VMIL 2011

The 5th Workshop on Virtual Machines and Intermediate Languages

Hridesh Rajan[λ], Michael Haupt[φ], Christoph Bockisch[β], Robert Dyer[λ]

[λ]Iowa State University, [φ]Oracle Labs, Potsdam, Germany, and [β]Universiteit Twente

[λ]{hridesh,rdyer}@cs.iastate.edu, [φ]michael.haupt@oracle.com, and [β]c.m.bockisch@cs.utwente.nl

Abstract

The VMIL workshop is a forum for research in virtual machines and intermediate languages. It is dedicated to identifying programming mechanisms and constructs that are currently realized as code transformations or implemented in libraries but should rather be supported at VM level. Candidates for such mechanisms and constructs include modularity mechanisms (aspects, context-dependent layers), concurrency (threads and locking, actors, software transactional memory), transactions, etc. Topics of interest include the investigation of which such mechanisms are worthwhile candidates for integration with the run-time environment, how said mechanisms can be elegantly (and reusably) expressed at the intermediate language level (e.g., in bytecode), how their implementations can be optimized, and how virtual machine architectures might be shaped to facilitate such implementation efforts.

Categories and Subject Descriptors D.3.4 [*Programming Languages*]: Processors—run-time environments

General Terms Design, Languages, Performance

Keywords Virtual machine, intermediate language

1. Motivations and Themes

An increasing number of high-level programming language implementations is realized using standard virtual machines. Recent examples of this trend include the Clojure (Lisp) and Potato (Squeak Smalltalk) projects, which are implemented on top of the Java Virtual Machine (JVM); and also F# (ML) and IronPython, which target the .NET CLR. Making diverse languages–possibly even adopting different paradigms–available on a robust and efficient common platform leverages language interoperability.

Vendors of standard virtual machine implementations have started to adopt extensions supporting this trend from the run-time environment side. For instance, the Sun standard JVM will include the *invokedynamic* instruction, which will facilitate a simpler implementation of dynamic programming languages on the JVM.

The observation that many language constructs are supported in library code, or through code transformations leading to over-generalized results, has led to efforts to make the core mechanisms of certain programming paradigms available at the level of the virtual machine implementation. Thus, dedicated support for language constructs enables sophisticated optimization by direct access to the running system. This approach has been adopted by several projects aiming at providing support for aspect-oriented programming or dynamic dispatch in general-purpose virtual machines (Steamloom, Nu, ALIA4J).

The main themes of this workshop are to investigate which programming language mechanisms are worthwhile candidates for integration with the run-time environment, how said mechanisms can be declaratively (and re-usably) expressed at the intermediate language level (e.g., in bytecode), how their implementations can be optimized, and how virtual machine architectures might be shaped to facilitate such implementation efforts. Possible candidates for investigation include modularity mechanisms (aspects, context-dependent layers), concurrency (threads and locking, actors, software transactional memory), transactions, paradigm-specific abstractions, and combinations of paradigms.

The areas of interest include, but are not limited to, compilation-based and interpreter-based virtual machines as well as intermediate-language designs with better support for investigated language mechanisms, compilation techniques from high-level languages to enhanced intermediate languages as well as native machine code, optimization strategies for reduction of run-time overhead due to either compilation or interpretation, advanced caching and memory management schemes in support of the mechanisms, and additional virtual machine components required to manage them.

SPLASH'11, October 22–27, 2011, Portland, Oregon, USA.
ACM 978-1-4503-0940-0/11/10.

2. Goals and Expected Results

We intend to solicit both technical and position papers. Our expectation is to receive contributions that, on the one hand, point out mechanisms and concepts worth to be supported at the level of the execution environment; and, on the other, provide more detailed descriptions of implementation approaches for such mechanisms and concepts. These papers should act as motivation for new researchers to include the topics of this workshop into their research. To accomplish this, we will make all papers available on the workshop web page; and we intend to publish the workshop proceedings—consisting of selected high-quality papers and extended abstracts of the remaining accepted papers—in the ACM digital library. The proceedings of the first four workshops in this series have already been published in the ACM digital library. We also intend to open the workshop to researchers without accepted papers.

It is our intention to receive submissions from researchers new to the field as well as experienced researchers and practitioners. For the former, we want to offer a platform for discussing their ideas and receiving feedback on them. This will be supported by question and answer sessions as well as by a session of group discussions. As in the past years, the program of VMIL 2011 will be complemented by high-quality invited talks.

3. Organizers

Hridesh Rajan is an Associate Professor of Computer Science at the Iowa State University. He received his Ph.D. from the University of Virginia in 2005. He is the recipient of a 2009 US NSF CAREER award and a 2010 Early Achievement in Research Award from Iowa State University.

Michael Haupt is a Principal Member of Technical Staff in the Maxine project at Oracle Labs. Previously, he has worked as a post-doctoral researcher in the Software Architecture Group at Hasso-Plattner-Institut in Potsdam. Michael holds a doctoral degree from Technische Universität Darmstadt.

Christoph Bockisch is an Assistant Professor on Software Composition with a research focus on the design and implementation of programming languages with advanced dispatching mechanisms. He received his doctoral degree from the Technische Universität Darmstadt in 2008. His PhD thesis was nominated for the German Dissertation Price 2009.

Robert Dyer is a fourth year Ph. D. student with a research focus on the design of intermediate language models and virtual machine support for advanced modularization techniques. He is the recepient of the 2009 Dr Robert Stewart Early Research Award and 2007 CRA Outstanding Undergraduate Award (Honorable Mention).

4. Program Committee

We are pleased to have assembled another excellent program committee for VMIL 2011. This year the program committee is chaired by Dr. Steve Blackburn from the Australian National University.

- Steve Blackburn (*Australian National U., Australia*)
- Cliff Click (*Azul Systems, USA*)
- David Grove (*IBM Research, USA*)
- Kim Hazelwood (*U. of Virginia, USA*)
- Antony Hosking (*Purdue U., USA*)
- Doug Lea (*State U. of New York, USA*)
- Guy Steele (*Oracle Labs, USA*)
- Ben Titzer (*Google, USA*)
- Olivier Zendra (*INRIA, France*)