

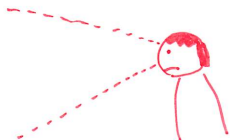
Hyper-lambda calculus for waitfree computation

Yoichi Hirai¹

Jun. 18, 2011 Eugene

¹Univ. of Tokyo, a JSPS fellow

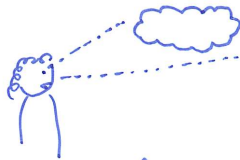
Yoichi Hirai wants to be a **traveling merchant**



Multicore
Programming

Library
Implementor

Concurrency
Theory

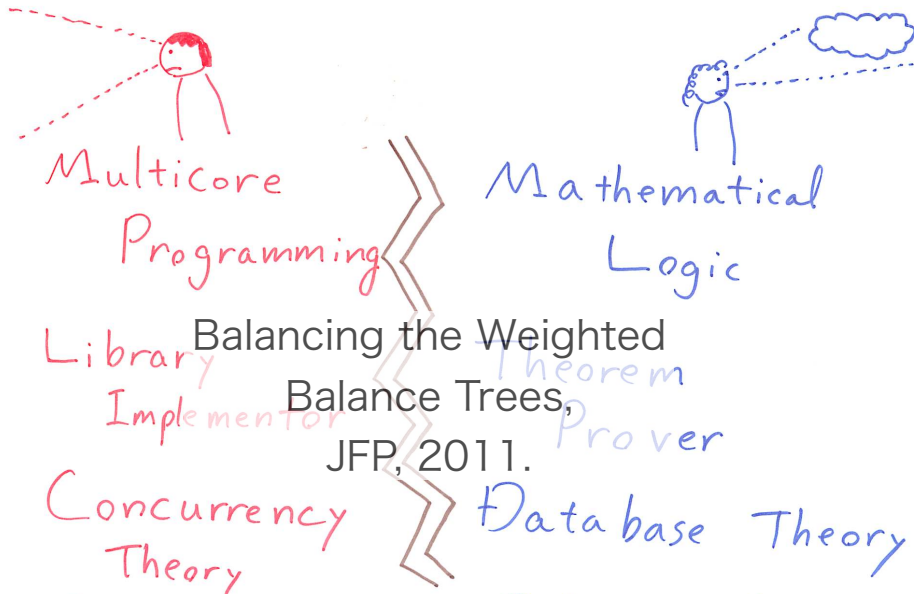


Mathematical
Logic

Theorem
Prover

Database Theory

Yoichi Hirai wants to be a **traveling merchant**



Gödel–Dummett logic

(nothing to do with completeness or incompleteness)

You are familiar with [Intuitionistic propositional logic](#)

Add $(\varphi \supset \psi) \vee (\psi \supset \varphi)$ as provable.

(And anything provable with this in intuitionistic logic)

Gödel–Dummett logic

(nothing to do with completeness or incompleteness)

You are familiar with **Intuitionistic propositional logic**

Add $(\varphi \supset \psi) \vee (\psi \supset \varphi)$ as provable.

(And anything provable with this in intuitionistic logic)

Q. What chooses this disjunction in the corresponding lambda calculus?

An oracle does: $M: (\varphi \supset \psi) \vee (\psi \supset \varphi)$.

$M \rightsquigarrow^* \text{inl}(\dots)$? OR $M \rightsquigarrow^* \text{inr}(\dots)$?

Choosing an interesting oracle leads to an interesting application.

idea: **formulas to inference rules!**

Hypersequent formulation [Avron, 1991]

Hypersequent is a finite sequence of sequents.

Instead of the axioms $(\varphi \supset \psi) \vee (\psi \supset \varphi)$, include the communication rule

$$\frac{\mathcal{H}_0 \mid \Gamma_0, \Delta_0 \vdash \varphi_0 \quad \mathcal{H}_1 \mid \Gamma_1, \Delta_1 \vdash \varphi_1}{\mathcal{H}_0 \mid \mathcal{H}_1 \mid \Gamma_0, \Delta_1 \vdash \varphi_0 \mid \Gamma_1, \Delta_0 \vdash \varphi_1}$$

- provable with cuts \implies provable without cuts (Gentzen's proof adopted)
- "it seems to us extremely important to determine the exact computational content of them [intermediate logics] — and to develop corresponding ' λ -calculi' " — Avron [1991].

Nobody has really done it

Hypersequent-style natural deduction [Baaz et al., 2001]

- normal form theorem proved via Avron's cut-elimination.
- reduction is not given

Parallel dialogue games [Fermüller, 2003]

- based on Lorenzen games.
- proof searching bottom-to-up with player knowing the answer proof tree and opponent directing for different parts of the proof.
- the global game state has local parts.
- for example, player can duplicate a local part into two (EC) 1
- already normalized proofs

Avron's question has not been answered.

Hypersequent-style natural deduction (modified Baaz et al. [2001])

$$\text{Ax} \frac{}{\varphi \vdash \varphi}$$

$$\supset \mathcal{I} \frac{\mathcal{H} \mid \varphi, \Gamma \vdash \psi}{\mathcal{H} \mid \Gamma \vdash \varphi \supset \psi}$$

$$\supset \mathcal{E} \frac{\mathcal{H} \mid \Gamma \vdash \varphi \supset \psi \quad \mathcal{H}' \mid \Gamma \vdash \varphi}{\mathcal{H} \mid \mathcal{H}' \mid \Gamma \vdash \psi}$$

... (NJ rules) ...

$$\text{com} \frac{\mathcal{H} \mid \varphi, \Gamma \vdash \psi \quad \mathcal{H}' \mid \theta, \Delta \vdash \tau}{\mathcal{H} \mid \mathcal{H}' \mid \Gamma \vdash \theta \supset \psi \mid \Delta \vdash \varphi \supset \tau}$$

$$\text{EC} \frac{\mathcal{H} \mid \Gamma \vdash \varphi \mid \Gamma \vdash \varphi}{\mathcal{H} \mid \Gamma \vdash \varphi}$$

$$\text{EW} \frac{\mathcal{H}}{\mathcal{H} \mid \Gamma \vdash \varphi}$$

and exchange rules.

$$\frac{}{x: \varphi \triangleright x: \varphi}$$

$$\frac{\mathcal{G} \mid x: \varphi, \Gamma \triangleright M: \psi}{\mathcal{G} \mid \Gamma \triangleright \lambda x. M: \varphi \supset \psi}$$

$$\frac{\mathcal{G} \mid \Gamma \vdash M: \varphi \supset \psi \quad \mathcal{G}' \mid \Gamma \vdash N: \varphi}{\mathcal{G} \mid \mathcal{G}' \mid \Gamma \vdash MN: \psi}$$

... (typed λ -cal. rules with $\mathcal{G} \mid$) ...

$$\frac{\mathcal{G} \mid x: \varphi, \Gamma \triangleright M: \psi \quad \mathcal{G}' \mid y: \theta, \Delta \triangleright N: \tau}{\mathcal{G} \mid \mathcal{G}' \mid \Gamma \triangleright \overrightarrow{l}x.M: \theta \supset \psi \mid \Delta \triangleright \overleftarrow{l}y.N: \varphi \supset \tau}$$

(l : fresh location)

$$\frac{\mathcal{G} \mid \Gamma \triangleright M: \varphi \mid \Gamma \triangleright N: \varphi}{\mathcal{G} \mid \Gamma \triangleright [M, N]: \varphi}$$

$$\frac{\mathcal{G}}{\mathcal{G} \mid \Gamma \triangleright \text{abort}: \varphi}$$

and exchange rules (internal and external).

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')$$

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')$$

$(\vec{\sigma}, \overleftarrow{\sigma}, (\lambda x.M)O) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M[O/x])$ and other rules of the underlying λ -cal.

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')$$

$(\vec{\sigma}, \overleftarrow{\sigma}, (\lambda x.M)O) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M[O/x])$ and other rules of the underlying λ -cal.

for N closed:

$$(\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N], M[O/x])$$

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')$$

$(\vec{\sigma}, \overleftarrow{\sigma}, (\lambda x.M)O) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M[O/x])$ and other rules of the underlying λ -cal.

for N closed:

$$(\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N], M[O/x])$$

$$(\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N], \text{abort})$$

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')$$

$(\vec{\sigma}, \overleftarrow{\sigma}, (\lambda x.M)O) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M[O/x])$ and other rules of the underlying λ -cal.

for N closed:

$$(\vec{\sigma}[l \mapsto O], \overleftarrow{\sigma}[l \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[l \mapsto O], \overleftarrow{\sigma}[l \mapsto N], M[O/x])$$

$$(\vec{\sigma}[l \mapsto \perp], \overleftarrow{\sigma}[l \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[l \mapsto \perp], \overleftarrow{\sigma}[l \mapsto N], \text{abort})$$

$$(\vec{\sigma}[l \mapsto O], \overleftarrow{\sigma}[l \mapsto N'], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[l \mapsto O], \overleftarrow{\sigma}[l \mapsto N'], M[O/x])$$

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')$$

$(\vec{\sigma}, \overleftarrow{\sigma}, (\lambda x.M)O) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M[O/x])$ and other rules of the underlying λ -cal.

for N closed:

$$(\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N], M[O/x])$$

$$(\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N], \text{abort})$$

$$(\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N'], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N'], M[O/x])$$

$$(\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N'], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N'], \text{abort})$$

Reduction: configuration \rightsquigarrow configuration

Configuration is a triple $(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G})$

for $\vec{\sigma}, \overleftarrow{\sigma} : L \rightarrow \text{Term} + \{\perp\}$ and a hyperterm \mathcal{G} .

We consider terms up to α -equivalence.

$$(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)$$

$$\frac{(\vec{\sigma}, \overleftarrow{\sigma}, M) \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', O)}{(\vec{\sigma}, \overleftarrow{\sigma}, \mathcal{G} \mid C[M] \mid \mathcal{G}') \rightsquigarrow (\vec{\sigma}', \overleftarrow{\sigma}', \mathcal{G} \mid C[O] \mid \mathcal{G}')}$$

$(\vec{\sigma}, \overleftarrow{\sigma}, (\lambda x.M)O) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M[O/x])$ and other rules of the underlying λ -cal.

for N closed:

$$(\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N], M[O/x])$$

$$(\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto \perp], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N], \text{abort})$$

$$(\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N'], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto O], \overleftarrow{\sigma}[I \mapsto N'], M[O/x])$$

$$(\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N'], (\overleftarrow{T} x.M)N) \rightsquigarrow (\vec{\sigma}[I \mapsto \perp], \overleftarrow{\sigma}[I \mapsto N'], \text{abort})$$

conversion rules involving [,]

Reduction (contd.)

$$(\vec{\sigma}, \overleftarrow{\sigma}, \lambda x.\text{abort}) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, \text{abort})$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \text{abort } N) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, \text{abort})$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, M\text{abort}) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, \text{abort})$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \overrightarrow{\lambda} x.\text{abort}) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, \text{abort})$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, \overleftarrow{\lambda} x.\text{abort}) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, \text{abort})$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, [\text{abort}, M]) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M)$$

$$(\vec{\sigma}, \overleftarrow{\sigma}, [M, \text{abort}]) \rightsquigarrow (\vec{\sigma}, \overleftarrow{\sigma}, M)$$

Example Typing

$$\frac{\frac{x: \text{Nat} \triangleright x: \text{Nat}}{\triangleright \vec{T}x.x: \text{Rat} \supset \text{Nat}} \quad \frac{y: \text{Rat} \triangleright y: \text{Rat}}{\triangleright \overleftarrow{T}y.y: \text{Nat} \supset \text{Rat}}}{\triangleright (\vec{T}x.x) \otimes (\overleftarrow{T}y.y)0: \text{Rat}} \quad \triangleright (\overleftarrow{T}y.y)0: \text{Rat}$$

$$\frac{\triangleright \text{inl}((\vec{T}x.x) \otimes (\overleftarrow{T}y.y)0): \text{Nat} \vee \text{Rat} \quad \triangleright \text{inr}((\overleftarrow{T}y.y)0): \text{Nat} \vee \text{Rat}}{\triangleright [\text{inl}((\vec{T}x.x) \otimes (\overleftarrow{T}y.y)0), \text{inr}((\overleftarrow{T}y.y)0)]: \text{Nat} \vee \text{Rat}}$$

$$\triangleright [\text{inl}((\vec{T}x.x) \otimes (\overleftarrow{T}y.y)0), \text{inr}((\overleftarrow{T}y.y)0)]: \text{Nat} \vee \text{Rat}$$

Example: reduction 1

$$\begin{aligned} & \left([], [], [\text{inl} \left((\overrightarrow{T} x.x) \right), \text{inr} \left((\overleftarrow{T} y.y)0 \right)] \right) \\ \rightsquigarrow & \left([I \mapsto \text{inl} \left((\overrightarrow{T} x.x) \right)], [], [\text{inl}(\text{abort}), \text{inr} \left((\overleftarrow{T} y.y)0 \right)] \right) \\ \rightsquigarrow & \left([I \mapsto \text{inl} \left((\overrightarrow{T} x.x) \right)], [], [\text{abort}, \text{inr} \left((\overleftarrow{T} y.y)0 \right)] \right) \\ \rightsquigarrow & \left([I \mapsto \text{inl} \left((\overrightarrow{T} x.x) \right)], [], \text{inr} \left((\overleftarrow{T} y.y)0 \right) \right) \\ \rightsquigarrow & \left([I \mapsto \text{inl} \left((\overrightarrow{T} x.x) \right)], [I \mapsto 0], \text{inr} \left((\overrightarrow{T} x.x) \right) \right) \end{aligned}$$

Example: reduction 2

$$\begin{aligned} & \left([], [], [\text{inl} \left((\overrightarrow{T} x.x) \text{🐼} \right), \text{inr} \left((\overleftarrow{T} y.y) 0 \right)] \right) \\ \rightsquigarrow & \left([], [I \mapsto 0], [\text{inl} \left((\overrightarrow{T} x.x) \text{🐼} \right), \text{inr} (\text{abort})] \right) \\ \rightsquigarrow & \left([], [I \mapsto 0], [\text{inl} \left((\overrightarrow{T} x.x) \text{🐼} \right), \text{abort}] \right) \\ \rightsquigarrow & \left([], [I \mapsto 0], \text{inl} \left((\overrightarrow{T} x.x) \text{🐼} \right) \right) \\ \rightsquigarrow & \left([I \mapsto \text{🐼}], [I \mapsto 0], \text{inl} (0) \right) \end{aligned}$$

The schedule chooses left or right.

Theorems

Rewriting system:

- 1 strongly normalizable (simulation)
- 2 not confluent

Asynchrony:

- 1 Sound for waitfree computation (by SN)
- 2 not complete for waitfree computation
 - ▶ add an modality representing each thread?
 - ▶ change the logical connectives (seems harder)

Demo

In Haskell:

$$\text{com}' \frac{\frac{\triangleright \text{getLine} : \text{String}}{\triangleright \mathbf{b}_a(\text{getLine}) : K_a \text{String}} \quad \frac{\triangleright \text{getLine} : \text{String}}{\triangleright \mathbf{b}_b(\text{getLine}) : K_b \text{String}}}{K_a(\text{String} \wedge \text{String}) \quad \mathbf{I} \quad K_b(\text{String} \wedge \text{String})}$$

Future work

(The traveling merchant shouts to)

- (computer scientists!)

Compare with lambda-mu [Parigot, 1992] and symmetric lambda [Barbanera and Berardi, 1994] .

- (logicians!)

classifying classical proofs on topology of schedules (Saks and Zaharoglou [1993])

- (engineers!)

Extend this logic (modal version) to higher-order to enable program extraction and verify some concurrent algorithms

- Arnon Avron. Hypersequents, logical consequence and intermediate logics for concurrency. *Ann. Math. Artif. Intell.*, 4:225–248, 1991.
- Matthias Baaz, Agata Ciabattoni, and Christian G. Fermüller. A natural deduction system for intuitionistic fuzzy logic. *Lectures on soft computing*, 2001.
- Christian Fermüller. Parallel dialogue games and hypersequents for intermediate logics. In *Automated Reasoning with Analytic Tableaux and Related Methods*, volume 2796 of *LNCS*, pages 48–64. Springer, 2003.
- Michel Parigot. $\lambda\mu$ -Calculus: An algorithmic interpretation of classical natural deduction. In *LNCS*, pages 190–201. Springer, 1992.
- Franco Barbanera and Stefano Berardi. A symmetric lambda calculus for classical program extraction. In *Theoretical Aspects of Computer Software*, volume 789 of *LNCS*, pages 495–515. Springer, 1994.
- Michael Saks and Fotios Zaharoglou. Wait-free k -set agreement is impossible: the topology of public knowledge. pages 101–110, 1993.