Evaluation of Mr. Pyzik's draft paper for Ph.D. dissertation

Yukiyoshi Kameyama

This document is my evaluation of Mr. Mateusz Pyzik's paper entitled "Reflection of Continuation-Passing Style in Calculi of Delimited Control".

1 General Comments

Studies on control operators have a long history, dating back to 1970s or even earlier. Among all, delimited-control operators have long been the major targets for programming-language theoreticians since their 'discovery' in late 1980s. The paper by Mr. Pyzik builds on previous works, and has made an important advance in this area, which previous studies have never achieved, or even tried. The subject of the paper is to establish a very precise correspondence between a source calculus with delimited-control operators and a target calculus without them via a CPS translation. The results are more precise than previous studies in that the paper establishes the correspondence between reduction theories of the source and target calculi (called Galois connection), rather than equational theories. More precisely, the paper gives three instances of Galois connections, for the delimited-control operators shift, shift0, and unary shift0.

These results are indeed remarkable. While the standard calculi for these control operators are shown to satisfy equational correspondence between Direct-Style terms (DS terms) and CPS terms, they do not form Galois connections as they are, since the standard CPS translations may not preserve the direction of reductions. To obtain a Galois connection, one needs to refine not only the CPS translation, but also the source and target calculi, while preserving the semantics of control operators. In fact, an existing work for a much simpler setting (the calculi without control operators) by Sabry and Wadler [SW97] involved a number of small adjustments as well as a lot of calculations to prove the results. Clearly, establishing a Galois connection for the calculi with control operators is far more difficult, that needs a deep understanding of the semantic nature of the target control operators and their CPS translation.

The paper has achieved the goal for the control operators shift and shift0. The former has been the most important control operators until 2010 or so, until the latter replaces its position by a series of works by Materzok and Biernacki around 2010, and is still an important one since it is closely related to algebraic effects with deep handlers, which is the most popular target of recent studies. Hence, the reported results are (and will be) highly significant in the theory of programming languages.

It is true that the piratical impact of the results is not completely clear. For instance, the presented calculi are minimal in the sense that they do not have constants, conditionals, recursions and so on. I do not think the mentality of the calculi is a problem; Sabry and Felleisen obtained similar, but less ambitious results for a minimal lambda calculus (and that with the control operator call/cc), but their results later became a basis to A-translation, which is useful in theory and in compiler practice.

Also, the studied calculi are not typed while most modern programming languages with sophisticated control operators have static type systems. Although I suspect some part of the presented results need to be adjusted or simplified (e.g., η-expansion will be possible in typed setting), I believe that the work in this paper can provide a good foundation even for typed control operators.

In summary, I think the work presented in this paper is timely, mathematically precise, and significant that will be cited by many other works in future.

I thereby conclude that the paper should be accepted. Since the presented results are remarkable, I recommend the university to nominate it for an award (if there is any).

2 Detailed Comments on Chapter 1

Chapter 1 gives an introduction, which provides a good summary of the works presented in the later chapters. I have only one serious comment. At the end of section 1.3, the author argues the practical usefulness of Galois connections, but I think the arguments should be strengthened; the current text says that Galois connection can be useful to do an optimization in some intermediate calculus B in an earlier intermediate calculus A. However, it may be done as well, if A and B have equational correspondence (not a Galois connection).

Smaller comments on Chapter 1 are as follows.

The paper's English should be polished, as occasionally I found problems, for instance, two sentences are sequenced with no connectives (such as 'and'). Also, for 'direct style transformation', I recommend to add one hyphen, i.e. 'direct-style transformation'.

Since Continuation-Composing style is one of the key ingredients in this paper (opposed to Continuation-Passing style), Section 1.2.3 should explain this notion in more depth, possibly using examples.

In Section 1.5, line -5 (from bottom), the set $\{M \mid \exists N.N^{*}M\}$ has some typo, and needs to be fixed.

In Section 1.6, the meaning of the phrase 'reflection into X' should be explained at the beginning.

3 Detailed Comments on Chapter 2

Chapter 2 is interesting and provides a very solid, and original result. I highly appreciate the achievements.

I have a few comments about Figure 3 in page 18:6.

First, it should be explicitly written that the image of shift (the constant S) cannot belong to the syntactic category for values (V), since the image contains continuation-composing style. Without such an explanation, one would be puzzled to see that the reduction ($\beta.S$) is independently given than ($\beta.v$).

Second, for the reduction rule (η.let), the variable condition (x is not free in K) should be written. (I found several other places where this condition is omitted.)

Third, it was not easy to understand your notation for '$_{(\Delta=k)}k$' in the category '$continuations_{\Delta}$', for the first reading. Also, I don't understand why the metavariable for continuations are J and K, and not J_{Δ} and K_{Δ}.

I also think the context notation such as 'K[V]' in Figure 5 needs more explanation, since it is not a part of syntax. (I know that the original article by Sabry and Wadler used this notation, and the author just followed it, but I believe that a dissertation should be self-contained.)

I have a question about Lemma 11 in page 18:8. Since the source calculus has a shift operator, continuations may be duplicated, so I am not sure if the conclusion of Lemma 11 can be up to one-step reductions.

Smaller comments.

In page 18:9, there are a few typos. In line 3, J should have a 'hat' (as in Lemma 10). A few lines before Corollary 13, the one-step reduction should be replaced by zero-or-one-step reduction (as in Lemma 11).

To confess, I did not understand Figure 12, which needs more explanation.

4 Detailed Comments on Chapter 3

Chapter 3 is also very interesting, and I enjoyed the presented results very much, as shift0 is nowadays the most important delimited-control operator.

I have two major comments on this chapter.

The chapter emphasizes the importance of having an η-hear-normal roots. I could not fully understand it, though I could understand the overall structure of the arguments by Section 5.1. I recommend the author to expand this section, possibly using more concrete examples (how to distinguish values from contexts etc.).

It is interesting that the proposed calculus $\lambda_{c\$}$ does not have a reduction that corresponds to $\eta_{\$}$ in $\lambda_{\$}$, while they are equivalent equationally. I think the author should spell out why a directed version of $\eta_{\$}$ is not justified by the reductions in $\lambda_{c\$}$, and explains why it is not problematic.

Smaller comments:

Line -20 (from bottom), the 6th page of Chapter 3: a comma is missing between the terms $lambdax.Vx$ and $lambdax.V'x$.

Line -18 (from bottom), the 6th page of Chapter 3: "by rule (START-2) $\lambda Vx \sim V''$". should be "by rule (START-2) $\lambda x.Vx \sim V''$".

Line 7, the 7th page of Chapter 3 (in the proof of Lemma 4.4): the similarity relation between M and $\iota(e)[...]$ is reversed.

The proof of Lemma 4.8: for the case of $V\$K[J[M]]$, I do not understand why the induction hypothesis can be used from the second line to the third line.

5 Detailed Comments on Chapter 4

Chapter 4 has a similar purpose as Chapter 3, but it gives a better result in that a slightly different calculus for shift0 and the full set of lambda terms form a reflection. As a result, the reflection in this Chapter gives an isomorphism between some call-by-value calculus with control operators and call-by-name lambda calculus via a CPS translation. This is an 'unusual' result, since the target of a CPS translation (or its closure under reductions) is usually a very restricted subset of lambda terms, for instance, no nested function calls are allowed. Since shift and shift0 allow us to compose continuations, this restriction is relaxed (thus, Continuation-Composing Style), yet, the image of the CPS- (or CCS-) translation is usually considered as a restricted subset, but here the author showed that it is not the case. I agree that this is a quite interesting, new result.

I have one reservation to fully appreciate this result. Namely, the source calculus in this chapter, $\Lambda_\$$, is a calculus with shift0, but it contains a very strange reduction rule called 'bind'; it reduces $J[P]$ to a let-term, but a let-term in this calculus is macro-defined by a combination of shift0 and two \$-operators. Hence, starting with a pure term in the form $J[P]$ (by 'pure', I mean a term without control operators), we immediately obtain a complicated term with three occurrences of control operators. Clearly, the reduction rule is not natural, and needs a detailed explanation (other than just saying that 'it was found by a trial-and-error manner). Unfortunately, the current description does not explain the rule itself, and I do not fully appreciate the beautiful result in this chapter until I can understand the source calculus $\Lambda_\$$.

In particular, I would like to know (1) how this reduction was obtained, (2) why the usual let-term (with natural reductions) does not work for this case, and (3) what happens if the source calculus has its own let-terms (with natural reductions) in addition to this macro-defined let-terms.

However, the above concern of mine is mostly about presentations, and it should not be interpreted as a serious problem about the result in this chapter. In fact, even though the source calculus has a somewhat mysterious reduction, it is proved to be equivalent (up to equality) to the standard calculus for shift0, therefore, there should be no big problem. The obtained results (reflection etc.) are indeed splendid.

Small comments:

Last line in page 14-2: it is explained that the spacing matters in $M\$(N)$ etc., but I do not agree with this choice. To avoid misunderstanding, I believe that spacing should not have a meaning.

Middle in page 14-5: It seems to be argued about the choice of let-terms, but I do not fully understand why the current form for let-terms was introduced.

Line -16, page 14-5: 'Figure 2' should be 'Figure 4', I think.

Line -10, page 14-11: 'Every lambda terms is in Continuation-Passing Style'. I do not think so, nor the paper does not establish this phrase. I believe a better phrase is '... is in Continuation-Composing Style'.

Example 4.16 in page 14-16: there are three places where the author writes 'xy' which should be 'xz'.

6 Concluding Remarks

The presented results in the paper are well-motivated, timely, mathematically precise (modulo some minor problems), significant, and has a certain practical importance to the theory and practice of programming-language studies. I do believe that it is a mile stone in the study of delimited-control operators and will be cited by many other researchers in future.

To conclude, I do recommend to accept this paper as a Ph. D. dissertation.

7 Questions to the author

I would like to ask the following questions (in the final defense).

- Galois connections are mathematically beautiful concepts, e.g., the former is an ajunction in Category Theory. Only little about the implications of Galois connections has been discussed in the paper; can you say more about it?
  In particular, I am curious about the added value from the equational correspondence, which may be sufficient for compiler optimizations (explained in the introduction.) For instance, if we want to induce a type system for the source calculus from the one for the target calculus via a DS-translation, is it possible to obtain the subject reduction property easily?
- The work presented in this paper considers so called 1-CPS translation, so that its image may not be a CPS term in the strict sense. An alternative CPS translation for shift is a 2-CPS translation, whose target terms are (strict) CPS terms. Can you relate, or compare, your work with those for 2-CPS translation?
- What happens if you introduce a type system to the calculi? Do you think you can give types to the source calculus from the target calculus via a DS-translation? One possible merit of introducing typing would be that you can η-expand terms in the typed setting; does it simplify the calculi?
- You mentioned that abortive control operators such as call/cc are (one of) the next target to study. Do you think any difficulty in this direction?

Yukiyoshi Kameyama
Professor
Department of Computer Scienence, University of Tsukuba
1-1-1, Tennodai, Tsukuba, 305-8573, JAPAN