Sequentiality and the $\pi$-Calculus

Martin Berger$^1$, Kohei Honda$^1$, and Nobuko Yoshida$^2$

$^1$ Queen Mary, University of London, U.K
$^2$ University of Leicester, U.K.

Abstract. We present a type discipline for the $\pi$-calculus which precisely captures the notion of sequential functional computation as a specific class of name passing interactive behaviour. The typed calculus allows direct interpretation of both call-by-name and call-by-value sequential functions. The precision of the representation is demonstrated by way of a fully abstract encoding of PCF. The result shows how a typed $\pi$-calculus can be used as a descriptive tool for a significant class of programming languages without losing the latter’s semantic properties. Close correspondence with games semantics and process-theoretic reasoning techniques are together used to establish full abstraction.

1 Introduction

This paper studies a type discipline for the $\pi$-calculus which precisely captures the notion of sequential functional computation. The precision of the representation is demonstrated by way of a fully abstract encoding of PCF. Preceding studies have shown that while operational encodings of diverse programming language constructs into the $\pi$-calculus are possible, they are rarely fully abstract [28,32]: we necessarily lose information by such a translation. The translation of a source term $M$ will generally result in a process containing more behaviour than $M$. Type disciplines for the $\pi$-calculus with significant properties such as linearity and deadlock-freedom have been studied before [9,16,21,22,29,30,37], but, to our knowledge, no previous typing system for the $\pi$-calculus has enabled a fully abstract translation of functional sequentiality. The present work shows that a relatively simple typing system suffices for this purpose. Despite its simplicity, the calculus is general enough to give clean interpretations of both call-by-name and call-by-value sequentiality, offering a basic articulation of functional sequentiality without relying on particular evaluation strategies. The core idea of the typing system is that affineness and stateless replication ensure deterministic computation. Sequentiality is guaranteed by controlling the number of threads through restricting the shape of processes. While the idea itself is simple, the result would offer a technical underpinning for the potential use of typed $\pi$-calculi as meta-languages for programming language study: having fully abstract descriptions in this setting means ensuring the results obtained in the meta-language to be transferable, in principle, to object languages. In a later exposition we wish to report how the proposed typed syntax can be a powerful tool for language analysis when coupled with process-theoretic reasoning techniques.
From the viewpoint of the semantic study of sequentiality [6,10,27], our work positions sequentiality as a sub-class of the general universe of name passing interactive behaviour. This characterisation allows us to delineate sequentiality against the background of a broad computational universe which, among others, includes concurrency and non-determinism, offering a uniform basis on which various semantic findings can be integrated and extensions considered. A significant point in this context is the close connection between the presented calculus and game semantics [3,20,23]: the structure of interaction of typed processes (with respect to typed environments) precisely conforms to the intensional structures of games introduced in [23] and studied in e.g. [2,11,20,25,26]. It is notable that the type discipline itself does not mention basic notions in game semantics such as visibility, well-bracketing and innocence (although it does use a syntactic form of IO-alternation): yet they are derivable as operational properties of typed processes. We use this correspondence combined with process-theoretic reasoning techniques to establish full abstraction. While we expect a direct behavioural proof would be possible, the correspondence, in addition to facilitating the proof, offers deeper understanding of the present type discipline and game semantics.

We briefly give comparisons with related work. Hyland and Ong [24] presented a $\pi$-calculus encoding of innocent strategies of their games and show operational correspondence with a $\pi$-calculus encoding of PCF. Fiore and Honda [11] propose another $\pi$-calculus encoding for call-by-value games [20]. Our work, while being built on these preceding studies, is novel in that it puts forward a general type discipline where typability ensures functional sequentiality. In comparison with game semantics, our approach differs as it is based on a syntactic calculus representing a general notion of concurrent, communicating processes. In spite of the difference, our results do confirm some of the significant findings in game semantics, such as the equal status owned by call-by-name and call-by-value evaluation. From a different viewpoint, our work shows an effective way to apply game semantics to the study of basic typing systems for the $\pi$-calculus, in particular for the proof of full abstraction of encodings. Concerning the use of the $\pi$-calculus as the target language for translations, [28] was the first to point out the difficulty of fully abstract embeddings of functional sequentiality and [32] showed that the same problems arise even with the higher-order $\pi$-calculus. While some preceding work studies the significance of replication and linearity of channels [9,16,22,29,31,34,37], none offers a fully abstract interpretation of functional sequentiality.

In the remainder, Section 2 and 3 introduce the typed calculus. Section 4 analyses operational structures of typed terms. Based on them Section 5 establishes full abstraction. The technical details, including proofs omitted from the main sections of the paper, can be found in the full version [4].

Acknowledgements. We thank Makoto Hasegawa and Vasco Vasconcelos for their comments. The work of the first two authors is partially supported by EPSRC grant GR/N/37633.