
ORACLE®

Building Reusable, Low-Overhead Tooling Support into a High-Performance Polyglot VM

MoreVMs 2017

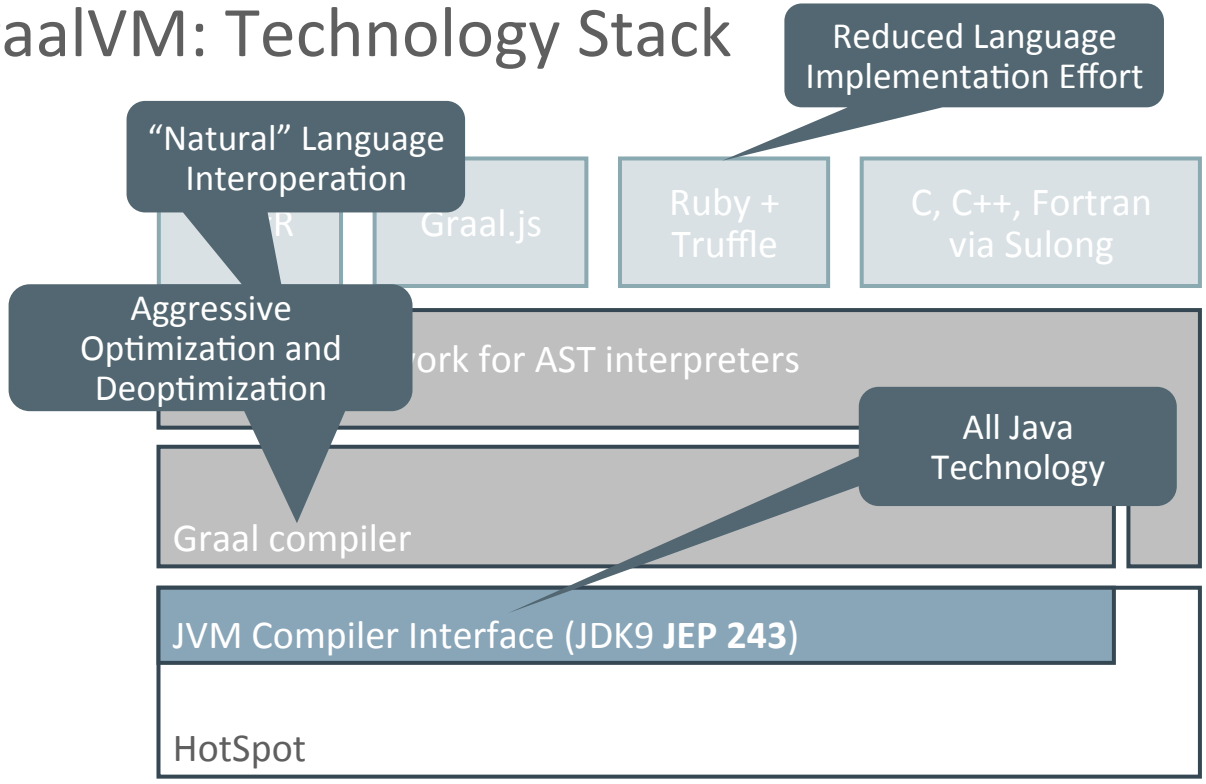
Michael L. Van De Vanter
Oracle Labs
April 3, 2017

ORACLE

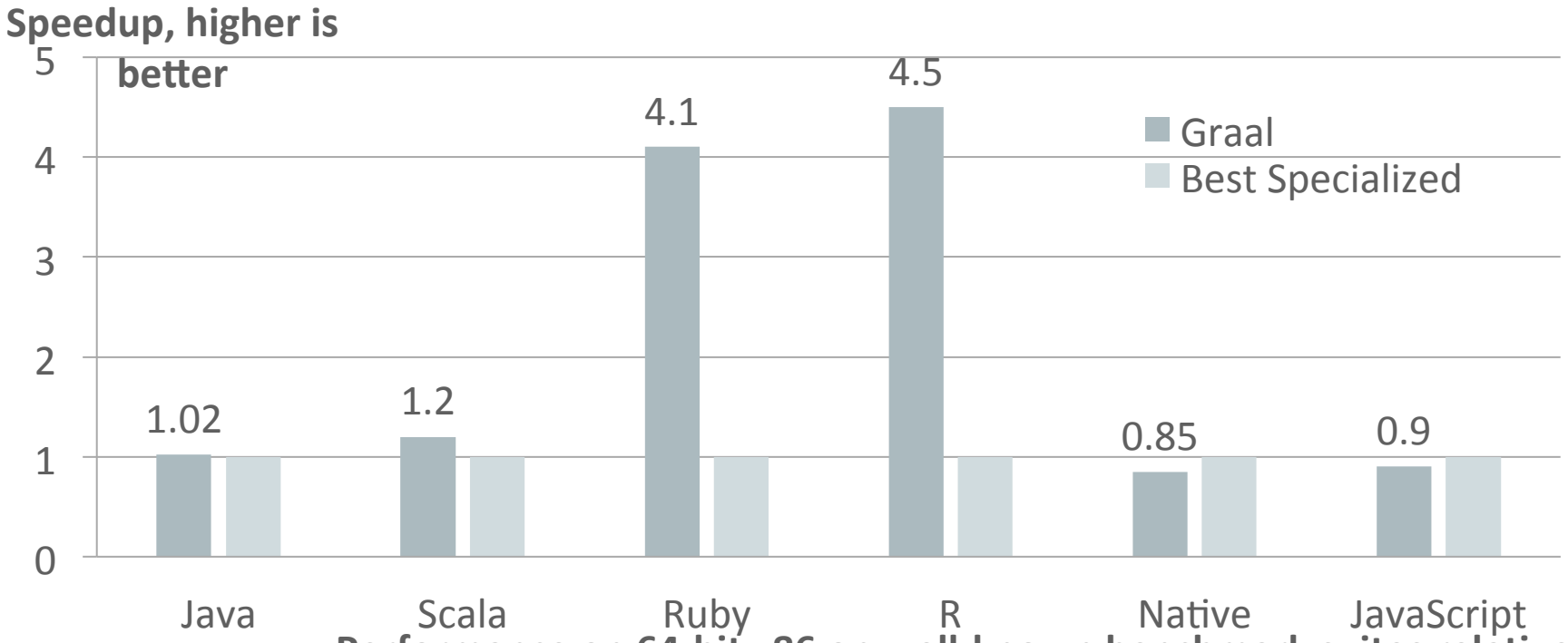
Safe Harbor Statement

The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

GraalVM: Technology Stack

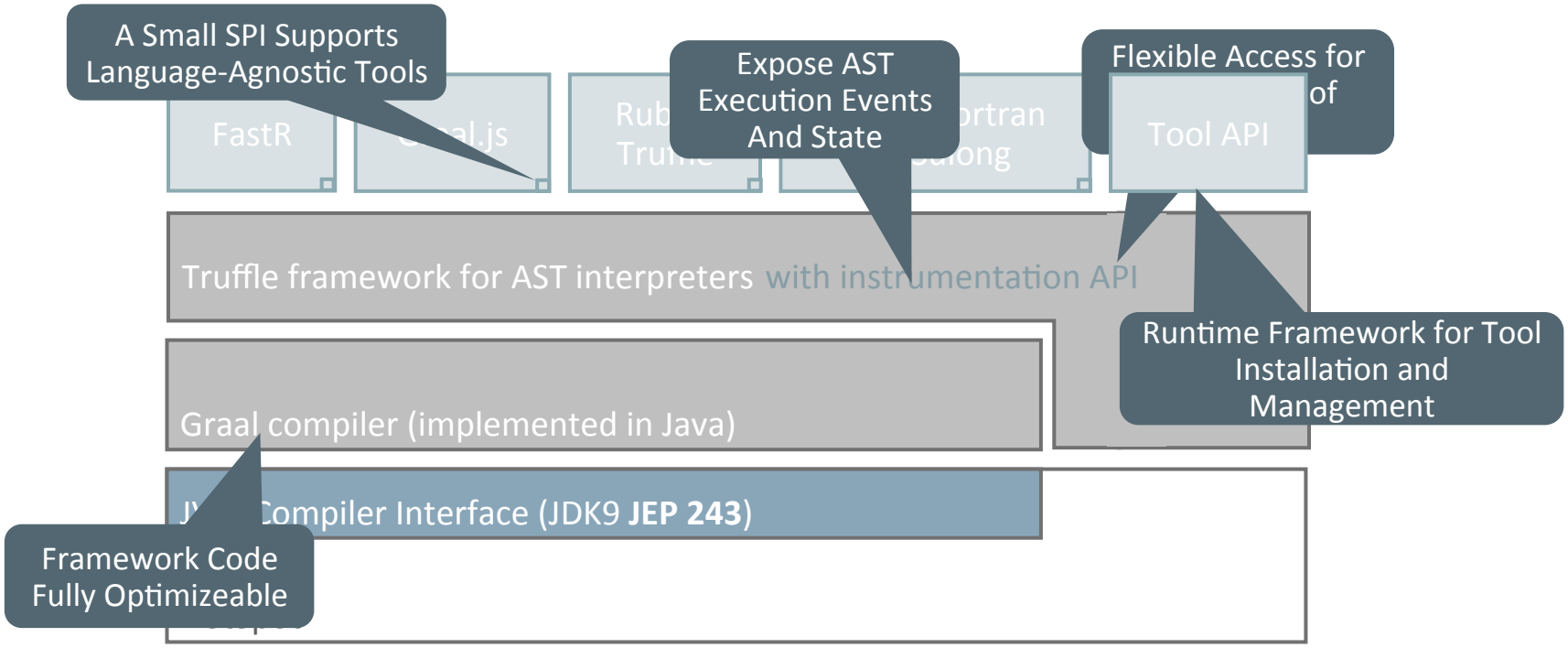


GraalVM: Peak Performance Results

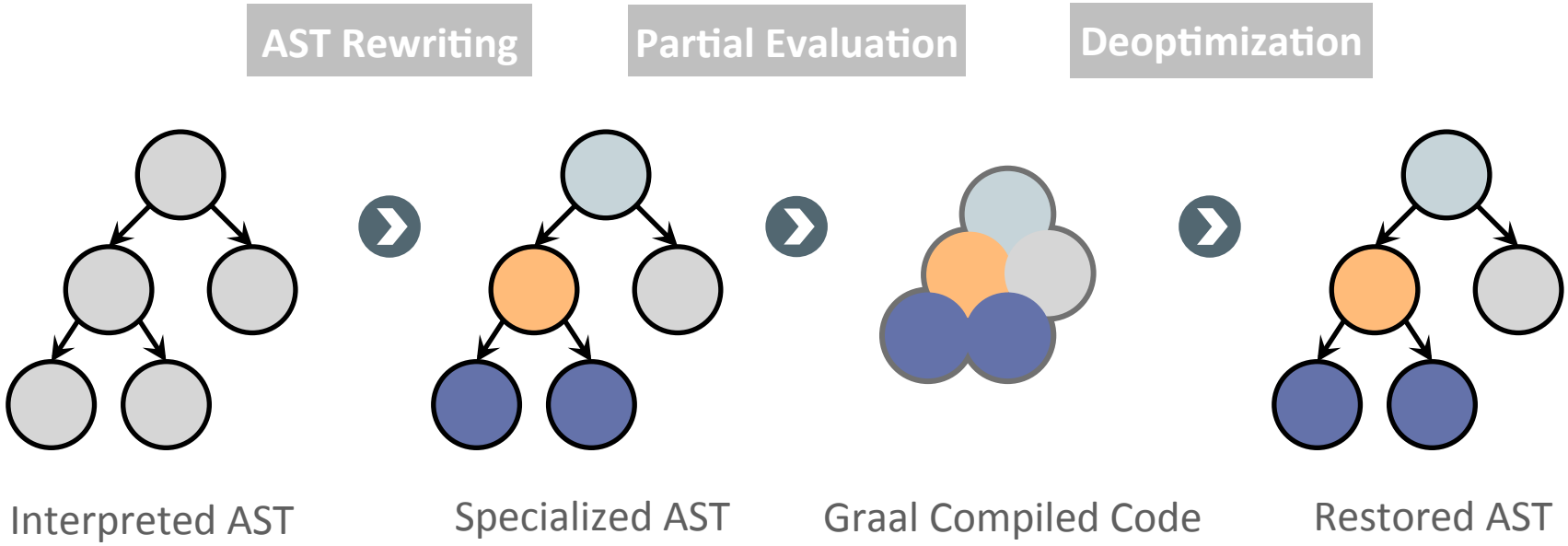


Performance on 64-bit x86 on well-known benchmark suites relative to: HotSpot/Server, JRuby, GNU R, LLVM AOT compiled, V8

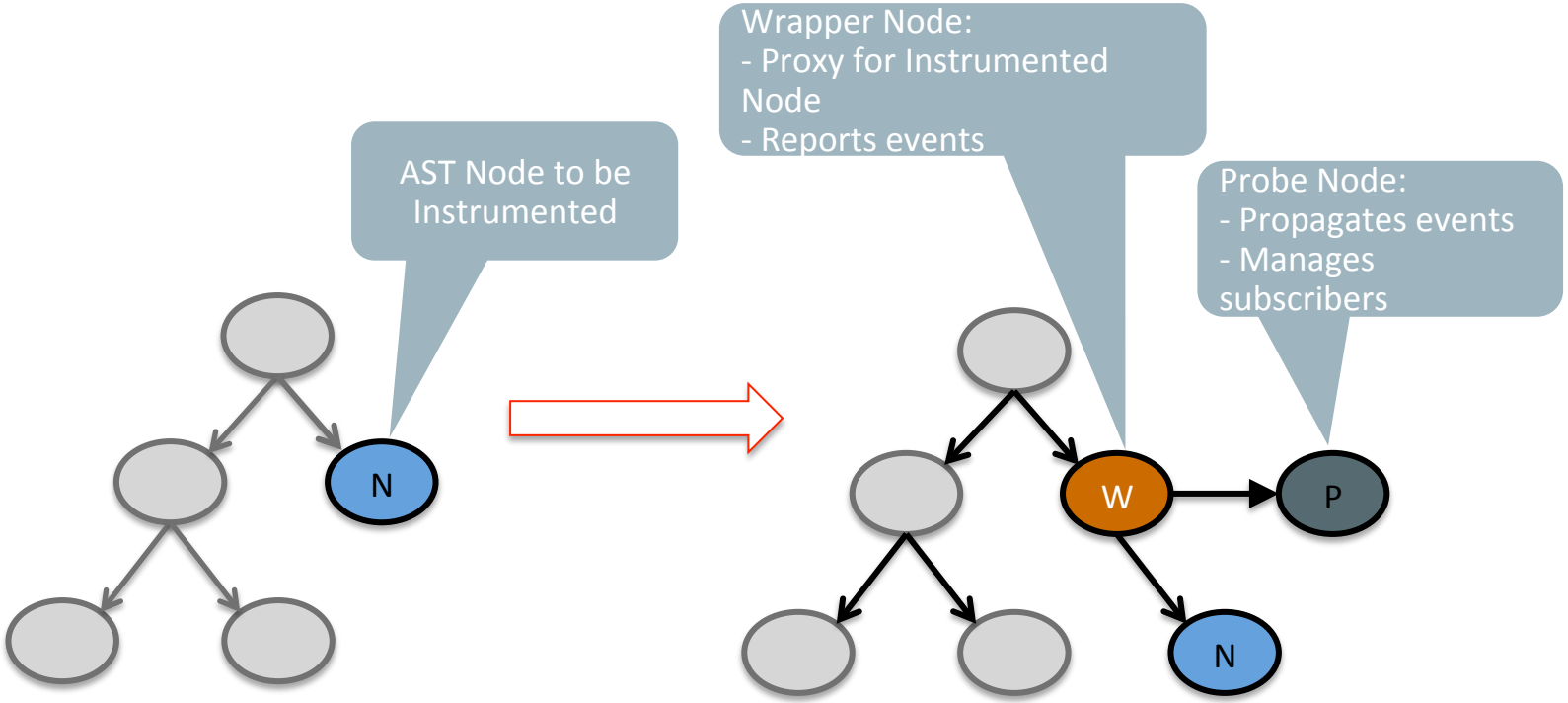
Instrumentation: Extended Truffle/Graal Technology



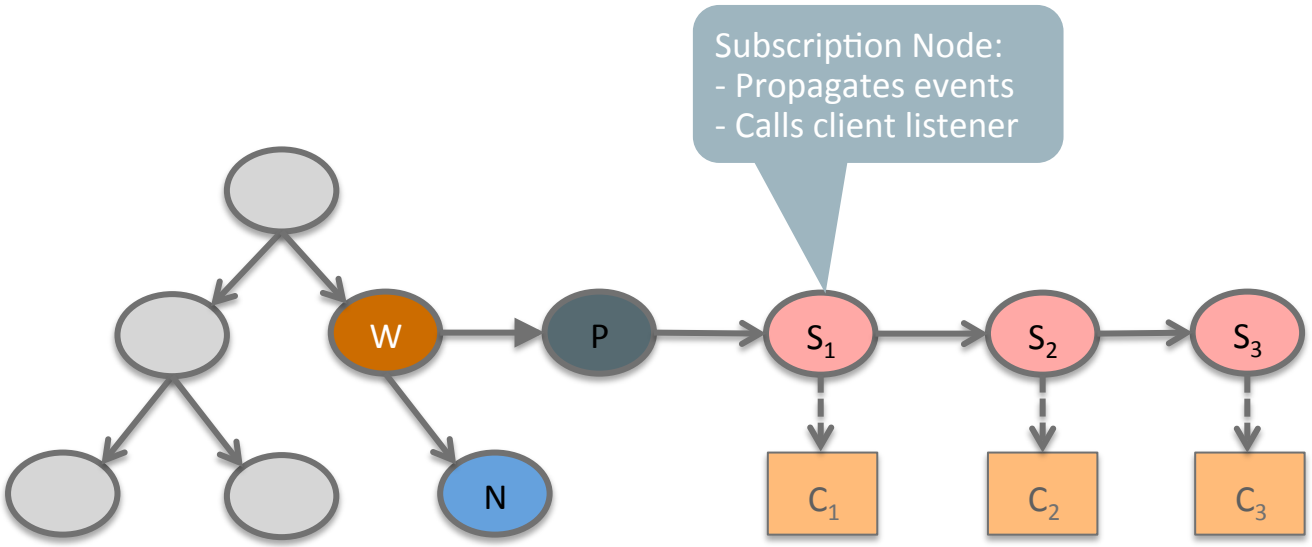
Graal Opt/Deopt: From interpreted AST to native & back



Probes: Event Capture by Node Insertion



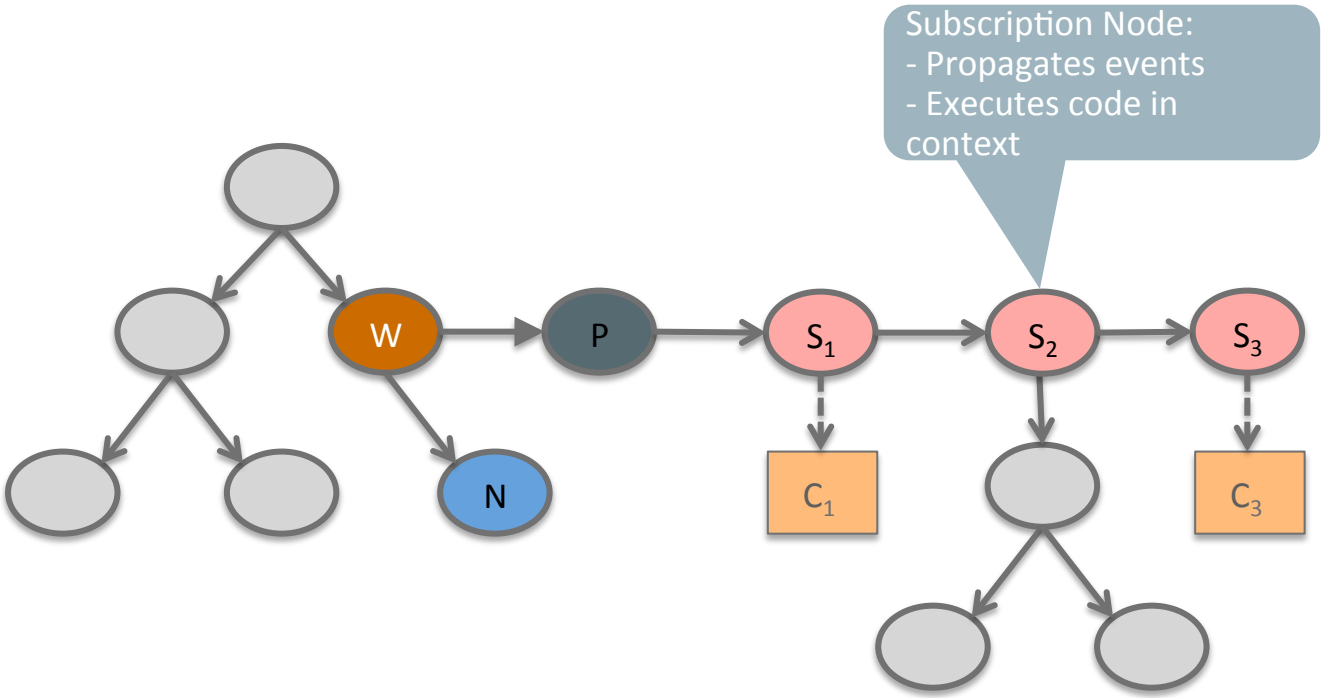
Subscriptions: Event Reporting to Clients



Subscriptions

- Locations specified by *query*:
  - For a breakpoint: “line 42 in mysource.js”
  - For stepping: “any STATEMENT” (nodes “tagged” by language implementation)
- Any number can be created/disposed dynamically
- Thread-safe
- *Deoptimizes* any affected fast-path code
  - But *lazily*, only when executed

Code Injection: Evaluation in Context



Language-Specific Support Requirements

- AST “markup”
  - Source attribution (detailed)
  - Node “tags”, e.g. *STATEMENT*, *EXPR*
- Visibility
  - Hide implementation artifacts (special frames, slot, sources, etc.)
- Presentation
  - How to display values, method names, etc.
- Dynamic eval in context of any stack frame
- Create a “patch” AST injection (e.g. for breakpoint conditions)

Performance Cost

Table 1 Performance times for `set_trace_func`, lower is better

	Disabled	Before	Empty	Increment	After
JRuby	0.555 ±0.004	15.928 ±0.062	125.371 ±0.0	338.526 ±0.0	16.707 ±0.047
TruffleRuby	0.044 ±0.001	0.044 ±0.001	0.085 ±0.001	2.096 ±0.006	0.044 ±0.0

Table 2 Performance times Ruby debugging, lower is better

	Disabled	Before	Not-taken	Conditional	After
JRuby	0.555 ±0.004	14.39 ±0.725	37.503 ±0.023	45.368 ±0.03	39.004 ±0.082
TruffleRuby	0.044 ±0.001	0.044 ±0.001	0.044 ±0.0	0.044 ±0.0	0.044 ±0.0

Applications (Known)

- Oracle Internal
  - Truffle Debugging API (clients: NetBeans, multi-language shell, web-based debugger)
  - Other APIs under develoment: Coverage, Profiling, …
  - Time boxing
  - Language features
    - Ruby set_trace_func
    - R stepping
- External
  - Event profiling, cross-language metrics (UC Irvine)
  - Code reloading (Univ. Tartu)
  - Specialized concurrency debugging, dynamic metrics (JKU Linz)

Acknowledgements

Oracle
Danilo Ansaloni
Stefan Anzinger
Cosmin Basca
Daniele Bonetta
Matthias Brantner
Petr Chalupa
Jürgen Christ
Laurent Daynès
Gilles Duboscq
Martin Entlicher
Brandon Fish
Bastian Hossbach
Christian Humer
Mick Jordan
Vojin Jovanovic
Peter Kessler
David Leopoldseder
Kevin Menard
Jakub Podlešák
Aleksandar Prokopec
Tom Rodriguez

Oracle (continued)
Roland Schatz
Chris Seaton
Doug Simon
Štěpán Šindelář
Zbyněk Šlajchrt
Lukas Stadler
Codrut Stancu
Jan Štola
Jaroslav Tulach
Michael Van De Vanter
Adam Welc
Christian Wimmer
Christian Wirth
Paul Wögerer
Mario Wolczko
Andreas Wöß
Thomas Würthinger

Oracle Interns
Brian Belleville
Miguel Garcia
Shams Imam
Alexey Karyakin
Stephen Kell
Andreas Kunft
Volker Lanting
Gero Leinemann
Julian Lettner
Joe Nash
David Piorkowski
Gregor Richards
Robert Seilbeck
Rifat Shariyar

Alumni
Erik Eckstein
Michael Haupt
Christos Kotselidis
Hyunjin Lee
David Leibs
Chris Thalinger
Till Westmann

JKU Linz
Prof. Hanspeter Mössenböck
Benoit Daloze
Josef Eisl
Thomas Feichtinger
Matthias Grimmer
Christian Häubl
Josef Haider
Christian Huber
Stefan Marr
Manuel Rigger
Stefan Rumzucker
Bernhard Urban

University of Edinburgh
Christophe Dubach
Juan José Fumero Alfonso
Ranjeet Singh
Toomas Remmelg

LaBRI
Floréal Morandat

University of California, Irvine
Prof. Michael Franz
Gulfem Savrun Yeniceri
Wei Zhang

Purdue University
Prof. Jan Vitek
Tomas Kalibera
Petr Maj
Lei Zhao

T. U. Dortmund
Prof. Peter Marwedel
Helena Kotthaus
Ingo Korb

University of California, Davis
Prof. Duncan Temple Lang
Nicholas Ulle

University of Lugano, Switzerland
Prof. Walter Binder
Sun Haiyang
Yudi Zheng

Code

- OTN product page including download:
  - www.oracle.com/technetwork/oracle-labs/program-languages
- Graal projects on github:
  - Compiler: github.com/graalvm/graal-core
  - Truffle: github.com/graalvm/truffle
  - Ruby github.com/graalvm/truffleruby
  - FastR: github.com/graalvm/fastr
  - Sulong: github.com/graalvm/sulong
- Graal on OpenJDK:
  - openjdk.java.net/projects/graal

ORACLE


ORACLE®