The atomic $\lambda\mu$ -calculus

Fanny He f.he@bath.ac.uk



The atomic $\lambda\mu$ -calculus

Classical logic and computation

Deep Inference and computation

The atomic $\lambda\mu$ -calculus: $\Lambda\mu S_a$

Properties of $\Lambda \mu S_a$

Classical Logic and Curry-Howard

Logic	Computation
Intuitionistic	
Formula	Program type
Proof	Program
Natural Deduction	The λ -calculus
$A \rightarrow (B \rightarrow A)$	$A \rightarrow (B \rightarrow A)$ types $\lambda x^A . \lambda y^B . x$
Classical	
Natural Deduction	?
$\neg \neg A \rightarrow A$?

Lafont's critical pair

$$\begin{array}{ccc} \Pi_1 & \Pi_2 \\ \vdots & \vdots \\ \hline \frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta} \text{Lwk} & \frac{\Gamma' \vdash \Delta'}{\Gamma' \vdash \Delta', A} \text{Rwk} \\ \hline \Gamma, \Gamma' \vdash \Delta, \Delta' & \text{Cut} \end{array}$$

There are two proofs of $\Gamma, \Gamma' \vdash \Delta, \Delta'$ without cuts:

$$\begin{array}{c}
\Pi_{1} \\
\vdots \\
\Gamma \vdash \Delta \\
\hline
\Gamma, \Gamma' \vdash \Delta, \Delta'
\end{array} wk$$

$$\begin{array}{c}
\Pi_{2} \\
\vdots \\
\hline
\Gamma, \Gamma' \vdash \Delta'
\end{array} wk$$

Curry-Howard correspondence

Logic	Computation
Intuitionistic	
Formula	Program type
Proof	Program
Natural Deduction	The λ -calculus
A o (B o A)	$A \rightarrow (B \rightarrow A)$ types $\lambda x^A . \lambda y^B . x$
Classical	
Natural Deduction	The $\lambda\mu$ -calculus [Parigot]
$ eg \neg A \rightarrow A \ (; \bot)$	Types $\lambda y.\mu\alpha.((y) \lambda x.\mu\delta.(x)\alpha)\phi$
$((A \rightarrow B) \rightarrow A) \rightarrow A$	Types $\lambda y.\mu\alpha.((y)\lambda x.\mu\delta.(x)\alpha)\alpha$
Double negation translation	CPS translation

Deep Inference and Curry-Howard

- Apply inference rules at any depth
- Open deduction [Guglielmi, Gundersen, Parigot]

$$\frac{A \wedge B}{A \wedge D} \equiv \frac{A \wedge B}{C \wedge D} \equiv \frac{A}{C} \wedge \frac{B}{D}$$

Logic	Computation
Intuitionistic	
Formula	Program type
Proof	Program
Open Deduction	The atomic λ -calculus Λ_a [Gundersen, Heijltjes, Parigot]
$A \rightarrow (B \rightarrow A)$	$A \rightarrow (B \rightarrow A)$ types $\lambda x^A . \lambda y^B . x [\leftarrow y]$
Classical	
Open Deduction	?
$\neg \neg A o A$?

The atomic λ -calculus Λ_a (informally)

- A linear λ-calculus
 - with sharings (similar to explicit substitutions)
 - with atomic duplications (similar to optimal reduction graphs)

Type system in Open Deduction:

(for
$$\rightarrow$$
, $A = C \rightarrow B$ and $C = A \rightarrow D$)

Deep Inference and Λ_a

Medial rule:

$$\frac{(A \lor B) \to (C \land D)}{(A \to C) \land (B \to D)} \mathsf{m}$$

Allows atomic contraction:

$$\frac{A \to B}{(A \to B) \land (A \to B)} \land \leadsto \frac{\frac{A \lor A}{A} \lor \to \frac{B}{B \land B} \land m}{(A \to B) \land (A \to B)}$$
m

The atomic λ -calculus allows duplication of individual constructors:

$$\frac{\lambda x.t:A \to B}{(\lambda x.t:A \to B) \land (\lambda x.t:A \to B)} \land \leadsto \frac{(x:A) \lor (x:A)}{(x:A)} \lor \to \frac{(t:B)}{(t:B) \land (t:B)} \land \underset{\mathbf{m}}{\triangle}$$

Properties of Λ_a

- Confluence, typed SN, subject reduction
- PSN
- Fully-lazy sharing
 - Share resources
 - Restrict duplications

Deep Inference and Curry-Howard

Logic	Computation
Rules reduced to their atomic forms	Sharing of individual constructors
Intuitionistic	
Formula	Program type
Proof	Program
Open Deduction	Λ_a
A o (B o A)	$A \rightarrow (B \rightarrow A)$ types $\lambda x^A . \lambda y^B . x [\leftarrow y]$
Classical	
Open Deduction	The atomic $\lambda\mu$ -calculus $\Lambda\mu S_a$
$\neg \neg A o A$	$\lambda y.\mu\alpha.((y) \lambda x.\mu\delta.(x)\alpha[\leftarrow \delta])\phi$

How to construct the atomic $\lambda\mu$ -calculus

Aims:

- Keep the good properties of $\lambda\mu$ and Λ_a
- Give a computational interpretation of a classical deep inference system

Approach:

- Naturally extend $\lambda\mu$ and Λ_a
- Adapt classical proofs with multiple conclusions to open deduction

Syntax of $\lambda\mu$ and Λ_a

■ The $\lambda\mu$ -calculus:

Terms
$$t, u ::= x \mid \lambda x.t \mid (t)u \mid \mu \alpha.n$$

 μ -variables $::= \alpha$
Names $n ::= (t)\alpha$
 $(\mu \alpha.n)u \longrightarrow_{\mu} \mu \alpha. n\{(w) u \alpha/(w) \alpha\}$

■ The atomic λ -calculus:

Terms
$$t, u ::= x \mid \lambda x.t \mid (t)u \mid u[\phi]$$

 λ -tuples $t^p ::= \langle t_1, \dots, t_p \rangle \mid t^p[\phi]$
Closures $[\phi], [\psi] ::= [\vec{x_q} \leftarrow t] \mid [\vec{x_q} \leftarrow \lambda y.t^q]$

Syntax of $\lambda\mu$ and Λ_a

■ The atomic λ -calculus:

Terms
$$t, u := x \mid \lambda x.t \mid (t)u \mid u[\phi]$$

 λ -tuples $t^p := \langle t_1, \dots, t_p \rangle \mid t^p[\phi]$
Closures $[\phi], [\psi] := [\vec{x_q} \leftarrow t] \mid [\vec{x_q} \leftarrow \lambda y.t^q]$

Examples:

$$(\lambda x.(y) \times x) t \to_{\beta} (y) t t$$

$$(\lambda x.(y) \times_1 \times_2 [x_1, x_2 \leftarrow x]) t \to_{\beta} (y) \times_1 \times_2 [x_1, x_2 \leftarrow t]$$

$$[x_1, x_2 \leftarrow \lambda z.z] \leadsto [x_1, x_2 \leftarrow \lambda z.\langle z_1, z_2 \rangle [z_1, z_2 \leftarrow z]]$$

$$\leadsto \{(\lambda z_1.z_1)/x_1\}\{(\lambda z_2.z_2)/x_2\}$$

First syntax towards $\Lambda \mu S_a$

■ How to define sharings with names/ μ -variables?

```
\mu-variables ::= \alpha
Names n ::= (t)\alpha \mid n[\phi]
Terms t, u ::= x \mid \lambda x.t \mid (t)u \mid \mu\alpha.n \mid u[\phi]
\lambda-tuples t^p ::= \langle t_1, \ldots, t_p \rangle \mid t^p[\phi]
Closures [\phi], [\psi] ::= [\vec{x_q} \leftarrow t] \mid [\vec{x_q} \leftarrow \lambda y.t^q] \mid ?????
```

First syntax towards $\Lambda \mu S_a$

An example:

 \blacksquare β -reduction in Λ_a :

$$(\lambda x.(y) \times x) t \to_{\beta} (y) t t$$
$$(\lambda x.(y) x_1 x_2[x_1, x_2 \leftarrow x]) t \to_{\beta} (y) x_1 x_2[x_1, x_2 \leftarrow t]$$

 \blacksquare μ -reduction with explicit sharings:

$$(\mu\alpha.(\mu\beta.(x)\alpha)\alpha)t \to_{\mu} \mu\alpha.(\mu\beta.(x)t\alpha)t\alpha$$
$$(\mu\alpha.(\mu\beta.(x)\alpha_1)\alpha_2[\alpha_1,\alpha_2\leftarrow\alpha])t \to_{\mu} ?????$$

• $[\alpha_1, \alpha_2 \leftarrow (t) \alpha]$ would not work

Syntax with streams

- Left associativity of application: no direct way to share names
- Three kinds of expressions: terms, names, μ -variables
- Solution: introduce streams [Saurin-Gaboardi], get rid of names
- The $\Lambda \mu S$ -calculus:

Streams
$$S, T := \alpha \mid t \circ S$$

Terms $t, u := x \mid \lambda x.t \mid (t)u \mid (t)S \mid \mu \alpha.t$
 $(\mu \alpha.t)u \longrightarrow_{\mu} \mu \alpha.t \{u \circ \alpha/\alpha\}$

Syntax with streams

■ The atomic $\lambda\mu$ -calculus $\Lambda\mu S_a$:

```
Closures [\phi], [\psi] ::= [\vec{x_q} \leftarrow t] \mid [\vec{\gamma_q} \leftarrow S] \mid [\vec{x_q} \leftarrow \lambda y.t^q] \mid [\vec{x_q} \leftarrow \mu \beta.t^q]

Streams S, T ::= \alpha \mid t \circ S \mid S[\phi]

Terms t, u ::= x \mid \lambda x.t \mid (t)u \mid (t)S \mid \mu \alpha.t \mid u[\phi]

\lambda-tuples t^p ::= \langle t_1, \dots, t_p \rangle \mid t^p[\phi]
```

Reduction rules of $\Lambda \mu S_a$

- β -reductions
- \blacksquare μ -reductions
- Sharing reductions \leadsto_s :
 - Moving closures, compounding sharings
 - Duplication rules: allows duplications of (smaller) subterms

Type system for Λ_a

$$\begin{array}{ccccc}
A & B & A & B & A & B \\
\downarrow & ::= & A & \downarrow & A & A & B & B & B \\
C & C & D & C & D & C & D & B
\end{array}$$

Inference rules:

$$\frac{A}{A \wedge \cdots \wedge A} \land \qquad \frac{B}{A \to (A \wedge B)} \land \qquad \frac{A \wedge (A \to B)}{B} \circ$$

Medial rule:

$$\frac{(A \vee B) \to (C \wedge D)}{(A \to C) \wedge (B \to D)}^{m}$$

Type system for $\Lambda \mu S_a$: first approach

Classical Natural Deduction:

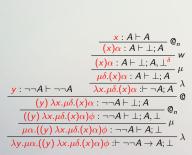
$$\frac{\Gamma \vdash C; \Delta}{\Gamma' \vdash C'; \Delta'} r$$

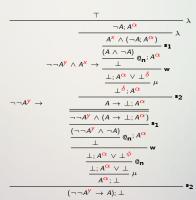
Multiple conclusions/disjunctions: needs switch rules

$$\frac{(A;\Delta) \wedge (B;\Delta')}{(A \wedge B);\Delta \vee \Delta'} s_1 \qquad \frac{A \rightarrow (B;\Delta)}{(A \rightarrow B);\Delta} s_2$$

Type system for $\Lambda \mu S_a$: first approach

Example (Double negation elimination):





Type system for $\Lambda \mu S_a$

- Multiple conclusions/disjunctions: needs switch rules
- Multiple conclusions come from μ -variables or body of μ -abstractions (of type \perp)

If
$$t \equiv \begin{array}{c} \Gamma \wedge A^{\times} \\ t \downarrow \\ C \end{array}$$

Then

$$\lambda x.t \equiv \frac{\Gamma}{A^{x} \to t}$$

$$C$$

and Ax remains untouched

We could similarly write:

$$\mu\alpha.t \equiv \frac{\Gamma}{\Gamma \wedge \neg A}^{\mu} \vee A^{\alpha}$$

$$\perp$$

$$A$$

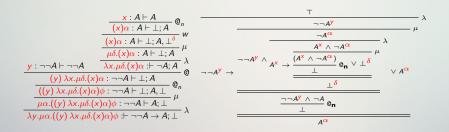
and A^{α} remains untouched

Type system

$$x \equiv A^{X} \qquad \alpha \equiv \neg A \qquad \lambda x.t \equiv \frac{\Gamma}{A^{X}} \xrightarrow{\Gamma \land A^{X}} \lambda \qquad \frac{\Gamma}{\Gamma \land \neg A} \xrightarrow{\mu} \mu$$

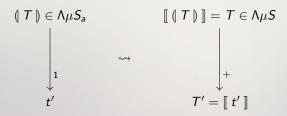
$$(t)u \equiv \frac{\mathbf{r}}{A} \xrightarrow{\Lambda} \frac{\Gamma}{B} \mathbf{u} \qquad (t)S \equiv \frac{\mathbf{r}}{A} \xrightarrow{\Lambda} \mathbf{u} \qquad (t)S$$

Example: Double negation elimination



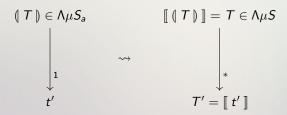
For any term $T \in \Lambda \mu S$, if T is strongly normalizing then its translation $(T) \in \Lambda \mu S_a$ is strongly normalizing.

We could show that one step of reduction in $\Lambda \mu S_a$ leads to at least one step in $\Lambda \mu S$:



Problem 1:

- Reductions in $\Lambda \mu S_a$ can be β , μ -reductions, or \leadsto_s
- For $t, u \in \Lambda \mu S_a$, if $t \leadsto_s u$ then $\llbracket t \rrbracket = \llbracket u \rrbracket$
- One step of \leadsto_s in $\Lambda \mu S_a$ corresponds to zero steps in $\Lambda \mu S_a$



Solution: \leadsto_s is strongly normalizing.

If (T) has an infinite reduction path, it will look like:

$$(\!(T)\!) \leadsto_s^* t_1 \to_{\beta,\mu}^+ t_2 \leadsto_s^* \ldots \leadsto_s^* t_n \to_{\beta,\mu}^\infty \ldots$$

lacktriangle We can concentrate on eta and μ -reductions

Problem 2: infinite reductions can occur inside weakenings.

- Example: $T = (\mu \alpha.(x) \beta) \Omega$
- $\blacksquare \ (\mid T \mid) = (\mu \alpha.(x) \beta [\leftarrow \alpha]) \Omega \longrightarrow_{\mu} t' = \mu \alpha.(x) \beta [\leftarrow \Omega \circ \alpha]$
- But $\llbracket t' \rrbracket = \mu \alpha.(x) \beta$ is in normal form
- \blacksquare There exists an infinite reduction path from t', but not from $[\![\ t'\]\!]$

An auxiliary calculus is introduced, the weakening calculus $\Lambda \mu S_w$, an extension of $\Lambda \mu S$ with explicit weakenings.

Find a strategy keeping infinite reductions outside of weakenings.

Other properties

- Fully-lazy sharing
- Typed calculus: strong normalization, subject reduction