(ii) A *basis* is a set of statements with only distinct (term) variables as subjects.

5.3. DEFINITION. Type *derivations* in the system $\lambda\!\!\rightarrow$ are built up from assumptions $x{:}\sigma$, using the following inference rules.

$$\frac{M : \sigma\!\rightarrow\!\tau \qquad N : \sigma}{MN : \tau} \qquad\qquad \frac{\begin{array}{c}\not{x : \sigma} \\ \vdots \\ M : \tau\end{array}}{\lambda x.M : \sigma\!\rightarrow\!\tau}$$

5.4. DEFINITION. (i) A statement $M : \sigma$ is *derivable from* a basis Γ, notation

$$\Gamma \vdash M : \sigma$$

(or $\Gamma \vdash_{\lambda\rightarrow} M : \sigma$ if we wish to stress the typing system) if there is a derivation of $M : \sigma$ in which all non-cancelled assumptions are in Γ.

(ii) We use $\vdash M : \sigma$ as shorthand for $\emptyset \vdash M : \sigma$.

5.5. Example. (i) Let $\sigma \in \mathbb{T}$. Then $\vdash \lambda fx.f(fx) : (\sigma\to\sigma)\to\sigma\to\sigma$, which is shown by the following derivation.

$$\dfrac{\dfrac{\dfrac{\cancel{f:\sigma\to\sigma}^{(2)} \qquad \dfrac{\cancel{f:\sigma\to\sigma}^{(2)} \qquad \cancel{x:\sigma}^{(1)}}{fx:\sigma}}{f(fx):\sigma}}{\lambda x.f(fx):\sigma\to\sigma}\,(1)}{\lambda fx.f(fx):(\sigma\to\sigma)\to\sigma\to\sigma}\,(2)$$

The indices (1) and (2) are bookkeeping devices that indicate at which application of a rule a particular assumption is being cancelled.

(ii) One has $\vdash \mathsf{K} : \sigma\to\tau\to\sigma$ for any $\sigma,\tau \in \mathbb{T}$, which is demonstrated as follows.

$$\dfrac{\dfrac{\cancel{x:\sigma}^{(1)}}{\lambda y.x:\tau\to\sigma}}{\lambda xy.x:\sigma\to\tau\to\sigma}\,(1)$$

(iii) Similarly one can show for all $\sigma \in \mathbb{T}$

$$\vdash \mathsf{I} : \sigma\to\sigma.$$

(iv) An example with a non-empty basis is the statement

$$y{:}\sigma \vdash \mathsf{I}y : \sigma.$$

Properties of $\lambda\to$

Several properties of type assignment in $\lambda\to$ are valid. The first one analyses how much of a basis is necessary in order to derive a type assignment.

5.6. Definition. Let $\Gamma = \{x_1{:}\sigma_1, \ldots, x_n{:}\sigma_n\}$ be a basis.

(i) Write $\mathrm{dom}(\Gamma) = \{x_1, \ldots, x_n\}$ and $\sigma_i = \Gamma(x_i)$. That is, Γ is considered as a partial function.

(ii) Let V_0 be a set of variables. Then $\Gamma \restriction V_0 = \{x{:}\sigma \mid x \in V_0 \,\&\, \sigma = \Gamma(x)\}$.

(iii) For $\sigma,\tau \in \mathbb{T}$ substitution of τ for α in σ is denoted by $\sigma[\alpha := \tau]$.

5.7. Basis Lemma. *Let Γ be a basis.*

(i) *If $\Gamma' \supseteq \Gamma$ is another basis, then*

$$\Gamma \vdash M : \sigma \;\Rightarrow\; \Gamma' \vdash M : \sigma.$$

(ii) $\Gamma \vdash M : \sigma \;\Rightarrow\; \mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$.

(iii) $\Gamma \vdash M : \sigma \;\Rightarrow\; \Gamma \restriction \mathrm{FV}(M) \vdash M : \sigma$.

Proof. (i) By induction on the derivation of $M : \sigma$. Since such proofs will occur frequently we will spell it out in this simple situation in order to be shorter later on.

Case 1. $M : \sigma$ is $x{:}\sigma$ and is element of Γ. Then also $x{:}\sigma \in \Gamma'$ and hence $\Gamma' \vdash M : \sigma$.

Case 2. $M : \sigma$ is $(M_1 M_2) : \sigma$ and follows directly from $M_1 : (\tau{\to}\sigma)$ and $M_2 : \tau$ for some τ. By the IH one has $\Gamma' \vdash M_1 : (\tau{\to}\sigma)$ and $\Gamma' \vdash M_2 : \tau$. Hence $\Gamma' \vdash (M_1 M_2) : \sigma$.

Case 3. $M : \sigma$ is $(\lambda x.M_1) : (\sigma_1{\to}\sigma_2)$ and follows directly from $\Gamma, x : \sigma_1 \vdash M_1 : \sigma_2$. By the variable convention it may be assumed that the bound variable x does not occur in $\mathrm{dom}(\Gamma')$. Then $\Gamma', x{:}\sigma_1$ is also a basis which extends $\Gamma, x{:}\sigma_1$. Therefore by the IH one has $\Gamma', x{:}\sigma_1 \vdash M_1 : \sigma_2$ and so $\Gamma' \vdash (\lambda x.M_1) : (\sigma_1{\to}\sigma_2)$.

(ii) By induction on the derivation of $M : \sigma$. We only treat the case that $M : \sigma$ is $(\lambda x.M_1) : (\sigma_1{\to}\sigma_2)$ and follows directly from $\Gamma, x{:}\sigma_1 \vdash M_1 : \sigma_2$. Let $y \in \mathrm{FV}(\lambda x.M_1)$, then $y \in \mathrm{FV}(M_1)$ and $y \not\equiv x$. By the IH one has $y \in \mathrm{dom}(\Gamma, x{:}\sigma_1)$ and therefore $y \in \mathrm{dom}(\Gamma)$.

(iii) By induction on the derivation of $M : \sigma$. We only treat the case that $M : \sigma$ is $(M_1 M_2) : \sigma$ and follows directly from $M_1 : (\tau{\to}\sigma)$ and $M_2 : \tau$ for some τ. By the IH one has $\Gamma \restriction \mathrm{FV}(M_1) \vdash M_1 : (\tau{\to}\sigma)$ and $\Gamma \restriction \mathrm{FV}(M_2) \vdash M_2 : \tau$. By (i) it follows that $\Gamma \restriction \mathrm{FV}(M_1 M_2) \vdash M_1 : (\tau{\to}\sigma)$and $\Gamma \restriction \mathrm{FV}(M_1 M_2) \vdash M_2 : \tau$ and hence $\Gamma \restriction \mathrm{FV}(M_1 M_2) \vdash (M_1 M_2) : \sigma$. □

The second property analyses how terms of a certain form get typed. It is useful among other things to show that certain terms have no types.

5.8. Generation Lemma. (i) $\Gamma \vdash x : \sigma \;\Rightarrow\; (x{:}\sigma) \in \Gamma$.
(ii) $\Gamma \vdash MN : \tau \;\Rightarrow\; \exists\sigma\, [\Gamma \vdash M : (\sigma{\to}\tau) \,\&\, \Gamma \vdash N : \sigma]$.
(iii) $\Gamma \vdash \lambda x.M : \rho \;\Rightarrow\; \exists\sigma,\tau\, [\Gamma, x{:}\sigma \vdash M : \tau \,\&\, \rho \equiv (\sigma{\to}\tau)]$.

Proof. By induction on the structure of derivations. □

5.9. Proposition (Typability of subterms). *Let M' be a subterm of M. Then*

$$\Gamma \vdash M : \sigma \;\Rightarrow\; \Gamma' \vdash M' : \sigma' \quad \textit{for some } \Gamma' \textit{ and } \sigma'.$$

The moral is: if M has a type, i.e. $\Gamma \vdash M : \sigma$ for some Γ and σ, then every subterm has a type as well.

Proof. By induction on the generation of M. □

5.10. Substitution Lemma.
(i) $\Gamma \vdash M : \sigma \;\Rightarrow\; \Gamma[\alpha := \tau] \vdash M : \sigma[\alpha := \tau]$.
(ii) *Suppose* $\Gamma, x{:}\sigma \vdash M : \tau$ *and* $\Gamma \vdash N : \sigma$. *Then* $\Gamma \vdash M[x := N] : \tau$.

Proof. (i) By induction on the derivation of $M : \sigma$.
(ii) By induction on the derivation showing $\Gamma, x{:}\sigma \vdash M : \tau$. □

The following result states that the set of $M \in \Lambda$ having a certain type in $\lambda{\to}$ is closed under reduction.

5.11. Subject Reduction Theorem. *Suppose $M \twoheadrightarrow_\beta M'$. Then*

$$\Gamma \vdash M : \sigma \;\Rightarrow\; \Gamma \vdash M' : \sigma.$$

PROOF. Induction on the generation of $\twoheadrightarrow_\beta$ using the Generation Lemma 5.8 and the Substitution Lemma 5.10. We treat the prime case, namely that $M \equiv (\lambda x.P)Q$ and $M' \equiv P[x := Q]$. Well, if

$$\Gamma \vdash (\lambda x.P)Q : \sigma$$

then it follows by the Generation Lemma that for some τ one has

$$\Gamma \vdash (\lambda x.P) : (\tau \to \sigma) \text{ and } \Gamma \vdash Q : \tau.$$

Hence once more by the Generation Lemma

$$\Gamma, x{:}\tau \vdash P : \sigma \text{ and } \Gamma \vdash Q : \tau$$

and therefore by the Substitution Lemma

$$\Gamma \vdash P[x := Q] : \sigma. \ \square$$

Terms having a type are not closed under expansion. For example,

$$\vdash \mathbf{I} : (\sigma \to \sigma), \text{ but } \not\vdash \mathbf{KI}\ (\lambda x.xx) : (\sigma \to \sigma).$$

See Exercise 5.1. One even has the following stronger failure of subject expansion, as is observed in van Bakel (1992).

5.12. OBSERVATION. There are $M, M' \in \Lambda$ and $\sigma, \sigma' \in \mathbb{T}$ such that $M' \twoheadrightarrow_\beta M$ and

$$\vdash M : \sigma, \qquad \vdash M' : \sigma',$$

but

$$\not\vdash M' : \sigma.$$

PROOF. Take $M \equiv \lambda xy.y$, $M' \equiv \mathbf{SK}$, $\sigma \equiv \alpha \to (\beta \to \beta)$ and $\sigma' \equiv (\beta \to \alpha) \to (\beta \to \beta)$; do Exercise 5.1. $\square$

All typable terms have a normal form. In fact, the so-called *strong normalization* property holds: if M is a typable term, then all reductions starting from M are finite.

Decidability of type assignment

For the system of type assignment several questions may be asked. Note that for $\Gamma = \{x_1{:}\sigma_1, \ldots, x_n{:}\sigma_n\}$ one has

$$\Gamma \vdash M : \sigma \ \Leftrightarrow \ \vdash (\lambda x_1{:}\sigma_1 \cdots \lambda x_n{:}\sigma_n.M) : (\sigma_1 \to \cdots \to \sigma_n \to \sigma),$$

therefore in the following one has taken $\Gamma = \emptyset$. Typical questions are

(1) Given M and σ, does one have $\vdash M : \sigma$?
(2) Given M, does there exist a σ such that $\vdash M : \sigma$?
(3) Given σ, does there exist an M such that $\vdash M : \sigma$?

These three problems are called *type checking*, *typability* and *inhabitation* respectively and are denoted by $M : \sigma?$, $M : ?$ and $? : \sigma$.

Type checking and typability are decidable. This can be shown using the following result, independently due to Curry (1969), Hindley (1969), and Milner (1978).

5.13. THEOREM. (i) *It is decidable whether a term is typable in* $\lambda\!\!\rightarrow$.

(ii) *If a term* M *is typable in* $\lambda\!\!\rightarrow$, *then* M *has a principal type scheme, i.e. a type* σ *such that every possible type for* M *is a substitution instance of* σ. *Moreover* σ *is computable from* M.

5.14. COROLLARY. *Type checking for* $\lambda\!\!\rightarrow$ *is decidable.*

PROOF. In order to check $M : \tau$ it suffices to verify that M is typable and that τ is an instance of the principal type of M. □

For example, a principal type scheme of **K** is $\alpha\!\rightarrow\!\beta\!\rightarrow\!\alpha$.

Polymorphism

Note that in $\lambda\!\!\rightarrow$ one has

$$\vdash \mathbf{I} : \sigma\!\rightarrow\!\sigma \qquad \text{for all } \sigma \in \mathbb{T}.$$

In the polymorphic lambda calculus this quantification can be internalized by stating

$$\vdash \mathbf{I} : \forall\alpha.\alpha\!\rightarrow\!\alpha.$$

The resulting system is the *polymorphic* of *second-order* lambda calculus due to Girard (1972) and Reynolds (1974).

5.15. DEFINITION. The set of *types* of $\lambda 2$ (notation $\mathbb{T} = \mathrm{Type}(\lambda 2)$) is specified by the syntax

$$\mathbb{T} = \mathbb{V} \mid \mathbb{B} \mid \mathbb{T}\!\rightarrow\!\mathbb{T} \mid \forall\mathbb{V}.\mathbb{T}.$$

5.16. DEFINITION. The rules of type assignment are those of $\lambda\!\!\rightarrow$, plus

$$\frac{M : \forall\alpha.\sigma}{M : \sigma[\alpha := \tau]} \qquad \frac{M : \sigma}{M : \forall\alpha.\sigma}$$

In the latter rule, the type variable α may not occur free in any assumption on which the premiss $M : \sigma$ depends.

5.17. EXAMPLE. (i) $\vdash \mathbf{I} : \forall\alpha.\alpha\!\rightarrow\!\alpha$.

(ii) Define $\mathrm{Nat} \equiv \forall\alpha.(\alpha\!\rightarrow\!\alpha)\!\rightarrow\!\alpha\!\rightarrow\!\alpha$. Then for the Church numerals $\boldsymbol{c}_n \equiv \lambda f x.f^n(x)$ we have $\vdash \boldsymbol{c}_n : \mathrm{Nat}$.

The following is due to Girard (1972).

5.18. THEOREM. (i) *The Subject Reduction property holds for* $\lambda 2$.

(ii) $\lambda 2$ *is strongly normalizing.*

Typability in $\lambda 2$ is *not* decidable; see Wells (1994).

Exercises

5.1. (i) Give a derivation of
$$\vdash \mathbf{SK} : (\alpha\to\beta)\to(\alpha\to\alpha).$$
(ii) Give a derivation of
$$\vdash \mathbf{KI} : \beta\to(\alpha\to\alpha).$$
(iii) Show that $\not\vdash \mathbf{SK} : (\alpha\to\beta\to\beta)$.
(iv) Find a common β-reduct of **SK** and **KI**. What is the most general type for this term?

5.2. Show that $\lambda x.xx$ and $\mathbf{KI}(\lambda x.xx)$ have no type in $\lambda\to$.

5.3. Find the most general types (if they exist) for the following terms.
(i) $\lambda xy.xyy$.
(ii) **SII**.
(iii) $\lambda xy.y(\lambda z.z(yx))$.

5.4. Find terms $M, N \in \Lambda$ such that the following hold in $\lambda\to$.
(i) $\vdash M : (\alpha\to\beta)\to(\beta\to\gamma)\to(\alpha\to\gamma)$.
(ii) $\vdash N : (((\alpha\to\beta)\to\beta)\to\beta)\to(\alpha\to\beta)$.

5.5. Find types in $\lambda 2$ for the terms in the exercises 5.2 and 5.3.

Chapter 1

The systems $\lambda_{\to}$

1.1. The $\lambda_{\to}$ systems *à la* Curry

Types in this part are syntactic objects built from atomic types using the operator $\to$. In order to classify untyped lambda terms, such types will be assigned to a subset of these terms. The main idea is that if M gets type $A{\to}B$ and N gets type A, then the application MN is 'legal' (as M is considered as a function from terms of type A to those of type B) and gets type B. In this way types help determining what terms fit together.

1.1.1. Definition. (i) Let $\mathbb{A}$ be a non-empty set of 'atomic types'. The set of *simple types over* $\mathbb{A}$, notation $\mathbb{T} = \mathbb{T}_{\mathbb{A}}$, is inductively defined as follows.

$$\begin{array}{rcll} \alpha \in \mathbb{A} & \Rightarrow & \alpha \in \mathbb{T} & \text{type atoms;} \\ A, B \in \mathbb{T} & \Rightarrow & (A{\to}B) \in \mathbb{T} & \text{function space types.} \end{array}$$

Such definitions will be used often and for these it is convenient to use the so called *abstract syntax*, see Waite and Goos [1984]. As an example we give the abstract syntax for $\mathbb{T} = \mathbb{T}_{\mathbb{A}}$.

$$\boxed{\mathbb{T} \quad = \quad \mathbb{A} \mid \mathbb{T} \to \mathbb{T}}$$

Figure 1.1: Simple types

(ii) Let $\mathbb{A}_o = \{o\}$. Then we write $\mathbb{T}_o = \mathbb{T}_{\mathbb{A}_o}$.
(iii) Let $\mathbb{A}_\infty = \{\alpha_0, \alpha_1, \alpha_2, \ldots\}$. Then we write $\mathbb{T}_\infty = \mathbb{T}_{\mathbb{A}_\infty}$

We consider that $o = \alpha_0$, hence $\mathbb{T}_o \subseteq \mathbb{T}_\infty$. If we write simply $\mathbb{T}$, then this refers to $\mathbb{T}_{\mathbb{A}}$ for an unspecified $\mathbb{A}$.

1.1.2. Notation. (i) If $A_1, \ldots, A_n \in \mathbb{T}$, then

$$A_1{\to}\ldots{\to}A_n \equiv (A_1{\to}(A_2{\to}\ldots{\to}(A_{n-1}{\to}A_n)..)).$$

That is, we use association to the right (here $\equiv$ denotes syntactic equality).

(ii) $\alpha, \boldsymbol{\beta}, \gamma, \ldots$ denote arbitrary elements of $\mathbb{A}$.
(iii) $A, B, C, \ldots$ denote arbitrary elements of $\mathbb{T}$.

Remember the untyped lambda calculus denoted by λ, see e.g. B[1984][1]. It consists of a set of terms Λ defined by the following abstract syntax.

$$\begin{array}{lcl} \mathsf{V} & = & x \,|\, \mathsf{V}' \\ \Lambda & = & \mathsf{V} \,|\, \lambda \mathsf{V}\ \Lambda \,|\, \Lambda\ \Lambda \end{array}$$

Figure 1.2: Untyped lambda terms

This makes $\mathsf{V} = \{x, x', x'', \ldots\} = \{x_0, x_1, x_2, \ldots\}$.

1.1.3. Notation. (i) $x, y, z, \ldots$ denote arbitrary term variables.
(ii) $M, N, L, \ldots$ denote arbitrary lambda terms.
(iii) $MN_1 \ldots N_k \equiv (..(MN_1) \ldots N_k)$.
(iv) $\lambda x_1 \ldots x_n.M \equiv (\lambda x_1(..(\lambda x_n(M))..))$.

1.1.4. Definition. On Λ the following equational theory $\boldsymbol{\lambda\beta\eta}$ is defined by the usual equality axiom and rules (reflexivity, symmetry, transitivity, congruence), inluding congruence with respect to abstraction:

$$M = N \ \Rightarrow\ \lambda x.M = \lambda x.N,$$

and the following special axiom(schemes)

$$\begin{array}{rcll} (\lambda x.M)N & = & M[x := N] & (\boldsymbol{\beta}\text{-rule}) \\ \lambda x.Mx & = & M, \quad \text{if } x \notin \mathrm{FV}(M) & (\eta\text{-rule}) \end{array}$$

Figure 1.3: The theory $\boldsymbol{\lambda\beta\eta}$

As is know this theory can be analyzed by a notion of reduction.

1.1.5. Definition. On Λ we define the following notions of reduction

$$\begin{array}{rcll} (\lambda x.M)N & \rightarrow & M[x{:=}N] & (\boldsymbol{\beta}) \\ \lambda x.Mx & \rightarrow & M, \quad \text{if } x \notin \mathrm{FV}(M) & (\boldsymbol{\eta}) \end{array}$$

Figure 1.4: $\boldsymbol{\beta\eta}$-contraction rules

As usual, see B[1984], these notions of reduction generate the corresponding reduction relations $\rightarrow_{\boldsymbol{\beta}}, \twoheadrightarrow_{\boldsymbol{\beta}}, \rightarrow_{\boldsymbol{\eta}}, \twoheadrightarrow_{\boldsymbol{\eta}}, \rightarrow_{\boldsymbol{\beta\eta}}$ and $\twoheadrightarrow_{\boldsymbol{\beta\eta}}$. Also there are the corresponding conversion relations $=_{\boldsymbol{\beta}}, =_{\boldsymbol{\eta}}$ and $=_{\boldsymbol{\beta\eta}}$. Terms in Λ will often be considered modulo $=_{\boldsymbol{\beta}}$ or $=_{\boldsymbol{\beta\eta}}$. If we write $M = N$, then we mean $M =_{\boldsymbol{\beta\eta}} N$ by default. (In B[1984] the default was $=_{\boldsymbol{\beta}}$.)

1.1.6. Proposition. *For all $M, N \in \Lambda$ one has*

$$\vdash_{\boldsymbol{\lambda\beta\eta}} M = N \iff M =_{\boldsymbol{\beta\eta}} N.$$

[1]This is an abbreviation fo the reference Barendregt [1984].

PROOF. See B[1984], Proposition 3.3.2. ■

One reason why the analysis in terms of the notion of reduction $\boldsymbol{\beta\eta}$ is useful is that the following holds.

1.1.7. THEOREM (Church-Rosser Theorem for $\lambda\boldsymbol{\beta\eta}$). *For the notions of reduction* $\twoheadrightarrow_{\boldsymbol{\beta}}$ *and* $\twoheadrightarrow_{\boldsymbol{\beta\eta}}$ *one has the following.*

(i) *Let* $M, N \in \Lambda$. *Then*

$$M =_{\boldsymbol{\beta}(\eta)} N \;\Rightarrow\; \exists Z \in \Lambda. M \twoheadrightarrow_{\boldsymbol{\beta}(\eta)} Z \;\&\; N \twoheadrightarrow_{\boldsymbol{\beta}(\eta)} Z.$$

(ii) *Let* $M, N_1, N_2 \in \Lambda$. *Then*

$$M \twoheadrightarrow_{\boldsymbol{\beta}(\eta)} N_1 \;\&\; M \twoheadrightarrow_{\boldsymbol{\beta}(\eta)} N_2 \;\Rightarrow\; \exists Z \in \Lambda. N_1 \twoheadrightarrow_{\boldsymbol{\beta}(\eta)} Z \;\&\; N_2 \twoheadrightarrow_{\boldsymbol{\beta}(\eta)} Z.$$

PROOF. (i) See Theorems 3.2.8 and 3.3.9 in B[1984].
(ii) By (i). ■

1.1.8. DEFINITION ($\lambda^{\mathrm{Cu}}_{\to}$). (i) A *(type assignment) statement* is of the form

$$M : A,$$

with $M \in \Lambda$ and $A \in \mathbb{T}$. This statement is pronounced as 'M in A'. The type A is the *predicate* and the term M is the *subject* of the statement.

(ii) A *declaration* is a statement with as subject a term variable.

(iii) A *basis* is a set of declarations with distinct variables as subjects.

(iv) A statement $M{:}A$ is *derivable from* a basis Γ, notation

$$\Gamma \vdash^{\mathrm{Cu}}_{\lambda\to} M{:}A$$

(or $\Gamma \vdash_{\lambda\to} M : A$, $\Gamma \vdash^{\mathrm{Cu}} M : A$ or even $\Gamma \vdash M{:}A$ if there is little danger of confusion) if $\Gamma \vdash M{:}A$ can be produced by the following rules.

$$(x{:}A) \in \Gamma \;\Rightarrow\; \Gamma \vdash x : A;$$

$$\Gamma \vdash M : (A \to B),\;\; \Gamma \vdash N : A \;\Rightarrow\; \Gamma \vdash (MN) : B;$$

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

These rules are usually written as follows.

$$\begin{array}{lll} \text{(axiom)} & \Gamma \vdash x : A, & \text{if } (x{:}A) \in \Gamma; \\ \text{($\to$-elimination)} & \dfrac{\Gamma \vdash M : (A \to B) \quad \Gamma \vdash N : A}{\Gamma \vdash (MN) : B}\,; & \\ \text{($\to$-introduction)} & \dfrac{\Gamma, x{:}A \vdash M : B}{\Gamma \vdash (\lambda x.M) : (A \to B)}\,. & \end{array}$$

Figure 1.5: The system $\lambda^{\mathrm{Cu}}_{\to}$ of type assignment *á la* Curry

This is the modification to the lambda calculus of the system in Curry [1934], as developed in Curry et al. [1958].

NOTATION. Another way of writing these rules is sometimes found in the literature.

$$\begin{array}{ll} \text{Introduction rule} & \dfrac{\begin{array}{c}\cancel{x : A} \\ \vdots \\ M : B\end{array}}{\lambda x.M : (A{\to}B)} \\ \text{Elimination rule} & \dfrac{M : (A \to B) \quad N : A}{MN : B} \end{array}$$

$\lambda^{\mathrm{Cu}}_{\to}$ alternative version

In this version the axiom is considered as implicit and is not notated. The notation

$$\begin{array}{c} x : A \\ \vdots \\ M : B \end{array}$$

denotes that $M : B$ can be derived from $x{:}A$. Striking through $x{:}A$ means that for the conclusion $\lambda x.M : A{\to}B$ the assumption $x{:}A$ is no longer needed; it is *discharged*.

1.1.9. DEFINITION. Let $\Gamma = \{x_1{:}A_1, \ldots, x_n{:}A_n\}$. Then
(i) $\mathrm{dom}(\Gamma) = \{x_1, \ldots, x_n\}$.
(ii) $x_1{:}A_1, \ldots, x_n{:}A_n \vdash M : A$ denotes $\Gamma \vdash M : A$.
(iii) In particular $\vdash M : A$ stands for $\emptyset \vdash M : A$.
(iv) $x_1, \ldots, x_n{:}A \vdash M : B$ stands for $x_1{:}A, \ldots, x_n{:}A \vdash M : B$.

1.1.10. EXAMPLE. (i) $\vdash (\lambda xy.x) : (A \to B \to A)$ for all $A, B \in \mathbb{T}$.
We will use the notation of version 1 of $\lambda_{\to}$ for a derivation of this statement.

$$\dfrac{\dfrac{x{:}A, y{:}B \vdash x : A}{x{:}A \vdash (\lambda y.x) : B{\to}A}}{\vdash (\lambda x \lambda y.x) : A{\to}B{\to}A}$$

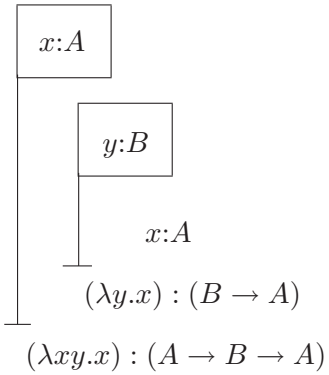
Note that $\lambda xy.x \equiv \lambda x \lambda y.x$ by definition.

(ii) A *natural deduction* derivation (for the alternative version of the system) of the same type assignment is the following.

$$\dfrac{\dfrac{\dfrac{\cancel{x{:}A}\ 2 \qquad \cancel{y{:}B}\ 1}{x{:}A}}{(\lambda y.x)\ :\ (B \to A)}\ 1}{(\lambda xy.x)\ :\ (A \to B \to A)}\ 2$$

The indices 1 and 2 are bookkeeping devices that indicate at which application of a rule a particular assumption is being discharged.

(iii) A more explicit way of dealing with cancellations of statements is the 'flag-notation' used by Fitch (1952) and in the languages AUTOMATH of de Bruijn (1980). In this notation the above derivation becomes as follows.



As one sees, the bookkeeping of cancellations is very explicit; on the other hand it is less obvious how a statement is derived from previous statements in case applications are used.

(iv) Similarly one can show for all $A \in \mathbb{T}$

$$\vdash (\lambda x.x) : (A \to A).$$

(v) An example with a non-empty basis is $y{:}A \vdash (\lambda x.x)y : A$.

In the rest of this chapter and in fact in the rest of this book we usually will introduce systems of typed lambda calculi in the style of the first variant of $\lambda_{\to}$.

1.1.11. DEFINITION. Let Γ be a basis and $A \in \mathbb{T}$. Then

(i) $\Lambda^{\Gamma}_{\to}(A) = \{M \in \Lambda \mid \Gamma \vdash_{\lambda_{\to}} M : A\}$.

(ii) $\Lambda^{\Gamma}_{\to} = \bigcup_{A \in \mathbb{T}} \Lambda^{\Gamma}_{\to}(A)$.

(iii) $\Lambda_{\to}(A) = \Lambda^{\emptyset}_{\to}(A)$.

(iv) $\Lambda_{\to} = \Lambda^{\emptyset}_{\to}$

1.1.12. DEFINITION. Let Γ be a basis, $A \in \mathbb{T}$ and $M \in \Lambda$. Then

(i) If $M \in \Lambda_{\to}(A)$, then we say that

M has type A or *A is inhabited by M.*

(ii) If $M \in \Lambda_{\to}$, then M is called *typable.*

(iii) If $M \in \Lambda_{\to}^{\Gamma}(A)$, then *$M$ has type A relative to Γ.*

(iv) If $M \in \Lambda_{\to}^{\Gamma}$, then M is called *typeable relative to Γ.*

(v) If $\Lambda_{\to}^{(\Gamma)}(A) \neq \emptyset$, then A is *inhabited (relative to Γ).*

1.1.13. EXAMPLE. We have

$$\begin{array}{rcl} \mathsf{K} & \in & \Lambda_{\to}^{\emptyset}(A{\to}B{\to}A); \\ \mathsf{K}x & \in & \Lambda_{\to}^{\{x:A\}}(B{\to}A). \end{array}$$

1.1.14. DEFINITION. Let $A \in \mathbb{T}(\lambda_{\to})$.

(i) The *depth* of A, notation $\mathrm{dpt}(A)$, is defined as follows.

$$\begin{array}{rcl} \mathrm{dpt}(\alpha) & = & 0 \\ \mathrm{dpt}(A{\to}B) & = & \max\{\mathrm{dpt}(A), \mathrm{dpt}(B)\} + 1 \end{array}$$

(ii) The *rank* of A, notation $\mathrm{rk}(A)$, is defined as follows.

$$\begin{array}{rcl} \mathrm{rk}(\alpha) & = & 0 \\ \mathrm{rk}(A{\to}B) & = & \max\{\mathrm{rk}(A) + 1, \mathrm{rk}(B)\} \end{array}$$

(iii) The *order* of A, notation $\mathrm{ord}(A)$, is defined as follows.

$$\begin{array}{rcl} \mathrm{ord}(\alpha) & = & 1 \\ \mathrm{ord}(A{\to}B) & = & \max\{\mathrm{ord}(A) + 1, \mathrm{ord}(B)\} \end{array}$$

(iv) The depth (rank or order) of a basis Γ is

$$\max_i\{\mathrm{dpt}(A_i) \mid (x_i{:}A_i) \in \Gamma\},$$

(similarly for the rank and order, respectively). Note that $\mathrm{ord}(A) = \mathrm{rk}(A) + 1$.

1.1.15. DEFINITION. For $A \in \mathbb{T}$ we define $A^k{\to}B$ by recursion on k:

$$\begin{array}{rcl} A^0{\to}B & = & B; \\ A^{k+1}{\to}B & = & A{\to}A^k{\to}B. \end{array}$$

Note that $\mathrm{rk}(A^k{\to}B) = \mathrm{rk}(A{\to}B)$, for all $k > 0$.

Several properties can be proved by induction on the depth of a type. This holds for example for Lemma 1.1.18(i).

The asymmetry in the definition of rank is intended because e.g. a type like $(o\to o)\to o$ is more complex than $o\to o\to o$, as can be seen by looking to the inhabitants of these types: functionals with functions as arguments versus binary function. Sometimes one uses instead of 'rank' the name *type level*. This notion will turn out to be used most of the times.

In logically motivated papers one finds the notion ord(A). The reason is that in first-order logic one deals with domains and their elements. In second order logic one deals with functions between first-order objects. In this terminology 0-th order logic can be identified with propositional logic.

The minimal and maximal systems $\lambda^o_\to$ and $\lambda^\infty_\to$

The collection $\mathbb{A}$ of type variables serves as set of base types from which other types are constructed. We have $\mathbb{T}_o = \{o\}$ with just one type atom and $\mathbb{T}_\infty = \{\alpha_0, \alpha_1, \alpha_2, \ldots\}$ with infinitely many of them. These two sets of atoms and their resulting type systems play a major role in this Part I of the book.

1.1.16. Definition. We define the following systems of type assignment.
(i) $\lambda^o_\to = \lambda^{\mathbb{T}_o}_\to$. This system is also called λ^τ in the literature.
(ii) $\lambda^\infty_\to = \lambda^{\mathbb{T}_\infty}_\to$.

If it becomes necessary to distinguish the set of atomic types, will use notations like $\Lambda_o(A) = \Lambda_{\mathbb{T}_o}(A)$ and $\Lambda_\infty(A) = \Lambda_{\mathbb{T}_\infty}(A)$.

Many of the interesting features of the 'larger' $\lambda_\to$ are already present in the minimal version $\lambda^o_\to$. The complexity of $\lambda_\to$ is already present in $\lambda^o_\to$.

1.1.17. Definition. (i) The following types of $\mathbb{T}_o \subseteq \mathbb{T}_\mathbb{A}$ are often used.

$$0 = o,\ 1 = 0\to 0,\ 2 = (0\to 0)\to 0,\ \ldots.$$

In general

$$0 = o \text{ and } k+1 = k\to 0.$$

Note that $\mathrm{rk}(n) = n$.

(ii) Define n_k by cases on n.

$$\begin{array}{rcl} o_k & = & o; \\ (n+1)_k & = & n^k\to o. \end{array}$$

For example

$$\begin{array}{rcl} 1_2 & = & o\to o\to o; \\ 2_3 & = & 1\to 1\to 1\to o. \end{array}$$

Notice that $\mathrm{rk}(n_k) = \mathrm{rk}(n)$, for $k > 0$.

1.1.18. LEMMA. (i) *Every type A of $\lambda^{\infty}_{\to}$ is of the form*

$$A = A_1 \to A_2 \to \ldots \to A_n \to \alpha.$$

(ii) *Every type A of $\lambda^{o}_{\to}$ is of the form*

$$A = A_1 \to A_2 \to \ldots \to A_n \to o.$$

(iii) $\mathrm{rk}(A_1 \to A_2 \to \ldots \to A_n \to \alpha) = \max\{\mathrm{rk}(A_i) + 1 \mid 1 \leq i \leq n\}$.

PROOF. (i) By induction on the structure (depth) of A. If $A = \alpha$, then this holds for $n = 0$. If $A = B \to C$, then by the induction hypothesis one has $C = C_1 \to \ldots \to C_n \to \gamma$. Hence $A = B \to C_1 \to \ldots \to C_n \to \gamma$.

(ii) By (i).

(iii) By induction on n. ■

1.1.19. NOTATION. Let $A \in \mathbb{T}_{\mathbb{A}}$ and suppose $A = A_1 \to A_2 \to \ldots \to A_n \to \alpha$. Then the A_i are called the *components* of A. We write

$$\begin{array}{rcll} \mathtt{arity}(A) & = & n, & \\ A(i) & = & A_i, & \text{for } 1 \leq i \leq n; \\ \mathtt{target}(A) & = & \alpha. & \end{array}$$

Iterated components are denoted as follows

$$A(i,j) = A(i)(j).$$

Different versions of $\lambda^{\mathbb{A}}_{\to}$

The system $\lambda^{\mathbb{A}}_{\to}$ that was introduced in Definition 1.1.8 assigns types to untyped lambda terms. These system will be referred to as the Curry system and be denoted by $\lambda^{\mathbb{A}}_{\to \mathrm{Cu}}$ or $\lambda^{\mathrm{Cu}}_{\to}$, as the set $\mathbb{A}$ often does not need to be specified. There will be introduced two variants of $\lambda^{\mathbb{A}}_{\to}$.

The first variant of $\lambda^{\mathrm{Cu}}_{\to}$ is the *Church* version of $\lambda^{\mathbb{A}}_{\to}$, denoted by $\lambda^{\mathbb{A}}_{\to \mathrm{Ch}}$ or $\lambda^{\mathrm{Ch}}_{\to}$. In this theory the types are assigned to embellished terms in which the variables (free and bound) come with types attached. For example the Curry style type assignments

$$\begin{array}{rll} & \vdash^{\mathrm{Cu}}_{\lambda_{\to}} \quad (\lambda x.x) : A \to A & (1_{\mathrm{Cu}}) \\ y{:}A & \vdash^{\mathrm{Cu}}_{\lambda_{\to}} \quad (\lambda x.xy) : (A \to B) \to A \to B & (2_{\mathrm{Cu}}) \end{array}$$

now becoming

$$\begin{array}{ll} (\lambda x^A.x^A) \in \Lambda^{\mathrm{Ch}}_{\to}(A \to A) & (1_{\mathrm{Ch}}) \\ (\lambda x^{A \to B}.x^{A \to B} y^A) : \Lambda^{\mathrm{Ch}}_{\to}((A \to B) \to A \to B) & (2_{\mathrm{Ch}}) \end{array}$$

The second variant of $\lambda_{\to}^{\mathrm{Cu}}$ is the *de Bruijn* version of $\lambda_{\to}^{\mathbb{A}}$, denoted by $\lambda_{\to\mathrm{dB}}^{\mathbb{A}}$ or $\lambda_{\to}^{\mathrm{dB}}$. Now only bound variables get ornamented with types, but only at the binding stage. The examples (1), (2) now become

$$\begin{array}{rll} & \vdash_{\lambda\to}^{\mathrm{dB}} & (\lambda x:A.x):A\to A \qquad (1_{\mathrm{dB}}) \\ y{:}A & \vdash_{\lambda\to}^{\mathrm{dB}} & (\lambda x:(A\to B).xy):(A\to B)\to A\to B \qquad (2_{\mathrm{dB}}) \end{array}$$

The reasons to have these variants will be explained later in Section 1.4. In the meantime we will work intuitively.

1.1.20. NOTATION. Terms like $(\lambda fx.f(fx))\in\Lambda^{\emptyset}(1\to o\to o)$ will often be written

$$\lambda f^1x^0.f(fx)$$

to indicate the types of the bound variables. We will come back to this notational issue in section 1.4.

1.2. Normal inhabitants

In this section we will give an algorithm that enumerates the set of closed terms in normal form of a given type $A\in\mathbb{T}$. Since we will prove in the next chapter that all typable terms do have a nf and that reduction preserves typing, we thus have an enumeration of essentially all closed terms of that given type. We do need to distinguish various kinds of nf's.

1.2.1. DEFINITION. Let $A=A_1\to\ldots A_n\to\alpha$ and suppose $\Gamma\vdash M:A$.

(i) Then M is in long-nf, notation lnf, if $M\equiv\lambda x_1^{A_1}\ldots x_n^{A_n}.xM_1\ldots M_n$ and each M_i is in lnf. By induction on the depth of the type of the closure of M one sees that this definition is well-founded.

(ii) M has a lnf if $M=_{\beta\eta}N$ and N is a lnf.

In Exercise 1.5.16 it is proved that if M has a $\boldsymbol{\beta}$-nf, which according to Theorem 2.2.4 is always the case, then it also has a unique lnf and will be its unique $\boldsymbol{\beta\eta}^{-1}$ nf. Here η^{-1} is the notion of reduction that is the converse of η.

1.2.2. EXAMPLES. (i) Note that $\lambda f^1.f=_{\beta\eta}\lambda f^1\lambda x^o.fx$ and that $\lambda f^1.f$ is a $\boldsymbol{\beta\eta}$-nf but not a lnf.

(ii) $\lambda f^1\lambda x^o.fx$ is a lnf, but not a $\boldsymbol{\beta\eta}$-nf.

(iii) $\lambda x{:}o.x$ is both in $\boldsymbol{\beta\eta}$-nf and lnf.

(iv) The $\boldsymbol{\beta}$-nf $\lambda F{:}2_2\lambda f{:}1.Ff(\lambda x{:}o.fx)$ is neither in $\boldsymbol{\beta\eta}$-nf nor lnf.

(v) A variable of atomic type α is a lnf, but of type $A\to B$ not.

(vi) A variable $f:1\to1$ has as lnf $\lambda g^1\lambda x^o.f(\lambda y^o.gy)x$.

1.2.3. PROPOSITION. *Every $\boldsymbol{\beta}$-nf M has a lnf M^ℓ such that $M^\ell\twoheadrightarrow_{\boldsymbol{\eta}}M$.*

(ix) *If M has the $\lambda\mathsf{K}^-$ property then M β-reduces to only finitely many N.* This follows by (vii) and (viii).

(x) *If M has the $\lambda\mathsf{K}^-$ property then M is strongly β-normalizable.* By (i), (iii) and (ix).

(xi) *If M has the $\lambda\mathsf{K}^-$ property then M is strongly $\beta\eta$-normalizable.* By (v) and (x).

(xii) *For each M there is an N with the $\lambda\mathsf{K}^-$ property such that $N \twoheadrightarrow_{\boldsymbol{\beta\eta}} M$.* First expand M by η expansion so that every subterm of M beginning with a lambda is a lambda prefix followed by a matrix of type 0. Let $a : \alpha$ and $f : 0{\to}(0{\to}0)$ be new variables. For each type $T = T_1{\to}\ldots{\to}T_t{\to}\alpha$ with $T_i = T_{i,1}{\to}\ldots{\to}T_{i,k_i}{\to}\alpha_i$ for $i = 1,\ldots,t$ define terms $U_A : T$ recursively by

$$\begin{aligned} U_0 &= a; \\ U_T &= \lambda x_1 \ldots x_t.f(x_1 U_{T_{1,1}} \ldots U_{T_{1,k_1}}) \ldots \\ &\quad (f(x_{t-1}U_{T_{t-1,1}} \ldots U_{T_{t-1,k_{t-1}}})(x_t U_{T_{t,1}} \ldots U_{T_{t,k_t}}))..). \end{aligned}$$

Now recursively replace each dummy λx occurring $\lambda x\lambda y \ldots \lambda z.X$ with $x : T$ and $X : 0$ by $\lambda x\lambda y \ldots \lambda z.KX(xU_{T_1} \ldots U_{T_t})$. Clearly the resulting N satisfies $N \twoheadrightarrow_{\boldsymbol{\beta\eta}} M$ and the $\lambda\mathsf{K}^-$ property, since all dummy lambdas appear in $K : 1_2$.

(xiii) *Every typable term is strongly $\beta\eta$ normalizable.* By (xi) and (xii). ■

Still another proof is to be found in de Vrijer [1987] in which for a typed term M a computation is given of the langest reduction path to $\boldsymbol{\beta}$-nf.

2.3. Checking and finding types

There are several natural problems concerning type systems.

2.3.1. Definition. (i) The problem of *type checking* consists of determining, given basis Γ, term M and type A whether $\Gamma \vdash M : A$.

(ii) The problem of *typeability* consists of given a term M determining whether M has some type with respect to some Γ.

(iii) The problem of *type reconstruction* ('finding types') consists of finding all possible types A and bases Γ that type a given M.

(iv) The *inhabitation problem* consists of finding out whether a given type A is inhabited by some term M in a given basis Γ.

(v) The *enumeration problem* consists of determining for a given type A and a given context Γ all possible terms M such that $\Gamma \vdash M : A$.

The five problems may be summarized stylistically as follows.

$$\begin{aligned} \Gamma \;&\vdash_{\lambda\to}\; M : A\,? && \textit{type checking}; \\ \exists A, \Gamma\,[\Gamma \;&\vdash_{\lambda\to}\; M : A]\,? && \textit{typeability}; \\ ? \;&\vdash_{\lambda\to}\; M : \;? && \textit{type reconstruction}; \\ \exists M\,[\Gamma \;&\vdash_{\lambda\to}\; M : A]\,? && \textit{inhabitation}; \\ \Gamma \;&\vdash_{\lambda\to}\; \;? : A && \textit{enumeration.} \end{aligned}$$

In another notation this is the following.

$$\begin{array}{rll} & M \in \Lambda^{\Gamma}_{\to}(A)\,? & \textit{type checking};\\ \exists A,\Gamma & M \in \Lambda^{\Gamma}_{\to}(A)? & \textit{typeability};\\ & M \in \Lambda^{?}_{\to}(?) & \textit{type reconstruction};\\ & \Lambda^{\Gamma}_{\to}(A) \neq \emptyset\,? & \textit{inhabitation};\\ & ? \in \Lambda^{\Gamma}_{\to}(A) & \textit{enumeration.}\end{array}$$

In this section we will treat the problems of type checking, typeability and type reconstruction for the three versions of $\lambda_{\to}$. It turns out that these problems are decidable for all versions. The solutions are essentially simpler for $\lambda^{\mathrm{Ch}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$ than for $\lambda^{\mathrm{Cu}}_{\to}$. The problems of inhabitation and enumeration will be treated in the next section.

One may wonder what is the role of the context Γ in these questions. The problem

$$\exists\Gamma\exists A\ \Gamma \vdash M : A.$$

can be reduced to one without a context. Indeed, for $\Gamma = \{x_1{:}A_1, \ldots, x_n{:}A_n\}$

$$\Gamma \vdash M : A \ \Leftrightarrow\ \vdash (\lambda x_1(:A_1)\ldots\lambda x_n(:A_n).M) : (A_1 \to \ldots \to A_n \to A).$$

Therefore

$$\exists\Gamma\exists A\ [\Gamma \vdash M : A] \iff \exists B\ [\vdash \lambda\vec{x}.M : B].$$

On the other hand the question

$$\exists\Gamma\exists M\ [\Gamma \vdash M : A]\,?$$

is trivial: take $\Gamma = \{x{:}A\}$ and $M \equiv x$. So we do not consider this question.

The solution of the problems like type checking for a fixed context will have important applications for the treatment of constants.

Checking and finding types for $\lambda^{\mathrm{dB}}_{\to}$ and $\lambda^{\mathrm{Ch}}_{\to}$

We will see again that the systems $\lambda^{\mathrm{Ch}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$ are essentially equivalent. For these systems the solutions to the problems of type checking, typeability and type reconstruction are easy. All of the solutions are computable with an algorithm of linear complexity.

2.3.2. Proposition (Type checking for $\lambda^{\mathrm{dB}}_{\to}$). *Let Γ be a basis of $\lambda^{\mathrm{dB}}_{\to}$. Then there is a computable function $\mathtt{type}_\Gamma : \Lambda_{\mathbb{T}} \to \mathbb{T} \cup \{\mathtt{error}\}$ such that*

$$M \in \Lambda^{\Gamma}_{\to\mathrm{Ch}}(A) \iff \mathtt{type}_\Gamma(M) = A.$$

Proof. Define

$$\begin{array}{rcll} \mathtt{type}_\Gamma(x) & = & \Gamma(x); & \\ \mathtt{type}_\Gamma(MN) & = & B, & \text{if } \mathtt{type}_\Gamma(M) = \mathtt{type}_\Gamma(N){\to}B,\\ & = & \mathtt{error}, & \text{else;}\\ \mathtt{type}_\Gamma(\lambda x{:}A.M) & = & A{\to}\mathtt{type}_{\Gamma\cup\{x:A\}}(M), & \text{if } \mathtt{type}_{\Gamma\cup\{x:A\}}(M) \neq \mathtt{error},\\ & = & \mathtt{error}, & \text{else.}\end{array}$$

Then the statement follows by induction on the structure of M. ■

2.3.3. Corollary. *Typeability and type reconstruction for $\lambda^{\mathrm{dB}}_{\to}$ are computable. In fact one has the following.*

(i) $M \in \Lambda^{\Gamma}_{\to\mathrm{dB}} \iff \mathtt{type}_{\Gamma}(M) \neq \mathtt{error}$.

(ii) *Each $M \in \Lambda^{\Gamma}_{\to\mathrm{dB}}(\mathtt{type}_{\Gamma})$ has a unique type; in particular*

$$M \in \Lambda^{\Gamma}_{\to\mathrm{dB}}(\mathtt{type}_{\Gamma}(M)).$$

Proof. By the proposition. ■

For $\lambda^{\mathrm{Ch}}_{\to}$ things are essentially the same, except that there are no bases needed, since variables come with their own types.

2.3.4. Proposition (Type checking for $\lambda^{\mathrm{Ch}}_{\to}$). *There is a computable function* $\mathtt{type} : \Lambda_{\to\mathrm{Ch}} \to \mathbb{T} \cup \{\mathtt{error}\}$ *such that*

$$M \in \Lambda_{\to\mathrm{Ch}}(A) \iff \mathtt{type}(M) = A.$$

Proof. Define

$$\begin{array}{rcll} \mathtt{type}(x^A) & = & A; & \\ \mathtt{type}(MN) & = & B, & \text{if } \mathtt{type}(M) = \mathtt{type}(N){\to}B, \\ & = & \mathtt{error}, & \text{else;} \\ \mathtt{type}(\lambda x^A.M) & = & A{\to}\mathtt{type}(M), & \text{if } \mathtt{type}(M) \neq \mathtt{error}, \\ & = & \mathtt{error}, & \text{else.} \end{array}$$

Then the statement follows again by induction on the structure of M. ■

2.3.5. Corollary. *Typeability and type reconstruction for $\lambda^{\mathrm{Ch}}_{\to}$ are computable. In fact one has the following.*

(i) $M \in \Lambda_{\to\mathrm{Ch}} \iff \mathtt{type}(M) \neq \mathtt{error}$.

(ii) *Each $M \in \Lambda_{\to\mathrm{Ch}}$ has a unique type; in particular $M \in \Lambda_{\to\mathrm{Ch}}(\mathtt{type}(M))$.*

Proof. By the proposition. ■

Checking and finding types for $\lambda^{\mathrm{Cu}}_{\to}$

We now will show the computability of the three questions for $\lambda^{\mathrm{Cu}}_{\to}$. This occupies 2.3.6 - 2.3.16 and in these items $\vdash$ stands for $\vdash^{\mathbb{T}^{\infty}}_{\lambda_{\to\mathrm{Cu}}}$.

Let us first make the easy observation that in $\lambda^{\mathrm{Cu}}_{\to}$ types are not unique. For example $\mathsf{I} \equiv \lambda x.x$ has as possible type $\alpha{\to}\alpha$, but also $(\boldsymbol{\beta}{\to}\boldsymbol{\beta}){\to}(\boldsymbol{\beta}{\to}\boldsymbol{\beta})$ and $(\alpha{\to}\boldsymbol{\beta}{\to}\boldsymbol{\beta}){\to}(\alpha{\to}\boldsymbol{\beta}{\to}\boldsymbol{\beta})$. Of these types $\alpha{\to}\alpha$ is the 'most general' in the sense that the other ones can be obtained by a substitution in α.

2.3.6. Definition. (i) A substitutor is an operation $* : \mathbb{T} \to \mathbb{T}$ such that

$$*(A \to B) \equiv *(A) \to *(B).$$

(ii) We write A^* for $*(A)$.

(iii) Usually a substitution $*$ has a finite support, that is, for all but finitely many type variables α one has $\alpha^* \equiv \alpha$ (the support of $*$ being

$$\sup(*) = \{\alpha \mid \alpha^* \not\equiv \alpha\}).$$

In that case we write

$$*(A) = A[\alpha_1 := \alpha_1^*, \ldots, \alpha_n := \alpha_n^*],$$

where $\{\alpha_1, \ldots, \alpha_n\} \supseteq \sup(*)$. We also write

$$* = [\alpha_1 := \alpha_1^*, \ldots, \alpha_n := \alpha_n^*]$$

and

$$* = [\,]$$

for the identity substitution.

2.3.7. Definition. (i) Let $A, B \in \mathbb{T}$. A *unifier* for A and B is a substitutor $*$ such that $A^* \equiv B^*$.

(ii) The substitutor $*$ is a *most general unifier for A and B if*

- $A^* \equiv B^*$
- $A^{*_1} \equiv B^{*_1} \;\Rightarrow\; \exists\, *_2 \,.\, *_1 \equiv *_2 \circ *$.

(iii) *Let $E = \{A_1 = B_1, \ldots, A_n = B_n\}$ be a finite set of equations between types. The equations do not need to be valid. A* unifier *for E is a substitutor $*$ such that $A_1^* \equiv B_1^* \;\&\; \cdots \;\&\; A_n^* \equiv B_n^*$. In that case one writes $* \models E$. Similarly one defines the notion of a most general unifier for E.*

2.3.8. Examples. The types $\beta \to (\alpha \to \beta)$ and $(\gamma \to \gamma) \to \delta$ have a unifier. For example $* = [\beta := \gamma \to \gamma,\ \delta := \alpha \to (\gamma \to \gamma)]$ or $*_1 = [\beta := \gamma \to \gamma,\ \alpha := \varepsilon \to \varepsilon,\ \delta := \varepsilon \to \varepsilon \to (\gamma \to \gamma)]$. The unifier $*$ is most general, $*_1$ is not.

2.3.9. Definition. A is a *variant* of B if for some $*_1$ and $*_2$ one has

$$A = B^{*_1} \text{ and } B = A^{*_2}.$$

2.3.10. Example. $\alpha \to \beta \to \beta$ is a variant of $\gamma \to \delta \to \delta$ but not of $\alpha \to \beta \to \alpha$.

Note that if $*_1$ and $*_2$ are both most general unifiers of say A and B, then A^{*_1} and A^{*_2} are variants of each other and similarly for B.

The following result due to Robinson (1965) states that unifiers can be constructed effectively.

2.3.11. THEOREM (Unification theorem). (i) *There is a recursive function U having (after coding) as input a pair of types and as output either a substitutor or* `fail` *such that*

$$\begin{array}{rcl} A \text{ and } B \text{ have a unifier} & \Rightarrow & U(A,B) \text{ is a most general unifier} \\ & & \text{for } A \text{ and } B; \\ A \text{ and } B \text{ have no unifier} & \Rightarrow & U(A,B) = \mathsf{fail}. \end{array}$$

(ii) *There is (after coding) a recursive function U having as input finite sets of equations between types and as output either a substitutor or* `fail` *such that*

$$\begin{array}{rcl} E \text{ has a unifier} & \Rightarrow & U(E) \text{ is a most general unifier for } E; \\ E \text{ has no unifier} & \Rightarrow & U(E) = \texttt{fail}. \end{array}$$

PROOF. Note that $A_1 \to A_2 \equiv B_1 \to B_2$ holds iff $A_1 \equiv B_1$ and $A_2 \equiv B_2$ hold.
(i) Define $U(A,B)$ by the following recursive loop, using case distinction.

$$\begin{array}{rcll} U(\alpha, B) & = & [\alpha := B], & \text{if } \alpha \notin \mathrm{FV}(B), \\ & = & [\,], & \text{if } B = \alpha, \\ & = & \texttt{fail}, & \text{else;} \end{array}$$

$$\begin{array}{rcl} U(A_1 \to A_2, \alpha) & = & U(\alpha, A_1 \to A_2); \\ U(A_1 \to A_2, B_1 \to B_2) & = & U(A_1^{U(A_2,B_2)}, B_1^{U(A_2,B_2)}) \circ U(A_2, B_2), \end{array}$$

where this last expression is considered to be `fail` if one of its parts is. Let $\#_{var}(A,B)$ ='the number of variables in $A \to B$' and $\#_{\to}(A,B)$='the number of arrows in $A \to B$'. By induction on $(\#_{var}(A,B), \#_{\to}(A,B))$ ordered lexicographically one can show that $U(A,B)$ is always defined. Moreover U satisfies the specification.

(ii) If $E = \{A_1 = B_1, \ldots, A_n = B_n\}$, then define $U(E) = U(A,B)$, where $A = A_1 \to \cdots \to A_n$ and $B = B_1 \to \cdots \to B_n$. ■

See [???] for more on unification. The following result due to Parikh [1973] for propositional logic (interpreted by the propositions-as-types interpretation) and Wand [1987] simplifies the proof of the decidability of type checking and typeability for $\lambda_{\to}$.

2.3.12. PROPOSITION. *For every basis Γ, term $M \in \Lambda$ and $A \in \mathbb{T}$ such that $\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$ there is a finite set of equations $E = E(\Gamma, M, A)$ such that for all substitutors $*$ one has*

$$\begin{array}{rcll} * \models E(\Gamma, M, A) & \Rightarrow & \Gamma^* \vdash M : A^*, & (1) \\ \Gamma^* \vdash M : A^* & \Rightarrow & *_1 \models E(\Gamma, M, A), & (2) \\ & & \text{for some } *_1 \text{ such that } * \text{ and } *_1 \text{ have the same} & \\ & & \text{effect on the type variables in } \Gamma \text{ and } A. & \end{array}$$

PROOF. Define $E(\Gamma, M, A)$ by induction on the structure of M:

$$\begin{array}{rcl} E(\Gamma, x, A) & = & \{A = \Gamma(x)\}; \\ E(\Gamma, MN, A) & = & E(\Gamma, M, \alpha{\to}A) \ \cup \ E(\Gamma, N, \alpha), \\ & & \quad \text{where } \alpha \text{ is a fresh variable;} \\ E(\Gamma, \lambda x.M, A) & = & E(\Gamma \cup \{x{:}\alpha\}, M, \beta) \ \cup \ \{\alpha{\to}\beta = A\}, \\ & & \quad \text{where } \alpha, \beta \text{ are fresh.} \end{array}$$

By induction on M one can show (using the generation lemma (2.1.3)) that (1) and (2) hold. ■

2.3.13. DEFINITION. (i) Let $M \in \Lambda$. Then (Γ, A) is a *principal pair* (pp) for M if

(1) $\Gamma \vdash M : A$.

(2) $\Gamma' \vdash M : A' \ \Rightarrow \ \exists * \, [\Gamma^* \subseteq \Gamma' \ \& \ A^* \equiv A']$.

Here $\{x_1{:}A_1, \ldots\}^* = \{x_1{:}A_1^*, \ldots\}$.

(ii) Let $M \in \Lambda$ be closed. Then A is a *principal type* (pt) for M if

(1) $\vdash M : A$

(2) $\vdash M : A' \ \Rightarrow \ \exists * \ [A^* \equiv A']$.

Note that if (Γ, A) is a *pp* for M, then every variant (Γ', A') of (Γ, A), in the obvious sense, is also a *pp* for M. Conversely if (Γ, A) and (Γ', A') are *pp*'s for M, then (Γ', A') is a variant of (Γ, A). Similarly for closed terms and *pt*'s. Moreover, if (Γ, A) is a *pp* for M, then $\mathrm{FV}(M) = \mathrm{dom}(\Gamma)$.

The following result is independently due to Curry (1969), Hindley (1969) and Milner (1978). It shows that for $\lambda_{\to}$ the problems of type checking and typeability are decidable.

2.3.14. THEOREM (Principal type theorem for $\lambda_{\to}^{\mathrm{Cu}}$). (i) *There exists a computable function pp such that one has*

$$\begin{array}{rcl} M \textit{ has a type} & \Rightarrow & pp(M) = (\Gamma, A), \textit{ where } (\Gamma, A) \textit{ is a pp for } M; \\ M \textit{ has no type} & \Rightarrow & pp(M) = \mathtt{fail}. \end{array}$$

(ii) *There exists a computable function pt such that for closed terms M one has*

$$\begin{array}{rcl} M \textit{ has a type} & \Rightarrow & pt(M) = A, \textit{ where } A \textit{ is a pt for } M; \\ M \textit{ has no type} & \Rightarrow & pt(M) = \mathsf{fail}. \end{array}$$

PROOF. (i) Let $\mathrm{FV}(M) = \{x_1, \ldots, x_n\}$ and set $\Gamma_0 = \{x_1{:}\alpha_1, \ldots, x_n{:}\alpha_n\}$ and $A_0 = \beta$. Note that

$$\begin{array}{rcl} M \text{ has a type} & \Rightarrow & \exists \Gamma \, \exists A \ \ \Gamma \vdash M : A \\ & \Rightarrow & \exists * \ \ \Gamma_0^* \vdash M : A_0^* \\ & \Rightarrow & \exists * \ \ * \models E(\Gamma_0, M, A_0). \end{array}$$

Define

$$\begin{array}{rcll} pp(M) & = & (\Gamma_0^*, A_0^*), & \text{if } U(E(\Gamma_0, M, A_0)) = *; \\ & = & \mathtt{fail}, & \text{if } U(E(\Gamma_0, M, A_0)) = \mathtt{fail}. \end{array}$$

Then $pp(M)$ satisfies the requirements. Indeed, if M has a type, then

$$U(E(\Gamma_0, M, A_0)) = *$$

is defined and $\Gamma_0^* \vdash M : A_0^*$ by (1) in proposition 2.3.12. To show that (Γ_0^*, A_0^*) is a pp, suppose that also $\Gamma' \vdash M : A'$. Let $\widetilde{\Gamma} = \Gamma' \upharpoonright \mathrm{FV}(M)$; write $\widetilde{\Gamma} = \Gamma_0^{*0}$ and $A' = A_0^{*0}$. Then also $\Gamma_0^{*0} \vdash M : A_0^{*0}$. Hence by (2) in proposition 2.3.12 for some $*_1$ (acting the same as $*_0$ on Γ_0, A_0) one has $*_1 \models E(\Gamma_0, M, A_0)$. Since $*$ is a most general unifier (proposition 2.3.11) one has $*_1 = *_2 \circ *$ for some $*_2$. Now indeed

$$(\Gamma_0^*)^{*_2} = \Gamma_0^{*_1} = \Gamma_0^{*0} = \widetilde{\Gamma} \subseteq \Gamma'$$

and

$$(A_0^*)^{*_2} = A_0^{*_1} = A_0^{*0} = A'.$$

If M has no type, then $\neg\exists * \ * \models E(\Gamma_0, M, A_0)$ hence

$$U(\Gamma_0, M, A_0) = \mathtt{fail} = pp(M).$$

(ii) Let M be closed and $pp(M) = (\Gamma, A)$. Then $\Gamma = \emptyset$ and we can put $pt(M) = A$. ■

2.3.15. Corollary. *Type checking and typeability for $\lambda_{\to}$ are decidable.*

Proof. As to type checking, let M and A be given. Then

$$\vdash M : A \iff \exists * \, [A = pt(M)^*].$$

This is decidable (as can be seen using an algorithm—*pattern matching*—similar to the one in Theorem 2.3.11).

As to the question of typeability, let M be given. Then M has a type iff $pt(M) \neq \mathtt{fail}$. ■

The following result is due to Hindley [1969].

2.3.16. Theorem (Second principal type theorem for $\lambda_{\to}^{\mathrm{Cu}}$). (i) *For every type $A \in \mathbb{T}$ one has*

$$\vdash M : A \ \Rightarrow \ \exists M' [M' \twoheadrightarrow_{\beta\eta} M \ \& \ \mathtt{pt}(M') = A].$$

(ii) *For every type $A \in \mathbb{T}$ there exists a basis Γ and term $M \in \Lambda$ such that (Γ, A) is a pp for M.*

PROOF. (i) We present a proof by examples. We choose three situations in which we have to construct an M' that are representative for the general case. Do exercise ?? for the general proof.

Case $M \equiv \lambda x.x$ and $A \equiv (\alpha{\to}\boldsymbol{\beta}){\to}\alpha{\to}\boldsymbol{\beta}$. Then $\mathtt{pt}(M) \equiv \alpha{\to}\alpha$. Take $M' \equiv \lambda xy.xy$. The η-expansion of $\lambda x.x$ to $\lambda xy.xy$ makes subtypes of A correspond to unique subterms of M'.

Case $M \equiv \lambda xy.y$ and $A \equiv (\alpha{\to}\gamma){\to}\boldsymbol{\beta}{\to}\boldsymbol{\beta}$. Then $\mathtt{pt}(M) \equiv \alpha{\to}\boldsymbol{\beta}{\to}\boldsymbol{\beta}$. Take $M' \equiv \lambda xy.\mathsf{K}y(\lambda z.xz)$. The $\boldsymbol{\beta}$-expansion forces x to have a functional type.

Case $M \equiv \lambda xy.x$ and $A \equiv \alpha{\to}\alpha{\to}\alpha$. Then $\mathtt{pt}(M) \equiv \alpha{\to}\boldsymbol{\beta}{\to}\alpha$. Take $M' \equiv \lambda xy.\mathsf{K}x(\lambda f.[fx, fy])$. The $\boldsymbol{\beta}$-expansion forces x and y to have the same types.

(ii) Let A be given. We know that $\vdash \mathsf{I} : A{\to}A$. Therfore by (i) there exists an $\mathsf{I}' \twoheadrightarrow_{\boldsymbol{\beta\eta}} \mathsf{I}$ such that $\mathtt{pt}(\mathsf{I}') = A{\to}A$. Then take $M \equiv \mathsf{I}'x$. We have $\mathtt{pp}(\mathsf{I}'x) = (\{x{:}A\}, A)$. ■

Complexity

The space and time complexity of finding a type for a typable term is exponential, see exercise 2.5.18.

In order to decide whether for two typed terms $M, N \in \Lambda_{\to}(A)$ one has

$$M =_{\beta\eta} N,$$

one can normalize both terms and see whether the results are syntactically equal (up to α-conversion). In exercise 2.5.17 it will be shown that the time and space costs of doing this is at least hyper-exponential (in the size of MN). The reason is that the type-free application of Church numerals

$$\mathbf{c}_n\mathbf{c}_m = \mathbf{c}_{m^n}$$

can be typed, even when applied iteratively

$$\mathbf{c}_{n_1}\mathbf{c}_{n_2}\ldots\mathbf{c}_{n_k}.$$

In exercise 2.5.16 it is shown that the costs are also at most hyper-exponential. The reason is that Turing's proof of normalization for terms in $\lambda_{\to}$ uses a succesive development of redexes of 'highest' type. Now the length of each such development depends exponentially on the length of the term, whereas the length of a term increases at most quadratically at each reduction step. The result even holds for typable terms $M, N \in \Lambda_{\to\mathrm{Cu}}(A)$, as the cost of finding types only ads a simple exponential to the cost.

One may wonder whether there is not a more efficient way to decide $M =_{\beta\eta} N$, for example by using memory for the reduction of the terms, rather than a pure reduction strategy that only depends on the state of the term reduced so far. The sharpest question is whether there is any Turing computable method, that has a better complexity class. In Statman [1979] it is shown that this is not the case, by showing that every elementary time bounded Turing machine computation can be coded as a a convertibility problem for terms of some type in $\lambda^o_{\to}$. A shorter proof of this result can be found in Mairson [1992].

The second variant of $\lambda_{\to}^{\mathrm{Cu}}$ is the *de Bruijn* version of $\lambda_{\to}^{\mathbb{A}}$, denoted by $\lambda_{\to\mathrm{dB}}^{\mathbb{A}}$ or $\lambda_{\to}^{\mathrm{dB}}$. Now only bound variables get ornamented with types, but only at the binding stage. The examples (1), (2) now become

$$\begin{array}{rll} & \vdash_{\lambda\to}^{\mathrm{dB}} & (\lambda x:A.x):A\to A \qquad (1_{\mathrm{dB}}) \\ y{:}A & \vdash_{\lambda\to}^{\mathrm{dB}} & (\lambda x:(A\to B).xy):(A\to B)\to A\to B \qquad (2_{\mathrm{dB}}) \end{array}$$

The reasons to have these variants will be explained later in Section 1.4. In the meantime we will work intuitively.

1.1.20. NOTATION. Terms like $(\lambda fx.f(fx))\in\Lambda^{\emptyset}(1\to o\to o)$ will often be written

$$\lambda f^1x^0.f(fx)$$

to indicate the types of the bound variables. We will come back to this notational issue in section 1.4.

1.2. Normal inhabitants

In this section we will give an algorithm that enumerates the set of closed terms in normal form of a given type $A\in\mathbb{T}$. Since we will prove in the next chapter that all typable terms do have a nf and that reduction preserves typing, we thus have an enumeration of essentially all closed terms of that given type. We do need to distinguish various kinds of nf's.

1.2.1. DEFINITION. Let $A=A_1\to\ldots A_n\to\alpha$ and suppose $\Gamma\vdash M:A$.

(i) Then M is in long-nf, notation lnf, if $M\equiv\lambda x_1^{A_1}\ldots x_n^{A_n}.xM_1\ldots M_n$ and each M_i is in lnf. By induction on the depth of the type of the closure of M one sees that this definition is well-founded.

(ii) M has a lnf if $M=_{\beta\eta}N$ and N is a lnf.

In Exercise 1.5.16 it is proved that if M has a $\boldsymbol{\beta}$-nf, which according to Theorem 2.2.4 is always the case, then it also has a unique lnf and will be its unique $\boldsymbol{\beta\eta}^{-1}$ nf. Here η^{-1} is the notion of reduction that is the converse of η.

1.2.2. EXAMPLES. (i) Note that $\lambda f^1.f=_{\beta\eta}\lambda f^1\lambda x^o.fx$ and that $\lambda f^1.f$ is a $\boldsymbol{\beta\eta}$-nf but not a lnf.

(ii) $\lambda f^1\lambda x^o.fx$ is a lnf, but not a $\boldsymbol{\beta\eta}$-nf.

(iii) $\lambda x{:}o.x$ is both in $\boldsymbol{\beta\eta}$-nf and lnf.

(iv) The $\boldsymbol{\beta}$-nf $\lambda F{:}2_2\lambda f{:}1.Ff(\lambda x{:}o.fx)$ is neither in $\boldsymbol{\beta\eta}$-nf nor lnf.

(v) A variable of atomic type α is a lnf, but of type $A\to B$ not.

(vi) A variable $f:1\to1$ has as lnf $\lambda g^1\lambda x^o.f(\lambda y^o.gy)x$.

1.2.3. PROPOSITION. *Every $\boldsymbol{\beta}$-nf M has a lnf M^ℓ such that $M^\ell\twoheadrightarrow_{\boldsymbol{\eta}}M$.*

PROOF. Define M^ℓ by induction on the depth of the type of the closure of M as follows.

$$M^\ell \equiv (\lambda\vec{x}.yM_1\dots M_n)^\ell = \lambda\vec{x}\vec{z}.yM_1^\ell\dots M_n^\ell\vec{z}^\ell.$$

Then M^ℓ does the job. ■

Now we will define a 2-level grammar for obtaining the collection of all lnf's of a given type A.

1.2.4. DEFINITION. Let $N = \{L(A;\Gamma) \mid A\in\mathbb{T}_{\mathbb{A}};\Gamma \text{ a context of } \lambda_{\rightarrow}\}$. Let Σ be the alphabet of the terms of the $\lambda_{\rightarrow}^{\mathrm{Ch}}$. Define the following two-level grammar, see van Wijngaarden et al. [1976], as a notion of reduction over words over $N\cup\Sigma$. The elements of N are the non-terminals (unlike in a context-free language there are now infinitely many of them).

$$\begin{array}{rcll} L(\alpha;\Gamma) & \Rightarrow & xL(B_1;\Gamma)\dots L(B_n;\Gamma), & \text{if } (x{:}\vec{B}{\rightarrow}\alpha)\in\Gamma;\\ L(A{\rightarrow}B;\Gamma) & \Rightarrow & \lambda x^A.L(B;\Gamma,x{:}A). & \end{array}$$

Typical productions of this grammar are the following.

$$\begin{array}{rcl} L(3;\emptyset) & \Rightarrow & \lambda F^2.L(o;F^2)\\ & \Rightarrow & \lambda F^2.FL(1;F^2)\\ & \Rightarrow & \lambda F^2.F(\lambda x^o.L(o;F^2,x^o))\\ & \Rightarrow & \lambda F^2.F(\lambda x^o.x). \end{array}$$

But one has also

$$\begin{array}{rcl} L(o;F^2,x^o) & \Rightarrow & FL(1;F^2,x^o)\\ & \Rightarrow & F(\lambda x_1^o.L(o;F^2,x^o,x_1^o))\\ & \Rightarrow & F(\lambda x_1^o.x_1). \end{array}$$

Hence ($\twoheadrightarrow$ denotes the transitive reflexive closure of $\Rightarrow$)

$$L(3;\emptyset) \twoheadrightarrow \lambda F^2.F(\lambda x^o.F(\lambda x_1^o.x_1)).$$

In fact, $L(3;\emptyset)$ reduces to all possible closed lnf's of type 3. Like in abstract syntax we do not produce parentheses from the $L(A;\Gamma)$, but write them when needed.

1.2.5. PROPOSITION. *Let Γ, M, A be given. Then*

$$L(A,\Gamma)\twoheadrightarrow M \iff \Gamma\vdash M:A \;\&\; M \textit{ is in lnf.}$$

Now we will modify the 2-level grammar and the inhabitation machines in order to produce all β-nf's.

1.2.6. Definition. The 2-level grammar N is defined as follows.

$$\begin{array}{rcll} N(A;\Gamma) & \Rightarrow & xN(B_1;\Gamma)\ldots N(B_n;\Gamma), & \text{if } (x{:}\vec{B}{\to}A)\in\Gamma; \\ N(A{\to}B;\Gamma) & \Rightarrow & \lambda x^A.N(B;\Gamma,x{:}A). & \end{array}$$

Now the $\boldsymbol{\beta}$-nf's are being produced. As an example we make the following production. Remember that $1 = o{\to}o$.

$$\begin{array}{rcl} L(1{\to}o{\to}o;\emptyset) & \Rightarrow & \lambda f^1.L(o{\to}o;f{:}o{\to}o) \\ & \Rightarrow & \lambda f^1.f. \end{array}$$

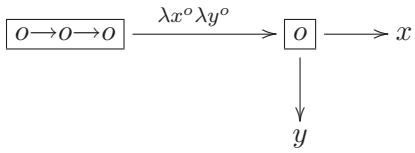
1.2.7. Proposition. *Let Γ, M, A be given. Then*

$$N(A,\Gamma) \twoheadrightarrow M \iff \Gamma \vdash M : A \;\&\; M \text{ is in } \boldsymbol{\beta}\text{-nf}.$$

Inhabitation machines

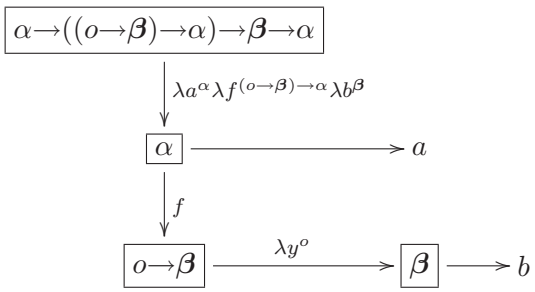
Inspired by this proposition one can introduce for each type A a machine M_A producing the set of closed terms of that type. If one is interested in terms containing variables $x_1^{A_1}, \ldots, x_n^{A_n}$, then one can also find these terms by considering the machine for the type $A_1{\to}\ldots{\to}A_n{\to}A$ and look at the subproduction at node A.

1.2.8. Examples. (i) $A = o{\to}o{\to}o$. Then M_A is



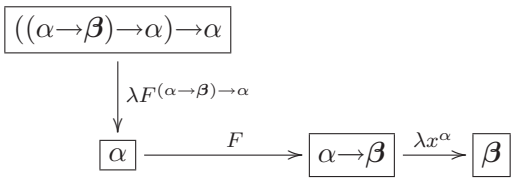
This shows that the type 1_2 has two closed inhabitants: $\lambda xy.x$ and $\lambda xy.y$. We see that the two arrows leaving $\boxed{\text{o}}$ represent a choice.

(ii) $A = \alpha{\to}((o{\to}\boldsymbol{\beta}){\to}\alpha){\to}\boldsymbol{\beta}{\to}\alpha$. Then M_A is



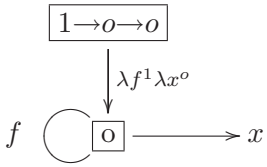
Again there are only two inhabitants, but now the production of them is rather different: $\lambda afb.a$ and $\lambda afb.f(\lambda x^o.b)$.

(iii) $A = ((\alpha{\rightarrow}\boldsymbol{\beta}){\rightarrow}\alpha){\rightarrow}\alpha$. Then M_A is



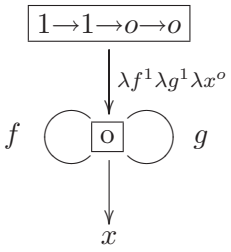
This type, corresponding to Peirce's law, does not have any inhabitants.

(iv) $A = 1{\rightarrow}o{\rightarrow}o$. Then M_A is



This is the type Nat having the Church's numerals $\lambda f^1 x^o.f^n x$ as inhabitants.

(v) $A = 1{\rightarrow}1{\rightarrow}o{\rightarrow}o$. Then M_A is

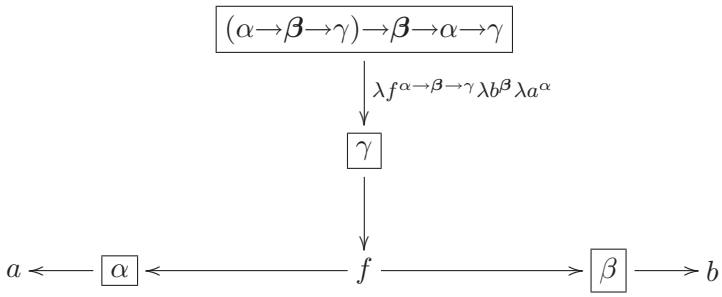


Inhabitants of this type represent words over the alphabet $\Sigma = \{f, g\}$, for example

$$\lambda f^1 g^1 x^o.fgfffgfggx,$$

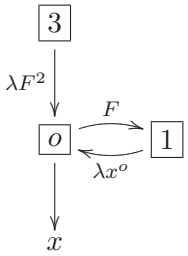
where we have to insert parentheses associating to the right.

(vi) $A = (\alpha{\rightarrow}\boldsymbol{\beta}{\rightarrow}\gamma){\rightarrow}\boldsymbol{\beta}{\rightarrow}\alpha{\rightarrow}\gamma$. Then M_A is



giving as term $\lambda f^{\alpha{\rightarrow}\boldsymbol{\beta}{\rightarrow}\gamma}\lambda b^{\boldsymbol{\beta}}\lambda a^{\alpha}.fab$. Note the way an interpretation should be given to paths going through f: the outgoing arcs (to $\boxed{\alpha}$ and $\boxed{\boldsymbol{\beta}}$) should be completed both separately in order to give f its two arguments.

(vii) $A = 3$. Then M_A is

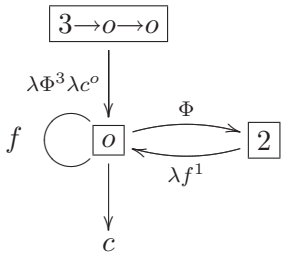


This type 3 has inhabitants having more and more binders:

$$\lambda F^2.F(\lambda x_0^o.F(\lambda x_1^o.F(\cdots(\lambda x_n^o.x_i)))).$$

The novel phenomenon that the binder λx^o may go round and round forces us to give new incarnations $\lambda x_0^o,\ \lambda x_1^o, \ldots$ each time we do this (we need a counter to ensure freshness of the bound variables). The 'terminal' variable x can take the shape of any of the produced incarnations x_k. As almost all binders are dummy, we will see that this potential infinity of binding is rather innocent and the counter is not yet really needed here.

(viii) $A = 3{\to}o{\to}o$. Then M_A is

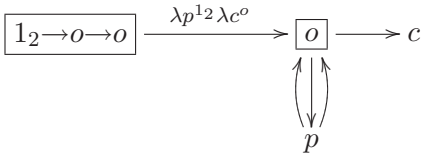


This type, called the *monster* M, does have a potential infinite amount of binding, having as terms e.g.

$$\lambda\Phi^3 c^o.\Phi\lambda f_1^1.f_1\Phi\lambda f_2^1.f_2 f_1\Phi\ldots\lambda f_n^1.f_n\ldots f_2 f_1 c,$$

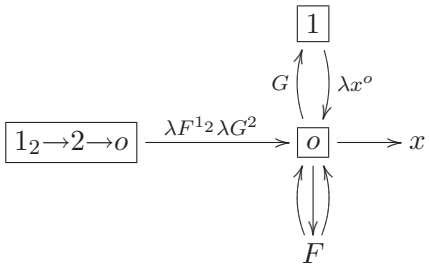
again with inserted parentheses associating to the right. Now a proper bookkeeping of incarnations (of f^1 in this case) becomes necessary, as the f going from $\boxed{o}$ to itself needs to be one that has already been incarnated.

(ix) $A = 1_2{\to}o{\to}o$. Then M_A is



This is the type of binary trees, having as elements, e.g. $\lambda p^{1_2} c^o.c$ and $\lambda p^{1_2} c^o.pc(pcc)$. Again, as in example (vi) the outgoing arcs from p (to $\boxed{o}$) should be completed both separately in order to give p its two arguments.

(x) $A = 1_2 \to 2 \to o$. Then M_A is



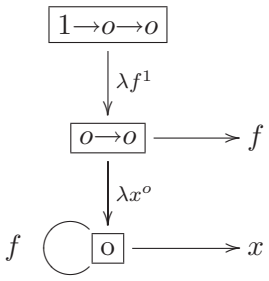
This is the type L corresponding to untyped lambda terms. For example the untyped terms $\omega \equiv \lambda x.xx$ and $\Omega \equiv (\lambda x.xx)(\lambda x.xx)$ can be translated to $(\omega)^t \equiv \lambda F^{1_2} G^2.G(\lambda x^o.Fxx)$ and

$$\begin{array}{rcl} (\Omega)^t & \equiv & \lambda F^{1_2} G^2.F(G(\lambda x^o.Fxx))(G(\lambda x^o.Fxx)) \\ & =_\beta & \lambda FG.F((\omega)^t FG)((\omega)^t FG) \\ & =_\beta & (\omega)^t \cdot_L (\omega)^t, \end{array}$$

where for $M, N \in L$ one defines $M \cdot_L N = \lambda FG.F(MFG)(NFG)$. All features of producing terms inhabiting types (bookkeeping bound variables, multiple paths) are present here.

Following the 2-level grammar N one can make inhabitation machines for $\boldsymbol{\beta}$-nf $M_A^{\boldsymbol{\beta}}$.

1.2.9. Example. We show how the production machine for $\boldsymbol{\beta}$-nf's differs from the one for lnf's. Let $A = 1 \to o \to o$. Then $\lambda f^1.f$ is the (unique) $\boldsymbol{\beta}$-nf of type A that is not a lnf. It will come out from the following machine $M_A^{\boldsymbol{\beta}}$.



So in order to obtain the $\boldsymbol{\beta}$-nf's, one has to allow output at types that are not atomic.

1.3. Representing data types

In this section it will be shown that first order algebraic data types can be represented in $\lambda^o_{\to}$. We start with several examples: Booleans, the natural numbers, the free monoid over n generators (words over a finite alphabet with n elements) and trees with at the leafs labels from a type A. The following definitions depend on a given type A. So in fact $\mathsf{Bool} = \mathsf{Bool}_A$ etcetera. Often one takes $A = o$.

Chapter 2

Properties

2.1. First properties

In this section we will treat simple properties of the various systems $\lambda_{\to}$. Deeper properties—like strong normalization of typeable terms—will be considered in Section 2.2.

Properties of $\lambda_{\to}^{\mathrm{Cu}}$, $\lambda_{\to}^{\mathrm{Ch}}$ and $\lambda_{\to}^{\mathrm{dB}}$

Unless stated otherwise, properties stated for $\lambda_{\to}$ apply to both systems.

2.1.1. Proposition (Weakening lemma for $\lambda_{\to}$).
Suppose $\Gamma \vdash M : A$ *and* Γ' *is a basis with* $\Gamma \subseteq \Gamma'$. *Then* $\Gamma' \vdash M : A$.

Proof. By induction on the derivation of $\Gamma \vdash M : A$. ■

2.1.2. Lemma (Free variable lemma). (i) *Suppose* $\Gamma \vdash M : A$. *Then* $FV(M) \subseteq \mathrm{dom}(\Gamma)$.
(ii) *If* $\Gamma \vdash M : A$, *then* $\Gamma \restriction \mathrm{FV}(M) \vdash A : M$, *where for a set* X *of variables one has* $\Gamma \restriction \mathrm{FV}(M) = \{x{:}A \in \Gamma \mid x \in X\}$.

Proof. (i), (ii) By induction on the generation of $\Gamma \vdash M : A$. ■

The following result is related to the fact that the system $\lambda_{\to}$ is 'syntax directed', i.e. statements $\Gamma \vdash M : A$ have a unique proof.

2.1.3. Proposition (Generation lemma for $\lambda_{\to}^{\mathrm{Cu}}$).

$$\begin{array}{lrcl}
\text{(i)} & \Gamma \vdash x : A & \Rightarrow & (x{:}A) \in \Gamma. \\
\text{(ii)} & \Gamma \vdash MN : A & \Rightarrow & \exists B \in \mathbb{T}\, [\Gamma \vdash M : B{\to}A \;\&\; \Gamma \vdash N : B]. \\
\text{(iii)} & \Gamma \vdash \lambda x.M : A & \Rightarrow & \exists B, C \in \mathbb{T}\, [A \equiv B{\to}C \;\&\; \Gamma, x{:}B \vdash M : C].
\end{array}$$

Proof. (i) Suppose $\Gamma \vdash x : A$ holds in $\lambda_{\to}$. The last rule in a derivation of this statement cannot be an application or an abstraction, since x is not of the right form. Therefore it must be an axiom, i.e. $(x{:}A) \in \Gamma$.
(ii), (iii) The other two implications are proved similarly. ■

2.1.4. PROPOSITION (Generation lemma for $\lambda^{\mathrm{dB}}_{\to}$).

$$\begin{array}{lrcl}
\text{(i)} & \Gamma \vdash x : A & \Rightarrow & (x{:}A) \in \Gamma. \\
\text{(ii)} & \Gamma \vdash MN : A & \Rightarrow & \exists B \in \mathbb{T}\, [\Gamma \vdash M : B{\to}A \;\&\; \Gamma \vdash N : B]. \\
\text{(iii)} & \Gamma \vdash \lambda x{:}B.M : A & \Rightarrow & \exists C \in \mathbb{T}\, [A \equiv B{\to}C \;\&\; \Gamma, x{:}B \vdash M : C].
\end{array}$$

PROOF. Similarly. ■

2.1.5. PROPOSITION (Generation lemma for $\lambda^{\mathrm{Ch}}_{\to}$).

$$\begin{array}{lrcl}
\text{(i)} & x^B \in \Lambda^{\mathrm{Ch}}_{\to}(A) & \Rightarrow & B = A. \\
\text{(ii)} & (MN) \in \Lambda^{\mathrm{Ch}}_{\to}(A) & \Rightarrow & \exists B \in \mathbb{T}.[M \in \Lambda^{\mathrm{Ch}}_{\to}(B{\to}A) \;\&\; N \in \Lambda^{\mathrm{Ch}}_{\to}(B)]. \\
\text{(iii)} & (\lambda x^B.M) \in \Lambda^{\mathrm{Ch}}_{\to}(A) & \Rightarrow & \exists C \in \mathbb{T}.[A = (B{\to}C) \;\&\; M \in \Lambda^{\mathrm{Ch}}_{\to}(C)].
\end{array}$$

PROOF. As before. ■

The following two results hold for $\lambda^{\mathrm{Cu}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$. Variants already have been proved for $\lambda^{\mathrm{Ch}}_{\to}$, Propositions 1.4.2 and 1.4.4(iii).

2.1.6. PROPOSITION (Substitution lemma for $\lambda^{\mathrm{Cu}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$).
(i) $\Gamma, x{:}A \vdash M : B \;\&\; \Gamma \vdash N : A \;\Rightarrow\; \Gamma \vdash M[x{:=}N] : B$.
(ii) $\Gamma \vdash M : A \;\Rightarrow\; \Gamma[\alpha := B] \vdash M : A[\alpha := B]$.

PROOF. The proof will be given for $\lambda^{\mathrm{Cu}}_{\to}$, for $\lambda^{\mathrm{dB}}_{\to}$ it is similar.
(i) By induction on the derivation of $\Gamma, x{:}A \vdash M : B$. Write $P^* \equiv P[x{:=}N]$.
Case 1. $\Gamma, x{:}A \vdash M : B$ is an axiom, hence $M \equiv y$ and $(y{:}B) \in \Gamma \cup \{x{:}A\}$.
Subcase 1.1. $(y{:}B) \in \Gamma$. Then $y \not\equiv x$ and $\Gamma \vdash M^* \equiv y[x{:}N] \equiv y : B$.
Subcase 1.2. $y{:}B \equiv x{:}A$. Then $y \equiv x$ and $B \equiv A$, hence $\Gamma \vdash M^* \equiv N : A \equiv B$.
Case 2. $\Gamma, x{:}A \vdash M : B$ follows from $\Gamma, x{:}A \vdash F : C{\to}B$, $\Gamma, x{:}A \vdash G : C$ and $FG \equiv M$. By the induction hypothesis one has $\Gamma \vdash F^* : C{\to}B$ and $\Gamma \vdash G^* : C$. Hence $\Gamma \vdash (FG)^* \equiv F^*G^* : B$.
Case 3. $\Gamma, x{:}A \vdash M : B$ follows from $\Gamma, x{:}A, y{:}D \vdash G : E$, $B \equiv D{\to}E$ and $\lambda y.G \equiv M$. By the induction hypothesis $\Gamma, y{:}D \vdash G^* : E$, hence $\Gamma \vdash (\lambda y.G)^* \equiv \lambda y.G^* : D{\to}E \equiv B$.
(ii) Similarly. ■

2.1.7. PROPOSITION (Subject reduction property for $\lambda^{\mathrm{Cu}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$). *Suppose* $M \twoheadrightarrow_{\boldsymbol{\beta\eta}} M'$. *Then* $\Gamma \vdash M : A \;\Rightarrow\; \Gamma \vdash M' : A$.

PROOF. The proof will be given for $\lambda^{\mathrm{dB}}_{\to}$, for $\lambda^{\mathrm{Cu}}_{\to}$ it is similar. Suppose $\Gamma \vdash M : A$ and $M \to M'$ in order to show that $\Gamma \vdash M' : A$; then the result follows by induction on the derivation of $\Gamma \vdash M : A$.
Case 1. $\Gamma \vdash M : A$ is an axiom. Then M is a variable, contradicting $M \to M'$. Hence this case cannot occur.
Case 2. $\Gamma \vdash M : A$ is $\Gamma \vdash FN : A$ and is a direct consequence of $\Gamma \vdash F : B{\to}A$ and $\Gamma \vdash N : B$. Since $FN \equiv M \to M'$ we can have three subcases.

Subcase 2.1. $M' \equiv F'N$ with $F \to F'$.

Subcase 2.2. $M' \equiv FN'$ with $N \to N'$.

In these two subcases it follows by the induction hypothesis that $\Gamma \vdash M' : A$.

Subcase 2.3. $F \equiv \lambda x{:}B.G$ and $M' \equiv G[x{:=}N]$. Since

$$\Gamma \vdash \lambda x.G : B{\to}A \;\&\; \Gamma \vdash N : B$$

it follows by the generation lemma 2.1.3 for $\lambda_{\to}$ that

$$\Gamma, x{:}B \vdash G : A \;\&\; \Gamma \vdash N : B.$$

Therefore by the substitution lemma 2.1.6 for $\lambda_{\to}$ it follows that
$\Gamma \vdash G[x{:=}N] : A$, i.e. $\Gamma \vdash M' : A$.

Case 3. $\Gamma \vdash M : A$ is $\Gamma \vdash \lambda x{:}B.N : B{\to}C$ and follows from $\Gamma, x{:}B \vdash N : C$. Since $M \to M'$ we have $M' \equiv \lambda x{:}B.N'$ with $N \to N'$. By the induction hypothesis one has $\Gamma, x{:}B \vdash N' : C$, hence $\Gamma \vdash \lambda x{:}B.N' : B{\to}C$, i.e. $\Gamma \vdash M' : A$. ■

The following result also holds for $\lambda^{\mathrm{Ch}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$, Exercise 2.5.4.

2.1.8. Corollary (Church-Rosser Theorem for $\lambda^{\mathrm{Cu}}_{\to}$). *On typable terms of $\lambda^{\mathrm{Cu}}_{\to}$ the Church-Rosser theorem holds for the notions of reduction $\twoheadrightarrow_{\boldsymbol{\beta}}$ and $\twoheadrightarrow_{\boldsymbol{\beta\eta}}$.*

(i) *Let $M, N \in \Lambda^{\Gamma}_{\to}(A)$. Then*

$$M =_{\boldsymbol{\beta(\eta)}} N \;\Rightarrow\; \exists Z \in \Lambda^{\Gamma}_{\to}(A).M \twoheadrightarrow_{\boldsymbol{\beta(\eta)}} Z \;\&\; N \twoheadrightarrow_{\boldsymbol{\beta(\eta)}} Z.$$

(ii) *Let $M, N_1, N_2 \in \Lambda^{\Gamma}_{\to}(A)$. Then*

$$M \twoheadrightarrow_{\boldsymbol{\beta\eta}} N_1 \;\&\; M \twoheadrightarrow_{\boldsymbol{\beta(\eta)}} N_2 \;\Rightarrow\; \exists Z \in \Lambda^{\Gamma}_{\to}(A).N_1 \twoheadrightarrow_{\boldsymbol{\beta(\eta)}} Z \;\&\; N_2 \twoheadrightarrow_{\boldsymbol{\beta(\eta)}} Z.$$

Proof. By the Church-Rosser theorems for $\twoheadrightarrow_{\boldsymbol{\beta}}$ and $\twoheadrightarrow_{\boldsymbol{\beta\eta}}$ on untyped terms, Theorem 1.1.7, and Proposition 2.1.7. ■

The following property of uniqueness of types only holds for the Church and de Bruijn versions of $\lambda_{\to}$. It is instructive to find out where the proof brakes down for $\lambda^{\mathrm{Cu}}_{\to}$ and also that the two contexts in (ii) should be the same.

2.1.9. Proposition (Unicity of types for $\lambda^{\mathrm{Ch}}_{\to}$ and $\lambda^{\mathrm{dB}}_{\to}$).

(i) $M \in \Lambda^{\mathrm{Ch}}_{\to}(A) \;\&\; M \in \Lambda^{\mathrm{Ch}}_{\to}(B) \;\Rightarrow\; A = B.$

(ii) $\Gamma \vdash^{\mathrm{dB}}_{\lambda_{\to}} M : A \;\&\; \Gamma \vdash^{\mathrm{dB}}_{\lambda_{\to}} M : B \;\Rightarrow\; A = B.$

Proof. (i), (ii) By induction on the structure of M, using the generation lemma 2.1.4. ■

Normalization

For several applications, for example for the problem to find all possible inhabitants of a given type, we will need the weak normalization theorem, stating that all typable terms do have a $\boldsymbol{\beta\eta}$-nf (normal form). The result is valid for all versions of $\lambda_{\rightarrow}$ and *a fortiori* for the subsystems $\lambda^o_{\rightarrow}$. The proof is due to Turing and is published posthumously in Gandy [1980]. In fact all typable terms in these systems are $\boldsymbol{\beta\eta}$ strongly normalizing, which means that all $\boldsymbol{\beta\eta}$-reductions are terminating. This fact requires more work and will be proved in §12.2.

The notion of 'abstract reduction system', see Klop [1992], is useful for the understanding of the proof of the normalization theorem.

2.1.10. Definition. (i) An *abstract reduction system* is a pair $(X, \rightarrow_R)$, where X is a set and $\rightarrow_R$ is a binary relation on X.

(ii) An element $x \in X$ is said to be in R-normal form (R-nf) if for no $y \in X$ one has $x \rightarrow_R y$.

(iii) (X, R) is called *weakly normalizing* (R-WN, or simply WN) if every element has an R-nf.

(iv) (X, R) is said to be *strongly normalizing* (R-SN, or simply SN) if every R-reduction path

$$x_0 \rightarrow_R x_1 \rightarrow_R x_2 \rightarrow_R \dots.$$

is finite.

2.1.11. Definition. (i) A *multiset over nat* can be thought of as a generalized set S in which each element may occur more than once. For example

$$S = \{3, 3, 1, 0\}$$

is a multiset. We say that 3 occurs in S with multiplicity 2; that 1 has multiplicity 1; etcetera.

More formally, the above multiset S can be identified with a function $f \in \mathbb{N}^{\mathbb{N}}$ that is almost everywhere 0, except

$$f(0) = 1, f(1) = 1, f(3) = 2.$$

This S is finite if f has *finite support*, where

$$\mathtt{support}(f) = \{x \in \mathbb{N} \mid f(x) \neq 0\}.$$

(ii) Let $\mathcal{S}(\mathbb{N})$ be the collection of all finite multisets over $\mathbb{N}$. $\mathcal{S}(\mathbb{N})$ can be identified with $\{f \in \mathbb{N}^{\mathbb{N}} \mid \mathtt{support}(f) \text{ is finite}\}$.

2.1.12. Definition. Let $S_1, S_2 \in \mathcal{S}(\mathbb{N})$. Write

$$S_1 \rightarrow_{\mathcal{S}} S_2$$

if S_2 results from S_1 by replacing some elements (just one occurrence) by finitely many lower elements (in the usual ordering of $\mathbb{N}$). For example

$$\{3, \underline{3}, 1, 0\} \to_{\mathcal{S}} \{3, \underline{2, 2, 2, 1}, 1, 0\}.$$

2.1.13. LEMMA. *We define a particular (non-deterministic) reduction strategy F on $\mathcal{S}(\mathbb{N})$. A multi-set S is contracted to $F(S)$ by taking a maximal element $n \in S$ and replacing it by finitely many numbers $< n$. Then F is a normalizing reduction strategy, i.e. for every $S \in \mathcal{S}(\mathbb{N})$ the $\mathcal{S}$-reduction sequence*

$$S \to_{\mathcal{S}} F(S) \to_{\mathcal{S}} F^2(S) \to_{\mathcal{S}} \dots$$

is terminating.

PROOF. By induction on the highest number n occuring in S. If $n = 0$, then we are done. If $n = k+1$, then we can successively replace in S all occurrences of n by numbers $\leq k$ obtaining S_1 with maximal number $\leq k$. Then we are done by the induction hypothesis. ■

In fact $(\mathcal{S}(\mathbb{N}), \twoheadrightarrow_{\mathcal{S}})$ is SN. Although we do not strictly need this fact, we will give even two proofs of it. In the first place it is something one ought to know; in the second place it is instructive to see that the result does not imply that $\lambda_{\to}$ satisfies SN.

2.1.14. LEMMA. *The reduction system $(\mathcal{S}(\mathbb{N}), \to_{\mathcal{S}})$ is SN.*

We will give two proofs of this lemma. The first one uses ordinals; the second one is from first principles.

PROOF$_1$. Assign to every $S \in \mathcal{S}(\mathbb{N})$ an ordinal $\#S < \omega^\omega$ as suggested by the following examples.

$$\begin{aligned} \#\{3, 3, 1, 0, 0, 0\} &= 2\omega^3 + \omega + 3; \\ \#\{3, 2, 2, 2, 1, 1, 0\} &= \omega^3 + 3\omega^2 + 2\omega + 1. \end{aligned}$$

More formally, if S is represented by $f \in \mathbb{N}^{\mathbb{N}}$ with finite support, then

$$\#S = \Sigma_{i \in \mathbb{N}} f(i) \cdot \omega^i.$$

Notice that

$$S_1 \to_{\mathcal{S}} S_2 \;\Rightarrow\; \#S_1 > \#S_2$$

(in the example because $\omega^3 > 3\omega^2 + \omega$). Hence by the well-foundedness of the ordinals the result follows. ■$_1$

PROOF$_2$. Define

$$\begin{aligned} \mathcal{F}_k &= \{f \in \mathbb{N}^{\mathbb{N}} \mid \forall n \geq k \, f(n) = 0\}; \\ \mathcal{F} &= \cup_{k \in \mathbb{N}} \mathcal{F}_k. \end{aligned}$$

The set $\mathcal{F}$ is the set of functions with finite support. Define on $\mathcal{F}$ the relation $>$ corresponding to the relation $\to_{\mathcal{S}}$ for the formal definition of $\mathcal{S}(\mathbb{N})$.

$$f > g \iff f(k) > g(k), \text{ where } k \in \mathbb{N} \text{ is largest such that } f(k) \neq g(k).$$

It is easy to see that $(\mathcal{F}, >)$ is a linear ordering. We will show that it is even a well-ordering, i.e. for every non-empty set $X \subseteq \mathcal{F}$ there is a least element $f_0 \in X$. This implies that there are no infinite descending chains in $\mathcal{F}$.

To show this claim it suffices to prove that each $\mathcal{F}_k$ is well-ordered, since

$$\ldots > (\mathcal{F}_{k+1} \setminus \mathcal{F}_k) > \mathcal{F}_k$$

element-wise. This will be proved by induction on k. If $k = 0$, then this is trivial, since $\mathcal{F}_0 = \{\lambda\!\!\lambda n.0\}$. Now assume (induction hypothesis) that $\mathcal{F}_k$ is well-ordered in order to show the same for $\mathcal{F}_{k+1}$. Let $X \subseteq \mathcal{F}_{k+1}$ be non-empty. Define

$$\begin{array}{rcl} X(k) & = & \{f(k) \mid f \in X\} \subseteq \mathbb{N}; \\ X_k & = & \{f \in X \mid f(k) \text{ minimal in } X(k)\} \subseteq \mathcal{F}_{k+1}; \\ X_k|k & = & \{g \in \mathcal{F}_k \mid \exists f \in X_k\, f|k = g\} \subseteq \mathcal{F}_k, \end{array}$$

where

$$\begin{array}{rcll} f|k(i) & = & f(i), & \text{if } i < k; \\ & = & 0, & \text{else.} \end{array}$$

By the induction hypothesis $X_k|k$ has a least element g_0. Then $g_0 = f_0|k$ for some $f_0 \in X_k$. This f_0 is then the least element of X_k and hence of X. $\blacksquare_2$

2.1.15. Remark. The second proof shows in fact that if $(D, >)$ is a well-ordered set, then so is $(\mathcal{S}(D), >)$, defined analogously to $(\mathcal{S}(\mathbb{N}), >)$. In fact the argument can be carried out in Peano Arithmetic, showing

$$\vdash_{\mathbf{PA}} \mathtt{TI}(\alpha) \to \mathtt{TI}(\alpha^\omega),$$

where $\mathtt{TI}(\alpha)$ is the principle of transfinite induction for the ordinal α. Since $\mathtt{TI}(\omega)$ is in fact ordinary induction we have in PA

$$\mathtt{TI}(\omega),\ \mathtt{TI}(\omega^\omega),\ \mathtt{TI}(\omega^{(\omega^\omega)}), \ldots.$$

This implies that the proof of $\mathtt{TI}(\alpha)$ can be carried out in Peano Arithmetic for every $\alpha < \epsilon_0$. Gentzen [1936] shows that $\mathtt{TI}(\epsilon_0)$, where $\epsilon_0 = \omega^{\omega^{\omega^{\cdots}}}$, cannot be carried out in PA.

In order to prove the $\lambda_\to$ is WN it suffices to work with $\lambda^{\mathrm{Ch}}_\to$. We will use the following notation. We write terms with extra type information, decorating each subterm with its type. For example, instead of $(\lambda x^A.M)N \in \mathtt{term}_B$ we write $(\lambda x^A.M^B)^{A\to B}N^A$.

2.1.16. DEFINITION. (i) Let $R \equiv (\lambda x^A.M^B)^{A\to B}N^A$ be a redex. The *depth* of R, notation $\#R$, is defined as follows.

$$\#R = \#(A\to B)$$

where $\#$ on types is defined inductively by

$$\begin{aligned} \#\alpha &= 0; \\ \#(A\to B) &= \max(\#A, \#B) + 1. \end{aligned}$$

(ii) To each M in $\lambda^{\mathrm{Ch}}_{\to}$ we assign a multi-set S_M as follows

$$S_M = \{\#R \,|\, R \text{ is a redex occurrence in } M\},$$

with the understanding that the multiplicity of R in M is copied in S_M.

In the following example we study how the contraction of one redex can duplicate other redexes or create new redexes.

2.1.17. EXAMPLE. (i) Let R be a redex occurrence in a typed term M. Assume

$$M \xrightarrow{R}_{\boldsymbol{\beta}} N,$$

i.e. N results form M by contracting R. This contraction can duplicate other redexes. For example (we write $M[P]$, or $M[P,Q]$ to display subterms of M)

$$(\lambda x.M[x,x])R_1 \to_{\boldsymbol{\beta}} M[R_1, R_1]$$

duplicates the other redex R_1.

(ii) (J.J. Lévy [1978]) Contraction of a $\boldsymbol{\beta}$-redex may also create new redexes. For example

$$\begin{aligned} (\lambda x^{A\to B}.M[x^{A\to B}P^A]^C)^{(A\to B)\to C}(\lambda y^A.Q^B) &\to_{\boldsymbol{\beta}} M[(\lambda y^A.Q^B)^{A\to B}P^A]^C; \\ (\lambda x^A.(\lambda y^B.M[x^A, y^B]^C)^{B\to C})^{A\to(B\to C)}P^AQ^B &\to_{\boldsymbol{\beta}} (\lambda y^B.M[P^A, y^B]^C)^{B\to C}Q^B; \\ (\lambda x^{A\to B}.x^{A\to B})^{(A\to B)\to(A\to B)}(\lambda y^A.P^B)^{A\to B}Q^A &\to_{\boldsymbol{\beta}} (\lambda y^A.P^B)^{A\to B}Q^A. \end{aligned}$$

2.1.18. LEMMA. *Assume $M \xrightarrow{R}_{\boldsymbol{\beta}} N$ and let R_1 be a created redex in N. Then $\#R > \#R_1$.*

PROOF. In Lévy [1978] it is proved that the three ways of creating redexes in example 2.1.17(ii) are the only possibilities. For a proof do exercise 14.5.3 in B[1984]. In each of three cases we can inspect that the statement holds. ■

2.1.19. THEOREM (Weak normalization theorem for $\lambda_{\to}$). *If $M \in \Lambda$ is typable in $\lambda_{\to}$, then M is $\boldsymbol{\beta\eta}$-WN, i.e. has a $\boldsymbol{\beta\eta}$-nf.*

PROOF. By Proposition 1.4.9(ii) it suffices to show this for terms in $\lambda^{\mathrm{Ch}}_{\rightarrow}$. Note $\boldsymbol{\eta}$-reductions decreases the length of a term; moreover, for $\boldsymbol{\beta}$-normal terms $\boldsymbol{\eta}$-contractions do not create $\boldsymbol{\beta}$-redexes. Therefore in order to establish $\boldsymbol{\beta\eta}$-WN it is sufficient to prove that M has a $\boldsymbol{\beta}$-nf.

Define the following $\boldsymbol{\beta}$-reduction strategy F. If M is in nf, then $F(M) = M$. Otherwise, let R be the *rightmost redex of maximal depth* n in M. Then

$$F(M) = N$$

where $M \xrightarrow{R}_{\boldsymbol{\beta}} N$. Contracting a redex can only duplicate other redexes that are to the right of that redex. Therefore by the choice of R there can only be redexes of M duplicated in $F(M)$ of depth $< n$. By lemma 2.1.18 redexes created in $F(M)$ by the contraction $M \rightarrow_{\boldsymbol{\beta}} F(M)$ are also of depth $< n$. Therefore in case M is not in $\boldsymbol{\beta}$-nf we have

$$S_M \rightarrow_{\mathcal{S}} S_{F(M)}.$$

Since $\rightarrow_{\mathcal{S}}$ is SN, it follows that the reduction

$$M \rightarrow_{\boldsymbol{\beta}} F(M) \rightarrow_{\boldsymbol{\beta}} F^2(M) \rightarrow_{\boldsymbol{\beta}} F^3(M) \rightarrow_{\boldsymbol{\beta}} \dots$$

must terminate in a $\boldsymbol{\beta}$-nf. ■

For $\boldsymbol{\beta}$-reduction this weak normalization theorem was first proved by Turing, see Gandy [1980b]. The proof does not really need SN for $\mathcal{S}$-reduction. One may also use the simpler result lemma 2.1.13.

It is easy to see that a different reduction strategy does not yield a $\mathcal{S}$-reduction chain. For example the two terms

$$\begin{array}{l}(\lambda x^A.y^{A\to A\to A}x^Ax^A)^{A\to A}((\lambda x^A.x^A)^{A\to A}x^A) \quad \rightarrow_{\boldsymbol{\beta}} \\ y^{A\to A\to A}((\lambda x^A.x^A)^{A\to A}x^A)((\lambda x^A.x^A)^{A\to A}x^A)\end{array}$$

give the multisets $\{1,1\}$ and $\{1,1\}$. Nevertheless, SN does hold for all systems $\lambda_{\rightarrow}$, as will be proved in Section 2.2. It is an open problem whether ordinals can be assigned in a natural and simple way to terms of $\lambda_{\rightarrow}$ such that

$$M \rightarrow_{\boldsymbol{\beta}} N \;\Rightarrow\; \mathtt{ord}(M) > \mathtt{ord}(N).$$

See Howard [1970] and de Vrijer [1987].

Applications of normalization

We will prove that normal terms inhabiting the represented data types (Bool, Nat, Σ^* and T_B) are standard, i.e. correspond to the intended elements. From WN for $\lambda_{\rightarrow}$ and the subject reduction theorem it then follows that all inhabitants of the mentioned data types are standard.

2.1.20. PROPOSITION. *Let $M \in \Lambda$ be in nf. Then $M \equiv \lambda x_1 \cdots x_n.yM_1 \dots M_m$, with $n, m \geq 0$ and the $M_1, \dots, M_m$ again in nf.*

PROOF. By induction on the structure of M. See Barendregt [1984], proposition 8.3.8 for some details if necessary. ■

2.1.21. PROPOSITION. *Let* $\mathsf{Bool} \equiv \mathsf{Bool}_\alpha$, *with* α *a type variable. Then for* M *in nf one has*

$$\vdash M : \mathsf{Bool} \;\Rightarrow\; M \in \{\mathsf{true}, \mathsf{false}\}.$$

PROOF. By repeated use of proposition 2.1.20, the free variable lemma 2.1.2 and the generation lemma for $\lambda^{\mathrm{Cu}}_{\to}$, proposition 2.1.3, one has the following chain of arguments.

$$\begin{array}{lll} \vdash M : \alpha{\to}\alpha{\to}\alpha & \Rightarrow & M \equiv \lambda x.M_1 \\ & \Rightarrow & x{:}\alpha \vdash M_1 : \alpha{\to}\alpha \\ & \Rightarrow & M_1 \equiv \lambda y.M_2 \\ & \Rightarrow & x{:}\alpha, y{:}\alpha \vdash M_2 : \alpha \\ & \Rightarrow & M_2 \equiv x \text{ or } M_2 \equiv y. \end{array}$$

So $M \equiv \lambda xy.x \equiv \mathsf{true}$ or $M \equiv \lambda xy.y \equiv \mathsf{false}$. ■

2.1.22. PROPOSITION. *Let* $\mathsf{Nat} \equiv \mathsf{Nat}_\alpha$. *Then for* M *in nf one has*

$$\vdash M : \mathsf{Nat} \;\Rightarrow\; M \in \{\ulcorner n \urcorner \mid n \in \mathbb{N}\}.$$

PROOF. Again we have

$$\begin{array}{lll} \vdash M : \alpha{\to}(\alpha{\to}\alpha){\to}\alpha & \Rightarrow & M \equiv \lambda x.M_1 \\ & \Rightarrow & x{:}\alpha \vdash M_1 : (\alpha{\to}\alpha){\to}\alpha \\ & \Rightarrow & M_1 \equiv \lambda f.M_2 \\ & \Rightarrow & x{:}\alpha, f{:}\alpha{\to}\alpha \vdash M_2 : \alpha. \end{array}$$

Now we have

$$\begin{array}{lll} x{:}\alpha, f{:}\alpha{\to}\alpha \vdash M_2 : \alpha & \Rightarrow & [M_2 \equiv x \vee \\ & & [M_2 \equiv f M_3 \;\&\; x{:}\alpha, f{:}\alpha{\to}\alpha \vdash M_3 : \alpha]]. \end{array}$$

Therefore by induction on the structure of M_2 it follows that

$$x{:}\alpha, f{:}\alpha{\to}\alpha \vdash M_2 : \alpha \;\Rightarrow\; M_2 \equiv f^n(x),$$

with $n \geq 0$. So $M \equiv \lambda x f.f^n(x) \equiv \ulcorner n \urcorner$. ■

2.1.23. PROPOSITION. *Let* $\mathsf{Sigma}^* \equiv \mathsf{Sigma}^*_\alpha$. *Then for* M *in nf one has*

$$\vdash M : \mathsf{Sigma}^* \;\Rightarrow\; M \in \{\underline{w} \mid w \in \Sigma^*\}.$$

PROOF. Again we have

$$\begin{array}{lll}
\vdash M:\alpha{\to}(\alpha{\to}\alpha)^k{\to}\alpha & \Rightarrow & M\equiv\lambda x.N\\
& \Rightarrow & x{:}\alpha\vdash N:(\alpha{\to}\alpha)^k{\to}\alpha\\
& \Rightarrow & N\equiv\lambda a_1.N_1\ \&\ x{:}\alpha,a_1{:}\alpha{\to}\alpha\vdash N_1:(\alpha{\to}\alpha)^{k-1}{\to}\alpha\\
& \dots & \\
& \Rightarrow & N\equiv\lambda a_1\cdots a_k.N\ \&\ x{:}\alpha,a_1,\dots,a_k{:}\alpha{\to}\alpha\vdash N_k:\alpha\\
& \Rightarrow & [N_k\equiv x\vee\\
& & [N_k\equiv a_{i_j}{N'}_k\ \&\ x{:}\alpha,a_1,\dots,a_k{:}\alpha{\to}\alpha\vdash {N_k}':\alpha]]\\
& \Rightarrow & N_k\equiv a_{i_1}(a_{i_2}(\cdots(a_{i_p}x)\cdot\cdot))\\
& \Rightarrow & M\equiv\lambda x a_1\cdots a_k.a_{i_1}(a_{i_2}(\cdots(a_{i_p}x)\cdot\cdot))\\
& & \equiv \underline{a_{i_1}a_{i_2}\cdots a_{i_p}}.\ \blacksquare
\end{array}$$

Before we can prove that inhabitants of $\mathsf{tree}[\boldsymbol{\beta}]$ are standard, we have to intoduce an auxiliary notion.

2.1.24. DEFINITION. Given $t\in T[b_1,\dots,b_n]$ define $[t]^{p,l}\in\Lambda$ as follows.

$$\begin{array}{rcl}
[b_i]^{p,l} & = & lb_i;\\
[P(t_1,t_2)]^{p,l} & = & p[t_1]^{p,l}[t_2]^{p,l}.
\end{array}$$

2.1.25. LEMMA. *For $t\in T[b_1,\dots,b_n]$ we have*

$$[t]=_\beta\lambda pl.[t]^{p,l}.$$

PROOF. By induction on the structure of t.

$$\begin{array}{rcll}
[b_i] & \equiv & \lambda pl.lb_i & \\
& \equiv & \lambda pl.[b_i]^{p,l}; & \\
[P(t_1,t_2)] & \equiv & \lambda pl.p([t_1]pl)([t_2]pl) & \\
& = & \lambda pl.p[t_1]^{p,l}[t_2]^{p.l}, & \text{by the IH,}\\
& \equiv & \lambda pl.[P(t_1,t_2)]^{p,l}.\ \blacksquare &
\end{array}$$

2.1.26. PROPOSITION. *Let $\mathsf{tree}[\boldsymbol{\beta}]\equiv\mathsf{tree}_\alpha[\boldsymbol{\beta}]$. Then for M in nf one has*

$$b_1,\dots,b_n{:}\boldsymbol{\beta}\vdash M:\mathsf{tree}[\boldsymbol{\beta}]\ \Rightarrow\ M\in\{[t]\mid t\in T[b_1,\dots,b_n]\}.$$

PROOF. We have $\vec{b}{:}\boldsymbol{\beta} \vdash M : (\alpha{\to}\alpha{\to}\alpha){\to}(\boldsymbol{\beta}{\to}\alpha){\to}\alpha \;\Rightarrow$

$$\begin{array}{lll}
\Rightarrow & M \equiv \lambda p.M' & \\
\Rightarrow & \vec{b}{:}\boldsymbol{\beta}, p{:}\alpha{\to}\alpha{\to}\alpha \vdash M' : (\boldsymbol{\beta}{\to}\alpha){\to}\alpha & \\
\Rightarrow & M' \equiv \lambda l.M'' & \\
\Rightarrow & \vec{b}{:}\boldsymbol{\beta}, p{:}(\alpha{\to}\alpha{\to}\alpha), l{:}(\boldsymbol{\beta}{\to}\alpha) \vdash M'' : \alpha & \\
\Rightarrow & M'' \equiv lb_i \;\vee\; [M'' \equiv pM_1M_2 \;\&\; & \\
 & \vec{b}{:}\boldsymbol{\beta}, p{:}(\alpha{\to}\alpha{\to}\alpha), l{:}(\boldsymbol{\beta}{\to}\alpha) \vdash M_j : \alpha], & \text{j=1,2,} \\
\Rightarrow & M'' \equiv [t]^{p,l}, \text{ for some } t \in T[\vec{b}], & \\
\Rightarrow & M \equiv \lambda pl.[t]^{p,l} =_\beta [t], & \text{by lemma 2.1.25. } \blacksquare
\end{array}$$

2.2. Proofs of strong normalization

We now will give two proofs showing that $\lambda_{\to}$ is strongly normalizing. The first one is the classical proof due to Tait [1967] that needs little technique, but uses set theoretic comprehension. The second proof due to Statman is elementary, but needs results about reduction.

2.2.1. THEOREM (SN for $\lambda_{\to}^{\mathrm{Ch}}$). *For all $A \in \mathbb{T}_\infty$, $M \in \Lambda_{\to}^{\mathrm{Ch}}(A)$ one has $\mathrm{SN}_{\boldsymbol{\beta\eta}}(M)$.*

PROOF. We use an induction loading. First we add to $\lambda_{\to}$ constants $d_\alpha \in \Lambda_{\to}^{\mathrm{Ch}}(\alpha)$ for each atom α, obtaining $\lambda_{\to\mathrm{Ch}}^{+}$. Then we prove SN for the extended system. It follows *a fortiori* that the system without the constants is SN.

One first defines for $A \in \mathbb{T}_\infty$ the following class $\mathcal{C}_A$ of *computable* terms of type A. We write SN for $\mathrm{SN}_{\boldsymbol{\beta\eta}}$.

$$\begin{array}{rcl}
\mathcal{C}_\alpha & = & \{M \in \Lambda_{\to\mathrm{Ch}}^{\emptyset}(\alpha) \mid \mathrm{SN}(M)\}; \\
\mathcal{C}_{A\to B} & = & \{M \in \Lambda_{\to\mathrm{Ch}}^{\emptyset}(A{\to}B) \mid \forall P \in \mathcal{C}_A.MP \in \mathcal{C}_B\}.
\end{array}$$

Then one defines the classes $\mathcal{C}_A^*$ of terms that are *computable under substitution*

$$\mathcal{C}_A^* = \{M \in \Lambda_{\to\mathrm{Ch}}(A) \mid \forall \vec{Q} \in \mathcal{C}.[M[\vec{x}{:=}\vec{Q}] \in \Lambda_{\to\mathrm{Ch}}^{\emptyset}(A) \;\Rightarrow\; M[\vec{x}{:=}\vec{Q}] \in \mathcal{C}_A]\}.$$

Write $\mathcal{C}^{(*)} = \bigcup\{\mathcal{C}_A^{(*)} \mid A \in \mathbb{T}(\lambda_{\to}^{+})\}$. For $A = A_1{\to}\ldots{\to}A_n{\to}\alpha$ define

$$d_A \equiv \lambda x_1{:}A_1 \ldots \lambda x_n{:}A_n.d_\alpha.$$

Then for A one has

$$M \in \mathcal{C}_A \iff \forall \vec{P} \in \mathcal{C}.M\vec{P} \in \mathrm{SN}, \tag{0}$$

$$M \in \mathcal{C}_A^* \iff \forall \vec{P}, \vec{Q} \in \mathcal{C}.M[\vec{x}{:=}\vec{Q}]\vec{P} \in \mathrm{SN}, \tag{1}$$

where the $\vec{P}, \vec{Q}$ should have the right types and $M\vec{P}$ and $M[\vec{x}{:=}\vec{Q}]\vec{P}$ are of type α, respectively. By an easy simultaneous induction on A one can show

$$M \in \mathcal{C}_A \;\Rightarrow\; \mathrm{SN}(M); \tag{2}$$

Chapter 14

An Exemplary System

31.10.2006:581

There are several systems that assign intersection types to untyped lambda terms. These will be collectively denoted by $\lambda_\cap$. In this section we consider one particular system of this family, $\lambda_\cap^{\mathrm{BCD}}$ in order to outline the concepts and related properties. Definitions and the statement of theorems will be given, but no proofs. These can be found in the next chapters of Part III.

One motivation for the system presented comes from trying to modify the system $\lambda_\to$ in such a way that not only subject reduction, but also subject expansion holds. The problem of subject expansion is the following. Suppose $\vdash_{\lambda_\to} M : A$ and that $M' \to\!\!\!\!\!\to_{\boldsymbol{\beta\eta}} M$. Does one have $\vdash_{\lambda_\to} M' : A$? Let us focus on one $\boldsymbol{\beta}$-step. So let $M \equiv (\lambda x.P)Q$ be a redex and suppose

$$\vdash_{\lambda_\to} P[x := Q] : A. \tag{1}$$

Do we have $\vdash_{\lambda_\to} (\lambda x.P)Q : A$? It is tempting to reason as follows. By assumption (1) also Q must have a type, say B. Then $(\lambda x.P)$ has a type $B \to A$ and therefore $\vdash_{\lambda_\to} (\lambda x.P)Q : A$. The mistake is that in (1) there may be several occurrences of Q, say $Q_1 \equiv Q_2 \equiv \ldots \equiv Q_n$, having as types respectively $B_1, \ldots, B_n$. It may be impossible to find a single type for all the occurrences of Q and this prevents us from finding a type for the redex. For example

$$\begin{array}{rcl} \vdash_{\lambda_\to} (\lambda x.\mathsf{I}(\mathsf{K}x)(\mathsf{I}x)) & : & A \to A, \\ \not\vdash_{\lambda_\to} (\lambda xy.x(\mathsf{K}y)(xy))\mathsf{I} & : & A \to A. \end{array}$$

The system introduced in this chapter with intersection types assigned to untyped lambda terms remedies the situation. The idea is that if the several occurrences of Q have to have different types $B_1, \ldots, B_n$, we give them all of these types:

$$\vdash Q : B_1 \cap \ldots \cap B_n,$$

implying that for all i one has $Q : B_i$. Then we have

$$\begin{array}{rcll} \vdash (\lambda x.P) & : & B_1 \cap \ldots \cap B_n \to A & \text{and} \\ \vdash ((\lambda x.P)Q) & : & A. & \end{array}$$

There is, however, a second problem. In the $\lambda\mathsf{K}$-calculus, with its terms $\lambda x.P$ such that $x \notin \mathrm{FV}(P)$ there is the extra problem that Q may not be

typable at all, as it may not occur in $P[x := Q]$! This is remedied by allowing $B_1 \cap \ldots \cap B_n$ also for $n = 0$ and writing this type as $\top$, to be considered as the universal type, i.e. assigned to all terms. Then in case $x \notin \mathrm{FV}(P)$ one has

$$\begin{array}{rcll} \vdash (\lambda x.P) & : & \top\to A & \text{and} \\ \vdash ((\lambda x.P)Q) & : & A. & \end{array}$$

This is the motivation to introduce a $\leq$ relation on types with largest element $\top$ and intersections such that $A\cap B \leq A, A\cap B \leq B$ and the extension of the type assignment by the sub-sumption rule $\Gamma \vdash M : A,\ A \leq B \ \Rightarrow\ \Gamma \vdash M : B$. It has as consequence that terms like $\lambda x.xx$ get as type $((A\to B) \cap A)\to B$, while $(\lambda x.xx)(\lambda x.xx)$ only gets $\top$ as type. Also we have subject conversion

$$\Gamma \vdash M : A \ \&\ M =_\beta N \ \Rightarrow\ \Gamma \vdash N : A.$$

This has as consequence that one can create a lambda model in which the meaning of a closed term consists of the collection of types it gets. In this way new lambda models will be obtained and new ways to study classical models as well.

The type assignement system $\lambda^{\mathrm{BCD}}_\cap$ will be introduced in Section 14.1 and the correspondig filter model in 14.2.

14.1. The system of type assignment $\lambda^{\mathrm{BCD}}_\cap$

A typical member of the family of intersection type assignment systems is $\lambda^{\mathrm{BCD}}_\cap$. This system is introduced in Barendregt et al. [1983] as an extension of the initial system in Coppo and Dezani-Ciancaglini [1980].

14.1.1. Definition. Let $\mathbb{A}$ be a set of type atoms.

(i) The *intersection type language* over $\mathbb{A}$, denoted by $\mathbb{T} = \mathbb{T}^{\mathbb{A}}_\cap$ is defined by the following abstract syntax.

$$\mathbb{T} = \mathbb{A} \mid \mathbb{T}\to\mathbb{T} \mid \mathbb{T} \cap \mathbb{T}$$

(ii) Write

$$\begin{array}{rcl} \mathbb{A}_\infty & = & \{\psi_0, \psi_1, \psi_2, \ldots\} \\ \mathbb{A}^\top_\infty & = & \mathbb{A}_\infty \cup \{\top\}, \end{array}$$

where the type atom $\top \notin \mathbb{A}_\infty$ is considered as a constant.

Notation. (i) A, B, C, D, E range over arbitrary types. When writing intersection types we shall use the following convention: the constructor $\cap$ takes precedence over the constructor $\to$ and it associates to the right. For example

$$(A\to B\to C) \cap A\to B\to C \equiv ((A\to(B\to C)) \cap A)\to(B\to C).$$

(ii) $\alpha, \beta, \ldots$ range over $\mathbb{A}$.

14.1.2. Remark. In Part III the set of syntactic types will be formed as above; for many of these systems the set $\mathbb{A}$ will be finite. In this Chapter, however, we take $\mathbb{A} = \mathbb{A}^{\top}_{\infty}$.

The following deductive system has as intention to introduce an appropriate pre-order on $\mathbb{T}$, compatible with the operator $\to$, such that $A \cap B$ is a greatest lower bound of A and B, for each A, B.

14.1.3. Definition (Intersection type preorder). On $\mathbb{T} = \mathbb{T}^{\mathbb{A}^{\top}_{\infty}}_{\cap}$ a binary relation $\leq$ '*is subtype of*' is defined by the following axioms and rules.

(refl)	$A \leq A$
(incl_L)	$A \cap B \leq A$
(incl_R)	$A \cap B \leq B$
(glb)	$\dfrac{C \leq A \quad C \leq B}{C \leq A \cap B}$
(trans)	$\dfrac{A \leq B \quad B \leq C}{A \leq C}$
($\top$)	$A \leq \top$
($\top\to$)	$\top \leq \top\to\top$
($\to\cap$)	$(A\to B) \cap (A\to C) \leq A\to(B \cap C)$
($\to$)	$\dfrac{A' \leq A \quad B \leq B'}{(A\to B) \leq (A'\to B')}$

14.1.4. Definition. The intersection type theory BCD is the set of all judgements $A \leq B$ derivable from the axioms and rules in Definition 14.1.3. For $(A \leq B) \in \mathrm{BCD}$ we write $A \leq_{\mathrm{BCD}} B$ or $\vdash_{\mathrm{BCD}} A \leq B$ (or often just $A \leq B$).

14.1.5. Remark. All systems in Part III have the first five axioms and rules of Definition 14.1.3. They differ in the extra axioms and rules and the set of constants.

14.1.6. Definition. Write $A =_{\mathrm{BCD}} B$ (or $A = B$) for $A \leq_{\mathrm{BCD}} B \;\&\; B \leq_{\mathrm{BCD}} A$. In BCD we usually work with $\mathbb{T}$ modulo $=_{\mathrm{BCD}}$. By rule ($\to$) one has

$$A = A' \;\&\; B = B' \;\Rightarrow\; (A\to B) = (A'\to B').$$

Moreover, $A \cap B$ becomes *the* glb of A, B.

14.1.7. Definition. (i) A *basis* is a finite set of statements of the shape $x{:}B$, where $B \in \mathbb{T}$, with all variables distinct.

(ii) The type assignment system $\lambda^{\mathrm{BCD}}_{\cap}$ for deriving statements of the form $\Gamma \vdash M : A$ with Γ a basis, $M \in \Lambda$ (the set of untyped lambda terms) and $A \in \mathbb{T}$

is defined by the following axioms and rules.

(Ax)	$\Gamma \vdash x{:}A$	if $(x{:}A) \in \Gamma$
($\to$I)	$\dfrac{\Gamma,\ x{:}A \vdash M : B}{\Gamma \vdash (\lambda x.M) : (A \to B)}$	
($\to$E)	$\dfrac{\Gamma \vdash M : (A \to B) \quad \Gamma \vdash N : A}{\Gamma \vdash (MN) : B}$	
($\cap$I)	$\dfrac{\Gamma \vdash M : A \quad \Gamma \vdash M : B}{\Gamma \vdash M : (A \cap B)}$	
($\leq$)	$\dfrac{\Gamma \vdash M : A}{\Gamma \vdash M : B}$	if $A \leq_{\mathrm{BCD}} B$
($\top$-universal)	$\Gamma \vdash M : \top$	

(iii) We say that a term M is *typable* from a given basis Γ, if there is a type $A \in \mathbb{T}$ such that the judgement $\Gamma \vdash M : A$ is derivable in $\lambda^{\mathrm{BCD}}_{\cap}$. In this case we write $\Gamma \vdash^{BCD}_{\cap^{\top}} M : A$ or just $\Gamma \vdash M : A$, if there is little danger of confusion.

14.1.8. Remark. All systems of type assignment in Part III have the first five axioms and rules of Definition 14.1.7.

In the following Proposition we need the notions of admissible and derived rule. Let us first informally define these notions for the simple logical theory of propositional logic.

14.1.9. Definition. Let $\vdash$ denote provability in propositional logic. Consider the rule

$$\frac{\Gamma \vdash A}{\Gamma \vdash B} \qquad (R)$$

(i) R is called *admissible* if one has

$$\Gamma \vdash A \ \Rightarrow\ \Gamma \vdash B$$

(ii) R is called *derived* if one has

$$\Gamma \vdash A \to B$$

For example we have that

$$\frac{\Gamma \vdash A \to A \to B}{\Gamma \vdash A \to B}$$

is derived. Also that for propositional variables ϑ, ϱ

$$\frac{\vdash \vartheta}{\vdash \varrho}$$

is admissible, simply because $\vdash \vartheta$ does not hold, but not derived. A derived rule is always admissible and the example shows that the converse does not hold. If

$$\frac{\Gamma \vdash A}{\Gamma \vdash B}$$

is a derived rule, then for all $\Gamma' \supseteq \Gamma$ one has that

$$\frac{\Gamma' \vdash A}{\Gamma' \vdash B}$$

is also derived. Hence derived rules are closed under theory extension.

We will only be concerned with admissible and derived rules for theories of type assignment.

14.1.10. Proposition. (i) *Notice that the rules ($\cap$E)*

$$\frac{\Gamma \vdash M : (A \cap B)}{\Gamma \vdash M : A} \qquad \frac{\Gamma \vdash M : (A \cap B)}{\Gamma \vdash M : B}$$

are derived in $\lambda^{\mathrm{BCD}}_{\cap}$.

(ii) *The following rules are admissible in the intersection type assignment system* $\lambda^{\mathrm{BCD}}_{\cap}$.

(*weakening*)	$\dfrac{\Gamma \vdash M : A \quad x \notin \Gamma}{\Gamma, x{:}B \vdash M : A}$
(*strengthening*)	$\dfrac{\Gamma, x{:}B \vdash M : A \quad x \notin FV(M)}{\Gamma \vdash M : A}$
(*cut*)	$\dfrac{\Gamma, x{:}B \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash (M[x := N]) : A}$
($\leq$-L)	$\dfrac{\Gamma, x{:}B \vdash M : A \quad C \leq B}{\Gamma, x{:}C \vdash M : A}$
($\to$L)	$\dfrac{\Gamma, y{:}B \vdash M : A \quad \Gamma \vdash N : C \quad x \notin \Gamma}{\Gamma, x{:}(C \to B) \vdash (M[y := xN]) : A}$
($\cap$L)	$\dfrac{\Gamma, x{:}A \vdash M : B}{\Gamma, x{:}(A \cap C) \vdash M : B}$

14.1.11. Theorem. *In (i) assume $A \neq \top$. Then*

$$\begin{array}{lrcl}
\text{(i)} & \Gamma \vdash x : A & \Leftrightarrow & \exists B \in \mathbb{TT}.[(x{:}B \in \Gamma \ \&\ B \leq A]. \\
\text{(ii)} & \Gamma \vdash (MN) : A & \Leftrightarrow & \exists B \in \mathbb{TT}.[\Gamma \vdash M : (B \to A) \ \&\ \Gamma \vdash N : B]. \\
\text{(iii)} & \Gamma \vdash \lambda x.M : A & \Leftrightarrow & \exists n{>}0 \exists B_1, \ldots, B_n, C_1, \ldots, C_n \in \mathbb{TT} \\
 & & & \forall i \in \{1, \ldots, n\}.[\Gamma, x{:}B_i \vdash M : C_i \ \& \\
 & & & (B_1 \to C_1) \cap \ldots \cap (B_n \to C_n) \leq A]. \\
\text{(iv)} & \Gamma \vdash \lambda x.M : B \to C & \Leftrightarrow & \Gamma, x{:}B \vdash M : C.
\end{array}$$

14.1.12. DEFINITION. Let R be a notion of reduction. We introduce the following rules:

$$(R\text{-}red)\quad \frac{\Gamma \vdash M : A \quad M \to_R N}{\Gamma \vdash N : A}$$

$$(R\text{-}exp)\quad \frac{\Gamma \vdash M : A \quad M \leftarrow_R N}{\Gamma \vdash N : A}$$

14.1.13. PROPOSITION. *The rules (β-red), (β-exp) and (η-red) are admissible in $\lambda^{\mathrm{BCD}}_{\cap}$. The rule ($\eta$-exp) is not.*

The following result characterizes notions related to normalization in terms of type assignment in the system $\lambda^{\mathrm{BCD}}_{\cap}$. The notation $\top \notin A$ means that $\top$ does not occur in A.

14.1.14. THEOREM. *Let $M \in \Lambda^{\emptyset}$.*

(i) *M has a head normal form* $\Leftrightarrow$ $\exists A \in \mathbb{T}.[A \neq_{\mathrm{BCD}} \top \;\&\; \vdash M : A]$.

(ii) *M has a normal form* $\Leftrightarrow$ $\exists A \in \mathbb{T}.[\top \notin A \;\&\; \vdash M : A]$.

Let M be a lambda term. For the notion 'approximant of M', see Barendregt [1984]. These are roughly obtained from the Böhm tree $\mathrm{BT}(M)$ of M by cutting of branches and replacing these by a new symbol $\bot$. The set of approximants of M is denoted by $\mathcal{A}(M)$. We have e.g. for the fixed-point combinator Y

$$\mathcal{A}(\mathsf{Y}) = \{\bot\} \cup \{\lambda f.f^n\bot \mid n{>}0\}.$$

Approximants are being typed by letting the typing rules be valid for approximants. For example one has

$$\begin{array}{l}
\vdash \bot : \top \\
\vdash \lambda f.f\bot : (\top{\to}A_1){\to}A_1 \\
\vdash \lambda f.f(f\bot) : (\top{\to}A_1) \cap (A_1{\to}A_2){\to}A_2 \\
\ldots \\
\vdash \lambda f.f^n\bot : (\top{\to}A_1) \cap (A_1{\to}A_2) \cap \ldots \cap (A_{n-1}{\to}A_n){\to}A_n \\
\ldots
\end{array}$$

The set of types of a term M coincides with the union of the sets of types of its approximants $P \in \mathcal{A}(M)$. This will give an Approximation Theorem for the filter model of next section.

14.1.15. THEOREM. $\Gamma \vdash M : A \;\Leftrightarrow\; \exists P \in \mathcal{A}(M).\Gamma \vdash P : A$.

For example since for all n $\lambda f.f^n\bot$ is an approximant of Y we have that all types of the shape $(\top{\to}A_1) \cap \ldots \cap (A_{n-1}{\to}A_n){\to}A_n$ can be derived for Y.

Finally the question whether an intersection type is inhabited is undecidable.

14.1.16. THEOREM. *The set $\{A \in \mathbb{T} \mid \exists M \in \Lambda^{\emptyset} \vdash M : A\}$ is undecidable.*

14.2. The filter model

14.2.1. Definition. (i) A *complete lattice* $(\mathcal{D}, \sqsubseteq)$ is a partial order which has arbitrary least upper bounds (sup's) (and hence has arbitrary inf's).

(ii) A subset $Z \subseteq \mathcal{D}$ is *directed* if $Z \neq \emptyset$ and

$$\forall x, y \in Z \exists z \in Z. x, y \sqsubseteq z.$$

(iii) An element $c \in \mathcal{D}$ is *compact* (in the literature also called *finite*) if for each directed $Z \subseteq \mathcal{D}$ one has

$$c \sqsubseteq \bigsqcup Z \;\Rightarrow\; \exists z \in Z. c \sqsubseteq z.$$

Let $\mathcal{K}(\mathcal{D})$ denote the set of compact elements of $\mathcal{D}$.

(iv) A complete lattice is *ω-algebraic* if $\mathcal{K}(\mathcal{D})$ is countable, and for each $d \in \mathcal{D}$, the set $\mathcal{K}(d) = \{c \in \mathcal{K}(\mathcal{D}) \mid c \sqsubseteq d\}$ is directed and $d = \bigsqcup \mathcal{K}(d)$.

(v) Let $(\mathcal{D}, \sqsubseteq)$ be an *ω-algebraic* complete lattice. The Scott topology on $\mathcal{D}$ contains as open sets the $U \subseteq \mathcal{D}$ such that

(1) $d \in U \;\&\; d \sqsubseteq e \;\Rightarrow\; e \in U$;

(2) if $Z \subseteq \mathcal{D}$ is directed then $\bigsqcup Z \in U \;\Rightarrow\; \exists z \in Z. z \in U$.

(vi) If $\mathcal{D}, \mathcal{E}$ are ω-algebraic complete lattices, then $[\mathcal{D} \to \mathcal{E}]$ denotes the set of continuous maps from $\mathcal{D}$ to $\mathcal{E}$. This set can be ordered pointwise

$$f \sqsubseteq g \;\Leftrightarrow\; \forall d \in \mathcal{D}. f(d) \sqsubseteq g(d)$$

and $\langle [\mathcal{D} \to \mathcal{E}], \sqsubseteq \rangle$ is again an ω-algebraic lattice.

(vii) The category **ALG** is the category whose objects are the ω-algebraic complete lattices and whose morphisms are the (Scott) continuous functions.

14.2.2. Definition. (i) A *filter* over $\mathbb{T} = \mathbb{T}^{\mathbb{A}^\top_\infty}_\cap$ is a non-empty set $X \subseteq \mathbb{T}$ such that

(1) $A \in X \;\&\; A \leq B \;\Rightarrow\; B \in X$;

(2) $A, B \in X \;\Rightarrow\; (A \cap B) \in X$.

(ii) $\mathcal{F}$ denotes the set of filters over $\mathbb{T}$.

14.2.3. Definition. (i) If $X \subseteq \mathbb{T}$ is non-empty, then the filter *generated by* X, notation $\uparrow X$, is the least filter containing X. Note that

$$\uparrow X = \{A \mid \exists n \geq 1 \, \exists B_1 \ldots B_n \in X. B_1 \cap \ldots \cap B_n \leq A\}.$$

(ii) A *principal* filter is of the form $\uparrow\{A\}$ for some $A \in \mathbb{T}$. We shall denote this simply by $\uparrow A$. Note that $\uparrow A = \{B \mid A \leq B\}$.

14.2.4. Proposition. (i) *$\mathcal{F} = \langle \mathcal{F}, \subseteq \rangle$ is an ω-algebraic complete lattice.*

(ii) *$\mathcal{F}$ has as bottom element $\uparrow\top$ and as top element $\mathbb{T}$.*

(iii) *The compact elements of $\mathcal{F}$ are exactly the principal filters.*

14.2.5. DEFINITION. Let $\mathcal{D}$ be an ω-algebraic lattice and let

$$\begin{array}{rcl} F & : & \mathcal{D}\to[\mathcal{D}\to\mathcal{D}] \\ G & : & [\mathcal{D}\to\mathcal{D}]\to\mathcal{D} \end{array}$$

be Scott continuous. $\mathcal{D}$ is called a *reflexive* via F, G if $F \circ G = \mathtt{id}_{[\mathcal{D}\to\mathcal{D}]}$.

A reflexive element of **ALG** is also a λ-model in which the term interpretation is naturally defined as follows (see Barendregt [1984], Section 5.4).

14.2.6. DEFINITION (Interpretation of terms). Let $\mathcal{D}$ be reflexive via F, G.

(i) A *term environment* in $\mathcal{D}$ is a map $\rho : \mathtt{Var}\to\mathcal{D}$.

(ii) If ρ is a term environment and $d\in\mathcal{D}$, then $\rho(x := d)$ is the term environment ρ' defined by

$$\begin{array}{rcll} \rho'(y) & = & \rho(y) & \text{if } y \not\equiv x; \\ \rho'(x) & = & d. & \end{array}$$

(iii) Given a term environment ρ, the interpretation $[\![\ ]\!]_\rho : \Lambda\to\mathcal{D}$ is defined as follows.

$$\begin{array}{rcl} [\![x]\!]^{\mathcal{D}}_\rho & = & \rho(x); \\ [\![MN]\!]^{\mathcal{D}}_\rho & = & F[\![M]\!]^{\mathcal{D}}_\rho[\![N]\!]^{\mathcal{D}}_\rho; \\ [\![\lambda x.M]\!]^{\mathcal{D}}_\rho & = & G(\boldsymbol{\lambda} d\in\mathcal{D}.[\![M]\!]^{\mathcal{D}}_{\rho(x:=d)}). \end{array}$$

(iv) The statement $M = N$, for M, N untyped lambda terms, is *true in* $\mathcal{D}$, notation $\mathcal{D} \models M = N$ iff

$$\forall\rho\in\mathsf{Env}_D.[\![M]\!]^{\mathcal{D}}_\rho = [\![N]\!]^{\mathcal{D}}_\rho.$$

14.2.7. THEOREM. *Let $\mathcal{D}$ be reflexive via F, G. Then $\mathcal{D}$ is a λ-model, in particular for all $M, N\in\Lambda$*

$$\mathcal{D} \models (\lambda x.M)N = M[x := N].$$

14.2.8. PROPOSITION. *Define maps $F : \mathcal{F}\to[\mathcal{F}\to\mathcal{F}]$ and $G : [\mathcal{F}\to\mathcal{F}]\to\mathcal{F}$ by*

$$\begin{array}{rcl} F(X)(Y) & = & \uparrow\{B \mid \exists A\in Y.(A\to B)\in X\} \\ G(f) & = & \uparrow\{A\to B \mid B\in f(\uparrow A)\}. \end{array}$$

Then $\mathcal{F}$ is reflexive via F, G. Therefore $\mathcal{F}$ is a λ-model.

An important property of the λ-model $\mathcal{F}$ is that the meaning of a term is the set of types which are deducible for it.

14.2.9. THEOREM. *For all λ-terms M one has*

$$[\![M]\!]^{\mathcal{F}}_\rho = \{A \mid \exists\Gamma \models \rho.\Gamma \vdash M : A\},$$

where $\Gamma \models \rho$ iff for all $(x{:}B)\in\Gamma$ one has $B\in\rho(x)$.

Lastly we notice that all continous functions are representable.

14.2.10. THEOREM.

$$[\mathcal{F}\to\mathcal{F}] = \{f : \mathcal{F}\to\mathcal{F} \mid f \textit{ is representable}\},$$

where $f \in \mathcal{F}\to\mathcal{F}$ *is called* representable *iff for some* $X \in \mathcal{F}$ *one has*

$$\forall Y \in \mathcal{F}.f(Y) = F(X)(Y).$$

14.3. Completeness of type assignment

14.3.1. DEFINITION (Interpretation of types). Let $\mathcal{D}$ be reflexive via F, G and hence a λ-model. For $F(d)(e)$ we also write (as usual) $d \cdot e$.

(i) A *type environment* in $\mathcal{D}$ is a map $\xi : \mathbb{A}_\infty\to\mathcal{P}(\mathcal{D})$.

(ii) For $X, Y \in \mathcal{P}(\mathcal{D})$ define

$$X\to Y = \{d\in\mathcal{D} \mid d \cdot X \subseteq Y\} = \{d\in\mathcal{D} \mid \forall x \in X.d \cdot x \in Y\}.$$

(iii) Given a type environment ξ, the interpretation $[\![\ ]\!]_\xi : \mathbb{T}\to\mathcal{P}(\mathcal{D})$ is defined as follows.

$$\begin{array}{rcll} [\![\top]\!]^{\mathcal{D}}_\xi & = & \mathcal{D}; & \\ [\![\alpha]\!]^{\mathcal{D}}_\xi & = & \xi(\alpha), & \text{for } \alpha\in\mathbb{A}_\infty; \\ [\![A\to B]\!]^{\mathcal{D}}_\xi & = & [\![A]\!]^{\mathcal{D}}_\xi\to[\![B]\!]^{\mathcal{D}}_\xi; & \\ [\![A\cap B]\!]^{\mathcal{D}}_\xi & = & [\![A]\!]^{\mathcal{D}}_\xi \cap [\![B]\!]^{\mathcal{D}}_\xi. & \end{array}$$

14.3.2. DEFINITION (Satisfaction). (i) Given a λ-model $\mathcal{D}$, a term environment ρ and a type environment ξ one defines the following.

$$\begin{array}{rcll} \mathcal{D},\rho,\xi \models M : A & \Leftrightarrow & [\![M]\!]^{\mathcal{D}}_\rho \in [\![A]\!]^{\mathcal{D}}_\xi. & \\ \mathcal{D},\rho,\xi \models \Gamma & \Leftrightarrow & \mathcal{D},\rho,\xi \models x : B, & \text{for all } (x{:}B)\in\Gamma. \end{array}$$

(ii) $\Gamma \models M : A \Leftrightarrow \forall\mathcal{D},\rho,\xi.[\mathcal{D},\rho,\xi \models \Gamma \Rightarrow \rho,\xi \models M : A]$.

14.3.3. THEOREM (Soundness).

$$\Gamma \vdash M : A \Rightarrow \Gamma \models M : A.$$

14.3.4. THEOREM (Completeness).

$$\Gamma \models M : A \Rightarrow \Gamma \vdash M : A.$$

The completeness proof is an application of the λ-model $\mathcal{F}$, see Barendregt et al. [1983].

Chapter 15

The Systems $\lambda^{\mathcal{T}}_{\bigcap}$ and $\lambda^{\mathcal{T}}_{\bigcap^{\top}}$ 31.10.2006:581

Intersection types are syntactic objects forming a free algebra $\mathbb{T}$, which is generated from a set of atoms $\mathbb{A}$, using the operators $\to$ and $\cap$. Postulating axioms and rules an *intersection type theory* results, which characterizes a preorder $\leq_{\mathcal{T}}$ on $\mathbb{T}$ with $\cap$ as set intersection, giving for two elements a greatest lower bound (glb). The class of these theories is abbreviated[1] as TT.

Taking into account the intuitive meaning of $\to$ as function space constructor one usually requires that the resulting equivalence relation $=_{\mathcal{T}}$ is a congruence. Then we speak of a *compatible* type theory, having a corresponding *type structure*

$$\langle \mathcal{S}, \leq, \cap, \to \rangle = \langle \mathbb{T}/=_{\mathcal{T}}, \leq, \cap, \to \rangle.$$

The collection of type structures is denoted by TS. Each type structure can be seen as coming from a compatible type theory and compatible type theories and type structures are basically the same. In the present Part III of this book both these syntactic and semantic aspects will be exploited.

TT $^{\top}$ is a subset of TT, the set of *top type theories*, where the set of atoms $\mathbb{A}$ has a top element $\top$. Similarly a top intersection type structure TS$^{\top}$ is of the form $\langle \mathcal{S}, \leq, \cap, \to, \top \rangle$.

The various type theories (and type structures) are introduced together in order to give reasonably uniform proofs of their properties as well of those of the corresponding type assignment systems and filter models.

Given a (top) type theory $\mathcal{T}$, one can define a corresponding type assignment system. These type assignment systems will be studied extensively in later chapters. We also introduce so-called *filters*, sets of types closed under intersection $\cap$ and preorder $\leq$. These play an important role in Chapter 17 to establish equivalences of categories and in Chapter 18 to build λ-models.

In Section 15.1 we define the notion of type theory and introduce 13 specific examples, including basic lemmas for these. In Section 15.2 the type assignment systems are defined. In Section 15.3 we discuss intersection type structures and introduce specific categories of lattices and type structures to accommodate these. Finally in Section 15.4 the filters are defined.

[1] Since all type theories in Part III of this book are using the intersection operator, we keep this implicit and often simply speak about *(top) type theories*, leaving 'intersection' implicit.

15.1. Type theories

As in Chapter 14 we will use as syntactic types $\mathbb{T} = \mathbb{T}^{\mathbb{A}}_{\cap}$ defined by

$$\mathbb{T} = \mathbb{A} \mid \mathbb{T}{\to}\mathbb{T} \mid \mathbb{T} \cap \mathbb{T}$$

as abstract syntax. This time we will use various sets of atoms $\mathbb{A}$. The letters $\alpha, \beta, \gamma, \ldots$ range over arbitrary atoms. If we need special atoms for a special purpose, like for example $\top, \omega, \varphi$, then we can identify them with some of the ψ_i, i.e. $\top = \psi_0,\ \omega = \psi_1,\ \varphi = \psi_2$.

15.1.1. Definition. (i) An *intersection type theory over a set of type atoms* $\mathbb{A}$ is a set of judgements $\mathcal{T}$ of the form $A \leq B$ (to be read: A is a subtype of B), with $A, B \in \mathbb{T}^{\mathbb{A}}_{\cap}$, satisfying the following axioms and rules.

(refl)	$A \leq A$
(incl_L)	$A \cap B \leq A$
(incl_R)	$A \cap B \leq B$
(glb)	$\dfrac{C \leq A \quad C \leq B}{C \leq A \cap B}$
(trans)	$\dfrac{A \leq B \quad B \leq C}{A \leq C}$

This means that e.g. $(A \leq A) \in \mathcal{T}$ and $(A \leq B), (B \leq C) \in \mathcal{T} \ \Rightarrow\ (A \leq C) \in \mathcal{T}$, for all A, B, C.

(ii) A *top intersection type theory* is an intersection type theory with an element $\top \in \mathbb{T}$ for which one can derive

($\top$)	$A \leq \top$

(iii) The notion '(top) intersection type theory' will be abbreviated as '(top) type theory', as the 'intersection' part is default.

(iv) TT stands for the set of type theories and $\text{TT}^{\top}$ for that of top type theories.

(v) If $\mathcal{T} \in \text{TT}^{(\top)}$ over $\mathbb{A}$, then we also write $\mathbb{T}^{\mathcal{T}}$ for $\mathbb{T}^{\mathbb{A}}_{\cap}$.

In this and the next section $\mathcal{T}$ ranges over elements of $\text{TT}^{(\top)}$. Most of them have some extra axioms or rules, the above set being the minimum requirement. For example the theory BCD over $\mathbb{A} = \mathbb{A}^{\top}_{\infty}$, defined in Chapter 14 is a $\text{TT}^{\top}$ and has the extra axioms $(\top{\to})$ and $({\to}\cap)$ and rule $({\to})$.

15.1.2. Notation. Let $\mathcal{T} \in \text{TT}$. We write the following.

(i) $A \leq_{\mathcal{T}} B$ or $\vdash_{\mathcal{T}} A \leq B$ for $(A \leq B) \in \mathcal{T}$.

(ii) $A =_{\mathcal{T}} B$ for $A \leq_{\mathcal{T}} B \leq_{\mathcal{T}} A$.

(iii) $A <_{\mathcal{T}} B$ for $A \leq B \ \&\ A \neq_{\mathcal{T}} B$.

(iv) If there is little danger of confusion and $\mathcal{T}$ is clear from the context, then we will write $\leq, =, <$ for respectively $\leq_{\mathcal{T}}, =_{\mathcal{T}}, <_{\mathcal{T}}$.

(v) We write $A \equiv B$ for syntactic identity. E.g. $A \cap B \equiv A \cap B$, but $A \cap B \not\equiv B \cap A$.

15.1.3. LEMMA. *For any $\mathcal{T}$ one has $A \cap B =_{\mathcal{T}} B \cap A$.*

PROOF. By (incl_L), (incl_R) and (glb). ■

15.1.4. DEFINITION. $\mathcal{T}$ is called *compatible* iff the following rule holds.

$$\boxed{(\to^{=}) \quad \frac{A = A' \quad B = B'}{(A \to B) = (A' \to B')}}$$

This means $A =_{\mathcal{T}} A' \;\&\; B =_{\mathcal{T}} B' \;\Rightarrow\; (A \to B) =_{\mathcal{T}} (A' \to B')$. One way to insure this is to adopt $(\to^{=})$ as rule determining $\mathcal{T}$.

15.1.5. REMARKS. (i) Let $\mathcal{T}$ be compatible. Then by Lemma 15.1.3 one has

$$(A \cap B) \to C = (B \cap A) \to C.$$

(ii) The rule (glb) implies that the following rule is admissible.

$$\boxed{(\text{mon}) \quad \frac{A \leq A' \quad B \leq B'}{A \cap B \leq A' \cap B'}}$$

A $\mathcal{T} \in \text{TT}$ can be seen as a structure with a pre-order

$$\mathcal{T} = \langle \mathbb{T}, \leq, \cap, \to \rangle.$$

This means that $\leq$ is reflexive and transitive, but not necessarily anti-symmetric

$$A \leq_{\mathcal{T}} B \;\&\; B \leq_{\mathcal{T}} A \;\not\Rightarrow\; A =_{\mathcal{T}} B.$$

If $\mathcal{T}$ is compatible one can go over to equivalence classes and obtain a type structure

$$\mathcal{T}/{=_{\mathcal{T}}} = \langle \mathbb{T}/{=_{\mathcal{T}}}, \leq, \cap, \to \rangle.$$

If moreover $\mathcal{T} \in \text{TT}^{\top}$, then $\mathcal{T}/{=_{\mathcal{T}}}$ has top $[\top]$. In this structure $A \cap B$ is $\inf\{A, B\}$, the greatest lower bound of A and B. If $\mathcal{T}$ is also compatible, then $\to$ can be properly defined on the equivalence classes. This will be done in Section 15.3.

Specific intersection type theories

Now we will construct several, in total thirteen, type theories that will play an important role in later chapters, by introducing the following axiom schemes, rule schemes and axioms. Only two of them are non-compatible, so we obtain eleven type structures.

In the following φ, ω and $\top$ are distinct atoms differing from those in $\mathbb{A}_\infty$.

15.1.6. NOTATION. We introduce names for axiom(scheme)s and rule(scheme)s in Figure 15.1. Using these names a list of well-studied type structures can be specified in Figure 15.2 as the set of judgements axiomatized by mentioned rule(scheme)s and axiom(scheme)s.

Axioms	
$(\omega_{\mathrm{Scott}})$	$(\top\to\omega) = \omega$
(ω_{Park})	$(\omega\to\omega) = \omega$
$(\omega\varphi)$	$\omega \leq \varphi$
$(\varphi\to\omega)$	$(\varphi\to\omega) = \omega$
$(\omega\to\varphi)$	$(\omega\to\varphi) = \varphi$
(I)	$(\varphi\to\varphi)\cap(\omega\to\omega) = \varphi$
Axiom schemes	
$(\top)$	$A \leq \top$
$(\top\to)$	$\top \leq (A\to\top)$
$(\top_{\mathrm{lazy}})$	$(A\to B) \leq (\top\to\top)$
$(\to\cap)$	$(A\to B)\cap(A\to C) \leq A\to B\cap C$
$(\to\cap^{=})$	$(A\to B)\cap(A\to C) = A\to B\cap C$
Rule schemes	
$(\to)$	$\dfrac{A' \leq A \quad B \leq B'}{(A\to B) \leq (A'\to B')}$
$(\to^{=})$	$\dfrac{A' = A \quad B = B'}{(A\to B) = (A'\to B')}$

Figure 15.1: Possible Axioms and Rules concerning $\leq$.

15.1.7. DEFINITION. In Figure 15.2 a collection of TTs is defined. For each name $\mathcal{T}$ a set of atoms $\mathbb{A}^{\mathcal{T}}$ and a set of rules and axiom(scheme)s are given. The type theory $\mathcal{T}$ is the smallest set of judgements of the form $A \leq B$ with $A, B \in \mathbb{T}^{\mathcal{T}} = \mathbb{T}^{\mathbb{A}^{\mathcal{T}}}_{\cap}$ which is closed under the axiom(scheme)s and the rule(scheme)s of Definition 15.1.1 and the corresponding ones in Figure 15.2.

15.1.8. REMARK. (i) Note that CDS and CD are non-compatible, while the other eleven are compatible.

$\mathcal{T}$	$\mathbb{A}^{\mathcal{T}}$	Rules	Axiom Schemes	Axioms
Scott	$\{\top,\omega\}$	$(\to)$	$(\to\cap),(\top),(\top\to)$	$(\omega_{\mathrm{Scott}})$
Park	$\{\top,\omega\}$	$(\to)$	$(\to\cap),(\top),(\top\to)$	(ω_{Park})
CDZ	$\{\top,\varphi,\omega\}$	$(\to)$	$(\to\cap),(\top),(\top\to)$	$(\omega\varphi),(\varphi\to\omega),(\omega\to\varphi)$
HR	$\{\top,\varphi,\omega\}$	$(\to)$	$(\to\cap),(\top),(\top\to)$	$(\omega\varphi),(\varphi\to\omega),(I)$
DHM	$\{\top,\varphi,\omega\}$	$(\to)$	$(\to\cap),(\top),(\top\to)$	$(\omega\varphi),(\omega\to\varphi),(\omega_{\mathrm{Scott}})$
BCD	$\mathbb{A}^{\top}_{\infty}$	$(\to)$	$(\to\cap),(\top),(\top\to)$	
AO	$\{\top\}$	$(\to)$	$(\to\cap),(\top)$	$(\top_{\mathrm{lazy}})$
Plotkin	$\{\top,\omega\}$	$(\to^{=})$	$(\top)$	–
Engeler	$\mathbb{A}^{\top}_{\infty}$	$(\to^{=})$	$(\to\cap^{=}),(\top),(\top\to)$	–
CDS	$\mathbb{A}^{\top}_{\infty}$	–	$(\top)$	–
HL	$\{\varphi,\omega\}$	$(\to)$	$(\to\cap)$	$(\omega\varphi),(\omega\to\varphi),(\varphi\to\omega)$
CDV	$\mathbb{A}_{\infty}$	$(\to)$	$(\to\cap)$	–
CD	$\mathbb{A}_{\infty}$	–	–	–

Figure 15.2: Various type theories

(ii) The first ten type theories of Figure 15.2 belong clearly to $\mathrm{TT}^{\top}$. In Lemma 15.1.14(i) we will see that also $\mathrm{HL}\in\mathrm{TT}^{\top}$ with φ as top. Instead CDS and CD do not belong to $\mathrm{TT}^{\top}$, as shown in Lemma 15.1.14(ii) and (iii).

In this list the given order is logical, rather than historical, and some of the references define the models directly, others deal with the corresponding filter models (see Sections 17 and 18): Scott [1972], Park [1976], Coppo et al. [1987], Honsell and Ronchi Della Rocca [1992], Dezani-Ciancaglini et al. [2005], Barendregt et al. [1983], Abramsky and Ong [1993], Plotkin [1993], Engeler [1981], Coppo et al. [1979], Honsell and Lenisa [1999], Coppo et al. [1981], Coppo and Dezani-Ciancaglini [1980]. These theories are denoted by names (respectively acronymes) of the author(s) who have first considered the λ-model induced by such a theory.

The expressive power of intersection types is remarkable. This will become apparent when we will use them as a tool for characterizing properties of λ-terms (see Sections 19.2 and 18.3), and for describing different λ-models (see Section 18). Much of this expressive power comes from the fact that they are endowed with a *preorder relation*, $\leq$, which induces, on the set of types modulo $=$, the structure of a meet semi-lattice with respect to $\cap$. This appears natural when we think of types as subsets of a domain of discourse D, which is endowed with a (partial) application $\cdot : D\times D\to D$, and interpret $\cap$ as set-theoretic intersection,

$\leq$ as set inclusion, and give $\rightarrow$ the *realizability interpretation*.

$$\begin{array}{rcl} [\![A]\!] & \subseteq & D \\ A \leq B & \Leftrightarrow & [\![A]\!] \subseteq [\![B]\!] \\ [\![A \cap B]\!] & = & [\![A]\!] \cap [\![B]\!] \\ [\![A{\rightarrow}B]\!] & = & [\![A]\!]{\rightarrow}[\![B]\!] = \{d \in D \mid d \cdot [\![A]\!] \subseteq [\![B]\!]\}. \end{array}$$

This semantics, due to Scott, will be studied in Section 19.1.

The type $\top{\rightarrow}\top$ is the set of functions which applied to an arbitrary element return again an arbitrary element. In that case axiom scheme $(\top{\rightarrow})$ expresses the fact that all the objects in our domain of discourse are total functions, i.e. that $\top$ is equal to $A{\rightarrow}\top$, hence $A{\rightarrow}\top = B{\rightarrow}\top$ for all A, B (Barendregt et al. [1983]). If now we want to capture only those terms which truly represent functions, as we do for example in the lazy λ-calculus, we cannot assume axiom $(\top{\rightarrow})$. One still may postulate the weaker property $(\top_{\text{lazy}})$ to make all functions total (Abramsky and Ong [1993]). It simply says that an element which is a function, because it maps A into B, maps also the whole universe into itself.

In Figure 15.3 below consider $\vdash^{\mathcal{T}}_{\cap^\top}$ for the ten type theories above the horizontal line and $\vdash^{\mathcal{T}}_{\cap}$ for the other three. Define $\mathcal{T}_1 \subseteq \mathcal{T}_2$ as

$$\forall \Gamma, M, A.\ [\Gamma \vdash^{\mathcal{T}_1} M : A \ \Rightarrow\ \Gamma \vdash^{\mathcal{T}_2} M : A].$$

If this is the case we have connected $\mathcal{T}_1$ with an edge towards the higher positioned $\mathcal{T}_2$. In Exercise 16.3.21 we will show that the edges denote strict inclusions.

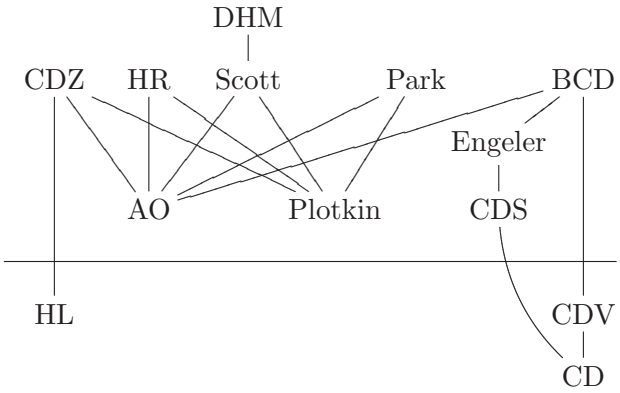


Figure 15.3: Inclusion among some intersection type theories.

The intended interpretation of arrow types also motivates axiom $(\rightarrow\cap)$, which implies that if a function maps A into B, and the same function maps also A into C, then, actually, it maps the whole A into the intersection between B and C (i.e. into $B \cap C$), see Barendregt et al. [1983].

Rule $(\rightarrow)$ is again very natural in view of the set-theoretic interpretation. It implies that the arrow constructor is contravariant in the first argument and covariant in the second one. It is clear that if a function maps A into B, and

we take a subset A' of A and a superset B' of B, then this function will map also A' into B', see Barendregt et al. [1983].

The rule $(\to\cap^=)$ is similar to the rule $(\to\cap)$. It capture properties of the graph models for the untyped lambda calculus, see Plotkin [1975] and Engeler [1981], as we shall discuss in Section 19.3.

In order to capture aspects of the λI-calculus we introduce TTs without an explicit mention of a top.

The remaining axioms express peculiar properties of D_∞-like inverse limit models, see Barendregt et al. [1983], Coppo et al. [1984], Coppo et al. [1987], Honsell and Ronchi Della Rocca [1992], Honsell and Lenisa [1993], Alessi, Dezani-Ciancaglini and Honsell [2004]. We shall discuss them in more detail in Section 19.3.

Some classes of type theories

Now we will consider some classes of TT. In order to do this, we list the relevant defining properties.

15.1.9. Definition. We define special subclasses of TT.

Class	Defining axiom(-scheme)(s) or rule
graph	$(\to^=), (\to\cap^=), (\top)$
lazy	$(\to), (\to\cap), (\top), (\top_{\text{lazy}})$
natural	$(\to), (\to\cap), (\top), (\top\to)$
proper	$(\to), (\to\cap)$

15.1.10. Notation. The sets of graph, lazy, natural and proper type theories are denoted by respectively $\text{GTT}^\top$, $\text{LTT}^\top$ $\text{NTT}^\top$ and PTT.

15.1.11. Remark. The type theories of Figure 15.2 are classified as follows.

non compatible	CD, CDS
$\text{GTT}^\top$	Plotkin, Engeler
$\text{LTT}^\top$	AO
$\text{NTT}^\top$	Scott, Park, CDZ, HR, DHM, BCD
PTT	CDV, HL

15.1.12. Remark. One has $\text{NTT}^\top \subseteq \text{LTT}^\top \subseteq \text{GTT}^\top \subseteq \text{TT}$ and $\text{LTT}^\top \subseteq \text{PTT} \subseteq \text{TT}$. These inclusions are sharp.

Some properties about specific TTs

Results about proper type theories

15.1.13. PROPOSITION. *Let $\mathcal{T}$ be a proper type theory. Then we have*

(i) $(A{\to}B)\cap(A'{\to}B')\leq(A\cap A'){\to}(B\cap B')$;

(ii) $(A_1{\to}B_1)\cap\ldots\cap(A_n{\to}B_n)\leq(A_1\cap\ldots\cap A_n){\to}(B_1\cap\ldots\cap B_n)$;

(iii) $(A{\to}B_1)\cap\ldots\cap(A{\to}B_n)=A{\to}(B_1\cap\ldots B_n)$.

PROOF. (i) $\begin{array}{rcl}(A{\to}B)\cap(A'{\to}B') &\leq& ((A\cap A'){\to}B)\cap((A\cap A'){\to}B')\\ &\leq& (A\cap A'){\to}(B\cap B'),\end{array}$
by respectively $(\to)$ and $(\to\cap)$.

(ii) Similarly (i.e. by induction on $n{>}1$, using (i) for the induction step).

(iii) By (ii) one has $(A{\to}B_1)\cap\ldots\cap(A{\to}B_n)\leq A{\to}(B_1\cap\ldots B_n)$. For $\geq$ use $(\to)$ to show that $A{\to}(B_1\cap\ldots\cap B_n)\leq(A{\to}B_i)$, for all i. ■

It follows that the mentioned equality and inequalities hold for Scott, Park, CDZ, HR, DHM, BCD, AO, HL and CDV.

Results about the type theories of Figure 15.2

15.1.14. LEMMA. (i) *φ is the top and ω the bottom element in* HL.

(ii) CDV *has no top element.*

(iii) CD *has no top element.*

PROOF. (i) By induction on the generation of $\mathbb{T}^{\mathrm{HL}}$ one shows that $\omega\leq A\leq\varphi$ for all $A\in\mathbb{T}^{\mathrm{HL}}$.

(ii) If α is a fixed atom and

$$\mathcal{B}_\alpha := \alpha \mid \mathcal{B}_\alpha\cap\mathcal{B}_\alpha$$

and $A\in\mathcal{B}_\alpha$, then one can show by induction on the generation of $\leq_{\mathrm{CDV}}$ that $A\leq_{\mathrm{CDV}}B\ \Rightarrow\ A\in\mathcal{B}_\alpha$. Hence if $\alpha\leq_{\mathrm{CDV}}B$, then $B\in\mathcal{B}_\alpha$. Since $\mathcal{B}_{\alpha_1}$ and $\mathcal{B}_{\alpha_2}$ are disjoint when α_1 and α_2 are two different atoms, we conclude that CDV has no top element.

(iii) Similar to (ii). ■

15.1.15. REMARK. By the above lemma φ turns out to be the top element in HL. But we will not use this and therefore denote it by the name φ and not $\top$.

In the following lemmas 15.1.16-15.1.20 we study the positions of the atoms ω, and φ in the compatible TTs introduced in Figure 15.2. The principal result is that $\omega<\varphi$ in HL and, as far as applicable,

$$\omega<\varphi<\top,$$

in the theories Scott, Park, CDZ, HR, DHM and Plotkin.

15.1.16. LEMMA. *Let* $\mathcal{T} \in \{\text{Scott}, \text{Park}, \text{CDZ}, \text{HR}, \text{DHM}, \text{BCD}, \text{Engeler}\}$ *be as defined in Figure 15.2. Define inductively the following collection of types*

$$\boxed{\mathcal{B} \;:=\; \top \mid \mathbb{T}^{\mathcal{T}} \to \mathcal{B} \mid \mathcal{B} \cap \mathcal{B}}$$

Then $\mathcal{B} = \{A \in \mathbb{T}^{\mathcal{T}} \mid A =_{\mathcal{T}} \top\}$.

PROOF. By induction on the generation of $A \leq_{\mathcal{T}} B$ one proves that $\mathcal{B}$ is closed upwards. This gives $\top \leq A \;\Rightarrow\; A \in \mathcal{B}$.

By induction on the definition of $\mathcal{B}$ one shows, using $(\top\to)$ and $(\to)$, that $A \in \mathcal{B} \;\Rightarrow\; \top \leq A$.

Therefore

$$A =_{\mathcal{T}} \top \;\Leftrightarrow\; \top \leq A \;\Leftrightarrow\; A \in \mathcal{B}. \blacksquare$$

15.1.17. LEMMA. *For* $\mathcal{T} \in \{\text{AO}, \text{Plotkin}\}$ *define inductively*

$$\boxed{\mathcal{B} \;:=\; \top \mid \mathcal{B} \cap \mathcal{B}}$$

Then $\mathcal{B} = \{A \in \mathbb{T}^{\mathcal{T}} \mid A =_{\mathcal{T}} \top\}$, *hence* $\top\to\top \neq_{\mathcal{T}} \top$.

PROOF. Similar to the proof of 15.1.14, but easier. ■

15.1.18. LEMMA. *For* $\mathcal{T} \in \{\text{CDZ}, \text{HR}, \text{DHM}\}$ *define by mutual induction*

$$\begin{aligned} \mathcal{B} &= \varphi \mid \top \mid \mathbb{T}^{\mathcal{T}} \to \mathcal{B} \mid \mathcal{H} \to \mathbb{T}^{\mathcal{T}} \mid \mathcal{B} \cap \mathcal{B} \\ \mathcal{H} &= \omega \mid \mathcal{B} \to \mathcal{H} \mid \mathcal{H} \cap \mathbb{T}^{\mathcal{T}} \mid \mathbb{T}^{\mathcal{T}} \cap \mathcal{H} \end{aligned}$$

Then

$$\begin{aligned} \varphi \leq B &\;\Rightarrow\; B \in \mathcal{B}, \\ A \leq \omega &\;\Rightarrow\; A \in \mathcal{H}. \end{aligned}$$

PROOF. By induction on $\leq_{\mathcal{T}}$ one shows

$$A \leq B \;\Rightarrow\; (A \in \mathcal{B} \;\Rightarrow\; B \in \mathcal{B}) \;\Rightarrow\; (B \in \mathcal{H} \;\Rightarrow\; A \in \mathcal{H}).$$

From this the assertion follows immediately. ■

15.1.19. LEMMA. *We work with the theory* HL.

(i) *Define by mutual induction*

$$\begin{aligned} \mathcal{B} &= \varphi \mid \mathcal{H} \to \mathcal{B} \mid \mathcal{B} \cap \mathcal{B} \\ \mathcal{H} &= \omega \mid \mathcal{B} \to \mathcal{H} \mid \mathcal{H} \cap \mathbb{T} \mid \mathbb{T} \cap \mathcal{H} \end{aligned}$$

Then

$$\begin{aligned} \mathcal{B} &= \{A \in \mathbb{T}^{\text{HL}} \mid A =_{\text{HL}} \varphi\}; \\ \mathcal{H} &= \{A \in \mathbb{T}^{\text{HL}} \mid A =_{\text{HL}} \omega\}. \end{aligned}$$

(ii) $\omega \neq_{\text{HL}} \varphi$ *and hence* $\omega <_{\text{HL}} \varphi$.

PROOF. (i) By induction on $\leq_{\mathcal{T}}$ one shows

$$A \leq B \;\Rightarrow\; (A \in \mathcal{B} \;\Rightarrow\; B \in \mathcal{B}) \;\&\; (B \in \mathcal{H} \;\Rightarrow\; A \in \mathcal{H}).$$

This gives

$$(\varphi \leq B \;\Rightarrow\; B \in \mathcal{B}) \;\&\; (A \leq \omega \;\Rightarrow\; A \in \mathcal{H}).$$

By simultaneous induction on the generation of $\mathcal{B}$ and $\mathcal{H}$ one shows, using that ω is the bottom element of HL, by Lemma 15.1.14(i),

$$(B \in \mathcal{B} \;\Rightarrow\; B = \varphi) \;\&\; (A \in \mathcal{H} \;\Rightarrow\; A = \omega).$$

Now the assertion follows immediately.

(ii) By (i). ■

15.1.20. PROPOSITION. *In* HL *we have* $\omega < \varphi$ *and as far as applicable we have for the other systems of Figure 15.2*

$$\omega < \varphi < \top.$$

More precisely,

(i) $\omega \leq \varphi$ *and* $\omega \neq \varphi$ *in* HL.

In all other systems

(ii) $\omega \leq \varphi$, $\omega \leq \top$, $\varphi \leq \top$;

(iii) $\omega \neq \varphi$, $\omega \neq \top$, $\varphi \neq \top$.

PROOF. (i) By $(\omega\varphi)$ and Lemma 15.1.19.

(ii) By $(\omega\varphi)$ and $(\top)$.

(iii) By Lemmas 15.1.16-15.1.18. ■

15.2. Type assignment

Assignment of types from type theories

In this subsection we define for a $\mathcal{T}$ in TT a type assignment system $\lambda^{\mathcal{T}}_{\cap}$, that assigns to untyped lambda terms a (possibly empty set of) types in $\mathbb{T}^{\mathcal{T}}$. For a $\mathcal{T}$ in TT$^\top$ we also define a type assignment system $\lambda^{\mathcal{T}}_{\cap^\top}$.

15.2.1. DEFINITION. (i) A $\mathcal{T}$-*statement* is of the form $M : A$ with the *subject* an untyped lambda term $M \in \Lambda$ and the *predicate* a type $A \in \mathbb{T}^{\mathcal{T}}$.

(ii) A $\mathcal{T}$-*declaration* is a $\mathcal{T}$-statement of the form $x : A$.

(iii) A $\mathcal{T}$-*basis* Γ is a finite set of $\mathcal{T}$-declarations, with all variables distinct.

(iv) A $\mathcal{T}$-*assertion* is of the form $\Gamma \vdash M : A$, where $M : A$ is a $\mathcal{T}$-statement and Γ is a $\mathcal{T}$-basis.

15.2.2. DEFINITION. (i) The *(basic) type assignment system* $\lambda^{\mathcal{T}}_{\cap}$ derives $\mathcal{T}$-assertions by the following axioms and rules.

$$
\begin{array}{lll}
\text{(Ax)} & \Gamma \vdash x{:}A & \text{if } (x{:}A \in \Gamma) \\
(\to\text{I}) & \dfrac{\Gamma, x{:}A \vdash M : B}{\Gamma \vdash \lambda x.M : A \to B} & \\
(\to\text{E}) & \dfrac{\Gamma \vdash M : A \to B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B} & \\
(\cap\text{I}) & \dfrac{\Gamma \vdash M : A \quad \Gamma \vdash M : B}{\Gamma \vdash M : A \cap B} & \\
(\leq) & \dfrac{\Gamma \vdash M : A \quad A \leq_{\mathcal{T}} B}{\Gamma \vdash M : B} &
\end{array}
$$

Figure 15.4: Basic type assignment system $\lambda^{\mathcal{T}}_{\cap}$.

(ii) If $\mathcal{T}$ has a top element $\top$, then the *$\top$-type assignment system* $\lambda^{\mathcal{T}}_{\cap^\top}$ is defined by adding the extra axiom to the basic system

$$\boxed{(\top\text{-universal}) \quad \Gamma \vdash M : \top}$$

Figure 15.5: The extra axiom for the top assignment system $\lambda^{\mathcal{T}}_{\cap^\top}$

15.2.3. Notation. (i) We write $\Gamma \vdash^{\mathcal{T}}_{\cap^\top} M : A$ or $\Gamma \vdash^{\mathcal{T}}_{\cap} M : A$ if $\Gamma \vdash M : A$ is derivable in $\lambda^{\mathcal{T}}_{\cap^\top}$ or $\lambda^{\mathcal{T}}_{\cap}$ respectively.

(ii) The assertion $\vdash^{\mathcal{T}}_{\cap^\top}$ may also be written as $\vdash^{\mathcal{T}}$, $\vdash_{\cap^\top}$ or simply $\vdash$ if by the context there is little danger of confusion. Similarly, $\vdash^{\mathcal{T}}_{\cap}$ may be written as $\vdash^{\mathcal{T}}$, $\vdash_{\cap}$ or $\vdash$.

(iii) $\lambda^{\mathcal{T}}_{\cap(^\top)}$ may be denoted by $\lambda_{\cap(^\top)}$.

15.2.4. Example. Let $\mathcal{T} \in \mathrm{TT}^{\top}$ with $A, B \in \Pi^{\mathcal{T}}$. Write $W \equiv (\lambda x.xx)$.

(i) $\vdash^{\mathcal{T}}_{\cap} W : A \cap (A \to B) \to B$.

$\vdash^{\mathcal{T}}_{\cap^\top} WW : \top$, but WW does not have a type in $\lambda^{\mathcal{T}}_{\cap}$.

(ii) Let $M \equiv \mathsf{KI}(WW)$. Then $\vdash M : (A \to A)$ in $\lambda^{\mathcal{T}}_{\cap^\top}$.

(iii) (van Bakel) Let $M \equiv \lambda yz.\mathsf{K}z(yz)$ and $N \equiv \lambda yz.z$. Then $M \twoheadrightarrow_{\beta} N$. We have $\vdash^{\mathcal{T}}_{\cap} N : B \to A \to A$, $\vdash^{\mathcal{T}}_{\cap^\top} M : B \to A \to A$, but $\not\vdash^{\mathcal{T}}_{\cap} M : B \to A \to A$.

(iv) $\not\vdash^{\mathrm{CD}}_{\cap} \mathsf{I} : ((A \cap B) \to C) \to ((B \cap A) \to C)$.

In general the type assignment systems $\lambda^{\mathcal{T}}_{\cap^\top}$ will be used for the the $\lambda\mathsf{K}$-calculus and $\lambda^{\mathcal{T}}_{\cap}$ for the $\lambda\mathsf{I}$-calculus.

15.2.5. Definition. Define the rules ($\cap$E)

$$\frac{\Gamma \vdash M : (A \cap B)}{\Gamma \vdash M : A} \qquad \frac{\Gamma \vdash M : (A \cap B)}{\Gamma \vdash M : B}$$

Notice that these rules are derived in $\lambda^{\mathcal{T}}_{\cap}$, $\lambda^{\mathcal{T}}_{\cap^\top}$ for all $\mathcal{T}$.

15.2.6. Lemma. *In $\lambda^{\mathcal{T}}_{\cap}$ one has the following.*

(i) $\Gamma \vdash M : A \;\Rightarrow\; \mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$.
(ii) $\Gamma \vdash M : A \;\Rightarrow\; (\Gamma \restriction \mathrm{FV}(M)) \vdash M : A$.
(iii) *If in $\mathcal{T}$ with top $\top$ one has $\top = \top\!\to\!\top$, then*

$$\mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma) \;\Rightarrow\; \Gamma \vdash M : \top.$$

PROOF. (i), (ii) By induction on the derivation.
(iii) By induction on M. ■

Notice that $\Gamma \vdash M : A \Rightarrow \mathrm{FV}(M) \subseteq \mathrm{dom}(\Gamma)$ does not hold in $\lambda^{\mathcal{T}}_{\cap^\top}$, since by axiom ($\top$ universal) we have $\vdash^{\mathcal{T}} M : \top$ for all $\mathcal{T}$ and all M.

15.2.7. REMARK. For the type theories of Figure 15.2 with $\top$ we have defined the type assignment systems $\lambda^{\mathcal{T}}_{\cap}$. For those system having a top, there is also the type assignment system $\lambda^{\mathcal{T}}_{\cap^\top}$. We will use for the type theories in Figure 15.2 only one of the two possibilities. For the first ten systems, i.e. Scott, Park, CDZ, HR, DHM, BCD, AO, Plotkin, Engeler and CDS, we only consider $\lambda^{\mathcal{T}}_{\cap^\top}$. For the other 3 systems, i.e. HL, CDV and CD, we will only consider $\lambda^{\mathcal{T}}_{\cap}$. In fact by Lemma 15.1.14(ii) and (iii) we know that CDV and CD have no top at all. The system HL has a top, but we will not use it, as we do not know interesting properties of $\lambda^{\mathrm{HL}}_{\cap^\top}$. So, for example, $\vdash^{\mathrm{Scott}}$ will be always $\vdash^{\mathrm{Scott}}_{\cap^\top}$, whereas $\vdash^{\mathrm{HL}}$ will be always $\vdash^{\mathrm{HL}}_{\cap}$. The reader will be reminded of this. We do not know wether there exist TTs where the interplay of $\lambda^{\mathcal{T}}_{\cap}$ and $\lambda^{\mathcal{T}}_{\cap^\top}$ yields results of interest.

Admissible rules

15.2.8. PROPOSITION. *The following rules are admissible in $\lambda^{\mathcal{T}}_{\cap(^\top)}$.*

(*weakening*)	$\dfrac{\Gamma \vdash M : A \quad x \notin \Gamma}{\Gamma, x{:}B \vdash M : A}$;
(*strengthening*)	$\dfrac{\Gamma, x{:}B \vdash M : A \quad x \notin FV(M)}{\Gamma \vdash M : A}$;
(*cut*)	$\dfrac{\Gamma, x{:}B \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash (M[x := N]) : A}$;
($\leq$-L)	$\dfrac{\Gamma, x{:}B \vdash M : A \quad C \leq_{\mathcal{T}} B}{\Gamma, x{:}C \vdash M : A}$;
($\to$L)	$\dfrac{\Gamma, y{:}B \vdash M : A \quad \Gamma \vdash N : C \quad x \notin \Gamma}{\Gamma, x{:}(C\!\to\!B) \vdash (M[y := xN]) : A}$;
($\cap$L)	$\dfrac{\Gamma, x{:}A \vdash M : B}{\Gamma, x{:}(A \cap C) \vdash M : B}$.

Figure 15.6: Various admissible rules.

PROOF. By induction on the structure of derivations. ■

Proofs later on in Part III will freely use the rules of the above proposition.

As we remarked earlier, there are various equivalent alternative presentations of intersection type assignment systems. We have chosen a natural deduction presentation, where $\mathcal{T}$-bases are additive. We could have taken, just as well, a sequent style presentation and replace rule ($\to$E) with the three rules ($\to$L), ($\cap$L) and (cut) occuring in Proposition 15.2.8, see Barbanera et al. [1995], Barendregt and Ghilezan [n.d.]. Next to this we could have formulated the rules so that $\mathcal{T}$-bases "multiply". Notice that because of the presence of the type constructor $\cap$, a special notion of *multiplication of $\mathcal{T}$-bases* can be given.

15.2.9. Definition (Multiplication of $\mathcal{T}$-bases).

$$\begin{array}{rcl} \Gamma \uplus \Gamma' & = & \{x{:}\, A \cap B \mid x{:}\, A \in \Gamma \text{ and } x{:}\, B \in \Gamma'\} \\ & \cup & \{x{:}\, A \mid x{:}\, A \in \Gamma \text{ and } x \notin \Gamma'\} \\ & \cup & \{x{:}\, B \mid x{:}\, B \in \Gamma' \text{ and } x \notin \Gamma\}. \ \blacksquare \end{array}$$

Accordingly we define:

$$\Gamma \Subset \Gamma' \ \Leftrightarrow \ \exists \Gamma''.\, \Gamma \uplus \Gamma'' = \Gamma'.$$

For example, $\{x{:}A, y{:}B\} \uplus \{x{:}C, z{:}D\} = \{x{:}A \cap C, y{:}B, z{:}D\}$.

15.2.10. Proposition. *The following rules are admissible in all $\lambda^{\mathcal{T}}_{\cap(\top)}$.*

$$\begin{array}{ll} (\textit{multiple weakening}) & \dfrac{\Gamma_1 \vdash M : A}{\Gamma_1 \uplus \Gamma_2 \vdash M : A} \\[2ex] (\textit{relevant} \to \text{E}) & \dfrac{\Gamma_1 \vdash M : A \to B \quad \Gamma_2 \vdash N : A}{\Gamma_1 \uplus \Gamma_2 \vdash MN : B} \\[2ex] (\textit{relevant} \cap \text{I}) & \dfrac{\Gamma_1 \vdash M : A \quad \Gamma_2 \vdash M : B}{\Gamma_1 \uplus \Gamma_2 \vdash M : A \cap B} \end{array}$$

Proof. By induction on derivations. ■

In Exercise 16.3.17, it will be shown that we can replace rule ($\leq$) with other more perspicuous rules. This is possible as soon as we will have proved appropriate "inversion" theorems for $\lambda^{\mathcal{T}}_{\cap(\top)}$. For some very special theories, one can even omit altogether rule ($\leq$), provided the remaining rules are reformulated "multiplicatively" with respect to $\mathcal{T}$-bases, see e.g. Di Gianantonio and Honsell [1993]. We shall not follow up this line of investigation.

In $\lambda^{\mathcal{T}}_{\cap(\top)}$, assumptions are allowed to appear in the basis without any restriction. Alternatively, we might introduce a *relevant* intersection type assignment system, where only "minimal-base" judgements are derivable, (see Honsell and Ronchi Della Rocca [1992]). Rules like (*relevant* $\to$E) and (*relevant* $\cap$ I), which exploit the above notion of multiplication of bases, are essential for this purpose. Relevant systems are necessary, for example, for giving finitary logical descriptions of qualitative domains as defined in Girard et al. [1989]. We will not follow up this line of research either. See Honsell and Ronchi Della Rocca [1992].

Special type assignment for call-by-value λ-calculus

We will study later the type theory EHR with $\mathbb{A}^{\mathrm{EHR}} = \{\nu\}$ and the extra rule $(\to)$ and axioms $(\to\cap)$ and

$$A \to B \leq \nu.$$

The type assignment system $\lambda^{\mathrm{EHR}}_{\cap\nu}$ is defined by the axiom and rules of $\lambda^{\mathcal{T}}_{\cap}$ in Figure 15.4 with the extra axiom

$$\boxed{(\nu \text{ universal}) \qquad \Gamma \vdash (\lambda x.M) : \nu.}$$

The type theory EHR has a top, namely ν, so one could consider it as an element of $\mathrm{TT}^\top$. This will not be done. Axiom (ν-universal) is different from ($\top$-universal) in Definition 15.2.2. This type assignment system has one particular application and will be studied in some exercises.

15.3. Type structures

Intersection type structures

Remember that a type algebra $\mathcal{A}$, see Definition **??**, is of the form $\mathcal{A} = \langle |\mathcal{A}|, \to \rangle$, i.e. just an arbitrary set $|\mathcal{A}|$ with a binary operation $\to$ on it.

15.3.1. Definition. (i) A *meet semi-lattice* is a structure

$$\mathcal{M} = \langle |\mathcal{M}|, \leq, \cap \rangle,$$

such that $\mathcal{M} = \langle |\mathcal{M}|, \leq, \cap \rangle$ is a partial order, for all $A, B \in |\mathcal{M}|$ the element $A \cap B$ (meet) is the greatest lower bound of A and B. MSL is the set of meet semi-lattices.

(ii) A *top meet semi-lattice* is a similar structure

$$\mathcal{M} = \langle |\mathcal{M}|, \leq, \cap, \top \rangle,$$

such that $\mathcal{M} = \langle |\mathcal{M}|, \leq, \cap \rangle$ is a MSL and $\top$ is the (unique) top of $\mathcal{M}$. $\mathrm{MSL}^\top$ is the set of top meet semi-lattices.

15.3.2. Definition. (i) An *(intersection) type structure* is a type algebra with the additional structure of a meet semi-lattice

$$\mathcal{S} = \langle |\mathcal{S}|, \to, \leq, \cap \rangle.$$

TS is the set of type structures. The relation $\leq$ and the operation $\to$ have a priori no relation with each other, but in special structures this will be the case.

(ii) A *top type structure* is a type algebra that is also a top meet semi-lattice

$$\mathcal{S} = \langle |\mathcal{S}|, \to, \leq, \cap, \top \rangle.$$

$\mathrm{TS}^\top$ is the set of top type structures.

NOTATION. (i) As 'intersection' is everywhere in this Part III, we will omit this word and only speak about a *type structure*.

(ii) *Par abus de language* we also use $A, B, C, \ldots$ to denote arbitrary elements of type structures and we write $A \in \mathcal{S}$ for $A \in |\mathcal{S}|$.

If $\mathcal{T}$ is a type theory that is not compatible, like CD and CDS, then $\to$ cannot be defined on the equivalence classes. But if $\mathcal{T}$ is compatible, then one can work on the equivalence classes and obtain a type structure in which $\leq$ is a partial order.

15.3.3. PROPOSITION. *Let $\mathcal{T}$ be a compatible type theory. Then $\mathcal{T}$ induces a type structure $\mathcal{T}/=$ defined as follows.*

$$\langle \mathbb{T}^{\mathcal{T}}/=_{\mathcal{T}}, \to, \leq, \cap \rangle,$$

by defining on the $=_{\mathcal{T}}$-equivalence classes

$$\begin{array}{rcl} [A] \to [B] & = & [A \to B]^2; \\ [A] \cap [B] & = & [A \cap B]; \\ [A] \leq [B] & \Leftrightarrow & A \leq_{\mathcal{T}} B. \end{array}$$

If moreover $\mathcal{T}$ has a top $\top$, then $\mathcal{T}/=$ is a top type structure with $[\top]$ as top.

PROOF. Here A, B, C range over $\mathbb{T}^{\mathcal{T}}$. Having realized this the rest is easy. Rule $(\to^{=})$ is needed to ensure that $\to$ is well-defined. ■

The (top) type structure $\mathcal{T}/=$, with $\mathcal{T}$ a type theory, is called a *syntactical* (top) type structure. In Proposition 15.3.6 we show that every type structure is isomorphic to a syntactical one.

Although essentially equivalent, type structures and type theories differ in the following. In the theories the types are freely generated from a fixed set of atoms and inequality can be controlled somewhat by choosing the right axioms and rules (this will be exploited in Section 19.3). In type structures one has the antisymmetric law $A \leq B \leq A \;\Rightarrow\; A = B$, which is in line with the common theory of partial orders (this will be exploited in Chapter 17).

Now the notion of type assignment will also be defined for intersection type structures. These structures arise naturally coming from algebraic lattices that are used towards obtaining a semantics for untyped lambda calculus.

15.3.4. DEFINITION. (i) Now let $\mathcal{S} \in \mathrm{TS}$. The notion of a $\mathcal{S}$-statement $M : A$, a $\mathcal{S}$-declaration $x : A$, a $\mathcal{S}$-basis and a $\mathcal{S}$-assertion $\Gamma \vdash M : A$ is as in Definition 15.2.1, now for $A \in \mathcal{S}$ an element of the type structure $\mathcal{S}$.

(ii) The notion $\Gamma \vdash^{\mathcal{S}}_{\cap} M : A$ is defined by the same set of axioms and rules as in Figure 15.4 where now $\leq_{\mathcal{S}}$ is the inequality of the structure $\mathcal{S}$. The assignment system $\lambda^{\mathcal{S}}_{\cap\top}$ has ($\top$-universal) as extra axiom.

[2] Here we misuse notation in a suggestive way, by using the same notation $\to$ for equivalence classes as for types.

The following result shows that for syntactic type structures type assignment is essentially the same as the one coming from the corresponding lambda theory.

15.3.5. PROPOSITION. *Let $\mathcal{T} \in TT^{(\top)}$ and let $[\mathcal{T}] = \langle \mathcal{T}/=_{\mathcal{T}}, \leq, \cap, \rightarrow(, \top)\rangle$ its corresponding (top) type structure. For a type $A \in \mathcal{T}$ write its equivalence class as $[A] \in [\mathcal{T}]$. For $\Gamma = \{x_1 : B_1, \ldots, x_n : B_n\}$ a $\mathcal{T}$-basis write $[\Gamma] = \{x_1 : [B_1], \ldots, x_n : [B_n]\}$, a $[\mathcal{T}]$-basis. Then*

$$\Gamma \vdash^{\mathcal{T}}_{\cap^{(\top)}} M : A \;\Leftrightarrow\; [\Gamma] \vdash^{[\mathcal{T}]}_{\cap^{(\top)}} M : [A].$$

PROOF. ($\Rightarrow$) By induction on the derivation of $\Gamma \vdash^{\mathcal{T}} M : A$. ($\Leftarrow$) Show by induction on the derivation of $[\Gamma] \vdash^{[\mathcal{T}]} M : [A]$ that for all $A' \in [A]$ and $\Gamma' = \{x_1 : B'_1, \ldots, x_n : B'_n\}$, with $B'_i \in [B_i]$ for all $1 \leq i \leq n$, one has

$$\Gamma' \vdash^{\mathcal{T}} M : A'. \blacksquare$$

Using this result we could have defined type assignment first for type structures and then for compatible type theories via translation to the type assignment for its corresponding syntactical type structure, essentially by turning the previous result into a definition.

15.3.6. PROPOSITION. *Every type structure is isomorphic to a syntactical one.*

PROOF. For a type structure $\mathcal{S}$, define $\mathcal{T}_{\mathcal{S}}$ as follows. Take $\mathbb{A} = \{\underline{c} \mid c \in \mathcal{S}\}$. Define $\leq_{\mathcal{T}_{\mathcal{S}}}$ on $\mathbb{T} = \mathbb{T}^{\mathbb{A}}_{\cap}$ as follows. We make every element of $\mathbb{T}$ equal to an element of $\mathbb{A}$ by requiring

$$(\underline{a} \cap \underline{b}) =_{\mathcal{T}_{\mathcal{S}}} \underline{a \cap b}, \;\&\; (\underline{a} \rightarrow \underline{b}) =_{\mathcal{T}_{\mathcal{S}}} \underline{a \rightarrow b}.$$

This means of course $(\underline{a} \cap \underline{b}) \leq_{\mathcal{T}_{\mathcal{S}}} \underline{a \cap b}$, $(\underline{a} \cap \underline{b}) \geq_{\mathcal{T}_{\mathcal{S}}} \underline{a \cap b}$, etcetera. We moreover require

$$\frac{a \leq_{\mathcal{S}} b}{\underline{a} \leq_{\mathcal{T}_{\mathcal{S}}} \underline{b}}.$$

As a consequence $\underline{a} \leq_{\mathcal{T}_{\mathcal{S}}} \underline{\top}$ if $\mathcal{S}$ is a top type structure. The axioms and rules (refl), (trans), ($\rightarrow^{=}$), (incl_L), (incl_R) and (glb) also hold automatically. Then $\mathcal{S} \cong \mathcal{T}_{\mathcal{S}}/=$. This can be seen as follows. Define $f : \mathcal{S} \rightarrow \mathcal{T}_{\mathcal{S}}/=$ by $f(a) = [\underline{a}]$. For the inverse, first define $g : \mathbb{T}^{\mathbb{A}}_{\cap} \rightarrow \mathcal{S}$ by

$$\begin{aligned} g(\underline{c}) &= c; \\ g(A \rightarrow B) &= g(A) \rightarrow g(B); \\ g(A \cap B) &= g(A) \cap g(B). \end{aligned}$$

Then show $A \leq_{\mathcal{T}_{\mathcal{S}}} B \;\Rightarrow\; g(A) \leq g(B)$. Finally set $f^{-1}([A]) = g(A)$, which is well defined. It is easy to show that f, f^{-1} constitute an isomorphism. $\blacksquare$

15.3.7. REMARK. Each of the eleven compatible type theories $\mathcal{T}$ in Figure 15.2 may be considered as the intersection type structure $\mathcal{T}/=$, also denoted as $\mathcal{T}$. For example Scott can be a name, a type theory or a type structure.

Categories of meet-semi lattices and type structures

For use in Chapter 17 we will introduce some categories related to given classes of type structures.

15.3.8. DEFINITION. (i) The category **MSL** has as objects at most countable meet semi-lattices and as morphisms maps $f : \mathcal{M} \to \mathcal{M}'$, preserving $\leq, \cap$:

$$\begin{array}{rcl} A \leq B & \Rightarrow & f(A) \leq' f(B); \\ f(A \cap B) & = & f(A) \cap' f(B). \end{array}$$

(ii) The category $\mathbf{MSL}^\top$ is as **MSL**, but based on top meet semi-lattices. So now also $f(\top) = \top'$ for morphisms.

The difference between **MSL** and $\mathbf{MSL}^\top$ is that, in the **MSL** case, the top element is either missing or not relevant (not preserved by morphisms).

15.3.9. DEFINITION. (i) The category **TS** has as objects the at most countable type structures and as morphisms maps $f : \mathcal{S} \to \mathcal{S}'$, preserving $\leq, \cap, \to$:

$$\begin{array}{rcl} A \leq B & \Rightarrow & f(A) \leq' f(B); \\ f(A \cap B) & = & f(A) \cap' f(B); \\ f(A \to B) & = & f(A) \to' f(B). \end{array}$$

(ii) The category $\mathbf{TS}^\top$ is as **TS**, but based on top type structures. Now also

$$f(\top) = \top'$$

for morphisms.

15.3.10. DEFINITION. We define four full subcategories of **TS** by specifying in each case the objects.

(i) $\mathbf{GTS}^\top$ with as objects the graph top type structures.
(ii) $\mathbf{LTS}^\top$ with as objects the lazy top type structures.
(iii) $\mathbf{NTS}^\top$ with as objects the natural top type structures.
(iv) **PTS** with as objects the proper type structures.

15.4. Filters

15.4.1. DEFINITION. (i) Let $\mathcal{T} \in \mathrm{TT}$ and $X \subseteq \mathbb{T}^{\mathcal{T}}$. Then X is a *filter* over $\mathcal{T}$ if the following hold.

(1) X is non-empty;
(2) $A \in X \ \& \ A \leq B \ \Rightarrow \ B \in X$;
(3) $A, B \in X \ \Rightarrow \ A \cap B \in X$.

(ii) Write $\mathcal{F}^{\mathcal{T}} = \{X \subseteq \mathcal{T} \mid X \text{ is a filter over } \mathcal{T}\}$.

We loosely say that filters are non-empty sets of types closed under $\leq$ and $\cap$.

15.4.2. DEFINITION. Let $\mathcal{T} \in \mathrm{TT}$.

(i) For $A \in \mathbb{T}^{\mathcal{T}}$ write $\uparrow A = \{B \in \mathbb{T}^{\mathcal{T}} \mid A \leq B\}$.

(ii) For a non-empty $X \subseteq \mathbb{T}^{\mathcal{T}}$ define $\uparrow X$ to be the least filter over $\mathbb{T}^{\mathcal{T}}$ containing X; it can be described explicitly by

$$\uparrow X = \{B \in \mathbb{T}^{\mathcal{T}} \mid \exists n \geq 1\, \exists A_1, \ldots, A_n \in X. A_1 \cap \ldots \cap A_n \leq B\}.$$

15.4.3. Remark. $C \in \uparrow \{B_i \mid i \in \mathcal{I} \neq \emptyset\} \Leftrightarrow \exists I \subseteq_{\text{fin}} \mathcal{I}.[I \neq \emptyset \,\&\, \bigcap_{i \in I} B_i \leq C]$.

15.4.4. Proposition. *Let $\mathcal{T} \in \mathrm{TT}^{\top}$.*

(i) *$\mathcal{F}^{\mathcal{T}} = \langle \mathcal{F}^{\mathcal{T}}, \subseteq \rangle$ is a complete lattice, with for $\mathcal{X} \subseteq \mathcal{F}^{\mathcal{T}}$ the sup is*

$$\begin{array}{lcll} \bigsqcup \mathcal{X} & = & \uparrow(\cup \mathcal{X}), & \text{if } X \neq \emptyset, \\ \bigsqcup \mathcal{X} & = & \{\top\}, & \text{else.} \end{array}$$

(ii) *For $A \in \mathbb{T}^{\mathcal{T}}$ one has $\uparrow A = \uparrow\{A\}$ and $\uparrow A \in \mathcal{F}^{\mathcal{T}}$.*

(iii) *For $A, B \in \mathbb{T}^{\mathcal{T}}$ one has $\uparrow A \sqcup \uparrow B = \uparrow(A \cap B)$.*

(iv) *For $X \in \mathcal{F}^{\mathcal{T}}$ one has*

$$\begin{array}{rcl} X & = & \bigsqcup\{\uparrow A \mid A \in X\} \\ & = & \bigsqcup\{\uparrow A \mid \uparrow A \subseteq X\} \\ & = & \bigcup\{\uparrow A \mid A \in X\} \\ & = & \bigcup\{\uparrow A \mid \uparrow A \subseteq X\}. \end{array}$$

(v) *$\{\uparrow A \mid A \in \mathbb{T}^{\mathcal{T}}\}$ is the set of finite elements of $\mathcal{F}^{\mathcal{T}}$.*

Proof. Easy. ■

15.4.5. Definition. Let $\mathcal{T} \in \mathrm{TT}$. Then $\mathcal{F}^{\mathcal{T}}_s = \mathcal{F}^{\mathcal{T}} \cup \{\emptyset\}$ is the extension of $\mathcal{F}^{\mathcal{T}}$ with the emptyset.

15.4.6. Proposition. *Let $\mathcal{T} \in \mathrm{TT}$.*

(i) *$\mathcal{F}^{\mathcal{T}}_s = \langle \mathcal{F}^{\mathcal{T}}_s, \subseteq \rangle$ is a complete lattice, with for $\mathcal{X} \subseteq \mathcal{F}^{\mathcal{S}}_s$ the sup is*

$$\bigsqcup \mathcal{X} = \begin{cases} \emptyset, & \text{if } \mathcal{X} = \emptyset \text{ or } \mathcal{X} = \{\emptyset\}, \\ \uparrow(\cup \mathcal{X}), & \text{else .} \end{cases}$$

(ii) *For $A \in \mathbb{T}^{\mathcal{T}}$ one has $\uparrow A = \uparrow\{A\}$ and $\uparrow A \in \mathcal{F}^{\mathcal{T}}_s$.*

(iii) *For $A, B \in \mathbb{T}^{\mathcal{T}}$ one has $\uparrow A \sqcup \uparrow B = \uparrow(A \cap B)$.*

(iv) *For $X \in \mathcal{F}^{\mathcal{T}}_s$ one has*

$$\begin{array}{rcl} X & = & \bigsqcup\{\uparrow A \mid A \in X\} \quad = \quad \bigsqcup\{\uparrow A \mid \uparrow A \subseteq X\} \\ & = & \bigcup\{\uparrow A \mid A \in X\} \quad = \quad \bigcup\{\uparrow A \mid \uparrow A \subseteq X\}. \end{array}$$

(v) *$\{\uparrow A \mid A \in \mathbb{T}^{\mathcal{T}}\} \cup \{\emptyset\}$ is the set of finite elements of $\mathcal{F}^{\mathcal{T}}_s$.*

Proof. Immediate. ■

15.4.7. REMARK. The items 15.1.9-15.2.10 and 15.4.1-15.4.6 are about type theories, but can be translated immediately to structures and if no $\to$ are involved to meet-semi lattices. For example Proposition 15.1.13 also holds for a proper type structure, hence it holds for Scott, Park, CDZ, HR, DHM, BCD, AO, HL and CDV considered as type structures. Also 15.1.14-15.1.20 immediately yield corresponding valid statements for the corresponding type structures, though the proof for the type theories cannot be translated to proofs for the type structures because they are by induction on the syntactic generation of $\mathbb{T}$ or $\leq$. Also 15.2.4-15.2.10 hold for type structures, as follows immediately from Propositions 15.3.5 and 15.3.6. Finally 15.4.1-15.4.6 can be translated immediately to type structures and meet semi-lattices. Therefore in the following chapters everywhere the type theories may be translated to type structures (or if no $\to$ is involved to meet semi-lattices). In Chapter 17 we work directly with meet semi-lattices and type structures and not with type theories, because there a partial order is needed.

15.5. Exercises 31.10.2006:581

15.5.1. Show that $\Gamma, x{:}\top \vdash^{\mathcal{T}}_{\cap^\top} M : A \;\Rightarrow\; \Gamma \vdash^{\mathcal{T}}_{\cap^\top} M : A$.

15.5.2. The system $\mathtt{K}$ and the type assignment system $\lambda^{\mathtt{K}}_{\cap}$ of Krivine [1990] are CD and $\lambda^{\mathrm{CD}}_{\cap}$, but with rule $(\leq)$ replaced by

$$(\cap\mathrm{E}) \quad \frac{\Gamma \vdash M : A \cap B}{\Gamma \vdash M : A} \qquad \frac{\Gamma \vdash M : A \cap B}{\Gamma \vdash M : B}$$

Similarly $\mathtt{K}^\top$ and $\lambda^{\mathtt{K}^\top}_{\cap^\top}$ are CDS and $\lambda^{\mathrm{CDS}}_{\cap^\top}$, with $(\leq)$ replaced by $(\cap\mathrm{E})$. Show that

(i) $\Gamma \vdash^{\mathtt{K}} M : A \quad\Leftrightarrow\quad \Gamma \vdash^{\mathrm{CD}}_{\cap} M : A$.

(ii) $\Gamma \vdash^{\mathtt{K}^\top} M : A \quad\Leftrightarrow\quad \Gamma \vdash^{\mathrm{CDS}}_{\cap^\top} M : A$.

15.5.3. (i) Show that $\lambda x.xxx$ and $(\lambda x.xx)\mathsf{I}$ are typable in system $\mathtt{K}$.
(ii) Show that all closed terms in normal forms are typable in system $\mathtt{K}$.

15.5.4. Show the following:

(i) $\vdash^{\mathtt{K}} \lambda z.\mathsf{KI}(zz) : (A{\to}B) \cap A{\to}C{\to}C$.

(ii) $\vdash^{\mathtt{K}^\top} \lambda z.\mathsf{KI}(zz) : \top{\to}C{\to}C$.

(iii) $\vdash^{\mathrm{BCD}}_{\cap^\top} \lambda z.\mathsf{KI}(zz) : \top{\to}(A{\to}B \cap C){\to}A{\to}B$.

15.5.5. For $\mathcal{T}$ a type theory, $M, N \in \Lambda$ and $x \notin \mathrm{dom}(\Gamma)$ show

(i) $\Gamma \vdash^{\mathcal{T}}_{\cap} M : A \quad\Rightarrow\quad \Gamma \vdash^{\mathcal{T}}_{\cap} M[x{:=}N] : A$;
(ii) $\Gamma \vdash^{\mathcal{T}}_{\cap^\top} M : A \quad\Rightarrow\quad \Gamma \vdash^{\mathcal{T}}_{\cap^\top} M[x{:=}N] : A$.

15.5.6. Show that

$$M \text{ is a closed term } \Rightarrow \vdash^{\mathrm{Park}}_{\cap^\top} M : \omega.$$

Later we will show the converse (Theorem 18.3.22).

15.5.7. Prove that for all types $A \in \mathbb{T}^{\mathrm{AO}}$ there is an n such that

$$\top^n \to \top \leq_{\mathrm{AO}} A.$$

15.5.8. Prove that if $(\omega\varphi), (\varphi \to \omega)$ and $(\omega \to \varphi)$ are axioms in $\mathcal{T}$, then for all M in normal form $\{x_1 : \omega, \ldots, x_n : \omega\} \vdash^{\mathcal{T}} M : \varphi$, where $\{x_1, \ldots, x_n\} \supseteq \mathrm{FV}(M)$.

15.5.9. Let $\mathcal{D} = \langle D, \cdot \rangle$ be an applicative structure, i.e. a set with an arbitrary binary operation on it. For $X, Y \subset D$ define

$$X \to Y = \{d \in D \mid \forall e \in X. d \cdot e \in Y\}.$$

Consider $(\mathcal{P}(D), \to, \subseteq, \cap, D)$, where $\mathcal{P}(D)$ is the power set of D, $\subseteq$ and $\cap$ are the usual set theoretic notions and D is the top of $\mathcal{P}(D)$. Show

- $(\mathcal{P}(\mathcal{D}), \to, \subseteq, \cap)$ is a proper type structure.
- $\mathcal{D} = \mathcal{D} \to \mathcal{D}$.
- $(\mathcal{P}(\mathcal{D}), \to, \subseteq, \cap, \mathcal{D})$ is a natural type structure.

Chapter 16

Basic Properties 31.10.2006:581

This Chapter is on type theories but, by Remark 15.4.7, applies as well to type structures. That is, everywhere $\mathcal{T}$, TT and TT$^{\top}$ may be replaced by $\mathcal{S}$, TS and TS$^{\top}$, respectively.

Let $\mathcal{T}$ be a type theory. We derive properties of $\vdash^{\mathcal{T}}$, where $\vdash^{\mathcal{T}}$ stands for $\vdash^{\mathcal{T}}_{\cap}$ or $\vdash^{\mathcal{T}}_{\cap^{\top}}$. Whenever we need to require extra properties about $\mathcal{T}$, this will be stated explicitly. Often $\mathcal{T}$ will be one of the theories from Figure 15.2.

The properties that will be studied are inversion theorems that will make it possible to predict when statements

$$\Gamma \vdash^{\mathcal{T}} M : A \tag{1}$$

are derivable, in particular from what other statments. This will be done in Section 16.1. Building upon this, in Section 16.2 conditions are given when type assignment statements remain valid after reducing or expanding the M according to $\boldsymbol{\beta}$ or $\boldsymbol{\eta}$-rules.

16.1. Inversion theorems

In the style of Coppo et al. [1984] and Alessi et al. [2003], [2005] we shall isolate special properties which allow to 'reverse' some of the rules of the type assignment system $\vdash^{\mathcal{T}}_{\cap}$, thereby achieving some form of 'generation' and 'inversion' properties. These state necessary and sufficient conditions when an assertion $\Gamma \vdash^{\mathcal{T}} M : A$ holds depending on the form of M and A, see Theorems 16.1.1 and 16.1.10.

16.1.1. Theorem (Inversion Theorem I). *If $\vdash$ is $\vdash^{\mathcal{T}}_{\cap}$, then the following statements hold unconditionally; if it is $\vdash^{\mathcal{T}}_{\cap^{\top}}$, then they hold under the assumption that*

$A \neq \top$ in (i) and (ii).

$$\begin{array}{lrcl}
\text{(i)} & \Gamma \vdash x : A & \Leftrightarrow & \Gamma(x) \leq A. \\
\text{(ii)} & \Gamma \vdash MN : A & \Leftrightarrow & \exists k \geq 1\, \exists B_1, \ldots, B_k, C_1, \ldots, C_k \\
 & & & [C_1 \cap \ldots \cap C_k \leq A \ \& \ \forall i \in \{1, \ldots, k\} \\
 & & & \Gamma \vdash M : B_i \to C_i \ \& \ \Gamma \vdash N : B_i]. \\
\text{(iii)} & \Gamma \vdash \lambda x.M : A & \Leftrightarrow & \exists k \geq 1\, \exists B_1, \ldots, B_k, C_1, \ldots, C_k \\
 & & & [(B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \leq A \\
 & & & \ \& \ \forall i \in \{1, \ldots, k\}. \Gamma, x{:}B_i \vdash M : C_i].
\end{array}$$

Proof. We only treat ($\Rightarrow$) in (i)-(iii), as ($\Leftarrow$) is trivial. Let first $\vdash$ be $\vdash^{\mathcal{T}}_{\cap}$.

(i) By induction on derivations. We reason according which axiom or rule has been used in the last step. Only axiom (Ax), and rules ($\cap$I), ($\leq$) could have been applied. In the first case one has $\Gamma(x) \equiv A$. In the other two cases the induction hypothesis applies.

(ii) By induction on derivations. By assumption on A and the shape of the term the last applied step has to be rule ($\to$E), ($\leq$) or ($\cap$I). In the first case the last applied rule is

$$(\to\text{E}) \quad \frac{\Gamma \vdash M : D \to A \quad \Gamma \vdash N : D}{\Gamma \vdash MN : A}.$$

We can take $k = 1$ and $C_1 \equiv A$ and $B_1 \equiv D$. In the second case the last rule applied is

$$(\leq) \quad \frac{\Gamma \vdash MN : B \quad B \leq A}{\Gamma \vdash MN : A}$$

and the induction hypothesis applies. In the last case $A \equiv A_1 \cap A_2$ and the last applied rule is

$$(\cap\text{I}) \quad \frac{\Gamma \vdash MN : A_1 \quad \Gamma \vdash MN : A_2}{\Gamma \vdash MN : A_1 \cap A_2}.$$

By the induction hypothesis there are B_i, C_i, D_j, E_j, with $1 \leq i \leq k,\ 1 \leq j \leq k'$, such that

$$\begin{array}{ll}
\Gamma \vdash M : B_i \to C_i, & \Gamma \vdash N : B_i, \\
\Gamma \vdash M : D_j \to E_j, & \Gamma \vdash N : D_j, \\
C_1 \cap \ldots \cap C_k \leq A_1, & E_1 \cap \ldots \cap E_{k'} \leq A_2.
\end{array}$$

Hence we are done, as $C_1 \cap \ldots \cap C_k \cap E_1 \cap \ldots \cap E_{k'} \leq A$.

(iii) Again by induction on derivations. We only treat the case $A \equiv A_1 \cap A_2$ and the last applied rule is ($\cap$I):

$$(\cap\text{I}) \quad \frac{\Gamma \vdash \lambda x.M : A_1 \quad \Gamma \vdash \lambda x.M : A_2}{\Gamma \vdash \lambda x.M : A_1 \cap A_2}.$$

By the induction hypothesis there are B_i, C_i, D_j, E_j with $1 \leq i \leq k,\ 1 \leq j \leq k'$ such that

$$\begin{array}{ll}
\Gamma, x{:}B_i \vdash M : C_i, & (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \leq A_1, \\
\Gamma, x{:}D_j \vdash M : E_j, & (D_1 \to E_1) \cap \ldots \cap (D_{k'} \to E_{k'}) \leq A_2.
\end{array}$$

We are done, since $(B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \cap (D_1 \to E_1) \cap \ldots \cap (D_{k'} \to E_{k'}) \leq A$.

Now we prove ($\Rightarrow$) in (i)-(iii) for $\lambda^{\mathcal{T}}_{\cap^\top}$.

(i) The condition $A \neq \top$ implies that axiom ($\top$ universal) cannot have been used in the last step. Hence the reasoning above suffices.

(ii), (iii) The only interesting rule is ($\cap$I). Condition $A \neq \top$ implies that we cannot have $A_1 = A_2 = \top$. In case $A_1 \neq \top$ and $A_2 \neq \top$ the result follows as above. The other cases are more easy. ■

Notice that as a consequence of this theorem the *subformula property* holds for all $\lambda^{\mathcal{T}}_{\cap(\top)}$.

16.1.2. Corollary (Subformula property). *Assume* $\Gamma \vdash^{\mathcal{T}}_{\cap(\top)} M : A$ *and let* N *be a subterm of* M. *Then* N *is typable in an extension* $\Gamma^+ = \Gamma, x_1{:}B_1, \ldots, x_n{:}B_n$ *in which also the variables* $\{x_1, \ldots, x_n\} = \mathrm{FV}(N) - \mathrm{FV}(M)$ *get a type assigned.*

Proof. If we have rule ($\top$-universal) the statement is trivial. Otherwise if N is a subterm of M, then we can write $M \equiv C[N]$. The statement is proved by induction on the structure of $C[\ ]$. ■

16.1.3. Proposition. *We have for fresh* y *(*$\notin \mathrm{dom}(\Gamma)$*) the following.*

$$\begin{array}{l} \exists B\,[\Gamma \vdash N : B \ \&\ \Gamma \vdash M[x{:=}N] : A] \ \Rightarrow \\ \qquad \exists B\,[\Gamma \vdash N : B \ \&\ \Gamma, y{:}B \vdash M[x{:=}y] : A]. \end{array}$$

Proof. By induction on the structure of M. ■

Under some conditions (that will hold for many TTs, notably the ones introduced in Section 15.1), the Inversion Theorem can be restated in a more memorable form. This will be done in Theorem 16.1.10.

16.1.4. Definition. $\mathcal{T}$ is called $\boldsymbol{\beta}$*-sound* if

$\forall k{\geq}1 \forall A_1, \ldots, A_k, B_1, \ldots, B_k, C, D.$

$$\left.\begin{array}{l} (A_1{\to}B_1) \cap \ldots \cap (A_k{\to}B_k) \leq (C{\to}D) \ \&\ D \neq \top \ \Rightarrow \\ C \leq A_{i_1} \cap \ldots \cap A_{i_p} \ \&\ B_{i_1} \cap \ldots \cap B_{i_p} \leq D, \\ \text{for some } p \geq 1 \text{ and } 1 \leq i_1, \ldots, i_p \leq k. \end{array}\right\} \qquad (*)$$

This definition immediately translates to type structures. The notion of $\boldsymbol{\beta}$-soundness is introduced to prove invertibility of the rule ($\to$I), which is important for the next section.

16.1.5. Lemma. *Let* $\mathcal{T}$ *satisfy* ($\top$) *and* ($\top\to$). *Suppose moreover that* $\mathcal{T}$ *is* $\boldsymbol{\beta}$*-sound. Then for all* A, B

$$A{\to}B = \top \ \Leftrightarrow \ B = \top.$$

Proof. ($\Rightarrow$) $\top{\to}\top \leq \top = A{\to}B$, by assumption; hence $\top \leq B$ ($\leq \top$), by $\boldsymbol{\beta}$-soundness. ($\Leftarrow$) By rule ($\top\to$). ■

Let $\mathcal{T}$ be $\boldsymbol{\beta}$-sound. Then $A{\to}B \leq A'{\to}B' \ \Rightarrow \ A' \leq A \ \ \& \ \ B \leq B'$ if B' is not the top element (but not in general).

In 16.1.6-16.1.8 we will show that all $\mathcal{T}$'s of Figures 15.2 are $\boldsymbol{\beta}$-sound.

16.1.6. Remark. Note that in a TT every type A can be written uniquely, apart from the order, as

$$A \equiv \alpha_1 \cap \ldots \cap \alpha_n \cap (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \qquad (+),$$

i.e. an intersection of atoms ($\alpha_i \in \mathbb{A}$) and arrow types.

For some of our $\mathcal{T}$ the shape $(+)$ in Remark 16.1.6 can be simplified.

16.1.7. Definition. For the type theories $\mathcal{T}$ of Figure 15.2 we define for each $A \in \mathbb{T}^{\mathcal{T}}$ its *canonical form*, notation $\mathrm{cf}(A)$, as follows.

(i) If $\mathcal{T} \in \{\mathrm{BCD}, \mathrm{AO}, \mathrm{Plotkin}, \mathrm{Engeler}, \mathrm{CDS}, \mathrm{CDV}, \mathrm{CD}\}$, then

$$\mathrm{cf}(A) \equiv A.$$

(ii) If $\mathcal{T} \in \{\mathrm{Scott}, \mathrm{Park}, \mathrm{CDZ}, \mathrm{HR}, \mathrm{DHM}, \mathrm{HL}\}$ then the definition is by induction on A. For an atom α the canonical form $\mathrm{cf}(\alpha)$ depends on the type theory in question; moreover the mapping cf preserves $\to, \cap$ and $\top$.

System $\mathcal{T}$	A	$\mathrm{cf}(A)$
Scott	ω	$\top \to \omega$
Park	ω	$\omega \to \omega$
CDZ, HL	ω	$\varphi \to \omega$
	φ	$\omega \to \varphi$
HR	ω	$\varphi \to \omega$
	φ	$(\omega \to \omega) \cap (\varphi \to \varphi)$
DHM	φ	$\omega \to \varphi$
	ω	$\top \to \varphi$
All systems except HL	$\top$	$\top$
All systems	$B \to C$	$B \to C$
All systems	$B \cap C$	$\mathrm{cf}(B) \cap \mathrm{cf}(C)$

16.1.8. Theorem. *All theories $\mathcal{T}$ of Figure 15.2 are $\boldsymbol{\beta}$-sound.*

Proof. We prove the following stronger statement (induction loading). Let

$$\begin{aligned} A &\leq A', \\ \mathrm{cf}(A) &\equiv \alpha_1 \cap \ldots \cap \alpha_n \cap (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k), \\ \mathrm{cf}(A') &\equiv \alpha'_1 \cap \ldots \cap \alpha'_{n'} \cap (B'_1 \to C'_1) \cap \ldots \cap (B'_{k'} \to C'_{k'}). \end{aligned}$$

Then

$$\forall j \in \{1, k'\}.[\, C_{j'} \neq \top \;\Rightarrow$$
$$\exists p \geq 1 \exists i_1, \ldots i_p \in \{1, k\}.[B'_j \leq B_{i_1} \cap \ldots \cap B_{i_p} \;\&\; C_{i_1} \cap \ldots \cap C_{i_p} \leq C'_j]].$$

The proof of the statement is by induction on the generation of $A \leq A'$. From it $\boldsymbol{\beta}$-soundness follows easily. ■

16.1.9. Remark. From the Theorem it follows immediately that for the compatible theories of Fig. 15.2 the corresponding type structures are $\boldsymbol{\beta}$-sound.

16.1.10. Theorem (Inversion Theorem II). *Of the following properties (i) holds in general, (ii) provided that $\mathcal{T}$ is proper and $A \neq \top$ if $\vdash$ is $\vdash^{\mathcal{T}}_{\cap^\top}$ and (iii) provided that $\mathcal{T}$ is $\boldsymbol{\beta}$-sound.*

$$\begin{array}{lrcl}
\text{(i)} & \Gamma, x{:}A \vdash x : B & \Leftrightarrow & A \leq B. \\
\text{(ii)} & \Gamma \vdash (MN) : A & \Leftrightarrow & \exists B\, [\Gamma \vdash M : (B{\to}A)\ \&\ \Gamma \vdash N : B]. \\
\text{(iii)} & \Gamma \vdash (\lambda x.M) : (B{\to}C) & \Leftrightarrow & \Gamma, x{:}B \vdash M : C.
\end{array}$$

Proof. The proof of each ($\Leftarrow$) is easy. So we only treat ($\Rightarrow$).

(i) If $B \neq \top$, then the conclusion follows from Theorem 16.1.1(i). If $B = \top$, then the conclusion holds trivially.

(ii) Suppose $\Gamma \vdash MN : A$. Then by Theorem 16.1.1(ii) there are $B_1, \ldots, B_k$, $C_1, \ldots, C_k$, with $k \geq 1$, such that $C_1 \cap \ldots \cap C_k \leq A$, $\Gamma \vdash M : B_i{\to}C_i$ and $\Gamma \vdash N : B_i$ for $1 \leq i \leq k$. Hence $\Gamma \vdash N : B_1 \cap \ldots \cap B_k$ and

$$\begin{array}{rl}
\Gamma \vdash M : & (B_1{\to}C_1) \cap \ldots \cap (B_k{\to}C_k) \\
& \leq (B_1 \cap \ldots \cap B_k){\to}(C_1 \cap \ldots \cap C_k) \\
& \leq (B_1 \cap \ldots \cap B_k){\to}A,
\end{array}$$

by Lemma 15.1.13. So we can take $B \equiv (B_1 \cap \ldots \cap B_k)$.

(iii) Suppose $\Gamma \vdash (\lambda x.M) : (B{\to}C)$. Then Theorem 16.1.1(iii) applies and we have for some $k \geq 1$ and $B_1, \ldots, B_k, C_1, \ldots, C_k$

$$\begin{array}{c}
(B_1{\to}C_1) \cap \ldots \cap (B_k{\to}C_k) \leq B{\to}C, \\
\Gamma, x{:}B_i \vdash M : C_i \text{ for all } i.
\end{array}$$

If $C = \top$, then the assertion holds trivially, so let $C \neq \top$. Then by $\boldsymbol{\beta}$-soundness there are $1 \leq i_1, \ldots, i_p \leq k$, $p \geq 1$ such that

$$\begin{array}{c}
B \leq B_{i_1} \cap \ldots \cap B_{i_p}, \\
C_{i_1} \cap \ldots \cap C_{i_p} \leq C.
\end{array}$$

Applying ($\leq$-L) we get

$$\begin{array}{c}
\Gamma, x{:}B \vdash M : C_{i_j},\ 1 \leq j \leq p, \\
\Gamma, x{:}B \vdash M : C_{i_1} \cap \ldots \cap C_{i_p} \leq C. \ \blacksquare
\end{array}$$

We give a simple example which shows that in general rule ($\to$E) cannot be reversed, i.e. that if $\Gamma \vdash MN : B$, then it is not always true that there exists A such that $\Gamma \vdash M : A{\to}B$ and $\Gamma \vdash N : A$.

16.1.11. Example. Let $\mathcal{T}$ = Engeler, one of the intersection type theories of Figure 15.2. Let $\Gamma = \{x{:}(\varphi_0{\to}\varphi_1) \cap (\varphi_2{\to}\varphi_3), y{:}(\varphi_0 \cap \varphi_2)\}$. Then one has $\Gamma \vdash^{\mathcal{T}}_{\cap^\top} xy : \varphi_1 \cap \varphi_3$. Nevertheless, it is not possible to find a type B such that $\Gamma \vdash^{\mathcal{T}}_{\cap^\top} x : B{\to}(\varphi_1 \cap \varphi_3)$ and $\Gamma \vdash^{\mathcal{T}}_{\cap^\top} y : B$. See Exercise ??.

16.1.12. REMARK. In general

$$\Gamma \vdash^{\mathcal{T}} (\lambda x.M) : A \not\Rightarrow \exists B,C.A = (B{\to}C) \;\&\; \Gamma, x{:}B \vdash^{\mathcal{T}} M : C.$$

A counterexample is $\vdash^{\mathrm{BCD}} \mathsf{I} : (\alpha_1{\to}\alpha_1) \cap (\alpha_2{\to}\alpha_2)$, with α_1, α_2 atomic.

16.1.13. PROPOSITION. *For $\mathcal{T} \in \{$Scott, Park, CDZ, HR, DHM, BCD, AO$\}$ the properties (i), (ii) and (iii) of Theorem 16.1.10 hold for $\vdash^{\mathcal{T}}_{\cap^\top}$, provided that in (ii) $A \neq \top$ for $\mathcal{T} = $ AO. For $\mathcal{T} \in \{\mathrm{HL}, \mathrm{CDV}\}$ the properties hold unconditionally for $\vdash^{\mathcal{T}}_{\cap}$.*

PROOF. For these $\mathcal{T}$ Theorem 16.1.10 applies since they are proper and $\boldsymbol{\beta}$-sound (by Theorem 16.1.8). Moreover, by axiom $(\to\top)$ we have $\Gamma \vdash^{\mathcal{T}}_{\cap^\top} M : \top \to \top$ for all Γ, M, hence we do not need to assume $A \neq \top$ for $\mathcal{T} \in \{$Scott, Park, CDZ, HR, DHM, BCD$\}$. ■

16.2. Subject reduction and expansion

Various subject reduction and expansion properties are proved, for the classical $\boldsymbol{\beta}$, $\boldsymbol{\beta}\mathsf{I}$ and $\boldsymbol{\eta}$ notions of reduction. Other results can be found in Alessi et al. [2003], Alessi et al. [2006]. We consider the following rules.

$$(R\text{-}red) \quad \frac{M \to_R N \quad \Gamma \vdash M : A}{\Gamma \vdash N : A}$$

$$(R\text{-}exp) \quad \frac{M \,{}_R{\leftarrow} N \quad \Gamma \vdash M : A}{\Gamma \vdash N : A}$$

where R is a notion of reduction, notably $\boldsymbol{\beta}$-, $\boldsymbol{\beta}\mathsf{I}$, or $\boldsymbol{\eta}$-reduction. If one of these rules holds in $\lambda^{\mathcal{T}}_{\cap(^\top)}$, we write $\lambda^{\mathcal{T}}_{\cap(^\top)} \models (R\text{-}\{exp, red\})$, respectively. If both hold we write $\lambda^{\mathcal{T}}_{\cap(^\top)} \models (R\text{-}cnv)$. These properties will be crucial in Chapters 17, 18 and 19, where we will discuss (untyped) λ-models induced by these systems.

Recall that $(\lambda x.M)N$ is a $\boldsymbol{\beta}\mathsf{I}$-*redex* if $x \in \mathrm{FV}(M)$, Curry and Feys [1958].

$\boldsymbol{\beta}$-conversion

We first investigate when $\lambda^{\mathcal{T}}_{\cap(^\top)} \models (\boldsymbol{\beta}(\mathsf{I})\text{-}red)$.

16.2.1. PROPOSITION. (i) $\lambda^{\mathcal{T}}_{\cap(^\top)} \models (\boldsymbol{\beta}\mathsf{I}\text{-}red) \;\Leftrightarrow$

$$[\Gamma \vdash^{\mathcal{T}} (\lambda x.M) : (B{\to}A) \;\&\; x \in \mathrm{FV}(M) \;\Rightarrow\; \Gamma, x{:}B \vdash^{\mathcal{T}} M : A].$$

(ii) $\lambda^{\mathcal{T}}_{\cap(^\top)} \models (\boldsymbol{\beta}\text{-}red) \;\Leftrightarrow\; [\Gamma \vdash^{\mathcal{T}} (\lambda x.M) : (B{\to}A) \;\Rightarrow\; \Gamma, x{:}B \vdash^{\mathcal{T}} M : A]$.

PROOF. (i) $(\Rightarrow)$ Assume $\Gamma \vdash \lambda x.M : B{\to}A \;\&\; x \in \mathrm{FV}(M)$, which implies $\Gamma, y{:}B \vdash (\lambda x.M)y : A$, by weakening and rule $(\to\mathrm{E})$ for a fresh y. Now rule $(\boldsymbol{\beta}\mathsf{I}\text{-}red)$ gives us $\Gamma, y{:}B \vdash M[x{:=}y] : A$. Hence $\Gamma, x{:}B \vdash M : A$.

$(\Leftarrow)$ Suppose $\Gamma \vdash (\lambda x.M)N : A \;\&\; x \in \mathrm{FV}(M)$, in order to show that $\Gamma \vdash M[x{:=}N] : A$. We may assume $A \neq \top$. Then Theorem 16.1.1(ii) implies $\Gamma \vdash \lambda x.M : B_i{\to}C_i$, $\Gamma \vdash N : B_i$ and $C_1 \cap \ldots \cap C_k \leq A$, for some

$B_1,\ldots,B_k,C_1,\ldots,C_k$. By assumption $\Gamma, x{:}B_i \vdash M : C_i$. Hence by rule (*cut*), Proposition 15.2.8, one has $\Gamma \vdash M[x{:=}N] : C_i$. Therefore $\Gamma \vdash M[x{:=}N] : A$, using rules ($\cap$I) and ($\leq$).

(ii) Similarly. ■

16.2.2. Corollary. *Let $\mathcal{T}$ be $\boldsymbol{\beta}$-sound. Then $\lambda^{\mathcal{T}}_{\cap(\top)} \models (\boldsymbol{\beta}\text{-}red)$.*

Proof. Using Theorem 16.1.10(iii). ■

16.2.3. Corollary. (i) *Let $\mathcal{T} \in \{$Scott, Park, CDZ, HR, DHM, BCD, AO, Plotkin, Engeler, CDS$\}$. Then $\lambda^{\mathcal{T}}_{\cap\top} \models (\boldsymbol{\beta}\text{-}red)$.*

(ii) *Let $\mathcal{T} \in \{$HL, CD, CDV$\}$. Then $\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\beta}\text{-}red)$.*

Proof. By Corollary 16.2.2 and Theorem 16.1.8. ■

In Definition 18.2.22 we will introduce a type theory that is not $\boldsymbol{\beta}$-sound, but nevertheless induces a type assignment system satisfying ($\boldsymbol{\beta}$-*red*).
Now we investigate when $\lambda^{\mathcal{T}}_{\cap(\top)} \models (\boldsymbol{\beta}\text{-}exp)$. As a warm-up, suppose that $\Gamma \vdash M[x{:=}N] : A$. Then we would like to conclude that N has a type, as it seems to be a subformula, and therefore $\Gamma \vdash (\lambda x.M)N : A$. There are two problems: N may occur several times in $M[x{:=}N]$, so that it has (should have) in fact several types. In the system $\lambda_{\to}$ this problem causes the failure of rule ($\boldsymbol{\beta}$-*exp*). But in the intersection type theories one has $N : B_1 \cap \ldots \cap B_k$ if $N : B_1, \ldots, N : B_k$. Therefore $(\lambda x.M)N$ has a type if $M[x{:=}N]$ has one. The second problem arises if N does not occur at all in $M[x{:=}N]$, i.e. if the redex is a $\lambda\mathsf{K}$-redex. We would like to assign as type to N the intersection over an empty sequence, i.e. the top $\top$. This makes ($\boldsymbol{\beta}$-*exp*) invalid in $\lambda^{\mathcal{T}}_{\cap}$, but valid in systems $\lambda^{\mathcal{T}}_{\cap\top}$.

16.2.4. Proposition. (i) *Suppose $\Gamma \vdash^{\mathcal{T}} M[x{:=}N] : A$. Then*

$$\Gamma \vdash^{\mathcal{T}} (\lambda x.M)N : A \;\Leftrightarrow\; N \textit{ is typable in context } \Gamma.$$

(ii) $\lambda^{\mathcal{T}}_{\cap(\top)} \models (\boldsymbol{\beta}\text{-}exp) \;\Leftrightarrow\; \forall \Gamma, M, N, A$

$$[\Gamma \vdash^{\mathcal{T}} M[x{:=}N] : A \;\Rightarrow\; N \textit{ is typable in context } \Gamma].$$

(iii) $\lambda^{\mathcal{T}}_{\cap(\top)} \models (\boldsymbol{\beta}\mathsf{I}\text{-}exp) \;\Leftrightarrow\; \forall \Gamma, M, N, A$ *with* $x \in \mathrm{FV}(M)$

$$[\Gamma \vdash^{\mathcal{T}} M[x{:=}N] : A \;\Rightarrow\; N \textit{ is typable in context } \Gamma].$$

Proof. (i) ($\Rightarrow$) By Theorem 16.1.1(ii). ($\Leftarrow$) Let $\Gamma \vdash M[x{:=}N] : A$ and suppose N is typable in context Γ. By proposition 16.1.3 for some B and a fresh y one has $\Gamma \vdash N : B$ & $\Gamma, y{:}B \vdash M[x{:}=y] : A$. Then $\Gamma \vdash \lambda x.M : (B{\to}A)$ and hence $\Gamma \vdash (\lambda x.M)N : A$.

(ii) ($\Rightarrow$) Assume $\Gamma \vdash M[x{:=}N] : A$. Then $\Gamma \vdash (\lambda x.M)N : A$, by ($\boldsymbol{\beta}$-*exp*), hence by (i) we are done. ($\Leftarrow$) Assume $\Gamma \vdash L' : A$, with $L \to_{\boldsymbol{\beta}} L'$. By induction on the generation of $L \to_{\boldsymbol{\beta}} L'$ we get $\Gamma \vdash L : A$ from (i) and Theorem 16.1.1.

(iii) Similar to (ii). ■

16.2.5. COROLLARY. (i) $\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\beta}\text{-}exp)$.
(ii) $\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\beta}\mathsf{I}\text{-}exp)$.

PROOF. (i) Trivial, since every term has type $\top$.
(ii) By the subformula property (Corollary 16.1.2). ■

Now we can harvest results towards closure under $\boldsymbol{\beta}$-conversion.

16.2.6. THEOREM. *Let $\mathcal{T} \in \mathrm{TT}$ be $\boldsymbol{\beta}$-sound.*
(i) *Let $\mathcal{T} \in \mathrm{TT}^\top$. Then $\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\beta}\text{-}cnv)$.*
(ii) $\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\beta}\mathsf{I}\text{-}cnv)$.

PROOF. (i) By Corollaries 16.2.2 and 16.2.5(i).
(ii) By Corollaries 16.2.2 and 16.2.5(ii). ■

16.2.7. COROLLARY. (i) *Let $\mathcal{T} \in \{\mathrm{Scott}, \mathrm{Park}, \mathrm{CDZ}, \mathrm{HR}, \mathrm{DHM}, \mathrm{BCD}, \mathrm{AO}, \mathrm{Plotkin}, \mathrm{Engeler}, \mathrm{CDS}\}$. Then $\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\beta}\text{-}cnv)$.*
(ii) *Let $\mathcal{T} \in \{\mathrm{HL}, \mathrm{CDV}, \mathrm{CD}\}$. Then $\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\beta}\mathsf{I}\text{-}cnv)$.*

PROOF. (i) By Theorem 16.2.6(i).
(ii) By Theorem 16.2.6(ii). ■

$\boldsymbol{\eta}$-conversion

First we give necessary and sufficient conditions for a system $\lambda^{\mathcal{T}}_{\cap(^\top)}$ to satisfy the rule $(\boldsymbol{\eta}\text{-}red)$.

16.2.8. THEOREM. (i) *Let $\mathcal{T} \in \mathrm{TT}^\top$. Then*

$$\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\eta}\text{-}red) \;\Leftrightarrow\; \mathcal{T} \text{ is natural.}$$

(ii) *Let $\mathcal{T} \in \mathrm{TT}$. Then*

$$\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-}red) \;\Leftrightarrow\; \mathcal{T} \text{ is proper.}$$

PROOF. (i) ($\Rightarrow$) Assume $\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\eta}\text{-}red)$ towards $(\to\cap)$, $(\to)$ and $(\top\to)$.
As to $(\to\cap)$, one has

$$x{:}(A{\to}B) \cap (A{\to}C), y{:}A \vdash xy : B \cap C,$$

hence by $(\to\mathrm{I})$ it follows that $x{:}(A{\to}B) \cap (A{\to}C) \vdash \lambda y.xy : A{\to}(B \cap C)$. Therefore $x{:}(A{\to}B) \cap (A{\to}C) \vdash x : A{\to}(B \cap C)$, by $(\boldsymbol{\eta}\text{-}red)$. By Theorem 16.1.10(i) one can conclude $(A{\to}B) \cap (A{\to}C) \leq A{\to}(B \cap C)$.

As to $(\to)$, suppose that $A \leq B$ and $C \leq D$, in order to show $B{\to}C \leq A{\to}D$. One has $x{:}B{\to}C, y{:}A \vdash xy : C \leq D$, so $x{:}B{\to}C \vdash \lambda y.xy : A{\to}D$. Therefore by $(\boldsymbol{\eta}\text{-}red)$ it follows that $x{:}B{\to}C \vdash x : A{\to}D$ and we are done as before.

As to $\top \leq \top{\to}\top$, notice that $x{:}\top, y{:}\top \vdash xy : \top$, so we have $x{:}\top \vdash \lambda y.xy : \top{\to}\top$. Therefore $x{:}\top \vdash x : \top{\to}\top$ and again we are done.

($\Leftarrow$) Let $\mathcal{T}$ be natural. Assume that $\Gamma \vdash \lambda x.Mx : A$, with $x \notin \mathrm{FV}(M)$, in order to show $\Gamma \vdash M : A$. If $A = \top$, we are done. Otherwise,

$$\begin{array}{lll} \Gamma \vdash \lambda x.Mx : A & \Rightarrow & \Gamma, x{:}B_i \vdash Mx : C_i, 1 \leq i \leq k, \ \& \\ & & (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \leq A, \\ & & \text{for some } B_1, \ldots, B_k, C_1, \ldots, C_k, \end{array}$$

by Theorem 16.1.1(iii). By Lemma 16.1.5 we omit the i such that $C_i = \top$. There is at least one $C_i \neq \top$, since otherwise $A \geq (B_1 \to \top) \cap \ldots \cap (B_k \to \top) = \top$, again by Lemma 16.1.5, and we would have $A = \top$. Hence by Theorem 16.1.10(ii)

$$\begin{array}{lll} \Rightarrow & \Gamma, x{:}B_i \vdash M : D_i \to C_i \text{ and} & \\ & \Gamma, x{:}B_i \vdash x : D_i, & \text{for some } D_1, \ldots, D_k, \\ \Rightarrow & B_i \leq D_i, & \text{by Theorem 16.1.10(i)}, \\ \Rightarrow & \Gamma \vdash M : (B_i \to C_i), & \text{by } (\leq\text{-L}) \text{ and } (\to), \\ \Rightarrow & \Gamma \vdash M : ((B_1 \to C_1) \cap \ldots \cap (B_k \to C_k)) \leq A. & \end{array}$$

(ii) Similarly, but simpler. ■

16.2.9. Corollary. (i) *Let* $\mathcal{T} \in \{\mathrm{Scott}, \mathrm{Park}, \mathrm{CDZ}, \mathrm{HR}, \mathrm{DHM}, \mathrm{BCD}\}$. *Then* $\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\eta}\text{-}red)$.

(ii) *Let* $\mathcal{T} \in \{\mathrm{HL}, \mathrm{CDV}\}$. *Then* $\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-}red)$.

In order to characterize the admissibility of rule ($\boldsymbol{\eta}$-*exp*), we need to introduce a further condition on type theories. This condition is necessary and sufficient to derive from the basis $x{:}A$ the same type A for $\lambda y.xy$, as we will show in the proof of Theorem 16.2.11.

16.2.10. Definition. Let $\mathcal{T} \in \mathrm{TT}$.
(i) $\mathcal{T}$ is called $\boldsymbol{\eta}$-*sound* iff for all A there are $k \geq 1$, $m_1, \ldots, m_k \geq 1$ and $B_1, \ldots, B_k$, $C_1, \ldots, C_k$,

$$\begin{pmatrix} D_{11} \ldots D_{1m_1} \\ \ldots \\ D_{k1} \ldots D_{km_k} \end{pmatrix} \text{ and } \begin{pmatrix} E_{11} \ldots E_{1m_1} \\ \ldots \\ E_{k1} \ldots E_{km_k} \end{pmatrix}$$

with

$$\begin{array}{l} (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \ \leq \ A \\ \& \ A \ \leq \ (D_{11} \to E_{11}) \cap \ldots \cap (D_{1m_1} \to E_{1m_1}) \cap \\ \qquad \ldots \\ \qquad (D_{k1} \to E_{k1}) \cap \ldots \cap (D_{km_k} \to E_{km_k}) \\ \& \ B_i \ \leq \ D_{i1} \cap \ldots \cap D_{im_i} \ \& \ E_{i1} \cap \ldots \cap E_{im_i} \ \leq \ C_i, \\ \quad \text{for } 1 \leq i \leq k. \end{array}$$

(ii) Let $\mathcal{T} \in \mathrm{TT}^\top$. Then $\mathcal{T}$ is called $\boldsymbol{\eta}^\top$-*sound* iff for all $A \neq \top$ at least one of the following two conditions holds.

(1) There are types $B_1, \ldots, B_n$ with $(B_1 \to \top) \cap \ldots \cap (B_n \to \top) \leq A$;

(2) There are $k \geq 1$, $m_1, \ldots, m_k \geq 1$ and $B_1, \ldots, B_k$, $C_1, \ldots, C_k$,

$$\begin{pmatrix} D_{11} \ldots D_{1m_1} \\ \ldots \\ D_{k1} \ldots D_{km_k} \end{pmatrix} \text{ and } \begin{pmatrix} E_{11} \ldots E_{1m_1} \\ \ldots \\ E_{k1} \ldots E_{km_k} \end{pmatrix}$$

with

$$\begin{array}{l} (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \cap \\ \cap\, (B_{k+1} \to \top) \cap \ldots \cap (B_n \to \top) \;\leq\; A \\ \& \;\; A \;\leq\; (D_{11} \to E_{11}) \cap \ldots \cap (D_{1m_1} \to E_{1m_1}) \cap \\ \qquad\quad \ldots \\ \qquad\quad (D_{k1} \to E_{k1}) \cap \ldots \cap (D_{km_k} \to E_{km_k}) \\ \& \;\; B_i \;\leq\; D_{i1} \cap \ldots \cap D_{im_i} \;\&\; E_{i1} \cap \ldots \cap E_{im_i} \;\leq\; C_i, \\ \qquad \text{for } 1 \leq i \leq k. \end{array}$$

This definition immediately translates to type structures. The validity of $\boldsymbol{\eta}$-expansion can be given as follows.

16.2.11. Theorem (Characterization of $\boldsymbol{\eta}$-*exp*).

$$\begin{array}{lll} \text{(i)} & \lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-}exp) & \Leftrightarrow \quad \mathcal{T} \textit{ is } \boldsymbol{\eta}\textit{-sound.} \\ \text{(ii)} & \lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\eta}\text{-}exp) & \Leftrightarrow \quad \mathcal{T} \textit{ is } \boldsymbol{\eta}^\top\textit{-sound.} \end{array}$$

Proof. (i) ($\Rightarrow$) Assume $\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-exp})$. As $x{:}A \vdash x : A$, by assumption we have $x{:}A \vdash \lambda y.xy : A$. From Theorem 16.1.1(iii) it follows that $x{:}A, y{:}B_i \vdash xy : C_i$ and $(B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \leq A$ for some B_i, C_i. By Theorem 16.1.1(ii) for each i there exist D_{ij}, E_{ij}, such that for each j one has $x{:}A, y{:}B_i \vdash x : (D_{ij} \to E_{ij})$, $x{:}A, y{:}B_i \vdash y : D_{ij}$ and $E_{i1} \cap \ldots \cap E_{im_i} \leq C_i$. Hence by Theorem 16.1.1(i) we have $A \leq (D_{ij} \to E_{ij})$ and $B_i \leq D_{ij}$ for all i and j. Therefore we obtain the condition of 16.2.10(i).

($\Leftarrow$) Suppose that $\Gamma \vdash M : A$ in order to show $\Gamma \vdash \lambda x.Mx : A$, with x fresh. By assumption A satisfies the condition of Definition 16.2.10(i).

$$\begin{array}{l} (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \;\leq\; A \\ \& \;\; A \;\leq\; (D_{11} \to E_{11}) \cap \ldots \cap (D_{1m_1} \to E_{1m_1}) \cap \\ \qquad\quad \ldots \\ \qquad\quad (D_{k1} \to E_{k1}) \cap \ldots \cap (D_{km_k} \to E_{km_k}) \\ \& \;\; B_i \;\leq\; D_{i1} \cap \ldots \cap D_{im_i} \;\&\; E_{i1} \cap \ldots \cap E_{im_i} \;\leq\; C_i, \\ \qquad \text{for } 1 \leq i \leq k. \end{array}$$

By rule ($\leq$) for all i, j we have $\Gamma \vdash M : D_{ij} \to E_{ij}$ and so $\Gamma, x{:}D_{ij} \vdash Mx : E_{ij}$ by rule ($\to$E). From ($\leq$ L), ($\cap$I) and ($\leq$) we get $\Gamma, x{:}B_i \vdash Mx : C_i$ and this implies $\Gamma \vdash \lambda x.Mx : B_i \to C_i$, using rule ($\to$I). So we can conclude by ($\cap$I) and ($\leq$) that $\Gamma \vdash \lambda x.Mx : A$.

(ii) The proof is nearly the same as for (i). ($\Rightarrow$) Again we get $x{:}A, y{:}B_i \vdash xy : C_i$ and $(B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \leq A$ for some B_i, C_i. If all $C_i = \top$, then A satisfies the first condition of Definition 16.2.10(ii). Otherwise, consider the i such that $C_i \neq \top$ and reason as in the proof of ($\Rightarrow$) for (i).

($\Leftarrow$) Suppose that $\Gamma \vdash M : A$ in order to show $\Gamma \vdash \lambda x.Mx : A$, with x fresh. If A satisfies the first condition of Definition 16.2.10(ii), that is $(B_1 \to \top) \cap \ldots \cap (B_n \to \top) \leq A$, then by rule $(\top)$ it follows that $\Gamma, x{:}B_i \vdash Mx : \top$, hence $\Gamma \vdash \lambda x.Mx : (B_1 \to \top) \cap \ldots \cap (B_n \to \top) \leq A$. Now let A satisfy the second condition. Then the proof is similar to that for ($\Leftarrow$) in (i). ■

For most intersection type theories of interest the condition of $\boldsymbol{\eta}(^\top)$-soundness is deduced from the following proposition.

16.2.12. Proposition. *Let $\mathcal{T} \in \mathrm{TT}$ with atoms $\mathbb{A}$ be proper.*

(i) $\mathcal{T}$ *is* $\boldsymbol{\eta}$*-sound* $\Leftrightarrow$ $\forall A \in \mathbb{A}\, \exists B_1, \ldots, B_k, C_1, \ldots, C_k\, \exists n \geq 1$ $A = (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k)$.

(ii) *Let $\mathcal{T} \in \mathrm{TT}^\top$. Then*

$$\begin{array}{ll} \mathcal{T} \text{ is } \boldsymbol{\eta}^\top\text{-sound} & \Leftrightarrow \quad \forall A \in \mathbb{A}[\top \to \top \leq A \vee \exists B_1, \ldots, B_k, C_1, \ldots, C_k \\ \exists k \geq 1 & \quad\; [(B_1 \to C_1) \cap \ldots \cap (B_k \to C_k) \cap (\top \to \top) \leq A \\ & \quad\; \& \; A \leq (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k)]]. \end{array}$$

(iii) *Let $\mathcal{T} \in NTT^\top$. Then*

$$\mathcal{T} \text{ is } \boldsymbol{\eta}^\top\text{-sound} \;\Leftrightarrow\; \mathcal{T} \text{ is } \boldsymbol{\eta}\text{-sound}.$$

Proof. (i) ($\Rightarrow$) Suppose $\mathcal{T}$ is $\boldsymbol{\eta}$-sound. Let $A \in \mathbb{A}$. Then A satisfies the condition of Definition 16.2.10(i), for some $B_1, \ldots, B_k$, $C_1, \ldots, C_k$, $D_{11}, \ldots, D_{1m_1}, \ldots, D_{k1}, \ldots, D_{km_1}$, $E_{11}, \ldots, E_{1m_1}, \ldots, E_{k1}, \ldots, E_{km_k}$. By $(\to\cap)$ and $(\to)$, using Proposition 15.1.13, it follows that

$$\begin{array}{rcl} A & \leq & (D_{11} \cap \ldots \cap D_{1m_1} \to E_{11} \cap \ldots \cap E_{1m_1}) \cap \ldots \cap \\ & & (D_{k1} \cap \ldots \cap D_{km_k} \to E_{k1} \cap \ldots \cap E_{km_k}) \\ & \leq & (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k), \end{array}$$

hence $A =_{\mathcal{T}} (B_1 \to C_1) \cap \ldots \cap (B_k \to C_k)$.

($\Leftarrow$) By induction on the generation of A one can show that A satisfies the condition of $\boldsymbol{\eta}$-soundness. The case $A_1 \to A_2$ is trivial and the case $A \equiv A_1 \cap A_2$ follows by the induction hypothesis and Rule (mon).

(ii) Similarly. Note that $(\top \to \top) \leq (B \to \top)$ for all B.

(iii) Immediately by (ii) using rule $(\top\to)$. ■

16.2.13. Corollary. (i) *Let $\mathcal{T} \in \{\mathrm{Scott}, \mathrm{Park}, \mathrm{CDZ}, \mathrm{HR}, \mathrm{DHM}, \mathrm{AO}\}$. Then $\mathcal{T}$ is $\boldsymbol{\eta}^\top$-sound.*

(ii) HL *is $\boldsymbol{\eta}$-sound.*

Proof. Easy. For AO in (i) one applies (ii) of the Proposition. ■

16.2.14. Corollary. (i) *Let $\mathcal{T} \in \{\mathrm{Scott}, \mathrm{Park}, \mathrm{CDZ}, \mathrm{HR}, \mathrm{DHM}, \mathrm{AO}\}$. Then*

$$\lambda^{\mathcal{T}}_{\cap^\top} \models (\boldsymbol{\eta}\text{-}exp).$$

(ii) *Let* $\mathcal{T} = \mathrm{HL}$*, then*

$$\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-}exp).$$

PROOF. By the previous Corollary and Theorem 16.2.11. ■

Exercise 16.3.15 shows that the remaining systems of Figure 15.2 do not satisfy ($\boldsymbol{\eta}$-*exp*).

Now we can harvest results towards closure under $\boldsymbol{\eta}$-conversion.

$\mathcal{T}$	$\boldsymbol{\beta}$-red	$\boldsymbol{\beta}$I-red	$\boldsymbol{\beta}$-exp	$\boldsymbol{\beta}$I-exp	$\boldsymbol{\eta}$-red	$\boldsymbol{\eta}$-exp
Scott	√	√	√	√	√	√
Park	√	√	√	√	√	√
CDZ	√	√	√	√	√	√
HR	√	√	√	√	√	√
DHM	√	√	√	√	√	√
BCD	√	√	√	√	√	↯
AO	√	√	√	√	↯	√
Plotkin	√	√	√	√	↯	↯
Engeler	√	√	√	√	↯	↯
CDS	√	√	√	√	↯	↯
HL	√	√	↯	√	√	√
CDV	√	√	↯	√	√	↯
CD	√	√	↯	√	↯	↯

Figure 16.1: Type theories versus reduction and expansion

16.2.15. THEOREM. (i) *Let* $\mathcal{T} \in \mathrm{TT}^{\top}$*. Then*

$$\lambda^{\mathcal{T}}_{\cap^{\top}} \models (\boldsymbol{\eta}\text{-}cnv) \;\Leftrightarrow\; \mathcal{T} \text{ is natural and } \boldsymbol{\eta}^{\top}\text{-sound.}$$

(ii) *Let* $\mathcal{T} \in \mathrm{TT}$*. Then*

$$\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-}cnv) \;\Leftrightarrow\; \mathcal{T} \text{ is proper and } \boldsymbol{\eta}\text{-sound.}$$

PROOF. (i) By Theorems 16.2.11(ii) and 16.2.8(i).

(ii) By Theorems 16.2.11(i) and 16.2.8(ii). ■

16.2.16. THEOREM. (i) *For* $\mathcal{T} \in \{\mathrm{Scott}, \mathrm{Park}, \mathrm{CDZ}, \mathrm{HR}, \mathrm{DHM}\}$ *one has*

$$\lambda^{\mathcal{T}}_{\cap^{\top}} \models (\boldsymbol{\eta}\text{-}cnv).$$

(ii) *For* $\mathcal{T} = \mathrm{HL}$ *one has*

$$\lambda^{\mathcal{T}}_{\cap} \models (\boldsymbol{\eta}\text{-}cnv).$$

PROOF. (i) By Corollaries 16.2.9(i) and 16.2.14(i).

(ii) By Corollaries 16.2.9(ii) and 16.2.14(ii). ■

Figure 16.1 summarises the results of this section and of the exercises in the following section for the type theories of Figure 15.2. The symbol '√' stands for "the property holds" and '↯' for "the property fails".

16.3. Exercises

16.3.1. Show that for each number $n \in \mathbb{N}$ there is a type $A_n \in \mathbb{T}^{\mathrm{CD}}$ such that for the Church numerals $\mathbf{c}_n$ one has $\Gamma \vdash^{\mathrm{CD}}_{\cap} \mathbf{c}_{n+1} : A_n$, but $\Gamma \not\vdash^{\mathrm{CD}}_{\cap} \mathbf{c}_n : A_n$.

16.3.2. Show that $\mathsf{S}(\mathsf{KI})(\mathsf{II})$ and $(\lambda x.xxx)\mathsf{S}$ are typable in $\vdash^{\mathrm{CD}}_{\cap}$.

16.3.3. Derive $\vdash^{\mathrm{CDZ}}_{\cap^\top} (\lambda x.xxx)\mathsf{S} : \varphi$ and $y{:}\omega, z{:}\omega \vdash^{\mathrm{CDZ}}_{\cap^\top} (\lambda x.xxx)(\mathsf{S}yz) : \omega$.

16.3.4. Find the relation between the following three types w.r.t. $\leq_{\mathrm{CDZ}}$.

$$(\omega{\to}(\varphi{\to}\varphi){\to}\omega) \cap ((\varphi{\to}\varphi){\to}\varphi), (\omega{\to}\omega){\to}\omega \text{ and } \varphi{\to}(\omega{\to}\omega){\to}\varphi.$$

16.3.5. Using the Inversion Theorems show the following.

(i) $\not\vdash^{\mathrm{CD}}_{\cap} \mathsf{1} : \alpha{\to}\alpha$, where α is any constant.

(ii) $\not\vdash^{\mathrm{HL}}_{\cap} \mathsf{K} : \omega$.

(iii) $\not\vdash^{\mathrm{Scott}}_{\cap^\top} \mathsf{I} : \omega$.

(iv) $\not\vdash^{\mathrm{Plotkin}}_{\cap^\top} \mathsf{I}x : \omega$.

16.3.6. We say that M and M' have the same types in Γ, notation $M \sim_\Gamma M'$ if

$$\forall A\, [\Gamma \vdash M : A \;\Leftrightarrow\; \Gamma \vdash M' : A].$$

Prove that $M \sim_\Gamma M' \;\Rightarrow\; M\vec{N} \sim_\Gamma M'\vec{N}$.

16.3.7. Show that $\mathcal{T} =$ Plotkin is $\boldsymbol{\beta}$-sound by checking that it satisfies the following stronger condition.

$$(A_1{\to}B_1) \cap \ldots \cap (A_n{\to}B_n) \leq C{\to}D \;\Rightarrow$$
$$\exists k \neq 0\, \exists i_1, \ldots, i_k . 1 \leq i_j \leq n \;\&\; C = A_{i_j} \;\&\; B_{i_1} \cap \ldots \cap B_{i_k} = D.$$

16.3.8. Show that $\mathcal{T} =$ Engeler is $\boldsymbol{\beta}$-sound by checking that it satisfies the following stronger condition:

$$(A_1{\to}B_1) \cap \ldots \cap (A_n{\to}B_n) \leq C{\to}D \& D \;\neq\; \top \;\Rightarrow$$
$$\exists k \neq 0\, \exists i_1, \ldots, i_k . 1 \leq i_j \leq n \;\&\; C = A_{i_j} \;\&\; B_{i_1} \cap \ldots \cap B_{i_k} = D.$$

16.3.9. Let $\mathbb{A}^{\mathcal{T}} = \{\top, \omega\}$ and $\mathcal{T}$ be defined by the axioms and rules of the theories Scott and Park together. Show that $\mathcal{T}$ is not $\boldsymbol{\beta}$-sound [Hint: show that $\top \neq \omega$].

16.3.10. Prove that Theorem 16.1.10(ii) still holds if the condition of properness is replaced by the following two conditions

$$A \leq_{\mathcal{T}} B \;\Rightarrow\; C{\to}A \leq_{\mathcal{T}} C{\to}B$$

$$(A{\to}B) \cap (C{\to}D) \leq_{\mathcal{T}} A \cap C{\to}B \cap D.$$

16.3.11. Show that the following condition

$$A \to B =_{\mathcal{T}} \top \to \top \;\Rightarrow\; B =_{\mathcal{T}} \top$$

is necessary for the admissibility of rule ($\boldsymbol{\beta}$-*red*) in $\lambda^{\mathcal{T}}_{\cap}$. [Hint: Use Proposition 16.2.1(ii).]

16.3.12. Remember that the systems $\lambda^{\mathtt{K}}_{\cap}$ and $\lambda^{\mathtt{K}^\top}_{\cap^\top}$ are defined in Exercise 15.5.2.
(i) Show that rules ($\boldsymbol{\beta}$-*red*) and ($\boldsymbol{\beta}$I-*exp*) are admissible in $\lambda^{\mathtt{K}}_{\cap}$, while ($\boldsymbol{\beta}$-*exp*) is not admissible.
(ii) Show that rules ($\boldsymbol{\beta}$-*red*) and ($\boldsymbol{\beta}$-*exp*) are admissible in $\lambda^{\mathtt{K}^\top}_{\cap^\top}$.

16.3.13. (i) Show that for $\mathcal{T} \in \{\text{AO}, \text{Engeler}, \text{Plotkin}, \text{CDS}\}$ one has

$$\lambda^{\mathcal{T}}_{\cap^\top} \not\models (\boldsymbol{\eta}\text{-red}).$$

(ii) Show that for $\mathcal{T} = \text{CD}$ one has

$$\lambda^{\mathcal{T}}_{\cap} \not\models (\boldsymbol{\eta}\text{-red}).$$

16.3.14. Verify the following.
(i) $\boldsymbol{\eta}$-soundness implies $\boldsymbol{\eta}\top$-soundness.
(ii) Let $\mathcal{T} \in \{\text{BCD}, \text{Plotkin}, \text{Engeler}, \text{CDS}\}$. Then $\mathcal{T}$ is not $\boldsymbol{\eta}\top$-sound.
(iii) Let $\mathcal{T} \in \{\text{AO}, \text{CDV}, \text{CD}\}$. Then $\mathcal{T}$ is not $\boldsymbol{\eta}$-sound.
Notice that AO is $\boldsymbol{\eta}\top$-sound (Corollary 16.2.13). **Comment:** it is very interesting that AO is $\boldsymbol{\eta}\top$-sound but not $\boldsymbol{\eta}$-sound, why do you propose to erase it?

16.3.15. (i) Show that for $\mathcal{T} \in \{\text{BCD}, \text{Engeler}, \text{Plotkin}, \text{CDS}\}$ one has

$$\lambda^{\mathcal{T}}_{\cap^\top} \not\models (\boldsymbol{\eta}\text{-exp}).$$

(ii) Show that for $\mathcal{T} \in \{\text{CDV}, \text{CD}\}$ one has

$$\lambda^{\mathcal{T}}_{\cap} \not\models (\boldsymbol{\eta}\text{-exp}).$$

16.3.16. Show that rules ($\boldsymbol{\eta}$-*red*) and ($\boldsymbol{\eta}$-*exp*) are not admissible in the systems $\lambda^{\mathtt{K}}_{\cap}$ and $\lambda^{\mathtt{K}^\top}_{\cap^\top}$ as defined in Exercises 15.5.2.

16.3.17. Let $\vdash$ denote derivability in the system obtained from the system $\lambda^{\text{CDV}}_{\cap}$ by replacing rule ($\leq$) by the rules ($\cap$E), see Definition 15.2.5, and adding the rule

$$(R\eta) \quad \frac{\Gamma \vdash \lambda x.Mx : A}{\Gamma \vdash M : A} \quad \text{if } x \notin \text{FV}(M).$$

Show that $\Gamma \vdash^{\text{CDV}}_{\cap} M : A \Leftrightarrow \Gamma \vdash M : A$.

16.3.18. (Barendregt et al. [1983]) Let $\vdash$ denote derivability in the system obtained from $\lambda^{\text{BCD}}_{\cap^\top}$ by replacing rule ($\leq$) by the rules ($\cap$E) and adding ($R\eta$) as defined in Exercise 16.3.17. Verify that

$$\Gamma \vdash^{\text{BCD}}_{\cap^\top} M : A \Leftrightarrow \Gamma \vdash M : A.$$

16.3.19. Let Δ be a basis that is allowed to be infinite. We define $\Delta \vdash M : A$ iff there exists a finite basis $\Gamma \subseteq \Delta$ such that $\Gamma \vdash M : A$.
(i) Show that all the typability rules are derivable except possibly for ($\to$I).

(ii) Suppose $\mathrm{dom}(\Delta)$ is the set of all the variables. Show that the rule ($\to$I) is derivable if it is reformulated as

$$\Delta_x, x{:}A \vdash M : B \;\Rightarrow\; \Delta \vdash (\lambda x.M) : (A \to B),$$

with Δ_x the result of removing any $x{:}C$ from Δ.

(iii) Reformulate and prove Propositions 15.2.8, 15.2.10, Theorems 16.1.1 and 16.1.10 for infinite bases.

16.3.20. A *multi-basis* Γ is a set of declarations, in which the requirement that

$$x{:}A, y{:}B \in \Gamma \;\Rightarrow\; x \equiv y \;\Rightarrow\; A \equiv B$$

is dropped. Let Δ be a (possibly infinite) multi-basis. We define $\Delta \vdash M : A$ iff there exists a singled (only one declaration per variable) basis $\Gamma \subseteq \Delta$ such that $\Gamma \vdash M : A$.

(i) Show that $x : \alpha_1, x : \alpha_2 \not\vdash^{\mathrm{CD}} x : \alpha_1 \cap \alpha_2$.

(ii) Show that $x : \alpha_1 \to \alpha_2, x : \alpha_1 \not\vdash^{\mathrm{CD}} xx : \alpha_2$.

(iii) Consider $\Delta = \{x : \alpha_1 \cap \alpha_2, x : \alpha_1\}$;
$A = \alpha_2$;
$B = (\alpha_1 \to \alpha_2 \to \alpha_3) \to \alpha_3$;
$M = \lambda y.yxx$.
Show that $\Delta, x : A \vdash^{\mathrm{CD}} M : B$, but $\Delta \not\vdash^{\mathrm{CD}} (\lambda x.M) : (A \to B)$.

(iv) We say that a multi-basis is closed under $\cap$ if for all $x \in \mathrm{dom}(\Delta)$ the set $\mathcal{X} = \Delta(x)$ is closed under $\cap$, i.e. $A, B \in \mathcal{X} \;\Rightarrow\; A \cap B \in \mathcal{X}$, up to equality of types in the TT under consideration.
Show that all the typability rules of Figures 15.4 and 15.6, except for ($\to$I), are derivable for (possibly infinite) multi-bases that are closed under $\cap$.

(v) Let Δ be closed under $\cap$. We define

$$\Delta[x := X] = \{y : \Delta(y) \mid y \neq x\} \cup \{x : A \mid A \in X\}.$$

Prove that the following reformulation of ($\to$I) using principal filters is derivable

$$\frac{\Delta[x :=\uparrow B] \vdash N : C}{\Delta \vdash \lambda x.N : B \to C}.$$

(vi) Prove Propositions 15.2.8, 15.2.10, Theorems 16.1.1 and 16.1.10 for (possible infinite) multi-bases reformulating the statements whenever it is necessary.

(vii) Prove that if $\Delta(x)$'s are filters then $\{A \mid \Delta \vdash x : A\} = \Delta(x)$.

16.3.21. Show that the inclusions suggested in 15.3 are strict.