# Introduction to Parigot's $\lambda\mu$-Calculus

Andreas Abel

March 30, 2001

**Abstract**

The first-order fragment of Parigot's $\lambda\mu$-Calculus is formalization of the implicational fragment of classical logic. The proof term syntax, proof reductions and normal forms are gently introduced in this article.

## 1 Proof Terms and Proof Rules

In his seminal 1992 paper Parigot [Par92] introduced the $\lambda\mu$-calculus. We will restrict on the first-order fragment, which assigns proof terms to classical natural deductions in propositional logic. Consider the $\to$-fragment plus the rule for absurdity elimination together with a rule how to obtain absurdity:

$$\dfrac{\begin{array}{c}\overline{A}^{\,x}\\ \vdots\\ B\end{array}}{A \to B}\to\mathcal{I}^x \qquad \dfrac{A\to B \qquad A}{B}\to\mathcal{E} \qquad \dfrac{\begin{array}{c}\overline{\neg A}^{\,a}\\ \vdots\\ \bot\end{array}}{A}\bot\mathcal{E}^a \qquad \dfrac{\neg A \qquad A}{\bot}$$

Parigot's insight was to introduce a notion of absurdity on the meta-level, which we will denote with $-$. Consequently negated propositions were interpreted as *passive* types or *cotypes* [Bie98], which we will denote with $\overline{A}$. This yields the following proof term assignment:

$$\dfrac{\begin{array}{c}\overline{x : A}\\ \vdots\\ M : B\end{array}}{\lambda x.M : A \to B}\,\mathsf{lam}^x \qquad \dfrac{M_1 : A \to B \qquad M_2 : A}{M_1\, M_2 : B}\,\mathsf{app}$$

$$\dfrac{\begin{array}{c}\overline{a : \overline{A}}\\ \vdots\\ N : -\end{array}}{\mu a.N : A}\,\mathsf{catch}^a \qquad \dfrac{K : \overline{A} \qquad M : A}{[K]M : -}\,\mathsf{throw}$$

The names $\mathsf{catch}$ and $\mathsf{throw}$ are borrowed from the realm of programming and can be justified by a computational interpretation. Note that a cotype never appears in the conclusion of a rule. Thus we can restrict the throw rule to variables of cotype. Furthermore we introduce contexts $\Gamma$ for $\lambda$-variables and $\Delta$ for $\mu$-variables:

$$\frac{x : A \in \Gamma}{\Gamma \vdash x : A, \Delta}\,\textsf{var} \qquad \frac{\Gamma, x : A \vdash M : B, \Delta}{\Gamma \vdash \lambda x.M : A \to B, \Delta}\,\textsf{lam}^x$$

$$\frac{\Gamma \vdash M_1 : A \to B, \Delta \qquad \Gamma \vdash M_2 : A, \Delta}{\Gamma \vdash M_1\, M_2 : B, \Delta}\,\textsf{app}$$

$$\frac{\Gamma \vdash N : -, \Delta, a : \overline{A}}{\Gamma \vdash \mu a.N : A, \Delta}\,\textsf{catch}^a \qquad \frac{a : \overline{A} \in \Delta \qquad \Gamma \vdash M : A, \Delta}{\Gamma \vdash [a]M : -, \Delta}\,\textsf{throw}$$

Traditionally the context $\Delta$ of $\mu$-variables is written on the right side. The reason for this convention is that the types in $\Delta$ are interpreted as the passive types of the term $M$ in the judgment; they can be activated by a throw.

As an example we give a deduction of the Pierce-formula and the corresponding $\lambda\mu$-term:

$$\dfrac{}{\overline{B}}\,b$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\overline{A}}\,a \qquad \dfrac{\dfrac{}{(A \to B) \to A}\,x \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{\overline{A}}\,a \quad \dfrac{}{A}\,y}{-}\,\textsf{throw}}{B}\,\textsf{catch}^b}{A \to B}\,\textsf{lam}^y}{A}\,\textsf{app}}{-}\,\textsf{throw}}{A}\,\textsf{catch}^a}{((A \to B) \to A) \to A}\,\textsf{lam}^x$$

From this deduction tree we can directly read off the proof term:

$$\cdot \vdash \lambda x.\mu a.[a](x\, \lambda y.\mu b.[a]y) : ((A \to B) \to A) \to A, \cdot$$

In the following we will consequently denote $\lambda$-variables by $x$, $y$, $z$, ... and $\mu$-variables by $a$, $b$, $c$, ...

## 2 Reductions

The $\lambda\mu$-calculus comes with two reductions both of which are required to normalize arbitrary proofs into proofs that have the subformula property. The first is ordinary $\beta$-reduction (Parigot's "logical reduction"):

$$\dfrac{\dfrac{\begin{matrix}\overline{A}^{\,x}\\ \vdots\,\mathcal{D}\\ B\end{matrix}}{A\to B}\,\mathsf{lam}^x \qquad \begin{matrix}\vdots\,\mathcal{E}\\ A\end{matrix}}{B}\,\mathsf{app} \qquad \longrightarrow_\beta \qquad \begin{matrix}\vdots\,\mathcal{E}\\ A\\ \vdots\,\mathcal{D}\\ B\end{matrix} \qquad\qquad rcl$$

$$(\lambda x.M(x))N \quad \longrightarrow_\beta \quad M(N)$$

$$(\lambda x.M)N \quad \longrightarrow_\beta \quad [N/x]M$$

Since the rules $\mathsf{catch}$ and $\mathsf{throw}$ do not increase the size of any type involved (unlike $\mathsf{lam}$), to maintain the subformula property there is no reduction necessary for the new constructs *themselves*. However, a $\mathsf{catch}$-$\mathsf{throw}$-pair could split an $\mathsf{app}$-$\mathsf{lam}$-pair, preventing $\beta$-reduction. To fix that problem, we introduce the following permutative conversion (Parigot's "structural reduction"):

$$\dfrac{\dfrac{\begin{matrix}\dfrac{\overline{\overline{A\to B}}^{\,a} \qquad \begin{matrix}\vdots\,\mathcal{D}\\ A\to B\end{matrix}}{-}\,\mathsf{throw}\\ \vdots\,\mathcal{E}\\ -\end{matrix}}{A\to B}\,\mathsf{catch}^a \qquad \begin{matrix}\vdots\,\mathcal{F}\\ A\end{matrix}}{B}\,\mathsf{app} \qquad \longrightarrow_\pi \qquad \dfrac{\begin{matrix}\dfrac{\overline{\overline{B}}^{\,b} \qquad \dfrac{\begin{matrix}\vdots\,\mathcal{D}\\ A\to B\end{matrix} \quad \begin{matrix}\vdots\,\mathcal{F}\\ A\end{matrix}}{B}\,\mathsf{app}}{-}\,\mathsf{throw}\\ \vdots\,\mathcal{E}\\ -\end{matrix}}{B}\,\mathsf{catch}^b$$

$$(\mu a.N([a]\bullet))\,M \quad \longrightarrow_\pi \quad \mu b.N([b](\bullet\,M))$$

$$(\mu a.N)\,M \quad \longrightarrow_\pi \quad \mu b.\,[[b](\bullet\,M)/[a]\bullet]\,N$$

Specifying this reduction for terms involves a new kind of substitution, which we will call $\mu$-substitution. It will replace a special *context* $[a]\bullet$ for another one, $C(\bullet)$—a context is a term with a hole "$\bullet$". In the $\longrightarrow_\pi$-reduction it is used for the special $C(\bullet) = [b](\bullet\,M)$. To simplify the notation and for reasons that will become more clear later we write $\mu$-substitution just

$$[C/a]M \qquad \text{instead of} \qquad [C(\bullet)/[a]\bullet]M$$

It can be defined by recursion as follows:

$$\begin{array}{lcl}
[C/a]([a]M) & = & C([C/a]M)\\
[C/a]([b]M) & = & [b]([C/a]M) \quad (b \neq a)\\
[C/a](\mu b.N) & = & \mu b.([C/a]N) \quad (b \neq a, b \notin \mu FV(C))\\
[C/a]x & = & x\\
[C/a](\lambda x.M) & = & \lambda x.([C/a]M)\\
[C/a](M_1\,M_2) & = & ([C/a]M_1)([C/a]M_2)
\end{array}$$

Note that the first case requires ordinary capture avoiding substitution: $C([C/a]M) = [[C/a]M/\bullet]C$.

Not motivated by the subformula property, are two more reductions that eliminate detours. The first deletes a throw-catch-pair. Since it is analogous to $\beta$-reduction, we call it $\longrightarrow_{\mu\beta}$ (Parigot's "renaming"):

$$\dfrac{\dfrac{}{\overline{A}}\,a \qquad \dfrac{\begin{array}{c}\vdots\,\mathcal{D}\\ -\end{array}}{A}\,\mathsf{catch}^b}{-}\,\mathsf{throw} \qquad \longrightarrow_{\mu\beta} \qquad \begin{array}{c}\vdots\,[a/b]\mathcal{D}\\ A\end{array}$$

$$[a]\mu b.N \quad \longrightarrow_{\mu\beta} \quad [a/b]N$$

The second one eliminates a throw-catch-pair. It resembles the $\eta$-rule, hence we call it $\longrightarrow_{\mu\eta}$. Parigot mentioned it but did not include it in the calculus:

$$\dfrac{\dfrac{\dfrac{}{\overline{A}}\,a \qquad \begin{array}{c}\vdots\,\mathcal{D}\\ A\end{array}}{-}\,\mathsf{throw}}{A}\,\mathsf{catch}^a \qquad \longrightarrow_{\mu\eta} \qquad \begin{array}{c}\vdots\,\mathcal{D}\\ A\end{array} \quad (a \notin \mathcal{D})$$

$$\mu a.[a]M \quad \longrightarrow_{\mu\eta} \quad M \qquad (a \notin \mu FV(M))$$

# 3 Normal Forms and Type-Checking

We can classify normal forms with regard to $\beta$ and $\pi$-reduction in the following calculus. This calculus implements bidirectional type-checking for the $\lambda\mu$-calculus. We introduce the following two judgments:

$$\begin{array}{ll} \Gamma \vdash M \downarrow A, \Delta & M \text{ is a neutral proof of } A \\ \Gamma \vdash M \Uparrow A, \Delta & M \text{ is a normal proof of } A \end{array}$$

We allow $M$ to be a named term $N$ and $A$ to be the meta-absurdity $-$. The following rules implement the judgments:

$$\dfrac{x : A \in \Gamma}{\Gamma \vdash x \downarrow A, \Delta} \downarrow \mathsf{var} \qquad \dfrac{\Gamma \vdash M_1 \downarrow A \to B, \Delta \qquad \Gamma \vdash M_2 \Uparrow A, \Delta}{\Gamma \vdash M_1\, M_2 \downarrow B, \Delta} \downarrow \mathsf{app}$$

$$\dfrac{a : \overline{A} \in \Delta \qquad \Gamma \vdash M \Uparrow A, \Delta}{\Gamma \vdash [a]M \downarrow -, \Delta} \downarrow \mathsf{throw} \qquad \dfrac{\Gamma \vdash M \downarrow A, \Delta}{\Gamma \vdash M \Uparrow A, \Delta} \Uparrow\!\downarrow$$

$$\dfrac{\Gamma, x : A \vdash M \Uparrow B, \Delta}{\Gamma \vdash \lambda x.M \Uparrow A \to B, \Delta} \Uparrow \mathsf{lam}^x \qquad \dfrac{\Gamma \vdash N \Uparrow -, \Delta, a : \overline{A}}{\Gamma \vdash \mu a.N \Uparrow A, \Delta} \Uparrow \mathsf{catch}^a$$

## References


[BHF01] Kensuke Baba, Sachio Hirokawa, and Ken-etsu Fujita. Parallel reduction in type free $\lambda\mu$-calculus. In *Proceedings of CATS 2001 (Computing: the Australasian Theory Symposium)*, volume 42 of *Electronic Notes in Theoretical Computer Science*. Elsevier Science B. V., 2001. Also appeared as Technical Report DOI-TR-177, Kyushu University, Fukuoka, Japan.

[Bie98] Gavin M. Bierman. A computational interpretation of the $\lambda\mu$-calculus. In L. Brim, J. Gruska, and J. Zlatuska, editors, *Proceedings of Symposium on Mathematical Foundations of Computer Science*, volume 1450 of *Lecture Notes in Computer Science*, pages 336–345, Brno, Czech Republic, August 1998.

[dG98] Philippe de Groote. An environment machine for the $\lambda\mu$-calculus. To appear in MSCS, 1998.

[Gri90] Timothy G. Griffin. A formulæ-as-types notion of control. In *Proceedings of the Seventeenth ACM/SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL)*. ACM Press, January 1990.

[HS01] Martin Hofmann and Thomas Streicher. Completeness of continuation models for $\lambda\mu$-calculus. *Information and Computation*, 2001. To appear.

[OS97] C.-H. L. Ong and C. A. Stewart. A Curry-Howard foundation for functional computation with control. In *Proceedings of ACM SIGPLAN-SIGACT Symposium on Principle of Programming Languages*, pages 215–227, Paris, Januar 1997. ACM Press.

[Par92] M. Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In A. Voronkov, editor, *Logic Programming and Automated Reasoning: Proc. of the International Conference LPAR'92*, pages 190–201. Springer, Berlin, Heidelberg, 1992.

[SR98] Thomas Streicher and Bernhard Reus. Classical logic, continuation semantics and abstract machines. *Journal of Functional Programming*, 8(6):543–572, 1998.