Available online at www.sciencedirect.com
SCIENCE DIRECT®

Information Processing Letters 99 (2006) 163–170

Information Processing Letters
www.elsevier.com/locate/ipl

Strong normalization proofs by CPS-translations

Satoshi Ikeda [*], Koji Nakazawa

Graduate School of Informatics, Kyoto University, Yoshida-Honmachi, Sakyo-ku, Kyoto 606-8501, Japan

Accepted 20 March 2006
Available online 27 April 2006
Communicated by G. Morrisett

Abstract

This paper proposes a new proof method for strong normalization of classical natural deduction and calculi with control operators. For this purpose, we introduce a new CPS-translation, *continuation and garbage passing style* (*CGPS*) translation. We show that this CGPS-translation method gives simple proofs of strong normalization of $\lambda\mu^{\to\wedge\vee\perp}$, which is introduced in [P. de Groote, Strong normalization of classical natural deduction with disjunction, in: S. Abramsky (Ed.), Typed Lambda Calculi and Applications, 5th International Conference, TLCA 2001, in: Lecture Notes in Comput. Sci., vol. 2044, Springer, Berlin, 2001, pp. 182–196] by de Groote and corresponds to the classical natural deduction with disjunctions and permutative conversions.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Programming calculi; Continuation passing style translation; Strong normalization; Classical natural deduction; Permutative conversion

1. Introduction

Since Griffin pointed out the correspondence between classical logic and programming languages with control operators in [7], many typed calculi corresponding to classical logic have been introduced. Parigot's $\lambda\mu$-calculus in [12] is one of such calculi, which is an extension of λ-calculus and corresponds to classical natural deduction with the rule for reduction to absurdity. As Ong and Stewart pointed out in [11], the $\lambda\mu$-calculus is also important as an abstract programming language. They introduced a call-by-value variant of the $\lambda\mu$-calculus and showed that several control operators can be simulated in the call-by-value $\lambda\mu$-calculus. De Groote introduced in [3] another calculus $\lambda\mu^{\to\wedge\vee\perp}$, which is an extension of $\lambda\mu$-calculus. It corresponds to the classical natural deduction with conjunctions, disjunctions and permutative conversions.

Strong normalization is one of the most important properties of typed calculi, because it guarantees termination of execution of well-typed terms for any evaluation strategy. In [13], Parigot proved the strong normalization of $\lambda\mu$-calculus in two ways. One is the reducibility method, and the other uses the *continuation passing style* (*CPS*) translation, which translates $\lambda\mu$-terms to λ-terms and preserves reduction relation and typability. By the CPS-translation any infinite reduction sequence from any well-typed $\lambda\mu$-term is translated to an infinite reduction sequence from a well-typed λ-term. Hence, we can reduce strong normalization of $\lambda\mu$-calculus to that of λ-calculus. This idea is also used for other calculi, $\lambda\mu^{\to\wedge\vee\perp}$ [3], call-by-value $\lambda\mu$-calculus [5], and a typed calculus $\lambda^{\to}_{exn}$ for exception handling [2].

[*] Corresponding author.
E-mail addresses: s-ikeda@kuis.kyoto-u.ac.jp (S. Ikeda), knak@kuis.kyoto-u.ac.jp (K. Nakazawa).

0020-0190/$ – see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.ipl.2006.03.009

$$\frac{x:\tau \in \Gamma}{\Gamma \vdash x:\tau;\, \Delta}\ (\mathrm{Ax}) \qquad \frac{\Gamma, x:\sigma \vdash M:\tau;\, \Delta}{\Gamma \vdash \lambda x.M:\sigma \to \tau;\, \Delta}\ (\mathrm{Abs}) \qquad \frac{\Gamma \vdash M:\tau \to \sigma;\, \Delta \quad \Gamma \vdash N:\tau;\, \Delta}{\Gamma \vdash MN:\sigma;\, \Delta}\ (\mathrm{App})$$

$$\frac{\Gamma \vdash M:\bot;\, \Delta, \alpha:\sigma}{\Gamma \vdash \mu\alpha.M:\sigma;\, \Delta}\ (\mathrm{MuAbs}) \qquad \frac{\Gamma \vdash M:\sigma;\, \Delta, \alpha:\sigma}{\Gamma \vdash \alpha M:\bot;\, \Delta, \alpha:\sigma}\ (\mathrm{Nam})$$

Fig. 1. Typing rules of λμ-calculus.

However, these proofs by CPS-translations do not work, as Nakazawa and Tatsuta [10] and Matthes [8] pointed out. In many cases, CPS-translations which preserve not only convertibility but also reduction relations are defined by reducing redexes of control operators. Therefore, if a subterm in the source program is eliminated by a reduction of control operators, it is not kept in the result of the CPS-translation. As for λμ-calculus and its reduction preserving CPS-translation in [13], for example, any *structural reduction* (*μ-reduction*) step is executed in the CPS-translation, that is, if M is reduced to N by structural reduction, then $\overline{M}$ and $\overline{N}$ are identical, where $\overline{M}$ denotes the CPS-translation of M. Therefore, the CPS-translation of $(\mu\alpha.x)N$ is identical with the CPS-translation of $\mu\alpha.x$ for any N; hence, any redex in N is erased by the CPS-translation. This is called *erasing-continuations* in [10]. Because of erasing-continuations, reduction relations of one or more steps are not necessarily preserved, which makes it difficult to prove strong normalization. All proofs by CPS-translations in [13,3,5,2] fail because of the same erasing-continuation problem.

In this paper, we introduce a new CPS-translation, *continuation and garbage passing style translation* (*CGPS-translation*). In CGPS-translation, not only continuations but also *garbage terms* are passed. Garbage terms contain all continuations passed in the CPS-translation. Because any part of the source program is kept in garbage terms, any redex in the source program remains in its CGPS-translation. Therefore, CGPS-translations enable us to avoid erasing-continuation problem and to prove strong normalization properties.

Furthermore, we show that CGPS-translations yield correct strong normalization proofs of λμ-calculus and λ$\mu^{\to\wedge\vee\bot}$.

2. Strong normalization proof by CPS-translation

In this section, we explain the idea and difficulty of strong normalization proofs by CPS-translations, taking the case of the λμ-calculus as an example.

The λμ-calculus is introduced in [12] as a term assignment system for second-order classical natural deduction with the rule for *reduction to absurdity*:

$$\frac{\Gamma;\, \Delta, \neg\sigma \vdash \bot}{\Gamma;\, \Delta \vdash \sigma},$$

where σ is a formula, Γ is a set of assumptions and Δ is a set of negated assumptions. In this paper, we adopt two-sided notation for judgments, where the above rule is denoted by

$$\frac{\Gamma \vdash \bot;\, \Delta, \sigma}{\Gamma \vdash \sigma;\, \Delta}.$$

Definition 2.1 (*λμ-calculus*). *Types* of the *λμ-calculus* are second-order propositional formulas

$$\sigma, \tau ::= X \mid \bot \mid \sigma \to \tau,$$

where X denotes type variables. The negation $\neg\sigma$ is defined as $\neg\sigma \equiv \sigma \to \bot$. The λ$\mu$-calculus has two distinct term-variables λ-*variables* $(x, y, \ldots)$ and μ-*variables* $(\alpha, \beta, \ldots)$. *Terms* are defined as

$$M, N ::= x \mid \lambda x.M \mid MN \mid \mu\alpha.M \mid \alpha M.$$

In λ-abstraction $\lambda x.M$ and μ-abstraction $\mu\alpha.M$, the variables x and α are bound, respectively. The name of bound variables is changed as usual when required.

Typing rules are given in Fig. 1, where Γ (Δ) denotes a set of types labeled by λ-variables (μ-variables) and is called λ-context (μ-context, respectively). It is not permitted that a context contains two different types labeled by a same variable. *Singular contexts* are defined as $\mathcal{C} ::= []M$. For $\mathcal{C} \equiv []M$, $\mathcal{C}[N]$ denotes the term NM.

The ordinary substitution to λ-variables $M[x := N]$ is defined as usual. We have the *structural substitution* $M[\alpha \Leftarrow \mathcal{C}]$, which is obtained by replacing each subterm of M of the form αN by $\alpha\mathcal{C}[N[\alpha \Leftarrow \mathcal{C}]]$. *Reduction rules* are the following.

(β) $(\lambda x.M)N \to_\beta M[x := N]$,

(μ) $\mathcal{C}[\mu\alpha M] \to_\mu \mu\alpha.M[\alpha \Leftarrow \mathcal{C}]$.

The one-step reduction relation $\to$ is the congruence relation defined by both reduction rules. Similarly, $\to_\beta$ is defined by only (β), and $\to_\mu$ by only (μ). $\to_\mu$ is called *structural reduction*. $\to^*$ is the reflexive transitive closure of $\to$, $\to^+$ is the transitive closure of $\to$, which we call *strict reduction relation*. The relations $\to_\beta^*$, $\to_\beta^+$, $\to_\mu^*$, and $\to_\mu^+$ are defined similarly.