Minimal Classical Logic and Control Operators

Zena M. Ariola
University of Oregon
USA

Hugo Herbelin
INRIA-Futurs
France

ICALP - 3 July 2003

Outline

- Introduction
  - Minimal, Intuitionistic and Classical Logic
  - λ-calculus control operators
- Minimal Classical Logic
- Computational interpretation of $\bot \to A$
- A new variant of Felleisen $\lambda_{\mathcal{C}}$ calculus

Minimal Logic and λ-calculus

$$A ::= X \mid A \to A$$

$$t, u ::= x \mid \lambda x.t \mid tu$$

$$\frac{}{\Gamma,\ x : A \vdash\ x : A}$$

$$\frac{\Gamma,\ x : A \vdash\ t : B}{\Gamma \vdash\ \lambda x.t : A \to B} \qquad \frac{\Gamma \vdash\ t : A \to B \quad \Gamma \vdash\ u : A}{\Gamma \vdash\ tu : B}$$

Control Operators

- $\mathcal{K}$ (callcc): $\lambda x.\mathcal{K}(\lambda k.1 + (\text{if } x = 0 \text{ then } k2 \text{ else } 3)) + 8$

  Name k is bound to the continuation which adds 8:

```
             acc := 1 + if x=0 then {acc:=2;goto k} else 3
 label k   : acc := acc+8
             return acc
```

  returns $2 + 8$ if $x = 0$ and $(1 + 3) + 8$ otherwise

- $\mathcal{C}$: $\lambda x.\mathcal{C}(\lambda k.1 + (\text{if } x = 0 \text{ then } k2 \text{ else } 3)) + 8$

  Additional "$\mathcal{A}bort$" to "toplevel" in case of non exceptional result

```
             acc := 1 + if x=0 then {acc:=2;goto k} else 3
             goto top
 label k   : acc := acc+8
 label top: return acc
```

  returns $2 + 8$ if $x = 0$ and $1 + 3$ otherwise

Intuitionistic - Classical Logic and Control Operators (Griffin '90)

- Intuitionistic: Minimal Logic + Ex Falso Quodlibet ($\bot \to A$)

$$\frac{\Gamma \vdash \ t : \bot}{\Gamma \vdash \ \mathcal{A}bort(t) : A}$$

- Classical: Intuitionistic + Double Negation ($\neg\neg A \to A$)

$$\frac{\Gamma \vdash \ t : \neg\neg A}{\Gamma \vdash \ \mathcal{C}(t) : A}$$

- Equivalent formulations of Classical Logic in Intuitionistic Logic

Excluded Middle	$A \vee \neg A$
Pierce law	$((A \rightarrow B) \rightarrow A) \rightarrow A$
Double negation	$\neg\neg A \rightarrow A$

- Felleisen: $\mathcal{C}$ is more expressive than $\mathcal{K}$ ($\mathcal{C}$ can't be defined from $\mathcal{K}$)

$$\mathcal{K}(\lambda k.M) = \mathcal{C}(\lambda k.kM)$$

$$\mathcal{C}(\lambda k.M) = \mathcal{K}(\lambda k.\mathcal{A}bort(M))$$

Are those axioms really equivalent?

$\Longrightarrow$ We study them in the context of Minimal Logic

$$\neg\neg A \to A$$

$$\Downarrow \quad \not\Uparrow$$

$$((A \to B) \to A) \to A$$

$$\Downarrow \quad \not\Uparrow$$

$$A \vee \neg A$$

$\Longrightarrow$ Minimal Classical Logic

Minimal Classical Logic

- Minimal Classical Logic:

  Minimal Logic + Pierce law

  λ-calculus + $\mathcal{K}$ (callcc)

- Classical Logic:

  $$\text{Minimal Logic} + \overbrace{\text{Peirce law} + (\bot \rightarrow A)}^{= \neg\neg A \rightarrow A}$$

  $$\lambda\text{-calculus} + \underbrace{\mathcal{K} + \mathcal{A}bort}_{= \ \mathcal{C}}$$

Minimal Classical Natural Deduction and Parigot $\lambda\mu$-calculus

Sequents: $\begin{cases} \Gamma \vdash A; \Delta & t, u ::= x \mid \lambda x.t \mid tu \mid \mu\alpha.c \ (terms) \\ \Gamma \vdash ; \Delta & c ::= [\alpha]t \qquad (commands) \end{cases}$

$$\frac{}{\Gamma, x : A \vdash_{\mathrm{MC}} x : A; \Delta}$$

$$\frac{\Gamma, x : A \vdash_{\mathrm{MC}} t : B; \Delta}{\Gamma \vdash_{\mathrm{MC}} \lambda x.t : A \to B; \Delta} \qquad \frac{\Gamma \vdash_{\mathrm{MC}} t : A \to B, \Delta \quad \Gamma \vdash_{\mathrm{MC}} u : A, \Delta}{\Gamma \vdash_{\mathrm{MC}} tu : B, \Delta}$$

$$\frac{c : (\Gamma \vdash_{\mathrm{MC}} ; A^{\alpha}, \Delta)}{\Gamma \vdash_{\mathrm{MC}} \mu\alpha.c : A; \Delta}\ \textit{Activate} \qquad \frac{\Gamma \vdash_{\mathrm{MC}} t : A; A^{\alpha}, \Delta}{[\alpha]t : (\Gamma \vdash_{\mathrm{MC}} ; A, \Delta)}\ \textit{Passivate}$$

Properties of Minimal Classical Natural Deduction

Prop: $\Gamma \vdash_{\mathrm{MC}} A$ iff $\Gamma, PL \vdash A$

Prop: there is no t of type $\neg\neg A \rightarrow A$

But we can derive $\vdash_{\mathrm{MC}} \lambda y.\mu\alpha.[\gamma](y\ \lambda x.\mu\delta.[\alpha]x) : \neg\neg A \rightarrow A; \bot^{\gamma}$

where a free continuation variable γ of type $\bot$ is needed.

Possible solution (Parigot, de Groote, Ong) : hide the variables of type $\bot$ in the typing system.

Our interpretation: this free variable denotes abortion to the "toplevel".

Classical Natural Deduction and Parigot $\lambda\mu$-*top*-calculus

$$\frac{}{\Gamma, x : A \vdash_{\mathrm{C}} x : A; \Delta}$$

$$\frac{\Gamma, x : A \vdash_{\mathrm{C}} t : B; \Delta}{\Gamma \vdash_{\mathrm{C}} \lambda x.t : A \to B; \Delta}$$

$$\frac{\Gamma \vdash_{\mathrm{C}} t : A \to B, \Delta \quad \Gamma \vdash_{\mathrm{C}} u : A, \Delta}{\Gamma \vdash_{\mathrm{C}} tu : B, \Delta}$$

$$\frac{c : (\Gamma \vdash_{\mathrm{C}}; A^{\alpha}, \Delta)}{\Gamma \vdash_{\mathrm{C}} \mu\alpha.c : A; \Delta} \ \textit{Activate}$$

$$\frac{\Gamma \vdash_{\mathrm{C}} t : A; A^{\alpha}, \Delta}{[\alpha]t : (\Gamma \vdash_{\mathrm{C}}; A, \Delta)} \ \textit{Passivate}$$

$$\frac{\Gamma \vdash_{\mathrm{C}} t : \bot; \Delta}{[top]t : (\Gamma \vdash_{\mathrm{MC}}; \Delta)}$$

$\mathcal{A}bort$, $\bot$, and the toplevel

Our analysis puts in the light the existence of an implicit continuation variable of type $\bot$

- its logical interpretation is the false formula
- its computational interpretation is the type of the toplevel

It allows to precisely decompose Felleisen's $\mathcal{A}bort$ operator as a "`throw`" to the toplevel:

$$\mathcal{A}bort = \lambda x.\mu d.[top]x : \bot \rightarrow A \quad \text{for } d \text{ fresh}$$

It pushes also to identify the logical formula $\bot$ with the toplevel type of a program (as done by Griffin). This conforms with the standard A-translation trick where $\bot$ is precisely replaced by the top statement of a proof.

From $\lambda\mu$-calculus to a variant of $\lambda_{\mathcal{C}}$-calculus

Parigot $\lambda\mu$-calculus:

- $\bot$ not needed to express minimal classical logic
- $\bot$ interpreted as an abstract type (which may be empty like in the standard logical interpretation, or inhabited as is the toplevel type in the standard computational interpretation).

One-conclusion classical logic needs another kind of $\bot$ to mimic the role of multiple conclusions. Let's call it $\bot^c$. This leads to a refinement of classical logic obtained by constraining the axiom scheme $\neg\neg A \rightarrow A$ to be of the form $\neg_c\neg_c A \rightarrow A$.

The formal formula $\bot^c$ is naturally interpreted as an empty type.

Prawitz Natural Deduction revisited

(an introduction to λ-$\mathcal{C}^{-}$-*top* calculus)

$$\frac{}{\Gamma, x : A \vdash_{\mathrm{RAA}} x : A}$$

$$\frac{\Gamma, x : A \vdash_{\mathrm{RAA}} t : B}{\Gamma \vdash_{\mathrm{RAA}} \lambda x.t : A \to B} \qquad \frac{\Gamma \vdash_{\mathrm{RAA}} t : A \to B \qquad \Gamma \vdash_{\mathrm{RAA}} u : A}{\Gamma \vdash_{\mathrm{RAA}} tu : B}$$

$$\frac{\Gamma, k : \neg_{\mathcal{C}} A \vdash_{\mathrm{RAA}} c : \bot^{\mathcal{C}}}{\Gamma \vdash_{\mathrm{RAA}} \mathcal{C}^{-}(\lambda k.c) : A}\ RAA_c \qquad \frac{\Gamma \vdash_{\mathrm{RAA}} t : \bot^{\mathcal{C}}}{\Gamma \vdash_{\mathrm{RAA}} \mathcal{C}^{-}(\lambda d.c) : A}\ RAA'_c$$

$$\frac{\Gamma \vdash_{\mathrm{RAA}} t : \bot}{\Gamma \vdash_{\mathrm{RAA}} \mathrm{top}\, t : \bot^{\mathcal{C}}}$$

The λ-$\mathcal{C}^{-}$-*top* calculus

$$t, u ::= x \mid \lambda x.t \mid tu \mid \mathcal{C}^{-}(\lambda k.c)$$

$$c ::= kt \mid \mathrm{top}\, t$$

Felleisen's operators in λ-$\mathcal{C}^{-}$-*top* calculus

$$\mathcal{K} = \lambda x.\mathcal{C}^{-}(\lambda k.k(xk))$$

$$\mathcal{C} = \lambda x.\mathcal{C}^{-}(\lambda k.\mathrm{top}\,(xk))$$

$$\mathcal{A}bort = \lambda x.\mathcal{C}^{-}(\lambda k.\mathrm{top}\, x)$$

Call-by-value reduction in λ-$\mathcal{C}^-$ and λ-$\mathcal{C}^-$-*top* calculi

$(v ::= x \mid \lambda x.t)$

$$
\begin{array}{lll}
\beta : & (\lambda x.t)v & \rightarrow t[x := v] \\
\mathcal{C}^-_{elim} : & \mathcal{C}^-(\lambda k.kt) & \rightarrow t \quad k \notin FV(t) \\
\mathcal{C}^-_{L} : & \mathcal{C}^-(\lambda k.t)u & \rightarrow \mathcal{C}^-(\lambda k.t[k\ (wu)/k\ w]) \\
\mathcal{C}^-_{R} : & v\mathcal{C}^-(\lambda k.t) & \rightarrow \mathcal{C}^-(\lambda k.t[k\ (vw)/k\ w]) \\
\mathcal{C}^-_{idem} : & \mathcal{C}^-(\lambda k.k'\mathcal{C}^-(\lambda q.u)) & \rightarrow \mathcal{C}^-(\lambda k.u[k'/q])
\end{array}
$$

Prop: The reduction rules are typable and enjoy subject reduction.

They are confluent. Similarly for call-by-name.