Proof-Theoretic Interconnections between Known Inference- and Axiom-Systems for the Equality and Subtype Relations on Recursive Types

Clemens Grabmayer

clemens@cs.vu.nl
www.cs.vu.nl/~clemens

Theoretical Computer Science group,
Vrije Universiteit Amsterdam

My Supervisors:

Jan Willem Klop,
Roel de Vrijer

Overview of the Talk

- Motivation for Recursive Types in Computer Science

A. Recursive Type Equality $=_\mu$

- Definition of $=_\mu$.
- The Proof-Systems $\mathbf{AC}^=$, $\mathbf{HB}^=$, $\mathbf{SM}^=$ for recursive type equality $=_\mu$.
  (Plus the variant-systems $\mathbf{AC}^=_\mathbf{1}$, $\mathbf{HB}^=_\mathbf{0}$ and $\mathbf{SM}^=_\mathbf{0}$).
- A "*Duality*" between "Consistency-Checks" in $\mathbf{SM}^=_\mathbf{0}$ and derivations in $\mathbf{HB}^=_\mathbf{0}$ (*Jan Willem Klop*).
- μ-COMPAT is a derived rule of $\mathbf{AC}^=_\mathbf{1}$.
- Translation of $\mathbf{AC}^=_\mathbf{1}$- (and also of $\mathbf{AC}^=$-)derivations into $\mathbf{HB}^=$-derivations.
- Transformation of $\mathbf{HB}^=$- into $\mathbf{AC}^=$-derivations.
- Some Side-Issues:
  - The FIX-rule.
  - $([g]^{(\mathrm{vs})}_{\underline{\leftrightarrow}}, \underline{\leftarrow})$ is not in general a complete lattice.

B. Subtyping Relation $\leq_\mu$ on Recursive Types

- Definition of $\leq_\mu$.
- The Proof-Systems $\mathbf{AC}^{\leq}$, $\mathbf{HB}^{\leq}$ and $\mathbf{SM}^{\leq}$ for $\leq_\mu$. (Plus the variant-systems $\mathbf{HB}_{\mathbf{0}}^{\leq}$, $\mathbf{HB}_{\mathbf{0,rst}}^{\leq}$ of $\mathbf{HB}^{\leq}$ and $\mathbf{SM}_{\mathbf{0}}^{\leq}$ of $\mathbf{SM}^{\leq}$).
- Duality between "consistency checks" in $\mathbf{SM}_{\mathbf{0}}^{\leq}$ and derivations in $\mathbf{HB}_{\mathbf{0}}^{\leq}$.
- Transformation of $\mathbf{AC}^{\leq}$- into $\mathbf{HB}^{\leq}$-derivations.
- Transformation of $\mathbf{HB}^{\leq}$- into $\mathbf{AC}^{\leq}$-derivations.
- Some Side-Issues.
  - The FIX-rule is a derived rule of $\mathbf{HB}^{\leq}$.
  - The rule μ-INTRO of $\mathbf{AC}^{\leq}$ is (in $\mathbf{HB}^{\leq}$) equivalent to the rules ARROW/FIX or FIX.

Type: A collection of values sharing a common shape or structure,

e.g. basic types: **Unit**, **Int**, **Float**

structured types: $\mathbf{Int} \to \mathbf{Int}$, $\mathbf{Int} \times \mathbf{Int}$, $\mathbf{Unit} + \mathbf{Int}$,

where **Unit** is a trivial type containing just one specific value val.

Recursive Type: A type, that satisfies a *recursive type equation* like

$\mathbf{Tree} = \mathbf{Int} + (\mathbf{Tree} \times \mathbf{Tree})$	type of binary trees with integer leaves
$\mathbf{A_List} = \mathbf{Unit} + (\mathbf{A} \times \mathbf{A_List})$	type of lists with members of type **A** .

For recursive type equations like

$$\tau = \mathbf{Unit} + (\mathbf{A} \times \tau) \qquad (1)$$

in theoretical contexts often *canonical solutions* in the form of a μ-expression

$$\mu\alpha.\,(\mathbf{Unit} + (\mathbf{A} \times \alpha)) \quad =:\ \sigma$$

are considered. Assuming the presence of a conversion

$$\mu\beta.\,\rho = \rho[\mu\beta.\,\rho/\beta] \ \text{(FOLD/UNFOLD)}$$

the μ-type σ can then be seen to satisfy equation (1) for τ:

$$\begin{aligned}\sigma = \mu\alpha.\,(\mathbf{Unit} + (\mathbf{A} \times \alpha)) &= (\mathbf{Unit} + (\mathbf{A} \times \alpha))\,[\sigma/\alpha] \\ &= \mathbf{Unit} + (\mathbf{A} \times \sigma) \ .\end{aligned}$$

There is also an intuitive need for identifying such solutions of recursive type equations that solve a common equation. E.g. in the case of the equations

$$\tau = \mathbf{Unit} + (\mathbf{A} \times \tau) \tag{2}$$

and

$$\tau = \mathbf{Unit} + (\mathbf{A} \times (\mathbf{Unit} + \mathbf{A} \times \tau)) \tag{3}$$

every solution to (2) also solves (3). – And intuitively they possess the same solution

$$\mathbf{A_List} := \{\mathrm{val}, (a_0, \mathrm{val}), (a_0, (a_1, \mathrm{val})), \ldots : a_0, a_1, \ldots \in A\}$$

(if $\mathbf{Unit} = \{\mathrm{val}\}$).

Therefore also

$$\mu\alpha.\,(\mathbf{Unit} + (\mathbf{A} \times \alpha)) = \\ = \mu\alpha.\,(\mathbf{Unit} + (\mathbf{A} \times (\mathbf{Unit} + \mathbf{A} \times \alpha)))$$

should hold. This means that canonical solutions that satisfy the same equation

$$\alpha = \tau(\alpha)$$

can—under certain restrictions on τ and α—be identified. In a formal system this leads to a rule

$$\frac{\tau_1 = \tau[\tau_1/\alpha] \qquad \tau_2 = \tau[\tau_2/\alpha]}{\tau_1 = \tau_2}\ \mathrm{UFP} \quad .$$

Recursive Types

Recursive Types μTp:

$$\tau ::= \alpha \mid \bot \mid \top \mid \tau_1 \to \tau_2 \mid \mu\alpha.\,\tau_0 \quad (\alpha \in TVar).$$

Recursive Types in canonical form can-μTp:

$$\tau ::= \alpha \mid \bot \mid \top \mid \tau_1 \to \tau_2 \mid \mu\alpha.\,\underbrace{\tau_1 \to \tau_2}_{\text{where } \alpha \in \mathrm{fv}(\tau_1 \to \tau_2)} \quad (\alpha \in TVar).$$

Definition 1. (Recursive Type Equality $=_\mu$ (Strong Recursive Type Equivalence))

For all $\tau, \sigma \in \mu Tp$:

$$\tau =_\mu \sigma \;:\Longleftrightarrow\; \mathbf{Tree}(\tau) = \mathbf{Tree}(\sigma)\,.$$

$\boxtimes$

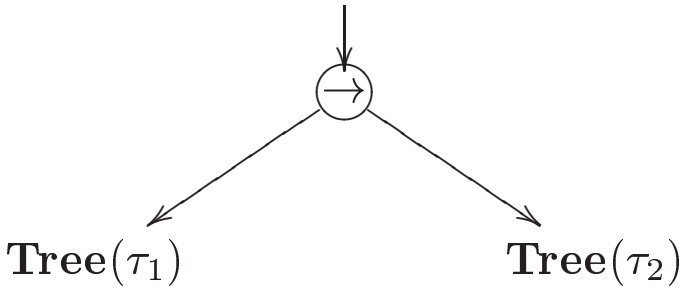
Definition 2. (**Tree-Unfolding of Recursive Types in canonical Form**)

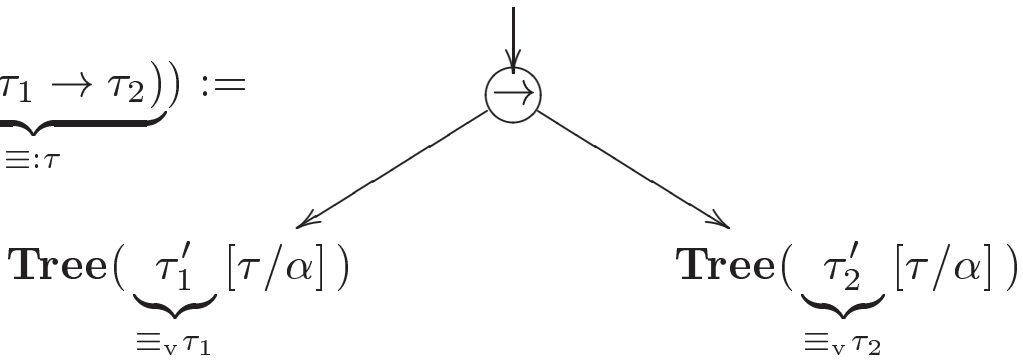
The function **Tree**

$$\mathbf{Tree}: \text{ can-}\mu Tp \longrightarrow \text{Labelled Binary Trees}$$
$$\tau \longmapsto \mathbf{Tree}(\tau)$$

which describes the *tree-unfolding* of recursive types in canonical form is defined by:

$\mathbf{Tree}(\bot) :=$; $\mathbf{Tree}(\top) :=$; $\mathbf{Tree}(\alpha) :=$;

$\mathbf{Tree}(\tau_1 \to \tau_2) :=$  ;

$\mathbf{Tree}(\underbrace{\mu\alpha.\,(\tau_1 \to \tau_2)}_{\equiv:\tau}) :=$ 

(where τ_1' and τ_2' are appropriate variants of τ_1 and τ_2 respectively to make the above two substitutions possible) .

$\boxtimes$

Definition 3. (Type-Trees $TpTrees$ = = Labelled Binary Trees).

A *type-tree* t is a partial function

$$t \,:\, \{0,1\}^* \longrightarrow \{\bot, \top, \rightarrow\} \cup TVar$$

with domain $\mathcal{D}(t) := \mathrm{dom}(t)$ s.th.

(i) $\mathcal{D}(t) \neq \emptyset$ and $\mathcal{D}(t)$ is prefix-closed, i.e.

$$\big(\forall\, p, p_1, p_2 \in \{0,1\}^*\big) \\ \big[\, p \equiv p_1 p_2 \;\wedge\; p \in \mathcal{D}(t) \;\Rightarrow\; p_1 \in \mathcal{D}(t) \,\big] \,.$$

(ii) $(\forall\, p \in \{0,1\}^*) \Big[\big(t(p) \equiv \rightarrow \;\Rightarrow\; p0, p1 \in \mathcal{D}(t) \big) \wedge$
$\wedge \big(t(p) \in \{\bot, \top\} \cup TVar \;\Rightarrow\; p0, p1 \notin \mathcal{D}(t) \big) \Big] \,.$

The set of all type-trees is denoted by $TpTrees$.

Definition 4 (Tree-Unfolding of Recursive Types).

The function

$$\begin{aligned}\mathbf{Tree} : \mu Tp &\longrightarrow TpTrees \\ \tau &\longmapsto \mathbf{Tree}(\tau) : \{0,1\}^* \to \{\bot, \top, \to\} \cup TVar \\ & \qquad p \longmapsto \mathbf{Tree}(\tau)(p)\end{aligned}$$

which to every recursive type $\tau \in \mu Tp$ assigns its *tree-unfolding* $\mathbf{Tree}(\tau)$ is defined as follows:

For every $\tau \in \mu Tp$ and every (path) $p \in \{0,1\}^*$ the symbol $\mathbf{Tree}(\tau)(p)$ is defined by induction on $|p|$ together with a subinduction on the number of leading μ-bindings in τ according to the following clauses:

(1) If $\tau \equiv \bot$ or $\tau \equiv \top$ or $\tau \equiv \alpha$ (for some $\alpha \in TVar$) , then

$$\mathbf{Tree}(\tau)(\epsilon) := \tau\,, \quad \mathbf{Tree}(\tau)(p) := \uparrow \text{ (for all } p \neq \epsilon).$$

(2) If $\tau \equiv \tau_1 \to \tau_2$ and $p = i\,p_0$ (for $i = 0, 1$), then

$$\mathbf{Tree}(\tau)(\epsilon) := \to\,, \quad \mathbf{Tree}(\tau)(p) := \mathbf{Tree}(\tau_i)(p_0) \text{ (for } i = 0, 1).$$

(3) If $\tau \equiv \mu\alpha_1 \ldots \alpha_n.\, \alpha$ with $n \in \mathbb{N}$, $\alpha \not\equiv \alpha_1, \ldots, \alpha_n$, then

$$\mathbf{Tree}(\tau)(\epsilon) := \alpha\,, \quad \mathbf{Tree}(\tau)(p) := \uparrow \text{ (for all } p \neq \epsilon).$$

(4) If $\tau \equiv \mu\alpha_1 \ldots \alpha_n.\, \alpha_i$ for $n \in \mathbb{N}$ and $i \in \{1, \ldots, n\}$, then

$$\mathbf{Tree}(\tau)(\epsilon) := \bot\,, \quad \mathbf{Tree}(\tau)(p) := \uparrow \text{ (for all } p \neq \epsilon).$$

(5) If $\tau \equiv \mu\alpha_1.\, \tau_0 \equiv \mu\alpha_1 \ldots \alpha_n.\, (\rho_1 \to \rho_2)$ with $n \in \mathbb{N}$, then

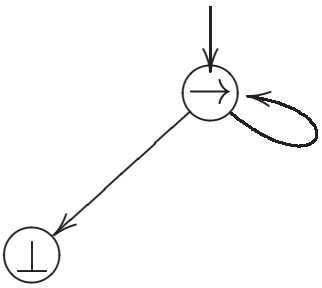
$$\mathbf{Tree}(\tau)(p) := \mathbf{Tree}(\tau_0'[\tau/\alpha_0])(p)$$
(for some variant τ_0' of τ_0). $\boxtimes$

Example 1 (Tree-Unfolding).

The recursive type

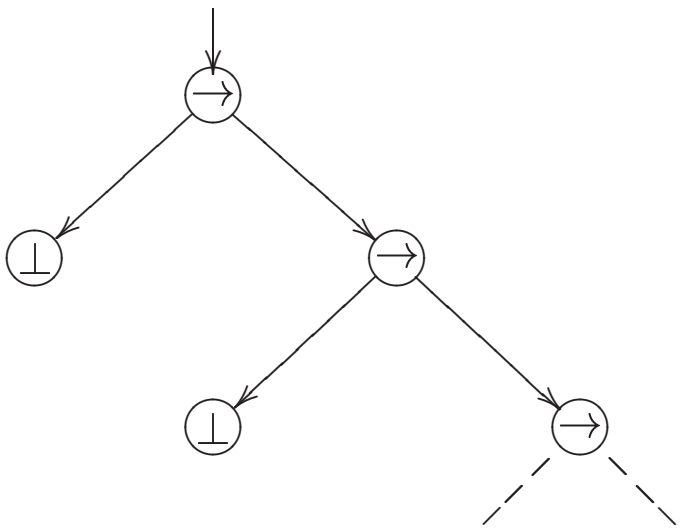
$$\tau := \mu\alpha.\,(\bot \to \alpha)$$

corresponds to the *cyclic term-graph*



and has the tree-unfolding

Tree(τ)=



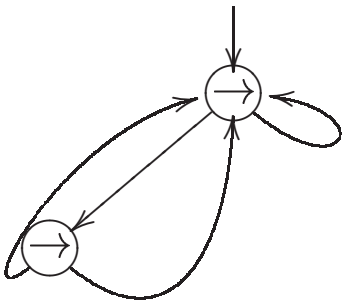
■

Example 2 (Tree-Unfolding).

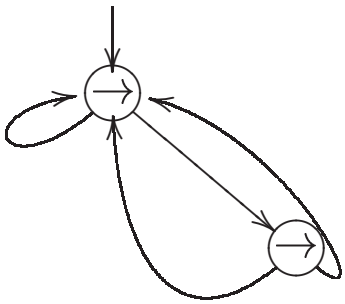
The recursive types

$$\tau := \mu\alpha.\,((\alpha \to \alpha) \to \alpha) \qquad \text{and}$$
$$\sigma := \mu\alpha.\,(\alpha \to (\alpha \to \alpha))$$

correspond respectively to the different cyclic term-graphs

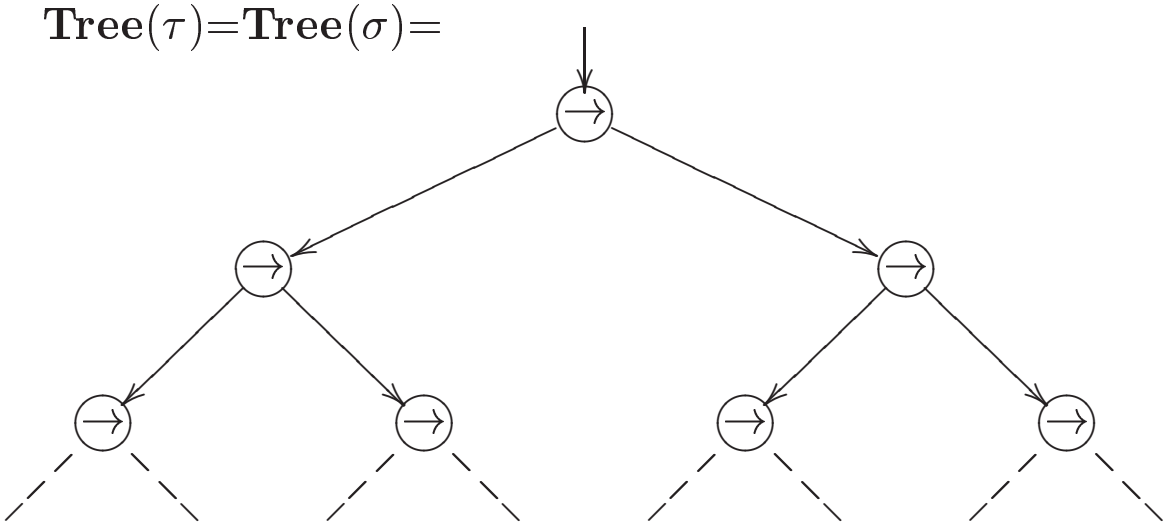


and



but possess the same tree-unfolding

Tree(τ)=**Tree**(σ)=



and hence $\tau =_\mu \sigma$ holds, i.e. recursive type equality holds between τ and σ.

■

Definition 5 (Syntactic Subterms of Recursive Types). A recursive type τ is said to be a *syntactic subterm* (or just a *subterm*) of a recursive type σ if and only if the formula $\tau \sqsubseteq \sigma$ is derivable in the following axiom-system **SUBT**:

Axioms:

$$\text{(REFL)}\ \frac{}{\tau \sqsubseteq \tau}$$

Rules:

$$\frac{\tau \sqsubseteq \sigma}{\tau \sqsubseteq \sigma'}\ \text{VAR (if } \sigma' \equiv_{\text{v}} \sigma\text{)}$$

$$\frac{\tau \sqsubseteq \sigma_1}{\tau \sqsubseteq \sigma_1 \to \sigma_2}\ \text{ARROW}_l \qquad \frac{\tau \sqsubseteq \sigma_2}{\tau \sqsubseteq \sigma_1 \to \sigma_2}\ \text{ARROW}_r$$

$$\frac{\tau \sqsubseteq \sigma_0[\mu\beta.\,\sigma_0/\beta]}{\tau \sqsubseteq \mu\beta.\,\sigma_0}\ \text{FOLD}_l$$

⊠

Lemma 1. *For every recursive type $\tau \in \mu Tp$ the number of its subterms is finite if in this count it is not distinguished between different variants of a recursive type.*

More precisely, for every recursive type $\tau \in \mu Tp$ there exists a finite set $A = \{\tau_1, \ldots, \tau_n\}$ (with $n \in \mathbb{N}$) of recursive types $\tau_1, \ldots, \tau_n \in \mu Tp$, such that every syntactic subterm of τ is a variant of a member of A.

Consequence. *The tree-unfolding* **Tree**(τ) *of every recursive type $\tau \in \mu Tp$ is a regular tree.*

Definition 6. (The axiom-system $\mathbf{AC}^{=}$ for recursive type equality $=_\mu$)

Axioms:

(REFL) $\overline{\tau = \tau}$ (VAR) $\overline{\tau = \tau'}$ (if $\tau \equiv_{\mathrm{v}} \tau'$)

$(\mu - \bot)$ $\overline{\mu\alpha.\,\alpha = \bot}$ (FOLD/UNFOLD) $\overline{\mu\alpha.\,\tau = \tau[\mu\alpha.\,\tau/\alpha]}$

Rules:

$$\frac{\sigma = \tau}{\tau = \sigma}\ \text{SYMM} \qquad \frac{\tau_1 = \sigma_1 \qquad \tau_2 = \sigma_2}{\tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}\ \text{ARROW}$$

$$\frac{\tau = \sigma}{\mu\alpha.\,\tau = \mu\alpha.\,\sigma}\ \mu\text{-COMPAT} \qquad \frac{\tau = \rho \qquad \rho = \sigma}{\tau = \sigma}\ \text{TRANS}$$

$$\frac{\tau_1 = \tau[\tau_1/\alpha] \qquad \tau_2 = \tau[\tau_2/\alpha]}{\tau_1 = \tau_2}\ \text{UFP} \quad (\text{if } \alpha \downarrow \tau)$$

(where $\alpha \downarrow \tau$ means that τ is *contractive* w.r.t. α). ⊠

Theorem 2 (R. Amadio, L. Cardelli, '93). *For all recursive types $\tau, \sigma \in \mu Tp$:*

$$\tau =_\mu \sigma \iff \vdash_{\mathbf{AC}^{=}} \tau = \sigma\ .$$

Notation. The notation $\alpha \downarrow \tau$ (α is *contractive* in τ) carries the meaning:

$$\Big[\, \alpha \notin \mathrm{fv}(\tau) \quad \text{or} \quad \tau \equiv \mu\alpha_1 \ldots \alpha_n.\, (\rho_1 \to \rho_2) \\ (\text{for some } n \in \mathbb{N}_0 \text{ and } \rho_1, \rho_2 \in \mu Tp \\ \text{and variables } \alpha_1 \ldots \alpha_n \in TVar) \,\Big] .$$

⊠

Lemma 3. *For all $\tau \in \mu Tp$ and variables α:*

$$\alpha \not\downarrow \tau \iff \left[\begin{array}{l} \tau \equiv \mu\alpha_1 \ldots \alpha_n.\, \alpha \ \textit{for some } n \in \mathbb{N}_0 \\ \textit{and variables } \alpha_1, \ldots, \alpha_n \textit{ distinct from } \alpha. \end{array} \right] .$$

Definition 7 (The variant-system $\mathbf{AC}_{\mathbf{1}}^{=}$ of $\mathbf{AC}^{=}$).

Axioms: All axioms of $\mathbf{AC}^{=}$ *plus*

$$(\mu - \bot)' \quad \overline{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha = \bot}$$

Rules: All rules of $\mathbf{AC}^{=}$ *except* μ-COMPAT. ⊠

The axioms $(\mu - \bot)'$ are "more general" than the axioms $(\mu - \bot)$: $\overline{\mu\alpha.\, \alpha = \bot}$ of $\mathbf{AC}^{=}$. In particular: Every equality

$$\mu\alpha.\, \tau_0 = \bot \quad (\text{if } \alpha \not\downarrow \tau_0)$$

(with arbitrary variables α and $\tau_0 \in \mu Tp$) is an axiom of the sort $(\mu - \bot)'$ of $\mathbf{AC}_{\mathbf{1}}^{=}$.

Frequently used derived rules of $\mathrm{AC}^{=}$:

$$\frac{\tau_0[\mu\alpha.\,\tau_0/\alpha] = \sigma}{\mu\alpha.\,\tau_0 = \sigma}\ \mathrm{FOLD}_l \qquad \frac{\tau = \sigma_0[\mu\beta.\,\sigma_0/\beta]}{\tau = \mu\beta.\,\sigma_0}\ \mathrm{FOLD}_r$$

$$\frac{\tau = \sigma}{\tau' = \sigma}\ \mathrm{VAR}_l \qquad \frac{\tau = \sigma}{\tau = \sigma'}\ \mathrm{VAR}_r \qquad \frac{\tau = \sigma}{\tau' = \sigma'}\ \mathrm{VAR}$$

(whenever it holds that $\tau' \equiv_{\mathrm{v}} \tau$ and $\sigma' \equiv_{\mathrm{v}} \sigma$)

$$\frac{\mu\alpha.\,\tau_0 = \sigma}{\tau_0[\mu\alpha.\,\tau_0/\alpha] = \sigma}\ \mathrm{UNFOLD}_l \qquad \frac{\tau = \mu\beta.\,\sigma_0}{\tau = \sigma_0[\mu\beta.\,\sigma_0/\beta]}\ \mathrm{UNFOLD}_r$$

$$\frac{\bot = \sigma}{\mu\alpha\alpha_1 \ldots \alpha_n.\,\alpha = \sigma}\ (\mu - \bot)^{\mathrm{inf}}_l \qquad +\ (\mu - \bot)^{\mathrm{inf}}_r$$

$$(\mu - \bot)^{\mathrm{der}}_l\ + \qquad \frac{\tau = \mu\beta\beta_1 \ldots \beta_n.\,\beta}{\tau = \bot}\ (\mu - \bot)^{\mathrm{der}}_r$$

$$\frac{\tau_1 = \tau[\tau_1/\alpha]}{\tau_1 = \mu\alpha.\,\tau}\ \mathrm{CONTRACT}\ \ (\text{if } \alpha \downarrow \tau)$$

Definition 8. (The axiom-system HB$^=$ for recursive type equality $=_\mu$)

Axioms/Marked assumptions:

$$\text{(REFL)}\ \frac{}{\tau = \tau} \qquad (\mu - \bot)'\ \frac{}{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha = \bot}$$

$$\text{(VAR)}\ \frac{}{\tau = \tau'}\ (\text{if } \tau \equiv_\text{v} \tau') \qquad \text{(FOLD/UNFOLD)}\ \frac{}{\mu\alpha.\, \tau = \tau[\mu\alpha.\, \tau/\alpha]}$$

$$\text{(HYP)}\quad (\tau = \sigma)^x$$

Rules:

$$\frac{\sigma = \tau}{\tau = \sigma}\ \text{SYMM} \qquad \frac{\tau = \rho \qquad \rho = \sigma}{\tau = \sigma}\ \text{TRANS}$$

$$\frac{\begin{array}{cc} \langle \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2 \rangle^x & \langle \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2 \rangle^x \\ \mathcal{D}_1 & \mathcal{D}_2 \\ \tau_1 = \sigma_1 & \tau_2 = \sigma_2 \end{array}}{\tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}\ (\text{ARROW/FIX})_x$$

⊠

Theorem 4 (M. Brandt (and F. Henglein), '98).

For all recursive types $\tau, \sigma \in \mu Tp$:

$$\tau =_\mu \sigma \iff \vdash_{\mathbf{HB}^=} \tau = \sigma\,.$$

Definition 9. (The syntactic-matching inference-system $\mathbf{SM}^{=}$ for consistency-checks concerning recursive type equality $=_\mu$)

Axioms:

$$\text{(REFL)}\ \frac{}{\tau = \tau} \qquad (\mu - \bot)'\ \frac{}{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha = \bot}$$

$$\text{(VAR)}\ \frac{}{\tau = \tau'}\ (\text{if } \tau \equiv_{\mathrm{v}} \tau') \qquad \text{(FOLD/UNFOLD)}\ \frac{}{\mu\alpha.\, \tau = \tau[\mu\alpha.\, \tau/\alpha]}$$

Rules:

$$\frac{\sigma = \tau}{\tau = \sigma}\ \text{SYMM} \qquad \frac{\tau = \rho \qquad \rho = \sigma}{\tau = \sigma}\ \text{TRANS}$$

$$\frac{\tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}{\tau_i = \sigma_i}\ \text{DECOMP}\ \ (i = 1, 2)$$

⊠

Definition 10. For all recursive types $\tau, \sigma \in \mu Tp$:

$\tau = \sigma$ is $\mathbf{SM}^{=}$-*consistent* : $\iff$

$\iff$ For all $\chi_1, \chi_2 \in \mu Tp$:

$$\tau = \sigma \vdash_{\mathbf{SM}^{=}} \chi_1 = \chi_2 \implies \mathcal{L}'(\chi_1) \equiv \mathcal{L}'(\chi_2)\,.$$

Hereby for all $\tilde{\tau} \in \mu Tp$ the notation $\mathcal{L}'(\tilde{\tau})$ designates a slight variant of the *leading symbol* $\mathcal{L}(\tilde{\tau})$ of $\tilde{\tau}$. ⊠

Theorem 5 ((∼) Z.M. Ariola, J.W. Klop, '95). *For all recursive types $\tau, \sigma \in \mu Tp$:*

$$\tau =_\mu \sigma \iff \tau = \sigma \text{ is } \mathbf{SM}^{=}\text{-consistent}.$$

Definition 11 (Leading Symbol of a recursive type).
The partial function

$$\begin{aligned} \mathcal{L} : \mu Tp &\longrightarrow \{\bot, \top, \to\} \cup \{\alpha : \alpha \in TVar\} \ , \\ \tau &\longmapsto \mathcal{L}(\tau) \end{aligned}$$

which to every recursive type τ assigns its *leading symbol* $\mathcal{L}(\tau)$ (if it exists), is defined by the following clauses:

$$\begin{aligned} &\mathcal{L}(\bot) :\equiv \bot \ , \ \mathcal{L}(\top) :\equiv \top \ , \\ &\mathcal{L}(\alpha) :\equiv \alpha \quad (\text{for } \alpha \in TVar) \ , \\ &\mathcal{L}(\tau_1 \to \tau_2) :\equiv \to \ , \\ &\mathcal{L}(\mu\alpha_1 \ldots \alpha_n . \, \alpha) :\equiv \alpha \quad (\text{if } \alpha \not\equiv \alpha_i \text{ for } 1 \leq i \leq n) \ , \\ &\mathcal{L}(\mu\alpha_1 \ldots \alpha_n . \, (\rho_1 \to \rho_2)) :\equiv \to \ . \end{aligned}$$

Only in the (single remaining) situation, that

$$\tau \equiv \mu\alpha_1 \ldots \alpha_n . \, \alpha_i \quad (\text{for } \ n \in \mathbb{N} \text{ and } \ 1 \leq i \leq n \,) \ ,$$

where $n \in \mathbb{N}$, $\mathcal{L}(\tau)$ is undefined. — The *total* function

$$\begin{aligned} \mathcal{L}' : \mu Tp &\longrightarrow \{\bot, \top, \to\} \cup \{\alpha : \alpha \in TVar\} \ , \\ \tau &\longmapsto \mathcal{L}'(\tau) \end{aligned}$$

which is a slight variant of $\mathcal{L}$ and is defined by the clauses in the definition of $\mathcal{L}$ and by the additional clause

$$\mathcal{L}'(\mu\alpha_1 \ldots \alpha_n . \, \alpha_i) :\equiv \bot \quad (\text{for } 1 \leq i \leq n) \ .$$

⊠

Remark. For all $\tau \in \mu Tp$ it holds: $\mathcal{L}'(\tau) = \mathbf{Tree}(\tau)(\epsilon)$.

Definition 12. **(The axiom-system $\mathbf{HB}_{\mathbf{0}}^{=}$ for recursive type equality $=_\mu$)**

Axs./M. Assms.: (REFL) $\dfrac{}{\tau = \tau}$ (HYP) $(\tau = \sigma)^x$

Rules:

$$\frac{\tau = \sigma}{\tau' = \sigma'}\ \text{VAR} \qquad \frac{\bot = \sigma}{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha = \sigma}\ (\mu - \bot)_l^{\text{inf}} + (\mu - \bot)_r^{\text{inf}}$$

$$\frac{\tau_0[\mu\alpha.\, \tau_0/\alpha] = \sigma}{\mu\alpha.\, \tau_0 = \sigma}\ \text{FOLD}_l \qquad \frac{\tau = \sigma_0[\mu\beta.\, \sigma_0/\beta]}{\tau = \mu\beta.\, \sigma_0}\ \text{FOLD}_r$$

$$\frac{\begin{array}{cc}\langle \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2 \rangle^x & \langle \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2 \rangle^x \\ \mathcal{D}_1 & \mathcal{D}_2 \\ \tau_1 = \sigma_1 & \tau_2 = \sigma_2\end{array}}{\tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}\ (\text{ARROW/FIX})_x$$

$\boxtimes$

Definition 13. **(The axiom-system $\mathbf{SM}_{\mathbf{0}}^{=}$ for recursive type equality $=_\mu$)**

Rules:

$$\frac{\tau = \sigma}{\tau' = \sigma'}\ \text{VAR} \qquad \frac{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha = \sigma}{\bot = \sigma}\ (\mu - \bot)_l^{\text{der}} + (\mu - \bot)_r^{\text{der}}$$

$$\frac{\mu\alpha.\, \tau_0 = \sigma}{\tau_0[\mu\alpha.\, \tau_0/\alpha] = \sigma}\ \text{UNFOLD}_l \qquad \frac{\tau = \mu\beta.\, \sigma_0}{\tau = \sigma_0[\mu\beta.\, \sigma_0/\beta]}\ \text{UNFOLD}_r$$

$$\frac{\tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}{\tau_i = \sigma_i}\ \text{DECOMP} \quad (i = 1, 2)$$

$\boxtimes$

Example 3 ("Consistency Check" in $\mathrm{SM}_0^{=}$).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{(\tau = \sigma)^{y}}\qquad \dfrac{\dfrac{}{\tau = \sigma \to \sigma}}{(\tau \to \tau) \to \tau = \sigma \to \sigma}}{(\tau \to \tau = \sigma)^{x}}}{\tau \to \tau = \sigma \to (\sigma \to \sigma)}}{(\tau \to \tau) \to \tau = \sigma \to (\sigma \to \sigma)}\qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\tau \to \tau = \sigma}}{\tau \to \tau = \sigma \to (\sigma \to \sigma)}}{(\tau = \sigma)^{y}}\qquad \dfrac{}{(\tau = \sigma)^{y}}}{(\tau \to \tau) \to \tau = \sigma \to \sigma}}{(\tau = \sigma \to \sigma)^{z}}}{\overbrace{\left(\overbrace{\mu\alpha.\,((\alpha \to \alpha) \to \alpha)}^{\equiv:\,\tau} = \overbrace{\mu\alpha.\,(\alpha \to (\alpha \to \alpha))}^{\equiv:\,\sigma}\right)}^{y}}$$

■

Example 4 (Proof in $\mathrm{HB}_0^{=}$).

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\dots)^{y}}{\tau=\sigma}\quad\dfrac{\dfrac{(\dots)^{x}}{\tau\to\tau=\sigma}\quad\dfrac{(\dots)^{y}}{\tau=\sigma}}{(\tau\to\tau)\to\tau=\sigma\to\sigma}}{\tau=\sigma\to\sigma}\,\mathrm{x}}{\tau\to\tau=\sigma\to(\sigma\to\sigma)}}{\tau\to\tau=\sigma}\qquad\dfrac{\dfrac{\dfrac{\dfrac{(\dots)^{y}}{\tau=\sigma}\quad\dfrac{(\dots)^{z}}{\tau=\sigma\to\sigma}}{\tau\to\tau=\sigma\to(\sigma\to\sigma)}\quad\dfrac{(\dots)^{y}}{\tau=\sigma}}{\tau\to\tau=\sigma}\,\mathrm{z}}{(\tau\to\tau)\to\tau=\sigma\to\sigma}}{\tau=\sigma\to\sigma}\,\mathrm{y}$$

$$\dfrac{(\tau\to\tau)\to\tau=\sigma\to(\sigma\to\sigma)}{\underbrace{\mu\alpha.\,((\alpha\to\alpha)\to\alpha)}_{\equiv:\,\tau}=\underbrace{\mu\alpha.\,(\alpha\to(\alpha\to\alpha))}_{\equiv:\,\sigma}}$$

■

Definition 14 (Finite Trees of Consequences in $\mathbf{SM}_{\mathbf{0}}^{=}$).

A *finite tree of consequences* (a *f.t.o.c.*) $\mathcal{C}$ of (or from) an equality $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ is the assemblage of finitely many derivations of finite length from $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ to the form of a derivation-tree.

More precisely, a finite tree of consequences $\mathcal{C}$ of an equality $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ is an object, which can be formed by a finite number of applications of the following three generation rules:

(i) $\mathcal{C}$ is the recursive type equality $\tau = \sigma$.

(ii) $\mathcal{C}$ is of the form

$$\dfrac{\begin{array}{c}\tau = \sigma \\ \mathcal{C}_0 \\ (\chi_{10} = \chi_{20})\end{array}}{\chi_1 = \chi_2}\, R \quad ,$$

given that

- $\mathcal{C}_0$ is a finite tree of consequences of $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ of the form

$$\begin{array}{c}\tau = \sigma \\ \mathcal{C}_0 \\ (\chi_{10} = \chi_{20})\end{array} \ .$$

- R denotes an application of one of the rules VAR, $\mathrm{UNFOLD}_{\mathrm{l/r}}$ or $(\mu - \bot)^{\mathrm{inf}}_{\mathrm{l/r}}$ of $\mathbf{SM}_{\mathbf{0}}^{=}$.

(iii) $\mathcal{C}$ is of the form

$$\dfrac{\begin{array}{c}\tau = \sigma \\ \mathcal{C}_0 \\ (\chi_1 \to \chi_2 = \chi_1' \to \chi_2')\end{array}}{\chi_1 = \chi_1' \qquad \chi_2 = \chi_2'}\ \text{DECOMP},$$

given that

- $\mathcal{C}_0$ is a finite tree of consequences of $\tau = \sigma$ of the form

$$\begin{array}{c}\tau = \sigma \\ \mathcal{C}_0 \\ (\chi_1 \to \chi_2 = \chi_1' \to \chi_2')\end{array}\ .$$

  Hereby the part

$$\dfrac{\chi_1 \to \chi_2 = \chi_1' \to \chi_2'}{\chi_1 = \chi_1' \qquad \chi_2 = \chi_2'}\ \text{DECOMP}$$

  is called a *branching*.

$\boxtimes$

Definition 15. (Consistency Checks for equalities between recursive types).

A *consistency check* for an equality $\tau = \sigma$ between recursive types $\sigma, \tau \in \mu Tp$ in $\mathbf{SM}_{\mathbf{0}}^{=\!=}$ is a finite tree of consequences $\mathcal{C}$ of $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=\!=}$, such that for all occurrences of equalities $\chi_1 = \chi_2$ as leaves at the bottom of $\mathcal{C}$, i.e. such that $\mathcal{C}$ can be written as $\begin{array}{c}\tau = \sigma\\ \mathcal{C}\\ (\chi_1 = \chi_2)\end{array}$, one of the following two conditions is true:

(i) $\chi_1 \equiv \chi_2$ holds;

(ii) In the thread in $\mathcal{C}$ from its root $\tau = \sigma$ downwards to the considered occurrence of $\chi_1 = \chi_2$ the equality $\chi_1 = \chi_2$ is encountered at least once before the considered leaf of $\mathcal{C}$ is reached; i.e. $\mathcal{C}$ with the considered leaf-occurrence can be written as

$$\begin{array}{c}\tau = \sigma\\ \mathcal{C}_0\\ (\chi_1 = \chi_2)\\ \mathcal{C}_1\\ (\chi_1 = \chi_2)\end{array}$$

for a f.t.o.c. $\mathcal{C}_0$ from $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=\!=}$ and a f.t.o.c. $\mathcal{C}_1$ of depth ≥ 1 of $\chi_1 = \chi_2$ in $\mathbf{SM}_{\mathbf{0}}^{=\!=}$. *Furthermore* the thread in $\mathcal{C}_1$ from its root down to the highlighted occurrence of $\chi_1 = \chi_2$ in $\mathcal{C}_1$ *crosses a branching* DECOMP *at least once*.

⊠

Lemma 6. *For all recursive types* $\tau, \sigma \in \mu Tp$ *:*

$\tau = \sigma$ *is* $\mathbf{SM}_{\mathbf{0}}^{=}$*-consistent* $\iff$

$\iff$ *There exists a consistency-check* $\mathcal{C}$ *of* $\tau = \sigma$ *in* $\mathbf{SM}_{\mathbf{0}}^{=}$.

Definition 16. (Reflection of $\mathbf{HB}_{\mathbf{0}}^{=}$-derivations and of finite trees of consequences in $\mathbf{SM}_{\mathbf{0}}^{=}$ at a horizontal line)

Let τ, σ be arbitrary recursive types.

(i) For every finite tree of consequences $\mathcal{C}$ of the recursive type equality $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ its *reflection at a horizontal line* (or just: its *reflection*) $\mathcal{D}(\mathcal{C})$ is defined as . . . .

(ii) For every derivation $\mathcal{D}$ of $\tau = \sigma$ in $\mathbf{HB}_{\mathbf{0}}^{=}$, in which open assumption-classes may occur, its *reflection at a horizontal line* (or just: its *reflection*) $\mathcal{C}(\mathcal{D})$ is defined as . . . .

Lemma 7. *Let τ, σ be arbitrary recursive types.*

(i) For every finite tree of consequences $\mathcal{C}$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ from the equality $\tau = \sigma$ between recursive types $\tau, \sigma \in \mu Tp$ its reflection $\mathcal{D}(\mathcal{C})$ at a horizontal line is a derivation of $\tau = \sigma$ in $\mathbf{HB}_{\mathbf{0}}^{=}$, in which as (labels for) leaves at the top of $\mathcal{D}(\mathcal{C})$ besides axioms (REFL) also unmarked assumptions may occur.

Furthermore, there is an obvious bijective correspondence between the unmarked assumptions at the top of $\mathcal{D}(\mathcal{C})$, which are not axioms (REFL), and those equalities between recursive types at the bottom of $\mathcal{C}$, which are not axioms (REFL).

(ii) For every derivation $\mathcal{D}$ of $\tau = \sigma$ in $\mathbf{HB}_{\mathbf{0}}^{=}$, which may contain open assumption-classes, its reflection $\mathcal{C}(\mathcal{D})$ at a horizontal line is a finite tree of consequences of $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$.

Theorem 8. [Idea by J.W. Klop]
(Duality between Derivations in $\mathbf{HB}_{\mathbf{0}}^{=}$ and Consistency Checks in $\mathbf{SM}_{\mathbf{0}}^{=}$).

Let τ, σ be arbitrary recursive types.

(1) For every derivation $\mathcal{D}$ in $\mathbf{HB}_{\mathbf{0}}^{=}$ with conclusion $\tau = \sigma$ and without open assumption-classes its reflection $\mathcal{C}(\mathcal{D})$ is a consistency-check for $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$.

(2) For every consistency-check $\mathcal{C}$ of $\tau = \sigma$ in $\mathbf{SM}_{\mathbf{0}}^{=}$ its reflection $\mathcal{D}(\mathcal{C})$ can be extended to a $\mathbf{HB}_{\mathbf{0}}^{=}$-derivation $\mathcal{D}^*(\mathcal{C})$ with conclusion $\tau = \sigma$ and with no open assumption-classes.

[Extending $\mathcal{D}(\mathcal{C})$ to $\mathcal{D}^*(\mathcal{C})$ hereby involves only:

- *Extending the proof-tree of $\mathcal{D}(\mathcal{C})$ above some of the occuring unmarked assumptions with finite chains of applications of rules VAR, $FOLD_l$ or $FOLD_r$, chains of applications, which are already present in $\mathcal{D}(\mathcal{C})$ at least once below each respective unmarked assumption, and*
- *Marking the newly formed (still unmarked) assumptions of the extended derivation-tree with appropriate markers and binding these back to respective applications of ARROW deeper down in $\mathcal{D}(\mathcal{C})$, thereby also changing such applications of ARROW to applications of ARROW/FIX.]*

Notation. ((Cyclic) Term-Graphs, Bisimularity, Homomorphisms, Bisim.-Equiv.-Classes).

Let Σ be a signature.

(i) $\mathcal{TG}(\Sigma)$ denotes the *set of (cyclic) term-graphs over* Σ.
$\mathcal{TG}^{(\mathrm{vs})}(\Sigma)$ denotes the *set of (cyclic) term-graphs over* Σ *with only vertical sharing*.

(ii) $\underline{\leftrightarrows}$ denotes the bisimularity-relation on term-graphs in $\mathcal{TG}(\Sigma)$ or in $\mathcal{TG}^{(\mathrm{vs})}(\Sigma)$.

$\underline{\rightrightarrows}$ and its inverse $\underline{\leftleftarrows}$ denote homomorphism-(partial-)orderings on term-graphs in $\mathcal{TG}(\Sigma)$ or in $\mathcal{TG}^{(\mathrm{vs})}(\Sigma)$:

For all $g, h \in \mathcal{TG}(\Sigma)$:

$$g \underline{\rightrightarrows} h \;:\Longleftrightarrow\; \text{There exists a homomorphism from g to h.}$$

(iii) Bisimularity equivalence-classes in $\mathcal{TG}(\Sigma)$ and $\mathcal{TG}^{(\mathrm{vs})}(\Sigma)$:

For all $g_1 \in \mathcal{TG}(\Sigma)$ and $g_2 \in \mathcal{TG}^{(\mathrm{vs})}(\Sigma)$:

$$[g_1]_{\underline{\leftrightarrows}} := \{g' \in \mathcal{TG}(\Sigma) : g' \underline{\leftrightarrows} g_1\},$$
$$[g_2]^{(\mathrm{vs})}_{\underline{\leftrightarrows}} := \{g' \in \mathcal{TG}^{(\mathrm{vs})}(\Sigma) : g' \underline{\leftrightarrows} g_2\}.$$

$\boxtimes$

It is a well-known fact, that:

Theorem 9. *For all $g \in \mathcal{TG}(\Sigma)$: $([g]_{\underline{\leftrightarrow}}, \underline{\Leftarrow})$ is a complete lattice.*

But:

Theorem 10. *The following variant of the statement in Theorem 9 applying to term-graphs with only vertical sharing and restricted bisimularity-equivalence-classes w.r.t. only such term-graphs* **is not true** *for all signatures Σ:*

For all $g \in \mathcal{TG}^{(vs)}(\Sigma)$:

$([g]^{(vs)}_{\underline{\leftrightarrow}}, \underline{\Leftarrow})$ *is a complete lattice.*

This is because:

Lemma 11. *There exists a term-graph $g \in \mathcal{TG}(\{\rightarrow\})$ and term-graphs $g_1, g_2, g_3, g_4 \in \mathcal{TG}^{(vs)}(\{\rightarrow\})$, which are bisimular to g (i.e. $g_i \in [g]_{\underline{\leftrightarrow}}$ (for $i = 1, 2, 3, 4$)), for which on the one hand*

$$l.u.b.\,(g_1, g_2) = g = g.l.b.\,(g_3, g_4)$$

is true, but for which on the other hand both

$$l.u.b.\,(g_1, g_2) \quad \text{and} \quad g.l.b.\,(g_3, g_4)$$

do not exist in $\mathcal{TG}^{(vs)}(\{\rightarrow\})$.

Handwritten slide

Proof of Lemma 11.

(Example of recursive types, for which the g.l.b. does not exists as recursive type (as μ-expression), and also example of recursive types, for which the l.u.b. does not exist as recursive type (μ-expression)).

Theorem 12. *μ-COMPAT is a derived rule of* $\mathbf{AC}_{\mathbf{1}}^{=}$.

Sketch of Proof. The lemma follows from the proof of the following stronger statement:

There is an effective methode to transform every derivation $\mathcal{D}$ of the form

$$\dfrac{\begin{array}{c}\mathcal{D}_0\\ \tau_1 = \tau_2\end{array}}{\mu\alpha.\,\tau_1 = \mu\alpha.\,\tau_2}\ \mu\text{-COMPAT}\quad , \tag{4}$$

where $\mathcal{D}_0$ is an $\mathbf{AC}_{\mathbf{1}}^{=}$-derivation, into an $\mathbf{AC}_{\mathbf{1}}^{=}$-derivation $\mathcal{D}'$ with the same conclusion:

Case 1: $\alpha \not\in \tau_1$ and $\alpha \not\in \tau_2$.

Then by a previous Lemma $\tau_1 \equiv \mu\alpha_1 \ldots \alpha_{n_1}.\,\alpha$ and $\tau_2 \equiv \mu\alpha'_1 \ldots \alpha'_{n_2}.\,\alpha$ for some $n_1, n_2 \in \mathbb{N}_0$ and variables $\alpha_1, \ldots, \alpha_{n_1}, \alpha'_1 \ldots \alpha'_{n_2} \not\equiv \alpha$. Then $\mu\alpha.\,\tau_1 = \bot$ and $\mu\alpha.\,\tau_2 = \bot$ are axioms of the scheme $(\mu - \bot)'$ of $\mathbf{AC}_{\mathbf{1}}^{=}$ and hence the derivation $\mathcal{D}'$

$$\dfrac{\dfrac{(\mu-\bot)'}{\mu\alpha.\,\tau_1 = \bot} \quad \dfrac{\dfrac{(\mu-\bot)'}{\mu\alpha.\,\tau_2 = \bot}\ \text{SYMM}}{\bot = \mu\alpha.\,\tau_2}}{\mu\alpha.\,\tau_1 = \mu\alpha.\,\tau_2}\ \text{TRANS}$$

is an $\mathbf{AC}_{\mathbf{1}}^{=}$-derivation with the same conclusion as $\mathcal{D}$.

Case 2: $\alpha \downarrow \tau_1$ or $\alpha \downarrow \tau_2$.

Suppose $\alpha \downarrow \tau_2$ (the case $\alpha \downarrow \tau_1$ can be handled quite analogously). Then the derivation (4) can be transformed into the derivation $\tilde{\mathcal{D}}$ of the form

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathcal{D}_0}{\tau_1 = \tau_2}\,\mathrm{SUBST}_{(\cdot)[\mu\alpha.\,\tau_1'/\alpha]}}{\tau_1'[\mu\alpha.\,\tau_1'/\alpha] = \tau_2'[\mu\alpha.\,\tau_1'/\alpha]}\,\mathrm{FOLD}_l}{\mu\alpha.\,\tau_1' = \tau_2'[\mu\alpha.\,\tau_1'/\alpha]}\,\mathrm{CONTRACT}}{\mu\alpha.\,\tau_1' = \mu\alpha.\,\tau_2'}\,\mathrm{VAR}_l,\ \mathrm{VAR}_r}{\mu\alpha.\,\tau_1 = \mu\alpha.\,\tau_2}\ ,$$

(where τ_1' and τ_2' are *appropriate* variants of τ_1 and τ_2 respectively) which can effectively be transformed into an $\mathbf{AC}_\mathbf{1}^=$-derivation $\mathcal{D}$: This is due to the fact that all rules

$$\dfrac{\tilde{\tau_1} = \tilde{\tau_2}}{\tilde{\tau_1}'[\sigma/\tilde{\alpha}] = \tilde{\tau_2}'[\sigma/\tilde{\alpha}]}\,\mathrm{SUBST}_{(\cdot)[\sigma/\tilde{\alpha}]}$$

(for arbitrary $\tilde{\alpha} \in TVar$ and $\tilde{\tau_1}, \tilde{\tau_2}, \sigma \in \mu Tp$, s.th. $\tilde{\tau_1}' \equiv_\mathrm{v} \tilde{\tau_1}$ and $\tilde{\tau_2}' \equiv_\mathrm{v} \tilde{\tau_2}$) are derived rules of $\mathbf{AC}^=$ and $\mathbf{AC}_\mathbf{1}^=$, which can effectively be removed from $\mathbf{AC}^=$- or $\mathbf{AC}_\mathbf{1}^=$-derivations.

□

Corollary 13. $\mathbf{AC}^=$ *and* $\mathbf{AC}_\mathbf{1}^=$ *are equivalent theories.*

Transformation: $\mathbf{AC}^{=}$-deriv. $\longmapsto$ ($\longmapsto$ $\mathbf{AC}_{\mathbf{1}}^{=}$-deriv.) $\longmapsto$ $\mathbf{HB}^{=}$-deriv.

Theorem 14. *Every* $\mathbf{AC}^{=}$*-derivation* $\mathcal{D}$ *can be transformed effectively into a* $\mathbf{HB}^{=}$*-derivation* $\mathcal{D}'$ *with the same conclusion.*

Proof. (1) Effective elimination of all rules μ-COMPAT in $\mathcal{D}$ with the result of a derivation $\mathcal{D}_1$ in $\mathbf{AC}_{\mathbf{1}}^{=}$.

(2) "Replacing" applications of UFP in $\mathcal{D}_1$ by applications of CONTRACT: The result is then a derivation $\mathcal{D}_2$ in

$$(\mathbf{AC}_{\mathbf{1}}^{=}\backslash\text{UFP})) + \text{CONTRACT}$$

as well as in (what is the same system:)

$$(\ \mathbf{HB}^{=}\backslash(\text{ARROW/FIX})) + \text{CONTRACT}.$$

(3) Eliminating successively all applications of CONTRACT from $\mathcal{D}_2$ (using the statement of Lemma 15 below), thereby building a $\mathbf{HB}^{=}$-derivation $\mathcal{D}'$ with the same conclusion as $\mathcal{D}$.

□

Lemma 15. *Every derivation* $\mathcal{D}$ *of the form*

$$\frac{\begin{array}{c}\mathcal{D}_0\\ \tau_1 = \tau[\tau_1/\alpha]\end{array}}{\tau_1 = \mu\alpha.\,\tau}\ \mathit{CONTRACT} \tag{5}$$

where $\mathcal{D}_0$ *is a* $\mathbf{HB}^{=}$*-derivation, can effectively be replaced by a* $\mathbf{HB}^{=}$*-derivation* $\mathcal{D}'$ *with the same conclusion.*

Lemma 16. *For every $\tau \in \mu Tp$ and every variable α s.th. $\alpha \downarrow \tau$ and $\alpha \in fv(\tau)$, and for every $\sigma_1, \sigma_2 \in \mu Tp$, that are substitutible for α in τ, there exists a $\mathbf{HB}_{\mathbf{0}}^{=}$-derivation $\mathcal{D}_{\tau,\alpha;\sigma_1,\sigma_2}$ of the form*

$$\begin{array}{l} \left[\sigma_1 = \sigma_2\right]^x \\ \mathcal{D}_{\tau,\alpha;\sigma_1,\sigma_2} \\ \vdots \\ \tau^{(0)}[\sigma_1/\alpha] = \tau^{(0)}[\sigma_2/\alpha] \\ \vdots \\ \tau[\sigma_1/\alpha] = \tau[\sigma_2/\alpha] \end{array} \quad \begin{array}{l} \\ \\ ARR.\ or\ ARR./FIX \\ \\ (FOLD_l,\ FOLD_r)^* \end{array} \tag{6}$$

with the single open assumption-class $[\sigma_1 = \sigma_2]^x$ ($\tau^{(0)} \in \mu Tp$ and x an assumption-marker).

Idea of the Proof. For every $\tau \in \mu Tp$ and every variable α s.th. $\alpha \downarrow \tau$ and $\alpha \in \mathrm{fv}(\tau)$ a derivation $\mathcal{D}_{\tau,\alpha}$ in $\mathbf{HB}_{\mathbf{0}}^{=}$ of the form

$$\begin{array}{l} \left[\overline{\alpha = \alpha}\right] \\ \mathcal{D}_{\tau,\alpha} \\ \vdots \\ \tau^{(0)} = \tau^{(0)} \\ \vdots \\ \tau = \tau \end{array} \quad \begin{array}{l} \\ \\ \text{ARROW or ARROW/FIX} \\ \\ (\text{FOLD}_l, \text{FOLD}_r)^* \end{array}$$

without undischarged assumption-classes exists; hereby the displayed class of axioms $[\overline{\alpha = \alpha}]$ denotes all axioms of the form $\overline{\alpha = \alpha}$ (that occur as leaves at the top of the proof-tree) in $\mathcal{D}$.

For arbitrary $\sigma_1, \sigma_2 \in \mu Tp$, which are substitutible in τ for α, the substitution of τ for α in each recursive type occuring in $\mathcal{D}_{\tau,\alpha}$ then results in a derivation $\mathcal{D}_{\tau;\sigma_1,\sigma_2}$ of the desired form (6). □

Proof of Lemma 15. The statement $\alpha \downarrow \tau$ must hold in the derivation $\mathcal{D}$ in (5), since this is the side-condition required for every application of CONTRACT.

– Now, if $\alpha \notin \mathrm{fv}(\tau)$ holds, then CONTRACT can be replaced by an application of FOLD_r ; but if instead $\alpha \in \mathrm{fv}(\tau)$ holds, then $\mathcal{D}$ can be replaced by the following $\mathbf{HB}_{\mathbf{0}}^{=}$+TRANS-derivation

$$\dfrac{\dfrac{\mathcal{D}_0}{\tau_1 = \tau[\tau_1/\alpha]} \qquad \dfrac{\dfrac{\dfrac{\

Handwritten Slide

depicting the motivation for the Tranformation of a $\mathbf{HB}_{\mathbf{0}}^{=}$-into an $\mathbf{AC}^{=}$-derivation.

Definition 17. (The annotated version ann-$\mathbf{HB}_{\mathbf{0}}^{=}$ of the system $\mathbf{HB}_{\mathbf{0}}^{=}$).

Axioms/Marked Assumptions:

$$\text{(REFL)}\ \frac{}{\tau : \tau = \tau} \qquad \text{(HYP)}\ \left(\alpha : \tau = \sigma\right)^{\alpha}$$

Rules:

$$\frac{\chi : \tau = \sigma}{\chi : \tau' = \sigma'}\ \text{VAR}\quad (\text{where } \tau' \equiv_{\text{v}} \tau \text{ and } \sigma' \equiv_{\text{v}} \sigma)$$

$$\frac{\chi : \bot = \sigma}{\chi : \mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha = \sigma}\ (\mu - \bot)_l^{\text{inf}} \qquad \frac{\chi : \tau = \bot}{\chi : \tau = \mu\beta\beta_1 \ldots \beta_n.\, \beta}\ (\mu - \bot)_r^{\text{inf}}$$

$$\frac{\chi : \tau_0[\mu\alpha.\, \tau_0/\alpha] = \sigma}{\chi : \mu\alpha.\, \tau_0 = \sigma}\ \text{FOLD}_l \qquad \frac{\chi : \tau = \sigma_0[\mu\beta.\, \sigma_0/\beta]}{\chi : \tau = \mu\beta.\, \sigma_0}\ \text{FOLD}_r$$

$$(\text{if side-condition } \mathbf{S_1})\ \frac{\begin{array}{cc}\mathcal{D}_1 & \mathcal{D}_2\\ \chi_1 : \tau_1 = \sigma_1 & \chi_2 : \tau_2 = \sigma_2\end{array}}{\chi_1 \to \chi_2 : \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}\ \text{ARROW}$$

$$(\text{if side-condition } \mathbf{S_2})\ \frac{\begin{array}{cc}\langle\alpha : \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2\rangle^{\alpha} & \langle\alpha : \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2\rangle^{\alpha}\\ \mathcal{D}_1 & \mathcal{D}_2\\ \chi_1 : \tau_1 = \sigma_1 & \chi_2 : \tau_2 = \sigma_2\end{array}}{\mu\alpha.\, \chi_1 \to \chi_2 : \tau_1 \to \tau_2 = \sigma_1 \to \sigma_2}\ (\text{AR./F.})_{\alpha}$$

Side-condition $\mathbf{S_1}$:

(i) *All marked hypotheses in $\mathcal{D}_1$ or $\mathcal{D}_2$, that belong to different undischarged assumption-classes, are marked by different variables and (hence also) contain different variables as annotations*, or, reversely and more precisely:

For all variables $\alpha \in TVar$ and all still undischarged assumptions $(\,\alpha : \rho_1 \to \rho_2\,)^\alpha$ and $(\,\alpha : \chi_1 \to \chi_2\,)^\alpha$ in $\mathcal{D}_1$ or $\mathcal{D}_2$ it holds, that

$$\rho_1 \equiv \chi_1 \quad \text{and} \quad \rho_2 \equiv \chi_2\,.$$

(ii) *Variables used as annotations in undischarged assumptions in $\mathcal{D}_1$ or $\mathcal{D}_2$ do not also occur in axioms used in $\mathcal{D}_1$ or $\mathcal{D}_2$*, more precisely:

For all still undischarged marked assumptions $(\,\alpha : \rho \to \chi\,)^\alpha$ and for all axioms $\overline{\tau : \tau = \tau}$ in $\mathcal{D}_1$ or $\mathcal{D}_2$ it holds, that

$$\alpha \notin \mathrm{fv}(\tau)\,.$$

Side-condition $\mathbf{S_2}$:

(i) Side-condition $\mathbf{S_1}$.

(ii) $[\,\alpha : \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,]^\alpha$ is an inhabitated assumption-class in $\mathcal{D}_1$ or in $\mathcal{D}_2$.

(iii) For every inhabitated assumption-class $[\,\beta : \rho \leq \chi\,]^\beta$ in $\mathcal{D}_1$ or $\mathcal{D}_2$, which is different from the assumption-class $[\,\alpha : \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,]^\alpha$, it holds, that

$$\alpha \notin \mathrm{fv}(\rho) \cup \mathrm{fv}(\chi)\,.$$

Transformation: ($\mathbf{HB^{=}}$-deriv. $\longmapsto$)

$\mathbf{HB_0^{=}}$-deriv. $\longmapsto$ **ann-**$\mathbf{HB_0^{=}}$-deriv. $\longmapsto$ $\mathbf{AC^{=}}$-deriv.

Relevant properties connected with **ann-**$\mathbf{HB_0^{=}}$ and steps in these transformations:

(1) Every $\mathbf{HB_0^{=}}$-derivation $\mathcal{D}$ can effectively be annotated to an **ann-**$\mathbf{HB_0^{=}}$-derivation $\hat{\mathcal{D}}$:

$$\begin{array}{c}\{\,[\,\tau_i = \sigma_i\,]^{x_i}\,\}_{i=1,\ldots,n} \\ \mathcal{D} \\ \tau = \sigma\end{array} \quad \longmapsto \quad \begin{array}{c}\{\,[\,\alpha'_i \,:\, \tau_i = \sigma_i\,]^{\alpha'_i}\,\}_{i=1,\ldots,n} \\ \hat{\mathcal{D}} \\ \chi \,:\, \tau = \sigma\end{array} \quad .$$

(2) Reversely, every **ann-**$\mathbf{HB_0^{=}}$-derivation $\mathcal{D}$ can be transformed (by basically stripping it of its annotations) into a $\mathbf{HB_0^{=}}$-derivation $\check{\mathcal{D}}$.

(This item is not needed for the transformations.)

(3) For every **ann-**$\mathbf{HB_0^{=}}$-derivation

$$\begin{array}{c}\{\,[\,\alpha_i \,:\, \tau_i = \sigma_i\,]^{\alpha_i}\,\}_{i=1,\ldots,n} \\ \mathcal{D} \\ \chi \,:\, \tau = \sigma\end{array} \quad ,$$

where the displayed family of assumption-classes comprises just all undischarged, inhabitated assumption-classes in $\mathcal{D}$, two $\mathbf{AC^{=}}$-derivations

$$\begin{array}{c}\mathcal{D}_1^* \\ \tau = \chi[\tau_1/\alpha_1, \ldots, \tau_n/\alpha_n]\end{array} \quad \text{and} \quad \begin{array}{c}\mathcal{D}_2^* \\ \sigma = \chi[\sigma_1/\alpha_1, \ldots, \sigma_n/\alpha_n]\end{array}$$

can effectively be constructed.

(The proof by induction is relatively straightforward.)

(4) For every **ann-HB$_0^=$**-derivation $\chi : \begin{matrix}\mathcal{D}\\ \tau = \sigma\end{matrix}$, which contains no open assm.-classes, an **AC$^=$**-derivation $\begin{matrix}\bar{\mathcal{D}}\\ \tau = \sigma\end{matrix}$ can effectively be found.
(This follows immediately from (3).)

(5) Every **HB$_0^=$**-derivation $\mathcal{D}$ with no open assumption-classes can effectively be transformed into an **AC$^=$**-derivation $\bar{\hat{\mathcal{D}}}$ with the same conclusion.
(This follows now by (1) and (4).)

(6) Every **HB$^=$**-deriv. can effectively be transformed into a **HB$_0^=$**-derivation with the the same conclusion.

(7) Ultimately, (5) and (6) together result in:

Theorem 17. *Every derivation $\mathcal{D}$ in* **HB$^=$** *with conclusion $\tau = \sigma$ (for some $\tau, \sigma \in \mu Tp$) and no open assumption-classes can be transformed by a sequence of effective steps into a derivation $\mathcal{D}'$ of $\tau = \sigma$ in* **AC$^=$** .

Handwritten Slide

sketching the possibility for a Transformation of a $\mathbf{HB}^{=}$-into a $\mathbf{HB}^{=}_{\mathbf{0}}$-derivation.

A *type-tree* $t \in TpTrees$ is a partial function

$$t \,:\, \{0,1\}^* \longrightarrow \{\bot, \top, \to\} \cup TVar$$

with domain $\mathcal{D}(t) := \mathrm{dom}(t)$ s.th. ... [cf. Definition 3].

Definition 18 (Partial order $\leq_T$ on Type-Trees).

The partial order $\leq_T$ on $TpTrees$ is for all type-trees t and s defined by

$$t \leq_T s \;\; :\Longleftrightarrow \;\; \big(\forall\, p \in \mathcal{D}(t) \cap \mathcal{D}(s)\big)\, [\, t(p) \leq_{\pi(p)} s(p)\,]\;\;,$$

where the function π is defined for all $p \in \{0,1\}^*$ as

$$\pi(p) := (\#\{0\text{'s occuring in } p\}) \mod 2\,,$$

and the partial orders $\leq_0$ and $\leq_1$ on $\{\bot, \top, \to\} \cup TVar$ are given by the respective transitive closure of the below two inequalities (for all $\alpha \in TVar$):

$$\bot \leq_0 \left\{ \begin{array}{c} \to \\ \alpha \end{array} \right\} \leq_0 \top \quad \text{and} \quad \top \leq_1 \left\{ \begin{array}{c} \to \\ \alpha \end{array} \right\} \leq_1 \bot\,.$$

$\boxtimes$

Definition 19. [Kozen, Palsberg, Schwartzbach]
(Subtyping relation $\leq_\mu$ on recursive types).

For all $\tau, \sigma \in \mu Tp$:

$$\tau \leq_\mu \sigma \;\; :\Longleftrightarrow \;\; \mathbf{Tree}(\tau) \leq_T \mathbf{Tree}(\sigma)\,.$$

$\boxtimes$

Definition 20. (The axiom-system $\mathbf{AC}^{\leq}$ for the subtyping relation $\leq_\mu$ between recursive types)

Axioms/Marked Assumptions:

$$\{\text{ Axioms of } \mathbf{AC}^{=} \} +$$

$$\text{(BOT)}\ \frac{}{\bot \leq \sigma} \qquad \text{(TOP)}\ \frac{}{\tau \leq \top} \qquad \text{(HYP)}\ (\alpha \leq \beta)^x$$

Rules:

$$\{\text{ Rules of } \mathbf{AC}^{=} \} +$$

$$\frac{\tau \leq \rho \qquad \rho \leq \sigma}{\tau \leq \sigma}\ \text{TRANS} \qquad \frac{\sigma_1 \leq \tau_1 \qquad \tau_2 \leq \sigma_2}{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ \text{ARROW}$$

$$\frac{\tau = \sigma}{\tau \leq \sigma}\ \text{EQUAL} \qquad \frac{\begin{array}{c}[\alpha \leq \beta]^x \\ \mathcal{D}_0 \\ \tau \leq \sigma\end{array}}{\mu\alpha.\,\tau \leq \mu\beta.\,\sigma}\ (\mu\text{-INTRO})_x \quad \begin{array}{l}(\text{if side-}\\ \text{condition S})\end{array}$$

where the *side-condition* S to the rule μ-INTRO is defined as follows:

$\alpha \notin \text{fv}(\sigma)$, $\beta \notin \text{fv}(\tau)$ and α and β do not occur in any undischarged marked assumption $(\gamma \leq \delta)^y$ of $\mathcal{D}_0$, which does not belong to the assumption-class $[\alpha \leq \beta]^x$ of $\mathcal{D}_0$.

$\boxtimes$

Theorem 18 (R. Amadio, L. Cardelli, '93). *For all recursive types $\tau, \sigma \in \mu Tp$:*

$$\tau \leq_\mu \sigma \iff \vdash_{\mathbf{AC}^{\leq}} \tau \leq \sigma\ .$$

Definition 21. (The axiom-system $\mathbf{HB}^{\leq}$ for the subtyping relation $\leq_\mu$ between recursive types)

Axioms/Marked Assumptions:

(BOT) $\dfrac{}{\bot \leq \sigma}$ (VAR) $\dfrac{}{\tau \leq \tau'}$ (if $\tau \equiv_{\mathrm{v}} \tau'$)

(TOP) $\dfrac{}{\tau \leq \top}$ $(\mu - \bot)'$ $\dfrac{}{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha \leq \bot}$ $(n \in \mathbb{N}_0)$

(REFL) $\dfrac{}{\tau \leq \tau}$ $(\mathrm{FOLD/UNFOLD})_1$ $\dfrac{}{\mu\alpha.\, \tau_0 \leq \tau_0[\mu\alpha.\, \tau_0/\alpha]}$

(HYP) $(\tau \leq \sigma)^x$ $(\mathrm{FOLD/UNFOLD})_2$ $\dfrac{}{\tau_0[\mu\alpha.\, \tau_0/\alpha] \leq \mu\alpha.\, \tau_0}$

Rules:

$$\frac{\tau \leq \rho \qquad \rho \leq \sigma}{\tau \leq \sigma}\ \mathrm{TRANS} \qquad \frac{\sigma_1 \leq \tau_1 \qquad \tau_2 \leq \sigma_2}{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ \mathrm{ARROW}$$

$$\frac{\begin{array}{cc}\langle \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2 \rangle^x & \langle \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2 \rangle^x \\ \mathcal{D}_1 & \mathcal{D}_2 \\ \sigma_1 \leq \tau_1 & \tau_2 \leq \sigma_2\end{array}}{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ (\mathrm{ARROW/FIX})_x$$

$\boxtimes$

Theorem 19. (M. Brandt and F. Henglein, '97)

For all recursive types $\tau, \sigma \in \mu Tp$:

$$\tau \leq_\mu \sigma \iff \vdash_{\mathbf{HB}^{\leq}} \tau \leq \sigma\,.$$

Definition 22. (The syntactic-matching inference-system $\mathbf{SM}^{\leq}$ for consistency-checks concerning the subtyping relation $\leq_\mu$ between recursive types)

Axioms:

(BOT) $\dfrac{}{\bot \leq \tau}$ (VAR) $\dfrac{}{\tau \leq \tau'}$ (if $\tau \equiv_{\mathrm{v}} \tau'$)

(TOP) $\dfrac{}{\top \leq \sigma}$ $(\mu - \bot)'$ $\dfrac{}{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha \leq \bot}$ $(n \in \mathbb{N}_0)$

(REFL) $\dfrac{}{\tau \leq \tau}$ $(\mathrm{FOLD/UNFOLD})_1$ $\dfrac{}{\mu\alpha.\, \tau_0 \leq \tau_0[\mu\alpha.\, \tau_0/\alpha]}$

$(\mathrm{FOLD/UNFOLD})_2$ $\dfrac{}{\tau_0[\mu\alpha.\, \tau_0/\alpha] \leq \mu\alpha.\, \tau_0}$

Rules:

$$\frac{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}{\sigma_1 \leq \tau_1}\ \mathrm{DECOMP}_l \qquad \frac{\tau \leq \rho \qquad \rho \leq \sigma}{\tau \leq \sigma}\ \mathrm{TRANS}$$

$$\frac{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}{\tau_2 \leq \sigma_2}\ \mathrm{DECOMP}_r$$

⊠

Definition 23. For all recursive types $\tau, \sigma \in \mu Tp$:

$$\tau \leq \sigma \text{ is } \mathbf{SM}^{\leq}\text{-}\textit{consistent} \ :\iff \text{For all } \chi_1, \chi_2 \in \mu Tp\text{:}$$
$$\Big[\tau \leq \sigma \vdash_{\mathbf{SM}\leq} \chi_1 \leq \chi_2 \ \Rightarrow\ \mathcal{L}'(\chi_1) \leq_0 \mathcal{L}'(\chi_2)\Big].$$

Theorem 20 ($\sim$ Theorem 5 about $\mathbf{SM}^{=}$).

For all recursive types $\tau, \sigma \in \mu Tp$:

$$\tau \leq_\mu \sigma \iff \tau \leq \sigma \textit{ is } \mathbf{SM}^{\leq}\textit{-consistent}.$$

Definition 24. (The axiom-system $\mathbf{HB}_{\mathbf{0}}^{\leq}$ for the subtyping relation $\leq_\mu$ between recursive types)

Axs./M. Assms.: $(\text{REFL})\ \dfrac{}{\tau \leq \tau}$ $(\text{HYP})\ \dfrac{}{(\tau \leq \sigma)^x}$

$(\text{BOT})\ \dfrac{}{\bot \leq \sigma}$ $(\text{TOP})\ \dfrac{}{\tau \leq \top}$

Rules:

$$\frac{\tau \leq \sigma}{\tau' \leq \sigma'}\ \text{VAR} \qquad \frac{\bot \leq \sigma}{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha \leq \sigma}\ (\mu - \bot)_l^{\text{inf}} + (\mu - \bot)_r^{\text{inf}}$$

$$\frac{\tau_0[\mu\alpha.\, \tau_0/\alpha] \leq \sigma}{\mu\alpha.\, \tau_0 \leq \sigma}\ \text{FOLD}_l \qquad \frac{\tau \leq \sigma_0[\mu\beta.\, \sigma_0/\beta]}{\tau \leq \mu\beta.\, \sigma_0}\ \text{FOLD}_r$$

$$\frac{\begin{array}{cc}\langle \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2 \rangle^x & \langle \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2 \rangle^x \\ \mathcal{D}_1 & \mathcal{D}_2 \\ \sigma_1 \leq \tau_1 & \tau_2 \leq \sigma_2\end{array}}{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ (\text{ARROW/FIX})_x$$

$\boxtimes$

Definition 25. (The checking-system $\mathbf{SM}_{\mathbf{0}}^{\leq}$ for the subtyping relation $\leq_\mu$ between recursive types)

Rules:

$$\frac{\tau \leq \sigma}{\tau' \leq \sigma'}\ \text{VAR} \qquad \frac{\mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha \leq \sigma}{\bot \leq \sigma}\ (\mu - \bot)_l^{\text{der}} + (\mu - \bot)_r^{\text{der}}$$

$$\frac{\mu\alpha.\, \tau_0 \leq \sigma}{\tau_0[\mu\alpha.\, \tau_0/\alpha] \leq \sigma}\ \text{UNFOLD}_l \qquad \frac{\tau \leq \mu\beta.\, \sigma_0}{\tau \leq \sigma_0[\mu\beta.\, \sigma_0/\beta]}\ \text{UNFOLD}_r$$

$$\frac{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}{\sigma_1 \leq \tau_1}\ \text{DECOMP}_{l/r} \qquad \frac{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}{\tau_2 \leq \sigma_2}$$

$\boxtimes$

Handwritten Slide

Example

An example (directed at indicating that there is also an analoguous "Duality" between consistency-checks in $\mathbf{SM}_{\mathbf{0}}^{\leq}$ and derivations in $\mathbf{HB}_{\mathbf{0}}^{\leq}$ similar as between consistency-checks in $\mathbf{SM}_{\mathbf{0}}^{=}$ and derivations in $\mathbf{HB}_{\mathbf{0}}^{=}$) about the difference between a derivation of $\tau \leq \sigma$ (or $\sigma \leq \tau$) in $\mathbf{HB}_{\mathbf{0}}^{\leq}$ (or, respectively, of $\tau = \sigma$ in $\mathbf{HB}_{\mathbf{0,rst}}^{=}$) and a derivation of $\tau = \sigma$ in $\mathbf{HB}_{\mathbf{0}}^{=}$.

Definition 26. (The variant-system $\mathbf{HB}^{\leq}_{\mathbf{0},\mathbf{rst}}$ of $\mathbf{HB}^{\leq}_{\mathbf{0}}$).

The variant-system $\mathbf{HB}^{\leq}_{\mathbf{0},\mathbf{rst}}$ of $\mathbf{HB}^{\leq}_{\mathbf{0}}$ with only restricted applications of ARROW/FIX

$$\text{(if side-condition S)}\ \frac{\begin{array}{c}\langle \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2 \rangle^x \\ \mathcal{D}_1 \\ \sigma_1 \leq \tau_1\end{array} \qquad \begin{array}{c}\langle \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2 \rangle^x \\ \mathcal{D}_2 \\ \tau_2 \leq \sigma_2\end{array}}{\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ (\text{ARR./F.})_x$$

has the same axioms and rules as $\mathbf{HB}^{\leq}_{\mathbf{0}}$, but all applications of ARROW/FIX in this system are subject to side-condition S:

(i) For every undischarged assm. $(\,\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,)^x$ in $\mathcal{D}_1$ the thread downwards to the conclusion $\sigma_1 \leq \tau_1$ of $\mathcal{D}_1$ crosses the *left* premise of an application of ARROW or ARROW/FIX an *odd* number of times.

(ii) For every undischarged assm. $(\,\tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,)^x$ in $\mathcal{D}_2$ the thread downwards to the conclusion $\tau_2 \leq \sigma_2$ of $\mathcal{D}_2$ crosses the *left* premise of an application of ARROW or ARROW/FIX an *even* number of times.

⊠

Lemma 21. *Every derivation in the system* $\mathbf{HB}^{\leq}_{\mathbf{0},\mathbf{rst}}$ *can be rebuilt effectively into a derivation in* $\mathbf{HB}^{\leq}_{\mathbf{0},\mathbf{rst}}$.

Hence the systems $\mathbf{HB}^{\leq}_{\mathbf{0}}$ *and* $\mathbf{HB}^{\leq}_{\mathbf{0},\mathbf{rst}}$ *are equivalent theories.*

Handwritten Slide

Proof of Lemma 20.

sketching the Transformation of a $\mathbf{HB}^{=}_{\mathbf{0}}$- into an $\mathbf{HB}^{=}_{\mathbf{0,rst}}$-derivation.

Definition 27. (The annotated version ann-$\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$ of the system $\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$).

Axioms/Marked Assumptions:

$$\text{(BOT)}\ \frac{}{(\bot, \sigma) : \bot \leq \sigma} \qquad \text{(TOP)}\ \frac{}{(\tau, \top) : \tau \leq \top}$$

$$\text{(REFL)}\ \frac{}{(\tau, \tau) : \tau \leq \tau} \qquad \text{(HYP)}\ \big((\alpha, \bar{\alpha}) : \tau \leq \sigma\big)^{\alpha} \ (\text{where } \bar{\alpha} \not\equiv \alpha)$$

Rules:

$$\frac{(\chi, \bar{\chi}) : \tau \leq \sigma}{(\chi, \bar{\chi}) : \tau' \leq \sigma'}\ \text{VAR} \quad (\text{if } \tau' \equiv_{\text{v}} \tau \text{ and } \sigma' \equiv_{\text{v}} \sigma)$$

$$\frac{(\chi, \bar{\chi}) : \bot \leq \sigma}{(\chi, \bar{\chi}) : \mu\alpha\alpha_1 \ldots \alpha_n.\, \alpha \leq \sigma}\ (\mu - \bot)^{\text{inf}}_{l/r} \qquad \frac{(\chi, \bar{\chi}) : \tau \leq \bot}{(\chi, \bar{\chi}) : \tau \leq \mu\beta\beta_1 \ldots \beta_n.\, \beta}$$

$$\frac{(\chi, \bar{\chi}) : \tau_0[\mu\alpha.\, \tau_0/\alpha] \leq \sigma}{(\chi, \bar{\chi}) : \mu\alpha.\, \tau_0 \leq \sigma}\ \text{FOLD}_l \qquad \frac{(\chi, \bar{\chi}) : \tau \leq \sigma_0[\mu\beta.\, \sigma_0/\beta]}{(\chi, \bar{\chi}) : \tau \leq \mu\beta.\, \sigma_0}\ \text{FOLD}_r$$

$$(\text{if side-condition } \mathbf{S'_1})\ \frac{\begin{array}{cc}\mathcal{D}_1 & \mathcal{D}_2 \\ (\bar{\chi_1}, \chi_1) : \sigma_1 \leq \tau_1 & (\chi_2, \bar{\chi_2}) : \tau_2 \leq \sigma_2\end{array}}{(\chi_1 \to \chi_2,\ \bar{\chi_1} \to \bar{\chi_2}) : \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ \text{ARROW}$$

$$\frac{\begin{array}{cc}\langle(\alpha, \bar{\alpha}) : \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\rangle^{\alpha} & \langle(\alpha, \bar{\alpha}) : \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\rangle^{\alpha} \\ \mathcal{D}_1 & \mathcal{D}_2 \\ (\bar{\chi_1}, \chi_1) : \sigma_1 \leq \tau_1 & (\chi_2, \bar{\chi_2}) : \tau_2 \leq \sigma_2\end{array}}{(\mu\alpha.\, \chi_1 \to \chi_2,\ \mu\bar{\alpha}.\, \bar{\chi_1} \to \bar{\chi_2}) : \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2}\ \begin{array}{l}(\text{if s.-c. } \mathbf{S'_2}) \\ (\text{AR./F.})_{\alpha}\end{array}$$

Side-condition $\mathbf{S'_1}$:

(i) *In different undischarged assumptions different sets of annotation-variables are used,* more precisely: For all undischarged assumption-classes $[\,(\alpha,\bar{\alpha}):\ \tau_1 \leq \tau_2\,]^{\alpha}$ and $[\,(\beta,\bar{\beta}):\ \sigma_1 \leq \sigma_2\,]^{\beta}$ in $\mathcal{D}_1$ or $\mathcal{D}_2$ it holds, that

$$(\,\alpha \in \{\beta,\bar{\beta}\}\ \vee\ \beta \in \{\alpha,\bar{\alpha}\}\,)\ \Longrightarrow$$
$$\Longrightarrow\ \ \alpha \equiv \beta\ \wedge\ \bar{\alpha} \equiv \bar{\beta}\ \wedge\ \tau_1 \equiv \sigma_1\ \wedge\ \tau_2 \equiv \sigma_2\ .$$

(ii) *Variables used as annotations in undischarged assumptions do not occur in axioms,* more precisely: For all undischarged assumption-classes $[\,(\alpha,\bar{\alpha}):\ \tau_1 \leq \tau_2\,]^{\alpha}$ in $\mathcal{D}_1$ or $\mathcal{D}_2$ and for all axioms $\dfrac{}{(\sigma_1,\sigma_2):\ \sigma_1 \leq \sigma_2}$ occuring as leaves at the top of $\mathcal{D}_1$ or $\mathcal{D}_2$ it holds that

$$\alpha,\bar{\alpha} \notin \mathrm{fv}(\sigma_1) \cup \mathrm{fv}(\sigma_2)\ .$$

Side-condition $\mathbf{S'_2}$:

(i) Side-condition $\mathbf{S'_1}$.

(ii) $[\,(\alpha,\bar{\alpha}):\ \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,]^{\alpha}$ is an inhabitated assumption-class in $\mathcal{D}_1$ or in $\mathcal{D}_2$.

(iii) For every inhabitated assumption-class $[\,(\beta,\bar{\beta}):\ \rho_1 \leq \rho_2\,]^{\beta}$ in $\mathcal{D}_1$ or $\mathcal{D}_2$, which is different from the assumption-class $[\,(\alpha,\bar{\alpha}):\ \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,]^{\alpha}$, it holds, that

$$\alpha,\bar{\alpha} \notin \mathrm{fv}(\rho_1) \cup \mathrm{fv}(\rho_2)\ .$$

(iv) Every thread in $\mathcal{D}_1$ $\left\{\text{ in } \mathcal{D}_2\right\}$ from an assumption $\big(\,(\alpha,\bar{\alpha}):\ \tau_1 \to \tau_2 \leq \sigma_1 \to \sigma_2\,\big)^{\alpha}$ down to the conclusion $(\bar{\chi_1},\chi_1):\ \sigma_1 \leq \tau_1$ of $\mathcal{D}_1$ $\left\{(\chi_2,\bar{\chi_2}):\ \tau_2 \leq \sigma_2 \text{ of } \mathcal{D}_2\right\}$ crosses the first premise of an application of ARROW or ARROW/FIX an *odd* $\left\{\textit{even}\right\}$ number of times.

Transformation: $\mathbf{HB}^{\leq}_{\mathbf{0}}$-deriv. $\longmapsto$

$\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-deriv. $\longmapsto$ **ann-**$\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-deriv. $\longmapsto$ $\mathbf{AC}^{\leq}$-deriv.

Relevant steps:

(1) Every $\mathbf{HB}^{\leq}_{\mathbf{0}}$-derivation $\mathcal{D}$ can effectively be transformed into a $\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-derivation $\tilde{\mathcal{D}}$ with the same conclusion and the same open assumption-classes.

(2) Every $\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-derivation $\mathcal{D}$ can effectively be annotated with an **ann-**$\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-derivation $\hat{\mathcal{D}}$ as result:

$$\begin{array}{c} \{\,[\,\tau_i \leq \sigma_i\,]^{x_i}\,\}_{i=1,\ldots,n} \\ \mathcal{D} \\ \tau \leq \sigma \end{array} \quad \longmapsto \quad \begin{array}{c} \{\,[\,(\alpha_i, \bar{\alpha}_i) \,:\, \tau_i \leq \sigma_i\,]^{\alpha_i}\,\}_{i=1,\ldots,n} \\ \hat{\mathcal{D}} \\ \chi \,:\, \tau \leq \sigma \end{array} .$$

(3) Reversely, every **ann-**$\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-derivation $\mathcal{D}$ can be rewritten (basically by stripping it of its annotations) into a $\mathbf{HB}^{\leq}_{\mathbf{0,rst}}$-derivation $\check{\mathcal{D}}$.
(This item is not needed for the transformations considered here.)

(4) For every $\mathbf{ann\text{-}HB}^{\leq}_{\mathbf{0,rst}}$-derivation

$$\frac{\{\,[\,(\alpha_i,\bar{\alpha}_i)\,:\,\tau_i\leq\sigma_i\,]^{\alpha_i}\,\}_{i=1,\ldots,n}}{\begin{array}{c}\mathcal{D}\\(\chi,\bar{\chi})\,:\,\tau\leq\sigma\end{array}}$$

where the displayed family of assumption-classes comprises just all yet undischarged, inhabitated assumption–classes in $\mathcal{D}$, two $\mathbf{AC}^{=}$-derivations

$$\begin{array}{c}\mathcal{D}_1^*\\ \tau=\chi[\ldots,\tau_i/\alpha_i,\ldots,\sigma_i/\bar{\alpha}_i,\ldots]\end{array} \quad\text{and}\quad \begin{array}{c}\mathcal{D}_2^*\\ \sigma=\bar{\chi}[\ldots,\tau_i/\alpha_i,\ldots,\sigma_i/\bar{\alpha}_i,\ldots]\end{array}$$

as well as an $\mathbf{AC}^{\leq}$-derivation $\mathcal{D}_3^*$ of the form

$$\frac{\{\,[\,\alpha_i\leq\bar{\alpha}_i\,]\,\}_{i=1,\ldots,n}}{\begin{array}{c}\mathcal{D}_3^*\\ \chi\leq\bar{\chi}\end{array}}$$

can effectively be constructed.

(The proof by induction is relatively straightforward.)

(5) For every $\mathbf{ann\text{-}HB}^{\leq}_{\mathbf{0,rst}}$-derivation $\begin{array}{c}\mathcal{D}\\ \chi\,:\,\tau=\sigma\end{array}$, which contains no open assm.-classes, an $\mathbf{AC}^{\leq}$-derivation $\begin{array}{c}\bar{\mathcal{D}}\\ \tau\leq\sigma\end{array}$ with no open assm.-classes can be found effectively.

(6) **Theorem 22.** *Every* $\mathbf{HB}^{\leq}_{\mathbf{0}}$*-derivation* $\mathcal{D}$ *with no open assm.-classes can effectively be transformed into an* $\mathbf{AC}^{\leq}$*-derivation* $\begin{array}{c}\bar{\bar{\mathcal{D}}}\\ \tau\leq\sigma\end{array}$ *with the same conclusion and no open assm.-classes.*

Transformation:

$$\mathbf{AC}^{\leq}\textbf{-derivation} \longmapsto \mathbf{HB}^{\leq}\textbf{-derivation}$$

Theorem 23. *Every* $\mathbf{AC}^{\leq}$*-derivation* $\mathcal{D}$ *can effectively be transformed into a* $\mathbf{HB}^{\leq}$*-derivation* $\mathcal{D}'$ *with the same conclusion.*

Route of the Proof:

(1) *Replacing* $\mathbf{AC}^{=}$*-subderivations by* $\mathbf{HB}^{\leq}$*-subderivations*:

Replacing all subderivations $\mathcal{D}_0$ of $\mathcal{D}$ of the form—where $\mathcal{D}_{00}$ is a derivation in $\mathbf{AC}^{=}$—

$$\frac{\begin{array}{c}\mathcal{D}_{00}\\ \tau_0 = \sigma_0\end{array}}{\tau_0 \leq \sigma_0}\,\text{EQUAL} \qquad \text{by derivations} \qquad \begin{array}{c}\tilde{\mathcal{D}}_0\\ \tau_0 \leq \sigma_0\end{array}$$

in $\mathbf{HB}^{\leq}$ (the derivations $\tilde{\mathcal{D}}_0$ can always be found effectively given derivations $\mathcal{D}_0$ of the here considered kind).

The result $\tilde{\mathcal{D}}$ of these replacements is then a derivation in $\mathbf{HB}^{\leq}+(\mu\text{-INTRO})$.

(2) *Eliminating applications of μ-INTRO*-rules:

Successively replacing—starting with topmost occurrences—every subderivation $\tilde{\mathcal{D}}_0$ of the form

$$\dfrac{\begin{array}{c}[\alpha \leq \beta]^x \\ \tilde{\mathcal{D}_{00}} \\ \tau \leq \sigma\end{array}}{\mu\alpha.\,\tau \leq \mu\beta.\,\sigma}\,(\mu\text{-INTRO})_x$$

in $\tilde{\mathcal{D}}$, where $\tilde{\mathcal{D}_{00}}$ is a $\mathbf{HB}^{\leq}$-derivation, by a derivation

$$\begin{array}{c}\tilde{\tilde{\mathcal{D}}}_0 \\ \mu\alpha.\,\tau \leq \mu\beta.\,\sigma\end{array}$$

in $\mathbf{HB}^{\leq}$, where $\tilde{\tilde{\mathcal{D}}}_0$ has the same open assumption-classes as $\tilde{\mathcal{D}}_0$ *except* $[\,\alpha \leq \beta\,]^x$, which does not appear any more in $\tilde{\tilde{\mathcal{D}}}_0$.

(These replacement-derivations can always be found effectively.)

The result of successively removing all applications of μ-INTRO from $\tilde{\mathcal{D}}$ (i.e. from it and from all the derivations occuring as intermediate results), is then a derivation $\mathcal{D}'$ in $\mathbf{HB}^{\leq}$ with the same conclusion and the same open assumption-classes as $\mathcal{D}$.

□

Handwritten Slide

showing the single steps in a (i.e. in one possible) transformation of an $\mathbf{AC}^{=}$- into a $\mathbf{HB}^{=}$-derivation.

Handwritten Slide

showing a way, how an elimination of μ-INTRO from derivation, whose immediate subderivation is a $\mathbf{HB}^{=}$-derivation, can be done effectively.

Handwritten Slide

(Continuation)

showing a way, how an elimination of μ-INTRO from derivation, whose immediate subderivation is a $\mathbf{HB}^{=}$-derivation, can be done effectively.

Example 5. (Elimination of a FIX-rule in a very easy case)

Consider the following derivation $\mathcal{D}$ in $\mathbf{HB}_{\mathbf{0}}^{\leq}$+FIX, in which there is only one application of FOLD_l between the one and only application of FIX and a preceeding application of ARROW, of the form

$$\dfrac{\dfrac{\dfrac{\dfrac{\langle \mu\alpha.\,\tau_0 \leq \sigma\rangle^x \quad \mathcal{D}_1}{\tau_0[\mu\alpha.\,\tau_0/\alpha] \leq \sigma}\,\mathrm{ARROW} \quad \langle \mu\alpha.\,\tau_0 \leq \sigma\rangle^x \quad \mathcal{D}_2}{\tau_0[\mu\alpha.\,\tau_0/\alpha] \leq \sigma}\,\mathrm{ARROW}}{\mu\alpha.\,\tau_0 \leq \sigma}\,\mathrm{FOLD}_l}{\mu\alpha.\,\tau_0 \leq \sigma}\,(\mathrm{FIX})_x$$

where $\mathcal{D}_0$ and $\mathcal{D}_1$ are $\mathbf{HB}_{\mathbf{0}}^{\leq}$-derivations. (The displayed application of FIX in $\mathcal{D}$ is justified, since its immediate subderivation is clearly *contractive*.)

$\mathcal{D}$ can be transformed into the following $\mathbf{HB}_{\mathbf{0}}^{\leq}$-derivation $\mathcal{D}'$:

$$\dfrac{\dfrac{\dfrac{(\tau_0[\mu\alpha.\,\tau_0/\alpha] \leq \sigma)^x}{\langle \mu\alpha.\,\tau_0 \leq \sigma\rangle}\,\mathrm{FOLD}_l \quad \mathcal{D}_1 \quad \dfrac{(\tau_0[\mu\alpha.\,\tau_0/\alpha] \leq \sigma)^x}{\langle \mu\alpha.\,\tau_0 \leq \sigma\rangle}\,\mathrm{FOLD}_l \quad \mathcal{D}_2}{\tau_0[\mu\alpha.\,\tau_0/\alpha] \leq \sigma}\,(\mathrm{ARROW}/\mathrm{FIX})_x}{\mu\alpha.\,\tau_0 \leq \sigma}\,\mathrm{FOLD}_l$$

■

Gathered Results

A. In the case of Recursive Type Equality $=_\mu$:

1. There is an immediate "duality" between "consistency checks" in $\mathbf{SM}_{\mathbf{0}}^{=}$ and derivations in $\mathbf{HB}_{\mathbf{0}}^{=}$ (*Jan Willem Klop*).
2. μ-COMPAT is a derived rule of $\mathbf{AC}_{\mathbf{1}}^{=}$.
3. An effective translation of $\mathbf{AC}^{=}$-derivations into $\mathbf{HB}^{=}$-derivations.
4. An effective translation of $\mathbf{HB}^{=}$- into $\mathbf{AC}^{=}$-derivations.
   a. A transformation of $\mathbf{HB}^{=}$- into $\mathbf{HB}_{\mathbf{0}}^{=}$-derivations.
   b. An effective transformation of $\mathbf{HB}_{\mathbf{0}}^{=}$- into $\mathbf{AC}^{=}$-derivations.
   (Use of the annotated version $\mathbf{ann\text{-}HB}_{\mathbf{0}}^{=}$ of $\mathbf{HB}_{\mathbf{0}}^{=}$).
5. a An effective proof for the eliminibility of the FIX-rule

$$\frac{\begin{array}{c}[\tau = \sigma]^x \\ \mathcal{D}_0 \\ \tau = \sigma\end{array}}{\tau = \sigma}\ \mathrm{FIX}_x \quad (\mathcal{D}_0 \text{ is } \textit{contractive})$$

   in $\mathbf{HB}^{=}$-derivations.

   b A semantical soundness-proof of $\mathbf{HB}^{=}$+FIX.

B. In the case of the Subtyping Relation $\leq_\mu$:

1. There is an immediate "duality" between "consistency checks" in $\mathbf{SM}_{\mathbf{0}}^{\leq}$ and derivations in $\mathbf{HB}_{\mathbf{0}}^{\leq}$.
2. An effective transformation of $\mathbf{AC}^{\leq}$-derivations into $\mathbf{HB}^{\leq}$-derivations.
3. An effective translation of $\mathbf{HB}^{\leq}$- into $\mathbf{AC}^{\leq}$-derivations.
   a. An effective Transformation of $\mathbf{HB}^{\leq}$- into $\mathbf{HB}_{\mathbf{0}}^{\leq}$-derivations.
   b. An effective Transformation of $\mathbf{HB}^{\leq}$- into $\mathbf{HB}_{\mathbf{0,rst}}^{\leq}$-derivations.
   c. An effective translation of $\mathbf{HB}_{\mathbf{0,rst}}^{\leq}$- into $\mathbf{AC}^{\leq}$-derivations.
   (Use of the annotated version $\mathbf{ann\text{-}HB}_{\mathbf{0,rst}}^{\leq}$ of $\mathbf{HB}_{\mathbf{0,rst}}^{\leq}$).
4. The FIX-rule is a derived rule of $\mathbf{HB}^{\leq}$, applications of which can be eliminated effectively from $\mathbf{HB}^{\leq}$-derivations.
5. The rule μ-INTRO of $\mathbf{AC}^{\leq}$ is (in $\mathbf{HB}^{\leq}$) equivalent to the rules ARROW/FIX or FIX.

References

[AmCa93] Amadio, R.M., Cardelli, L.: "Subtyping Recursive Types", *ACM Transactions on Programming Languages and Systems 15 (4)*, pp. 575–631, 1993.

[ArKl95] Ariola, Z.H., Klop, J.W.: "Equational Term Graph Rewriting", *Fundamenta Informaticae 26 (3,4)*, pp. 207–240, June 1996; extended version as: *University of Oregon Technical Report CIS–TR–95–16*, 1995; and as: *Vrije Universiteit Amsterdam Technical Report IR–391*, September 1995.

[BrHe98] Brandt, M., Henglein, F.: "Coinductive Axiomatization of Recursive Type Equality and Subtyping", *Fundamenta Informaticae 33*, pp. 1–30, 1998.

[KPS95] Kozen, D., Palsberg, J., Schwartzbach, M.I.: "Efficient Recursive Subtyping", *Mathematical Structures in Computer Science 5*, pp. 113-125, 1995.