Higher-Order Symb Comput (2011) 24:1–2
DOI 10.1007/s10990-011-9069-9

EDITORIAL

Editorial

Julia Lawall · Germán Puebla · Germán Vidal

Published online: 3 June 2011

Partial evaluation and Program Manipulation (PEPM) has been held as a conference, symposium or workshop most years since 1991, under the sponsorship of ACM SIGPLAN. This volume comprises extended versions of five of the papers presented at the PEPM 2009 workshop [1], held in Savannah, Georgia, USA, in January 2009. All papers were rigorously reviewed by at least three reviewers. Reflecting the wide scope of PEPM, this special issue includes articles on type inference for Java and for Verilog, on the definition of program analyses, on program optimization, and on aspect-oriented programming.

The problem of incomprehensible error messages has long plagued users, particularly new users, of statically typed functional languages, and with the introduction of generics, now plagues Java programmers. "Improving type error messages for Generic Java," by Nabil El Boustani and Jurriaan Hage, shows a number of examples in which the behavior of standard compilers is difficult to understand or even inconsistent. It then proposes a postprocessor to be integrated into the Java type system that collects information about type constraints in order to produce more comprehensible type error messages. Numerous examples are provided and the approach has been implemented in JastAdd.

Abstract interpretation is a well known approach for program analysis: first, the operational semantics of a programming language is specified and, then, a terminating approximation of this semantics is derived. However, it is often the case that both the specification of the semantics and the associated approximation should be changed in order to obtain different analyses, and this may require considerable ingenuity. In "Logical approximation for program analysis," Robert Simmons and Frank Pfenning introduce a novel approach

J. Lawall (✉)
DIKU, University of Copenhagen, Copenhagen, Denmark
e-mail: julia@diku.dk

G. Puebla
Universidad Politécnica de Madrid, Boadilla del Monte, Spain

G. Vidal
MiST, DSIC, Universitat Politécnica de Valéncia, Valencia, Spain

based on ordered logic to specify interpreters for programming languages in the style of substructural operational semantics. These specifications can then be automatically translated to linear logical specifications in such a way that control flow information is exposed and can be manipulated by an eventual approximation. A meta-approximation theorem is established that ensures that approximations compute abstractions when interpreted as saturated logic programs. The relative ease of encoding of two different analyses suggests the new technique can be used to derive other program analyses.

Hardware description involves complex repetitive low-level coding, making it an attractive target for program generation. Indeed, the Verilog hardware description language provides generative features. Nevertheless, generative code is not checked until it is elaborated, which is a time-consuming process, and Verilog is tolerant of some type inconsistency errors, even though these are likely bugs. "Static consistency checking for Verilog wire interconnects," by Cherif Salama, Gregory Malecha, Walid Taha, Jim Grundy and John O'Leary, tackles this problem by defining a dependent type system for Verilog generative code, and by resolving the constraints induced by this type system using satisfiability modulo theories (SMT) technology. They prove type preservation and type safety for a core language, and show that type inference is efficient in practice for circuits implementing a number of standard arithmetic operations.

A common pattern in functional programming relies on the well known producer/consumer scheme, where one function produces some intermediate data structure that is then consumed by another function. Despite the conceptual advantages of this programming pattern, producing intermediate data structures may also degrade the efficiency of the program. An alternative approach in a lazy functional setting is based on using circular programs, where the argument of a function call may also appear in its result. In "Shortcut fusion rules for the derivation of circular and higher-order programs," Alberto Pardo, João Fernandes and João Saraiva present new shortcut fusion rules for the derivation of circular and higher-order programs from programs that follow the producer/consumer pattern. The transformed programs construct no intermediate data structures and are thus more efficient in general. Moreover, the formulation of the rules is general enough to be instantiated for a wide class of algebraic data types and monads.

In recent years, there have been a number of proposals for integrating aspect-oriented programming into pure, lazy functional languages. As an aspect represents an orthogonal concern to that of the base program, the structure of the base program should not be influenced by the possibility of integrating an aspect. This poses a complication in the case of pure functional languages, as the side effects that are performed by many common uses of aspect-oriented programming, such as logging, require that the entire program be written in a monadic style. "Side-effect localization for lazy, purely functional languages via aspects," by Kung Chen, Shu-Chun Wang, Jia-Yin Lin, Meng Wang, and Siau-Cheng Khoo, proposes a technique for automatically introducing the use of monads in this case. They prove that this monadification preserves the property of noninterference between the aspect and the base program. In this, they propose an approach to ensure that an aspect does not affect the evaluation order of the program, preserving the lazy semantics.

We would like to thank all of the reviewers who helped with the paper selection process, both for the PEPM 2009 workshop and for this special issue.

References

1. Puebla, G., Vidal, G. (eds.): In: ACM SIGPLAN Workshop on Partial Evaluation and Program Manipulation (PEPM 2009), Savannah, Georgia, USA, January 2009. ACM Press