Parallel Reduction in Type Free Lambda-mu-Calculus

Baba, Kensuke
Graduate School of Information Science and Electrical Engineering, Kyushu University

Hirokawa, Sachio
Computing and Communications Center, Kyushu University

Fujita, Ken-etsu
Shimane University

出版情報：Electronic Notes in Theoretical Computer Science. 42, pp.52-66, 2001-01. Elsevier Science
バージョン：
権利関係：

Parallel Reduction in Type Free $\lambda\mu$-calculus*

Kensuke Baba† Sachio Hirokawa Ken-etsu Fujita

Abstract

The typed $\lambda\mu$-calculus is known to be strongly normalizing and weakly Church-Rosser, and hence becomes confluent. In fact, Parigot formulated a parallel reduction to prove confluence of the typed $\lambda\mu$-calculus by "Tait-and-Martin-Löf" method. However, the diamond property does not hold for his parallel reduction. The confluence for type-free $\lambda\mu$-calculus cannot be derived from that of the typed $\lambda\mu$-calculus and is not confirmed yet as far as we know. We analyze granularity of the reduction rules, and then introduce a new parallel reduction such that both renaming reduction and consecutive structural reductions are considered as one step parallel reduction. It is shown that the new formulation of parallel reduction has the diamond property, which yields a correct proof of the confluence for type free $\lambda\mu$-calculus. The diamond property of the new parallel reduction is also applicable to a call-by-value version of the $\lambda\mu$-calculus containing the symmetric structural reduction rule.

1 Introduction

Parigot's $\lambda\mu$-calculus [14] is an extension of the typed λ-calculus with two new term constructors $[\alpha]M$ and $\mu\alpha.M$. The term $[\alpha]M$ indexes α to the type of M and the term $\mu\alpha.M$ has the type indexed by α in M. These constructors provide classical proofs and can at the same time be considered as control operators for functional programming language. Approximately, we can think of $[\alpha]M$ and $\mu\alpha.M$ as `(throw a M)` and `(catch a M)` in terms of Common Lisp.

The $\lambda\mu$-terms M are defined as follows:

$$M ::= x \mid \lambda x.M \mid MM \mid \mu\alpha.M \mid [\alpha]M.$$

The calculus has the following basic reduction rules:
β-reduction:

$$(\lambda x.M)N \to M[N/x],$$

Structural reduction:

$$(\mu\alpha.M)N \to \mu\alpha.M[[\alpha](wN)/[\alpha]w],$$

Renaming:

$$[\beta](\mu\alpha.M) \to M[\beta/\alpha].$$

We assume some familiarity to λ-calculus [2, 11, 10]. In the structural reduction, the substitution is defined as follows.

1. $x[[\alpha](wN)/[\alpha]w] = x$
2. $(\lambda x.M)[[\alpha](wN)/[\alpha]w] = \lambda x.M[[\alpha](wN)/[\alpha]w]$
3. $(MM)[[\alpha](wN)/[\alpha]w]$
   $= M[[\alpha](wN)/[\alpha]w]M[[\alpha](wN)/[\alpha]w]$
4. $(\mu\beta.M)[[\alpha](wN)/[\alpha]w] = \mu\beta.M[[\alpha](wN)/[\alpha]w]$

5-1. $([\beta]M)[[\alpha](wN)/[\alpha]w]$
   $= [\beta](M[[\alpha](wN)/[\alpha]w]N)$ if $\alpha = \beta$

5-2. $([\beta]M)[[\alpha](wN)/[\alpha]w] = [\beta]M[[\alpha](wN)/[\alpha]w]$ if $\alpha \neq \beta$

In [14], Parigot outlined the proof for confluence of the $\lambda\mu$-calculus. He formulated the parallel reduction and claimed the diamond property for the parallel reduction:

$$\text{If} \quad M \Rightarrow N \quad \text{then} \quad N \Rightarrow M^*.$$

Here M^* is a term obtained by reducing all the redexes in M. M^* is usually referred as the "complete development [2]" of M. The formulation of the parallel reduction is based on "Tait-and-Martin-Löf" method, which is explained clearly in [17]. The method is applicable to prove confluence of many reduction systems. However, the direct application of the method does not work for the $\lambda\mu$-calculus. In fact, the diamond property does not hold for the formulation of parallel reduction in [14], see also observations in [7]. It is not so trivial to prove the confluence as it seems to be.

The $\lambda\mu$-calculus is known to be strongly normalizing [15, 16] and weak Church-Rosser. For notions of deduction, these two properties yield the confluence [2] for the typed terms [8]. However type free $\lambda\mu$-calculus is not strongly normalizing. For instance, the term $(\lambda x.xx)(\lambda x.xx)$ does not have a normal form. It is quite embarrassing that a correct proof of the confluence for type free $\lambda\mu$-calculus has never been published as far as we know.

We think that the reason why the diamond property does not hold for the parallel reduction comes

*An edited version of this report was published in: *Electronic Notes in Theoretical Computer Science, vol.42, pp.52–66, Elsevier Science, Jan, 2001.*

†Research and Development Division, Kyushu University Library, *baba@lib.kyushu-u.ac.jp*

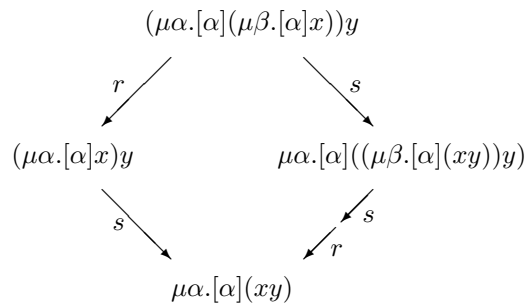

Figure 1: A counter example of the diamond property of the reduction rules in $\lambda\mu$-calculus, where arrows with s and r mean structural reduction and renaming, respectively.

from the sequential nature of the structural reduction rule. Consider a term $M = (\mu\alpha.[\alpha](\mu\beta.[\alpha]x))y$ which has a renaming redex and a structural redex. We have the terms $N_1 = (\mu\alpha.[\alpha]x)y$ and $N_2 = \mu\alpha.[\alpha]((\mu\beta.[\alpha](xy))y)$ by a renaming and a structural reductions, respectively. Then we have $M \Rightarrow N_1$ and $M \Rightarrow N_2$. If the diamond property would hold, then N_1 and N_2 were reducible to the same term M^* in one step reduction. However, this is impossible here. After the structural reduction, the "residual" of renaming redex in M is no longer a renaming redex in N_2. To make the residual back to a renaming redex, we need another step of structural reduction (Fig. 1). We consider such a successive sequence of structural reductions as a one step parallel reduction. With such a formulation, we can prove the strong diamond property for the parallel reduction.

We consider the $\lambda\mu$-calculus as a programming language and reduction as computation. The reduction rules of the $\lambda\mu$-calculus capture the mechanism of functional programming languages with control [6, 9, 4]. However we can not apply an arbitrary reduction for implementation of programming language. Usually we fix a reduction strategy. The call-by-value $\lambda\mu$-calculus $\lambda\mu_v$ was first considered by Ong and Stewart [13]. The $\lambda\mu$-calculus contains another reduction rule called "symmetric structural reduction" such that:

$$N(\mu\alpha.M) \to \mu\alpha.M[[\alpha](Nw)/[\alpha]w].$$

Note that a subsystem is not always confluent even if the whole system is confluent. Therefore, the confluence of $\lambda\mu_v$ does not yield the confluence of $\lambda\mu$, even if we ignore the symmetric structural reduction rule. We shall also formulate an appropriate parallel reduction for a call-by-value version of the $\lambda\mu$-calculus and prove the strong diamond property. In this paper, we deal with type free $\lambda\mu$-calculi, and the definition of the $\lambda\mu$-terms is distinct from that of the original ones. For example, the well formed term $([\alpha]M)N$ is not a term in the original $\lambda\mu$-calculus, since $[\alpha]M$ is not an unnamed but a named term. Does the confluence proof for type free $\lambda\mu$-calculus still work for that of the original one? The parallel reduction relation we define in the following sections is included in the transitive and reflexive closure of reduction rules, and therefore the complete development of an original term is also an original term. Hence our proof method is still and all sound for the case of the original $\lambda\mu$-calculus.

2 Parallel Reduction in $\lambda\mu$-Calculus

We define the parallel reduction in the following. The rules from 1 to 8 are obtained by a straightforward application of Tait-and-Martin-Löf method to β-reduction, structural reduction, and renaming. The last inference rule 9 is introduced in the present paper. It combines a renaming and a consecutive sequence of structural reductions. It is easy to see that the transitive and reflexive closure of "$\to$" is identical to the transitive closure of "$\Rightarrow$".

Definition 1 *1.* $x \Rightarrow x$

2. $\dfrac{M \Rightarrow M'}{\lambda x.M \Rightarrow \lambda x.M'}$

3. $\dfrac{M \Rightarrow M' \quad N \Rightarrow N'}{MN \Rightarrow M'N'}$

4. $\dfrac{M \Rightarrow M'}{\mu\alpha.M \Rightarrow \mu\alpha.M'}$

5. $\dfrac{M \Rightarrow M'}{[\alpha]M \Rightarrow [\alpha]M'}$

6. $\dfrac{M \Rightarrow M' \quad N \Rightarrow N'}{(\lambda x.M)N \Rightarrow M'[N'/x]}$

7. $\dfrac{M \Rightarrow M' \quad N \Rightarrow N'}{(\mu\alpha.M)N \Rightarrow \mu\alpha.M'[[\alpha](wN')/[\alpha]w]}$

8. $\dfrac{M \Rightarrow M'}{[\beta](\mu\alpha.M) \Rightarrow M'[\beta/\alpha]}$

9.
$$\frac{M \Rightarrow M' \quad N_1 \Rightarrow N_1' \quad \cdots \quad N_k \Rightarrow N_k'}{[\beta]((\mu\alpha.M)N_1 \cdots N_k) \Rightarrow M'[[\beta](wN_1 \cdots N_k)/[\alpha]w].}$$

We define the complete development M^* of a term M as follows:

Definition 2 *1.* $M = x$. *Then* $M^* = x$.

2. $M = \lambda x.M_1$. *Then* $M^* = \lambda x.M_1^*$.

3. $M = M_1 M_2$.

3-1. $M_1 = \lambda x.M_3$. *Then* $M^* = M_3^*[M_2^*/x]$.

3-2. $M_1 = \mu\alpha.M_3$.
Then $M^* = \mu\alpha.M_3^*[[\alpha](wM_2^*)/[\alpha]w]$.

3-3. $M^* = M_1^*M_2^*$ *o.w.*

4. $M = \mu\alpha.M_1$. *Then* $M^* = \mu\alpha.M_1^*$.

5. $M = [\alpha]M_1$.

5-1. $M_1 = \mu\beta.M_2$. *Then* $M^* = M_2^*[\alpha/\beta]$.

5-2. $M_1 = (\mu\beta.M_2)N_1 \cdots N_k$.
Then $M^* = M_2^*[[\alpha](wN_1^* \cdots N_k^*)/[\beta]w]$.

5-3. $M^* = [\alpha]M_1^*$ *o.w.*

3 Diamond Property of Parallel Reduction

A gap in the proof of confluence in [14] can be supplied by (2) of the following lemma. Without the rule 9 in Definition 1, (2) does not hold true.

Lemma 1 (1) *If* $M \Rightarrow M'$ *and* $N \Rightarrow N'$, *then* $M[N/x] \Rightarrow M'[N'/x]$.
(2) *If* $M \Rightarrow M'$ *and* $N \Rightarrow N'$, *then* $M[[\alpha](wN)/[\alpha]w] \Rightarrow M'[[\alpha](wN')/[\alpha]w]$.
(3) *If* $M \Rightarrow M'$, *then* $M[\alpha/\beta] \Rightarrow M'[\alpha/\beta]$.

Proof. (1) can be easily shown by induction on the structure of $M \Rightarrow (M'$. 3) is trivial. (2) can also be proved by induction on (the structure of $M \Rightarrow M'$. Most cases are routine. (Non-trivial cases are when the last inference of $M \Rightarrow M'$ is (either 8 or 9 in Definition 1. To save the space of the (paper, we explain only the case of 8.

By the definition of $M \Rightarrow M'$, M and M' have the form $M = [\beta](\mu\gamma.M_1)$ and $M' = M_1'[\beta/\gamma]$, and then $M \Rightarrow M'$ has the following form:

$$\frac{M_1 \Rightarrow M_1'}{[\beta](\mu\gamma.M_1) \Rightarrow M_1'[\beta/\gamma]}\ 8$$

Since γ is a bound variable, we can assume $\gamma \neq \alpha$.

If $\alpha = \beta$, then we have

$$\begin{aligned} & M[[\alpha](wN)/[\alpha]w] \\ &= ([\alpha](\mu\gamma.M_1))[[\alpha](wN)/[\alpha]w] \\ &= [\alpha]((\mu\gamma.M_1[[\alpha](wN)/[\alpha]w])N) \end{aligned}$$

and

$$\begin{aligned} & M'[[\alpha](wN')/[\alpha]w] \\ &= M_1'[\alpha/\gamma][[\alpha](wN')/[\alpha]w] \\ &= M_1'[[\alpha](wN')/[\alpha]w][[\gamma](wN')/[\gamma]w][\alpha/\gamma]. \end{aligned}$$

By induction hypothesis for $M_1 \Rightarrow M_1'$, we have $M_1[[\alpha](wN)/[\alpha]w] \Rightarrow M_1'[[\alpha](wN')/[\alpha]w]$. Thus, we have $[\alpha]((\mu\gamma.M_1[[\alpha](wN)/[\alpha]w])N) \Rightarrow M_1'[[\alpha](wN')/[\alpha]w][[\alpha](wN')/[\gamma]w][\alpha/\gamma]$ by the rule 9. Hence Lemma holds.

If $\alpha = \beta$, then we have

$$\begin{aligned} M[[\alpha](wN)/[\alpha]w] &= ([\beta](\mu\gamma.M_1))[[\alpha](wN)/[\alpha]w] \\ &= [\beta](\mu\gamma.M_1[[\alpha](wN)/[\alpha]w]) \end{aligned}$$

and

$$\begin{aligned} M'[[\alpha](wN')/[\alpha]w] &= M_1'[\beta/\gamma][[\alpha](wN')/[\alpha]w] \\ &= M_1'[[\alpha](wN')/[\alpha]w][\beta/\gamma]. \end{aligned}$$

By induction hypothesis for M_1, we have $M_1[[\alpha](wN)/[\alpha]w] \Rightarrow M_1'[[\alpha](wN')/[\alpha]w]$. Therefore we have $[\beta](\mu\gamma.M_1[[\alpha](wN)/[\alpha]w]) \Rightarrow M_1'[[\alpha](wN')/[\alpha]w][\beta/\gamma]$ by the rule 8. Thus, Lemma holds. □

Theorem 1 *For any* $\lambda\mu$*-term* M *and* M', *if* $M \Rightarrow M'$ *then* $M' \Rightarrow M^*$.

The proof is by induction on the structure of $M \Rightarrow M'$.

Put $M_3 = M^*$, then the following theorem holds by the previous theorem.

Theorem 2 *If* $M \Rightarrow M_1$ *and* $M \Rightarrow M_2$, *then there exists some* M_3 *such that* $M_1 \Rightarrow M_3$ *and* $M_2 \Rightarrow M_3$.

Since the transitive and reflexive closure of "$\rightarrow$" is identical to the transitive closure of "$\Rightarrow$", we have the confluence of the type free $\lambda\mu$-calculus.

Theorem 3 *The type free* $\lambda\mu$*-calculus is confluent.*

4 Parallel Computation in Call-by-Value $\lambda\mu$-Calculus

A call-by-value version of the $\lambda\mu$-calculus was first provided by Ong and Stewart [13]. As compared with the call-by-name system [14, 15, 16], one can adopt some reduction rule more in the call-by-value system; so-called symmetric structural reduction [13] such that $N(\mu/alpha.M) \rightarrow \mu\alpha.M[[\alpha](Nw)/[\alpha]w]$. It is known that adding such a reduction rule breaks down the confluence unless the above term N is in the form of a value. In this section, the notion of values as an extended form is introduced based on observations in [7, 8].

$$V ::= x \mid \lambda x.M \mid [\alpha]M$$

This notion is closed under both value-substitutions and substitutions induced by structural reduction or symmetric structural reduction defined below.

A context $\mathcal{E}[\ ]$ with a hole $[\ ]$ is defined as usual, such that

$$\mathcal{E} ::= [\ ] \mid \mathcal{E}M \mid V\mathcal{E}.$$

Let $n \geq 0$ and M be a term. We will write $\mathcal{E}_n[\mathcal{E}_{n-1}[\cdots\mathcal{E}_1[N]\cdots]]$ for $\mathcal{E}[N]$, where each $\mathcal{E}_i \neq [\ ]$ is either in the form of $V[\ ]$ or $[\ ]M$. For simplicity, such $\mathcal{E}_i$ also denotes the value V or the term M.

The call-by-value $\lambda\mu$-calculus consists of the following reduction rules:
β_v-reduction:

$$(\lambda x.M)V \to_v M[V/x]$$

Structural reduction:

$$(\mu\alpha.M1)M2 \to_v \mu\alpha.M1[[\alpha](wM2)/[\alpha]w]$$

Symmetric structural reduction:

$$V(\mu\alpha.M) \to_v \mu\alpha.M[[\alpha](Vw)/[\alpha]w]$$

Renaming reduction:

$$[\beta](\mu\alpha.V) \to_v V[\beta/\alpha]$$

This renaming rule is different from that in [13]. The distinction is essential under the extended form of values, and this form of renaming would also be natural from the viewpoint of CPS-translation such as in [7, 8].

We will show that the new parallel reduction can also be applicable to proving confluence for the call-by-value system of the $\lambda\mu$-calculus, contrary to the straightforward use of parallel reduction in [13]. To prove this, we define parallel reduction $\gg$ as follows:

Definition 3 *1.* $x \gg x$

2. $\dfrac{M \gg M'}{\lambda x.M \gg \lambda x.M'}$

3. $\dfrac{M \gg M' \quad N \gg N'}{MN \gg M'N'}$

4. $\dfrac{M \gg M'}{\mu\alpha.M \gg \mu\alpha.M'}$

5. $\dfrac{M \gg M'}{[\alpha]M \gg [\alpha]M'}$

6. $\dfrac{M \gg M' \quad V \gg N'}{(\lambda x.M)V \gg M'[N'/x]}$

7. $\dfrac{M \gg M' \quad N \gg N'}{(\mu\alpha.M)N \gg \mu\alpha.M'[[\alpha](wN')/[\alpha]w]}$

8. $\dfrac{M \gg M' \quad V \gg N'}{V(\mu\alpha.M) \gg \mu\alpha.M'[[\alpha](N'w)/[\alpha]w]}$

9. $\dfrac{V \gg M' \quad \mathcal{E}_1 \gg \mathcal{E}'_1 \quad \cdots \quad \mathcal{E}_n \gg \mathcal{E}'_n}{[\alpha](\mathcal{E}[\mu\beta.V]) \gg M'[[\alpha](\mathcal{E}'[w])/[\beta]w]}$, *where* $\mathcal{E}[\ ] = \mathcal{E}_n[\cdots\mathcal{E}_1[\ ]\cdots]$ *and* $\mathcal{E}'[\ ] = \mathcal{E}'_n[\cdots\mathcal{E}'_1[\ ]\cdots]$.

Now it can be seen that the transitive and reflexive closure of $\to_v$ is equivalent to the transitive closure of $\gg$.

Lemma 2 (1) *If* $V \gg M$, *then* M *is also in the form of a value.*
(2) *If* $M \gg N$ *and* $V \gg N'$, *then* $M[V/x] \gg N[N'/x]$.
(3) *If* $M \gg M'$ *and* $N \gg N'$, *then* $M[[\alpha](wN)/[\alpha]w] \gg M'[[\alpha](wN')/[\alpha]w]$.
(4) *If* $M \gg M'$ *and* $V \gg N'$, *then* $M[[\alpha](Vw)/[\alpha]w] \gg M'[[\alpha](N'w)/[\alpha]w]$.
(5) *If* $M \gg M'$, *then* $M[\alpha/\beta] \gg M'[\alpha/\beta]$.
(6) *Let* $n \geq 0$. *Let* $\mathcal{E}[\ ] = \mathcal{E}_n[\cdots\mathcal{E}_1[\ ]\cdots]$ *and* $\mathcal{E}'[\ ] = \mathcal{E}'_n[\cdots\mathcal{E}'_1[\ ]\cdots]$. *If* $M \gg M'$ *and* $\mathcal{E}_i \gg \mathcal{E}'_i$ *for* $1 \leq i \leq n$, *then* $M[[\alpha](\mathcal{E}[w])/[\alpha]w] \gg M'[[\alpha](\mathcal{E}'[w])/[\alpha]w]$.

Proposition 1 *For any* $\lambda\mu$*-term* M, *there exists* M^* *such that for any* N, $N \gg M^*$ *whenever* $M \gg N$.

Proof. By induction on the derivation of $\gg$. Here, the complete development M^* can be given inductively as follows:

Definition 4 *1.* $M = x$. *Then* $M^* = x$.

2. $M = \lambda x.M_1$. *Then* $M^* = \lambda x.M_1^*$.

3. $M = M_1M_2$.

3-1. $M_1 = \lambda x.M_3$ *and* $M_2 = V_2$. *Then* $M^* = M_3^*[V_2^*/x]$.

3-2. $M_1 = \mu\alpha.M_3$.
Then $M^* = \mu\alpha.M_3^*[[\alpha](wM_2^*)/[\alpha]w]$.

3-3. $M_1 = V_1$ *and* $M_2 = \mu\alpha.M_4$. *Then* $M^* = \mu\alpha.M_4^*[[\alpha](V_1^*w)/[\alpha]w]$.

3-4. $M^* = M_1^*M_2^*$ *o.w.*

4. $M = \mu\alpha.M_1$. *Then* $M^* = \mu\alpha.M_1^*$.

5. $M = [\alpha]M_1$.

5-1. $M_1 = \mathcal{E}[\mu\beta.V_2]$.
Then $M^* = V_2^*[[\alpha](\mathcal{E}^*[w])/[\beta]w]$, *where* $\mathcal{E}^*[\ ]$ *is defined as* $\mathcal{E}_n^*[\cdots\mathcal{E}_1^*[\ ]\cdots]$ *for* $\mathcal{E}[\ ] = \mathcal{E}_n[\cdots\mathcal{E}_1[\ ]\cdots]$ *and* $n \geq 0$.

5-2. $M^* = [\alpha]M_1^*$ *o.w.*

We show only the case M of $[\alpha]M_1$. The remaining cases can also be justified following a similar pattern.
1. Case $[\alpha]M_1$ of $[\alpha](\mathcal{E}[\mu\beta.V])$:
1-1. $M \gg N = [\alpha]N_1$ is derived from $\mathcal{E}[\mu\beta.V] \gg N_1$ by 5:
1-1-1. $\mathcal{E}[\mu\beta.V] = \mu\beta.V$:

In this case, $\mu\beta.V \gg N_1 = \mu\beta.N_2$ is derived from $V \gg N_2$ by 4, where N_2 is also a value. From the induction hypothesis, we have $N_2 \gg V^*$, and hence $N = [\alpha](\mu\beta.N_2) \gg V^*[\alpha/\beta] = M^*$ is obtained by 9.

1-1-2. $\mathcal{E}[\mu\beta.V] = \mathcal{E}_n[\cdots\mathcal{E}_1[\mu\beta.V]\cdots]$ for $n \geq 1$:

Since $\mathcal{E}_1[\mu\beta.V]$ is not a value, $\mathcal{E}_n[\cdots\mathcal{E}_1[\mu\beta.V]\cdots] \gg N_1$ must be derived from $\mathcal{E}_1[\mu\beta.V] \gg N_2'$ and $\mathcal{E}_j \gg \mathcal{E}_j'$ for $2 \leq j \leq n$ by the successive use of 3, where $N_1 = \mathcal{E}_n'[\cdots\mathcal{E}_2'[N_2']\cdots]$. Here, we have two cases for $\mathcal{E}_1$ and two derivations for each of those.

1-1-2-1. $\mathcal{E}_1[\mu\beta.V] = V_1(\mu\beta.V)$:

1-1-2-1-1. $V_1(\mu\beta.V) \gg N_2' = N_3'N_4'$ is derived from $\mu\beta.V \gg N_4'$ and $V_1 \gg N_3'$ by 3:

Since $\mu\beta.V \gg N_4' = \mu\beta.N_5'$ must be derived from $V \gg N_5'$ by 4, we have $N_5' \gg V^*$ by the induction hypothesis, where N_5' is also a value. Let $\mathcal{E}_1'$ be $N_3'[\ ]$, where N_3' is a value. Then the induction hypothesis gives $N_3' \gg V_1^*$ abbreviated as $\mathcal{E}_1' \gg \mathcal{E}_1^*$. From the induction hypotheses for $\mathcal{E}_j$ for $2 \leq j \leq n$ we also have $\mathcal{E}_j' \gg \mathcal{E}_j^*$, and then $[\alpha](\mathcal{E}_n'[\cdots\mathcal{E}_1'[\mu\beta.N_5']\cdots]) \gg V^*[[\alpha](\mathcal{E}_n^*[\cdots\mathcal{E}_1^*[w]\cdots])/[\beta]w]$ is obtained by 9.

1-1-2-1-2. $V_1(\mu\beta.V) \gg N_2' = N_3'[[\alpha](N_4'w)/[\alpha]w]$ is derived from $V_1 \gg N_3'$ and $V \gg N_4'$ by 8:

The induction hypotheses give $N_3' \gg V_1^*$ and $N_4' \gg V^*$. From the substitution lemma, we have $N_4'[[\beta](N_3'w)/[\beta]w] \gg V^*[[\beta](V_1^*w)/[\beta]w]$, where N_4' is also a value and values are closed under the substitutions. The induction hypotheses for $\mathcal{E}_j$ for $2 \leq j \leq n$ also give $\mathcal{E}_j' \gg \mathcal{E}_j^*$. Hence, the use of 9 derives $[\alpha](\mathcal{E}_n'[\cdots\mathcal{E}_2'[\mu\beta.N_4'[[\beta](N_3'w)/[\beta]w]]\cdots]) \gg (V^*[[\beta](V_1^*w)/[\beta]w])[[\alpha](\mathcal{E}_n^*[\cdots\mathcal{E}_2^*[w]\cdots])/[\beta]w]$, whose right-hand side is equivalent to $V^*[[\alpha](\mathcal{E}_n^*[\cdots\mathcal{E}_1^*[w]\cdots])/[\beta]w]$, where $\mathcal{E}_1^*$ is $V_1^*[\ ]$.

1-1-2-2. $\mathcal{E}_1[\mu\beta.V] = (\mu\beta.V)M_2$:

In this case, we have two derivations for $(\mu\beta.V)M_2 \gg N_2'$ by the use of 3 or 7. Each case can be verified following a similar pattern to the above two cases.

1-2. $M \gg N = N'[[\beta](\mathcal{E}'[w])/[\alpha]w]$ is derived from $V \gg N'$ and $\mathcal{E}_i \gg \mathcal{E}_i'$ for $1 \leq i \leq n$, where $\mathcal{E}[\ ] = \mathcal{E}_n[\cdots\mathcal{E}_1[\ ]\cdots]$ and $\mathcal{E}'[\ ] = \mathcal{E}_n'[\cdots\mathcal{E}_1'[\ ]\cdots]$:

The successive application of the substitution lemma to the induction hypotheses.

2. Otherwise:

The straightforward use of the induction hypothesis. □

Finally, the confluence for the call-by-value $\lambda\mu$-calculus can be confirmed, since $\gg$ has the diamond property.

Theorem 4 *The type free $\lambda\mu$-calculus of call-by-value has the confluence property.*

5 Related Work and Further Problems

5.1 Sequentiality of Reduction

Parallel reduction is a very clear and intuitive idea which means to reduce a number of redexes (existing in the term) simultaneously. It is often applied to prove confluence of reduction system. However, a naive formulation of parallel reduction does not always work. The $\lambda\mu$-calculus is one of such reduction systems. We have reasoned that the difficulty comes from the sequentiality of the structural reductions. Therefore we proposed that a consecutive sequence of the structural reductions should be considered as one step of parallel reduction. As pointed out in Takahashi [17], the idea does not work for $\lambda\eta^{-1}$ either, that is, λ-calculus with η-expansion: $M \to \lambda x.Mx$. The confluence of $\lambda\eta^{-1}$ is proved in [12, 1]. Jay and Ghani [12] proved the confluence by introducing "parallel expansion" which includes, roughly speaking, a consecutive application of η^{-1} $M \to \lambda x_1.Mx_1 \to \lambda x_1x_2.Mx_1x_2 \to \cdots$. van Raamsdonk [18] introduced the notion of "superdevelopment" to prove confluence of the orthogonal combinatory reduction systems. The superdevelopment is a reduction sequence in which some redexes that are created during reduction may be contracted, besides redexes that descend from the initial term. A key idea of these works is to overcome some sequentiality of reductions. We cannot show, at the current stage, what kind of reduction contains such sequentiality in general. To find some criteria of such sequentiality is one of further work.

5.2 Another Reduction Rule

Parigot's $\lambda\mu$-calculus [14] has one more reduction rule η^*:

$$\mu\alpha.[\alpha]M \to M$$

if α has no free occurrence in M. Consider a term $M = \mu\alpha.[\alpha]((\mu\beta.[\gamma]x)yz)$. Then M has η^*-redex and the redex with respect to the rule 9 of Definition 1. The reduction of each redex is represented by "η^*" and "rs" in Figure 2:

By "rs", we reach $\mu\alpha.[\gamma]x$ with one step parallel reduction. If we apply η^* first, we have $(\mu\beta.[\gamma]x)yz$ from which we cannot reach $\mu\alpha.[\gamma]x$ with one step parallel reduction. However, we can overcome this situation by counting a series of structural reductions as one step. On the basis of this idea, we have a formulation of parallel reduction with η^* as well, although the formal description is skipped here for simplicity.

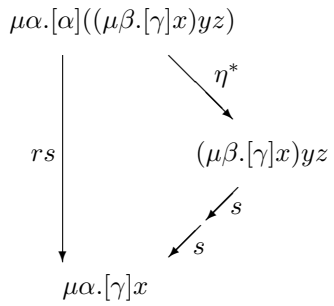

Figure 2: An example of the reduction rule η^*, where arrows with s and rs mean structural reduction and the reduction for the rule 9, respectively.

5.3 Practical Application of Type Free $\lambda\mu$-Calculi

In order to make the $\lambda\mu$-calculi in practical application, we need to realize some machines. In fact, Bierman [3] and de Groote [5] proposed such abstract machines and analyzed their behavior. However, all those machines are sequential in nature. We expect that our parallel reduction could yield natural extensions of those machines.

References

[1] Y. Akama. On mints's reduction for CCC-calculus. In *Lecture Notes in Computer Science*, volume 664, pages 1–15, 1993.

[2] H. P. Barendregt. *The Lambda Calculus, 2nd ed.* North-Holland, 1984.

[3] G. M. Bierman. A computational interpretation of the $\lambda\mu$-calculus. In *University of Cambridge Computer Laboratory Technical Report*, volume 448, 1998.

[4] P. de Groote. On the relation between the $\lambda\mu$-calculus and the syntactic theory of sequential control. In *Lecture Notes in Artificial Intelligence*, volume 822, pages 31–43, 1994.

[5] P. de Groote. An environment machine for the $\lambda\mu$-calculus. *Mathematical Structure in Computer Science*, 8:637–669, 1998.

[6] M. Felleisen, D. P. Friedman, E. Kohlbecker, and B. Duba. A syntactic theory of sequential control. *Theoretical Computer Science*, 52:205–237, 1987.

[7] K. Fujita. Calculus of classical proofs I. In *Lecture Notes in Computer Science*, volume 1345, pages 321–335, 1997.

[8] K. Fujita. Explicitly typed $\lambda\mu$-calculus for polymorphism and call-by-value. In *Lecture Notes in Computer Science*, volume 1581, pages 162–176, 1999.

[9] T. Griffin. A formulae-as-types notion of control. In *Conference Record of 17th ACM Symposium on Principles of Programmin Language*, pages 47–58, 1990.

[10] J. R. Hindley. *Basic Simple Type Theory.* Cambridge University Press, 1997.

[11] J. R. Hindley and J. P. Seldin. *Introduction to Combinators and λ-Calculus.* Cambridge University Press, 1986.

[12] C. B. Jay and N. Ghani. The virtues of eta-expansion. In *LFCS report*, volume 243, 1992.

[13] C.-H. L. Ong and C. A. Stewart. A curry-howard foundation for functional computation with control. In *Proc. 24th Annual ACM Symposium of Principles of Programming Language*, 1997.

[14] M. Parigot. $\lambda\mu$-calculus: An algorithmic interpretation of classical natural deduction. In *Lecture Notes in Artificial Intelligence*, volume 624, pages 190–201, 1992.

[15] M. Parigot. Strong normalization for second order classical natural deduction. In *Proc. 8th Annual IEEE Symposium on Logic in Computer Science*, pages 39–46, 1993.

[16] M. Parigot. Proofs of strong normalization for second order classical natural deduction. *Journal of Symbolic Logic*, 62(4):1461–1479, 1997.

[17] M. Takahashi. Parallel reduction in λ-calculus. *Information and Computation*, 118:120–127, 1995.

[18] F. van Raamsdonk. Confluence and superdevelopment. In *Lecture Notes in Computer Science*, volume 690, pages 168–182, 1993.