# Proof-Theoretic Interconnections between Known Inference- and Axiom-Systems for Recursive Type Equality

*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*

**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 $\mathbf{Unit} := \{\text{val}\}$.

**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 $\mathbf{A}$ . |

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 $\mu$-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 $\mu$-type $\sigma$ can then be seen to satisfy equation (1) for $\tau$:

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

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

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

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

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

can—under certain restrictions on $\tau$ and $\alpha$—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 $\mu Tp$:**

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

**Recursive Types in canonical form can-$\mu Tp$:**

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

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

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

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

$\boxtimes$

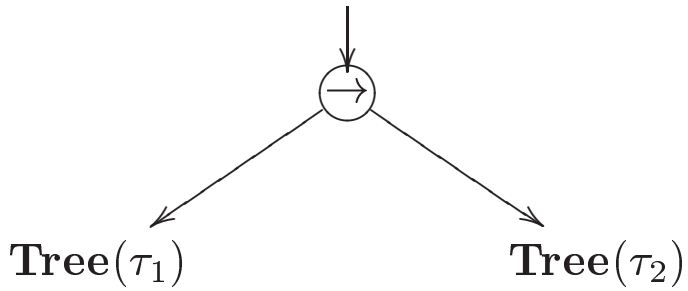
**Definition 2.** (**Tree-Unfolding of Recursive Types in canonical Form**)

The function **Tree**

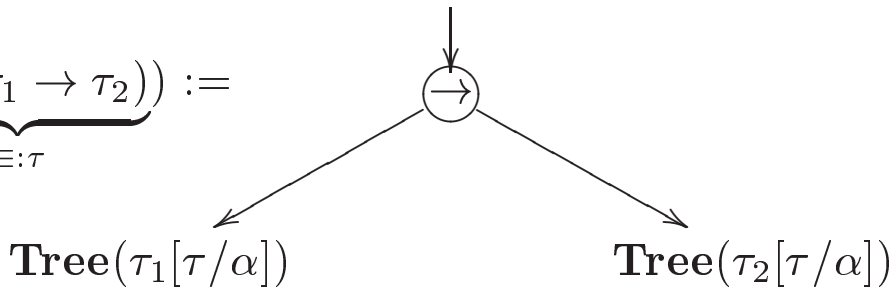
$$\mathbf{Tree}: \text{ can-}\mu Tp \longrightarrow \text{Labelled Binary Trees}$$
$$\tau \longmapsto \mathbf{Tree}(\tau)$$

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

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

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

;

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

.

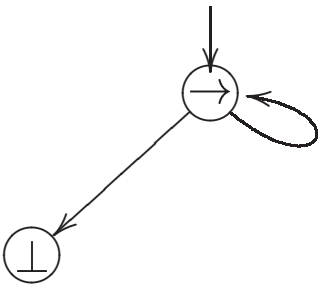
⊠

**Example 1 (Tree-Unfolding).**

The recursive type

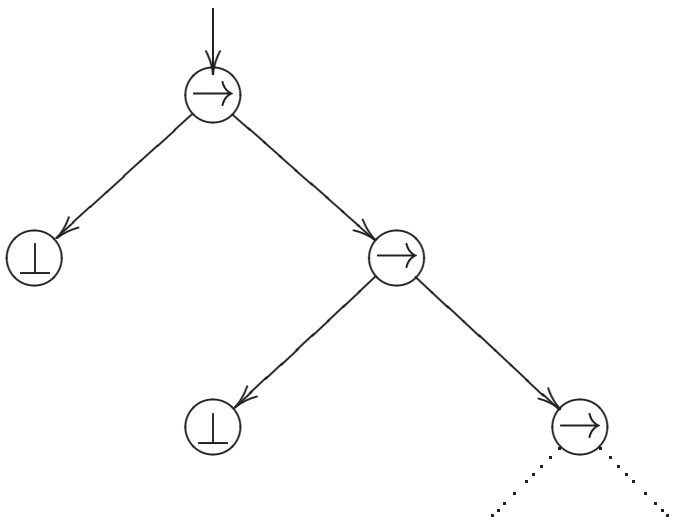
$$\tau := \mu\alpha.\,(\bot \to \alpha)$$

corresponds to the *cyclic term-graph*




and has the tree-unfolding

**Tree**$(\tau)$=



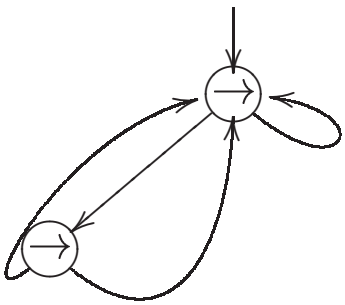

■

**Example 2 (Tree-Unfolding).**

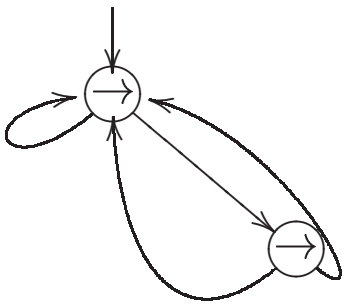
The recursive types

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

correspond respectively to the different cyclic term-graphs

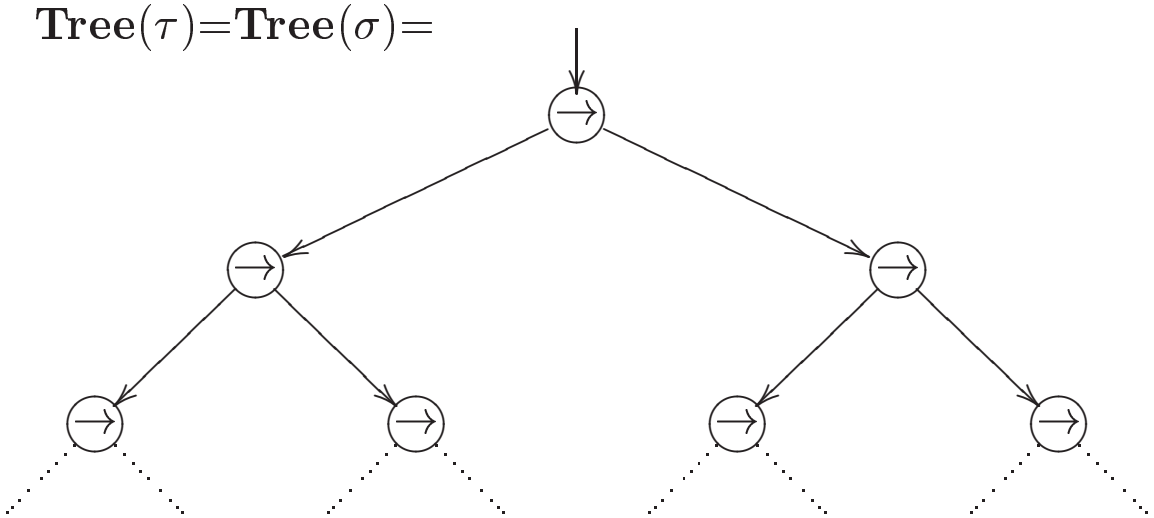


and



but possess the same tree-unfolding

**Tree**$(\tau)$=**Tree**$(\sigma)$=



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

■

**Definition 3. (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 $\tau$ is *contractive* w.r.t. $\alpha$). ⊠

**Theorem 1 (R. Amadio, L. Cardelli, '93).** *For all recursive types* $\tau, \sigma \in \mu Tp$: $\tau =_\mu \sigma \iff \vdash_{AC^=} \tau = \sigma$ .

**Definition 4. (The axiom-system HB$^=$ for recursive type equality $=_\mu$)**

*Axioms:*

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

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

*Rules:*

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

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

$\boxtimes$

**Theorem 2 (M. Brandt ( and F. Henglein), '98).**

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

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

**Definition 5. (The syntactic-matching inference-system $\mathbf{SM}^{=}$ for checks concerning recursive type equality $=_\mu$)**

*Axioms:*

(REFL) $\dfrac{}{\tau = \tau}$ $\quad (\mu - \bot)'$ $\dfrac{}{\mu\alpha\alpha_1 \ldots \alpha_n . \alpha = \bot}$

(VAR) $\dfrac{}{\tau = \tau'}$ (if $\tau \equiv_{\mathrm{v}} \tau'$) $\quad$ (FOLD/UNFOLD) $\dfrac{}{\mu\alpha . \tau = \tau[\mu\alpha . \tau / \alpha]}$

*Rules:*

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

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

⊠

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

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

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

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

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

**Theorem 3 ($\sim$) Z.M. Ariola, J.W. Klop, 1995).** *For all recursive types $\tau, \sigma \in \mu Tp$:*

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

**Definition 7 (Leading Symbol of a Recursive Type).** For every recursive type $\tau \in \mu Tp$ its *leading symbol* $\mathcal{L}(\tau)$ is formally defined by induction on $\tau$:

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

$\boxtimes$

**Definition 8. (The axiom-system $\mathbf{HB}_0^=$ for recursive type equality $=_\mu$)**

*Axioms:* $\overline{\tau = \tau}$ (REFL)

*Rules:*

$$\frac{\tau = \sigma}{\tau' = \sigma'} \text{ VAR} \qquad \frac{\bot = \sigma}{\mu\alpha_1\alpha_2 \ldots \alpha_n.\, \alpha_1 = \sigma} \; (\mu - \bot)_{\text{l}}^{\text{inf}} + (\mu - \bot)_{\text{r}}^{\text{inf}}$$

$$\frac{\tau_0[\mu\alpha.\, \tau_0/\alpha] = \sigma}{\mu\alpha.\, \tau_0 = \sigma} \text{ FOLD}_{\text{l}} \qquad \frac{\tau = \sigma_0[\mu\alpha.\, \sigma_0/\alpha]}{\tau = \mu\alpha.\, \sigma_0} \text{ FOLD}_{\text{r}}$$

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

$\boxtimes$

**Definition 9. (The axiom-system $\mathbf{SM}_0^=$ for recursive type equality $=_\mu$)**

*Rules:*

$$\frac{\tau = \sigma}{\tau' = \sigma'} \text{ VAR} \qquad \frac{\mu\alpha_1\alpha_2 \ldots \alpha_n.\, \alpha_1 = \sigma}{\bot = \sigma} \; (\mu - \bot)_{\text{l}}^{\text{der}} + (\mu - \bot)_{\text{r}}^{\text{der}}$$

$$\frac{\mu\alpha.\, \tau_0 = \sigma}{\tau_0[\mu\alpha.\, \tau_0/\alpha] = \sigma} \text{ UNFOLD}_{\text{l}} \qquad \frac{\tau = \mu\alpha.\, \sigma_0}{\tau = \sigma_0[\mu\alpha.\, \sigma_0/\alpha]} \text{ UNFOLD}_{\text{r}}$$

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

$\boxtimes$

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

$$\dfrac{\dfrac{\dfrac{

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

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

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

■

# "Results"

1. There is an immediate "duality" between "consistency checks" in $\mathrm{SM}_0^=$ and derivations in $\mathrm{HB}_0^=$ (*Jan Willem Klop*).
2. $\mu$-COMPAT is a derived rule of $\mathrm{AC}_1^=$.
3. An effective translation of $\mathrm{AC}^=$-derivations into $\mathrm{HB}^=$-derivations.
4. An effective translation of $\mathrm{HB}^=$- into $\mathrm{AC}^=$-derivations.
   a. A Transformation of $\mathrm{HB}^=$- into $\mathrm{HB}_0^=$-derivations.
   b. An effective translation of $\mathrm{HB}_0^=$- into $\mathrm{AC}^=$-derivations.
5. A "*Liberalisation*" e-$\mathrm{HB}^=$ of $\mathrm{HB}^=$ with rules such as e.g.
$$\frac{\begin{array}{c}[\tau = \mu\alpha.\,\sigma_0]^x \\ \mathcal{D} \\ \tau = \sigma_0[\mu\alpha.\,\sigma_0/\alpha]\end{array}}{\tau = \mu\alpha.\,\sigma_0}\,(\mathrm{FOLD/FIX})_x \quad (\mathcal{D} \text{ is } \textit{contractive}) \;.$$
6. Transferring these results to the more general situation of analoguous proof-theoretic interconnections between formal systems $\mathrm{AC}^\leq$, $\mathrm{HB}^\leq$ and $\mathrm{SM}^\leq$ for the *subtype-relation* between recursive types.

   Some further observations concerning details about the connections between derivations in these more general systems.

# 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.