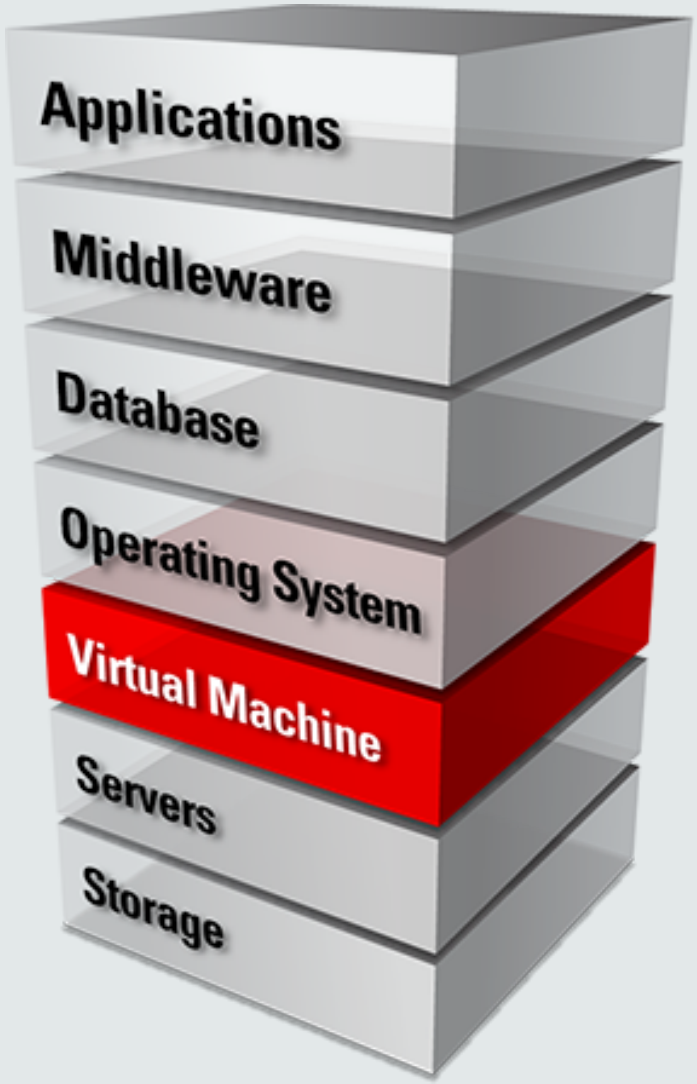

ORACLE®

Oracle Labs VM Research

2014-Sep-11

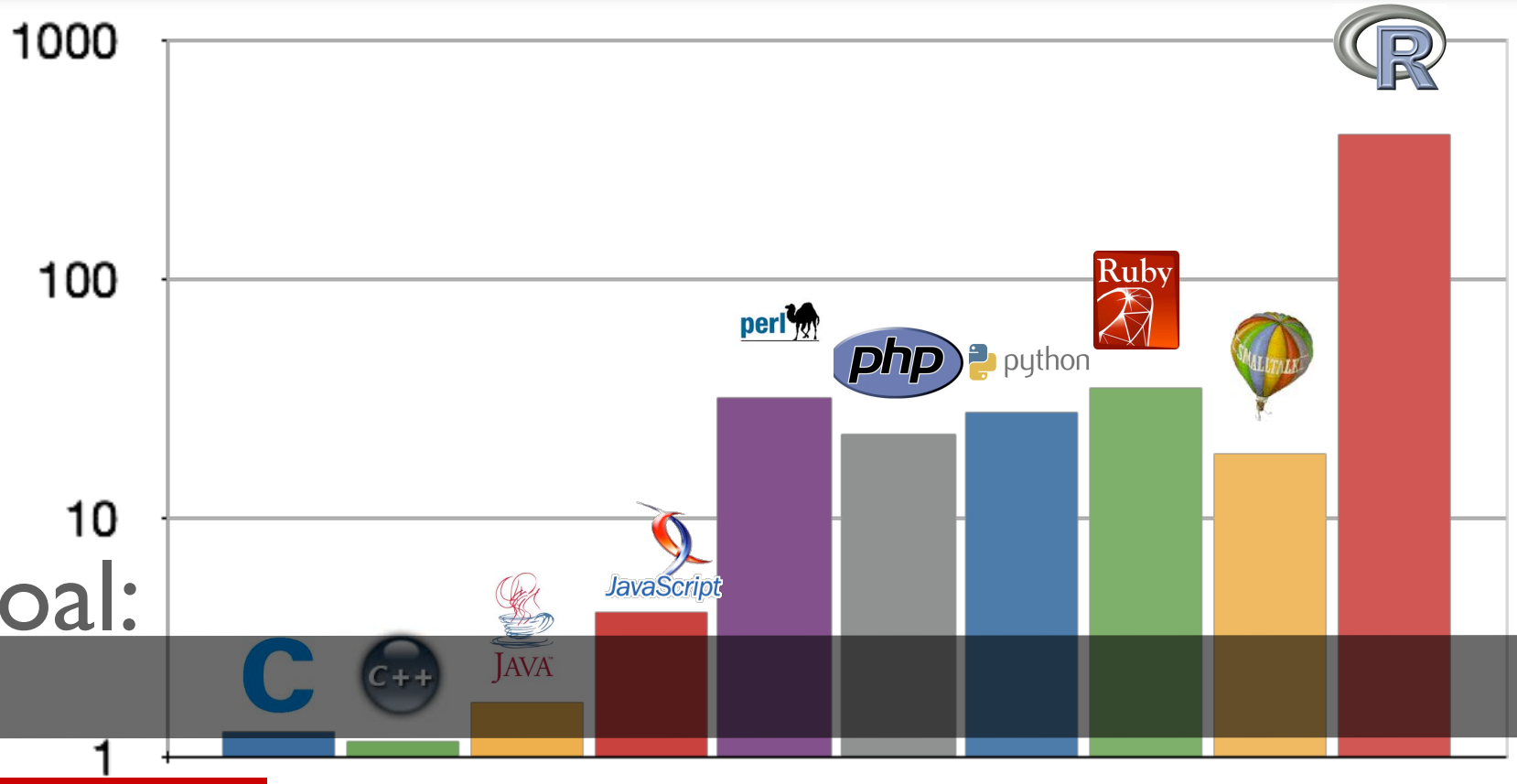
Thomas Wuerthinger
Oracle Labs



One VM to Rule Them All

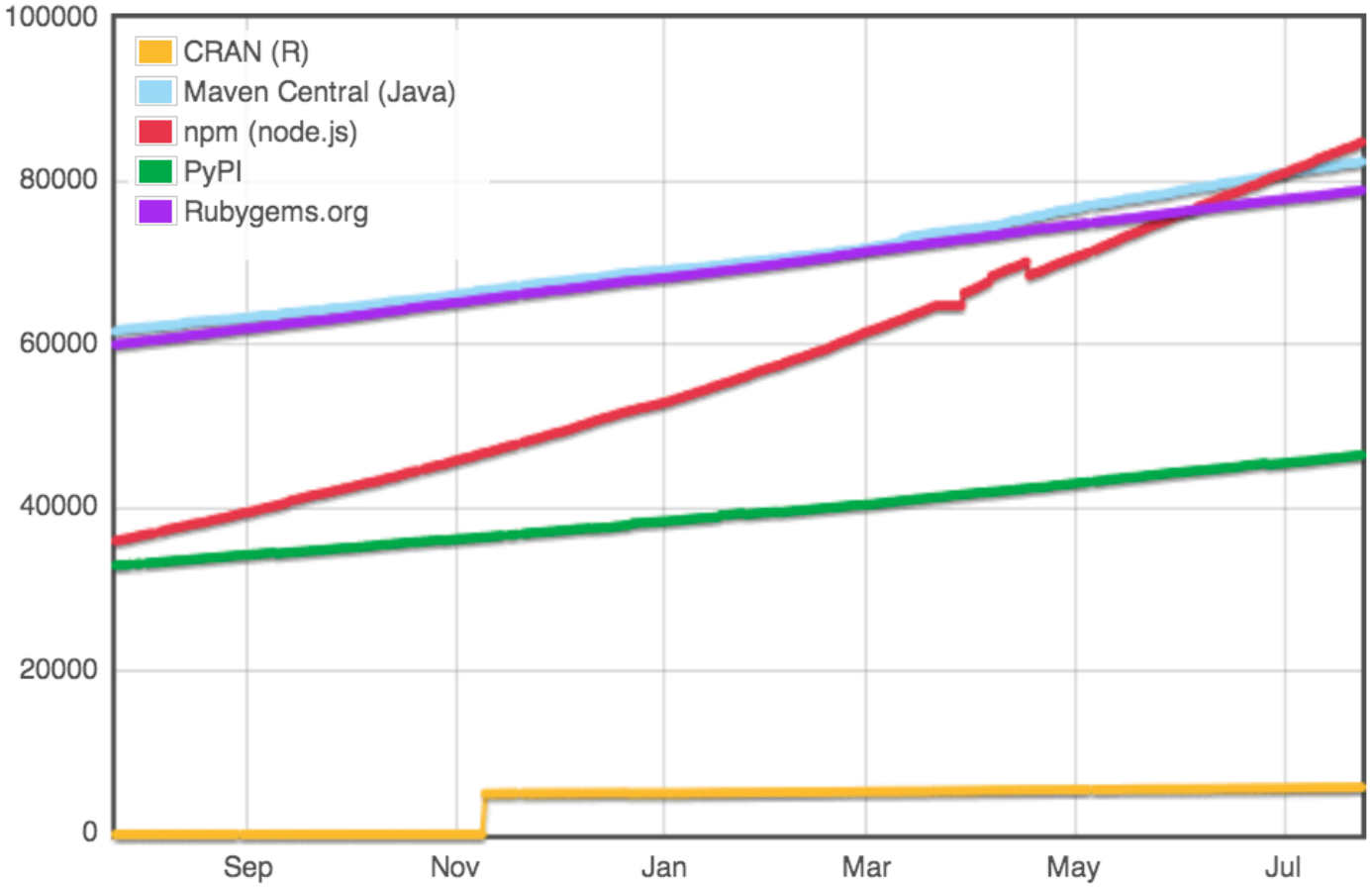
Thomas Würthinger* Christian Wimmer* Andreas Wöß† Lukas Stadler†
Gilles Duboscq† Christian Humer† Gregor Richards§ Doug Simon* Mario Wolczko*

*Oracle Labs †Institute for System Software, Johannes Kepler University Linz, Austria §S³ Lab, Purdue University

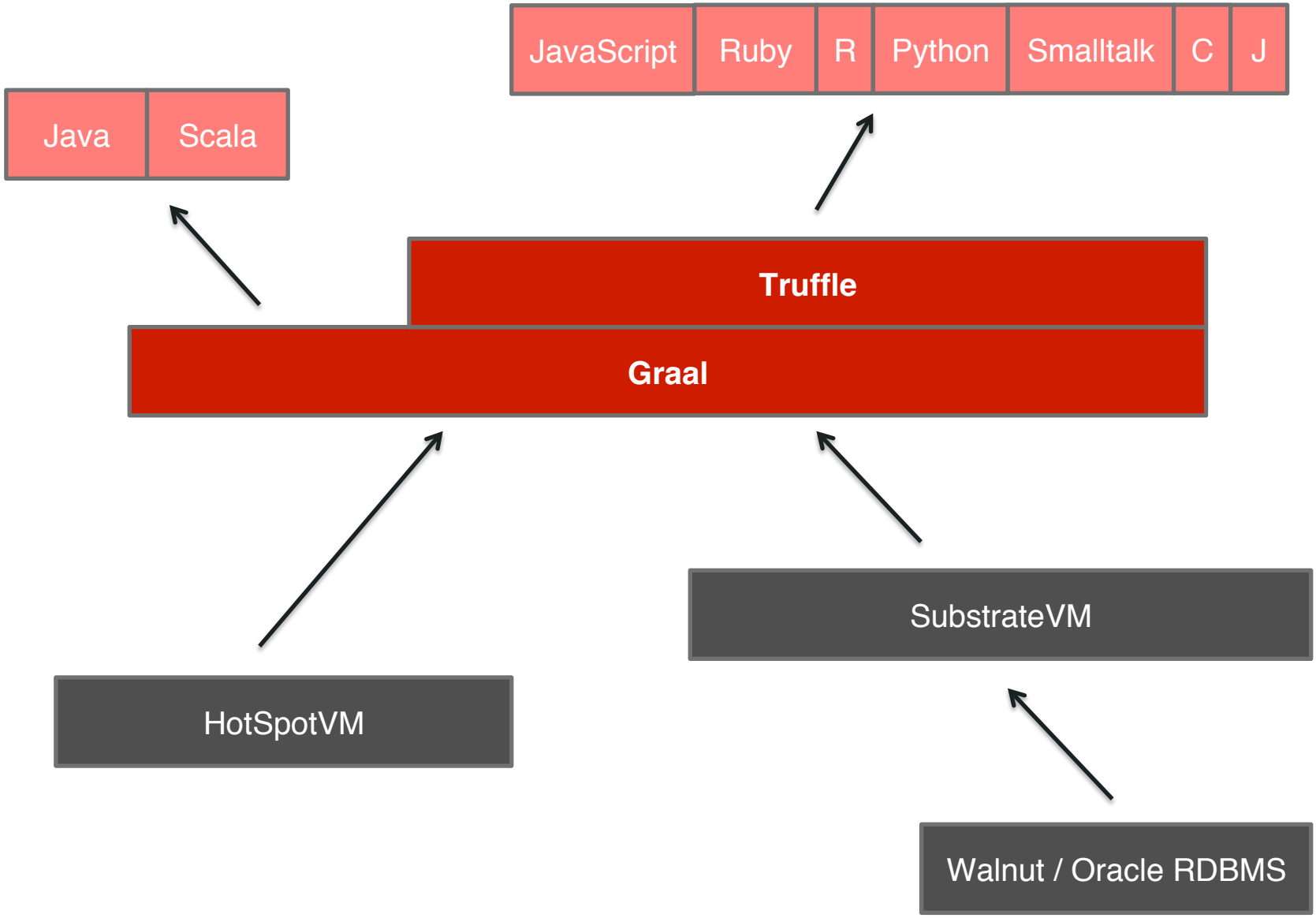


Goal:

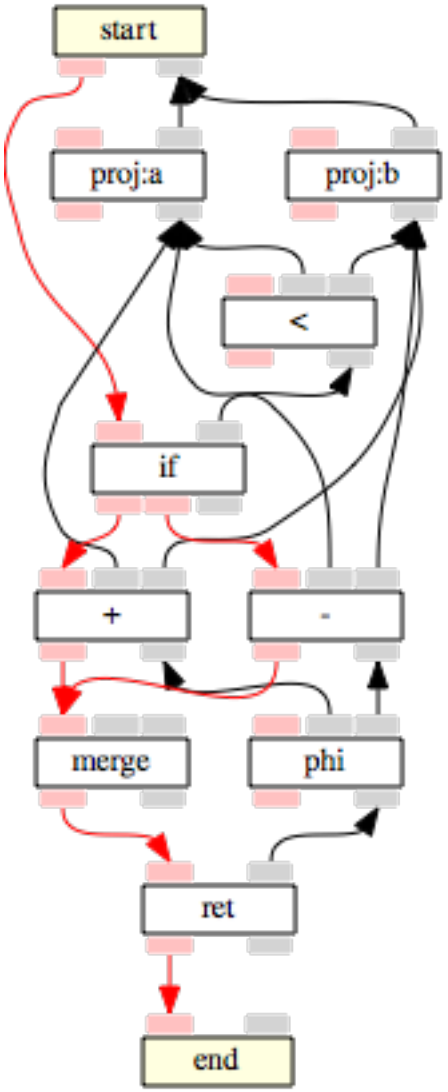
Module Counts



http://modulecounts.com/



Graal



Why Java?

Robustness: Runtime exceptions not fatal.

Reflection: Annotations instead of macros.

Meta-Evaluation: IR subgraph expressible in Java code.

Extensibility: No language barrier to the application.

Tooling: Java IDEs speed up the development process.

Graal Innovations

- IR as hybrid between CFG and PDG.
- Extensions to the concept of snippets over Maxine/Jikes.
- Floating deoptimization guards and optimistic guard movement.
- Two stage execution reassigning deoptimization information.
  - Enables more aggressive optimizations in the context of speculative execution.
- Partial escape analysis.
  - Sinking allocation and initialization to the latest possible points in the program.
- Heterogeneous computing support (developed in cooperation with AMD).

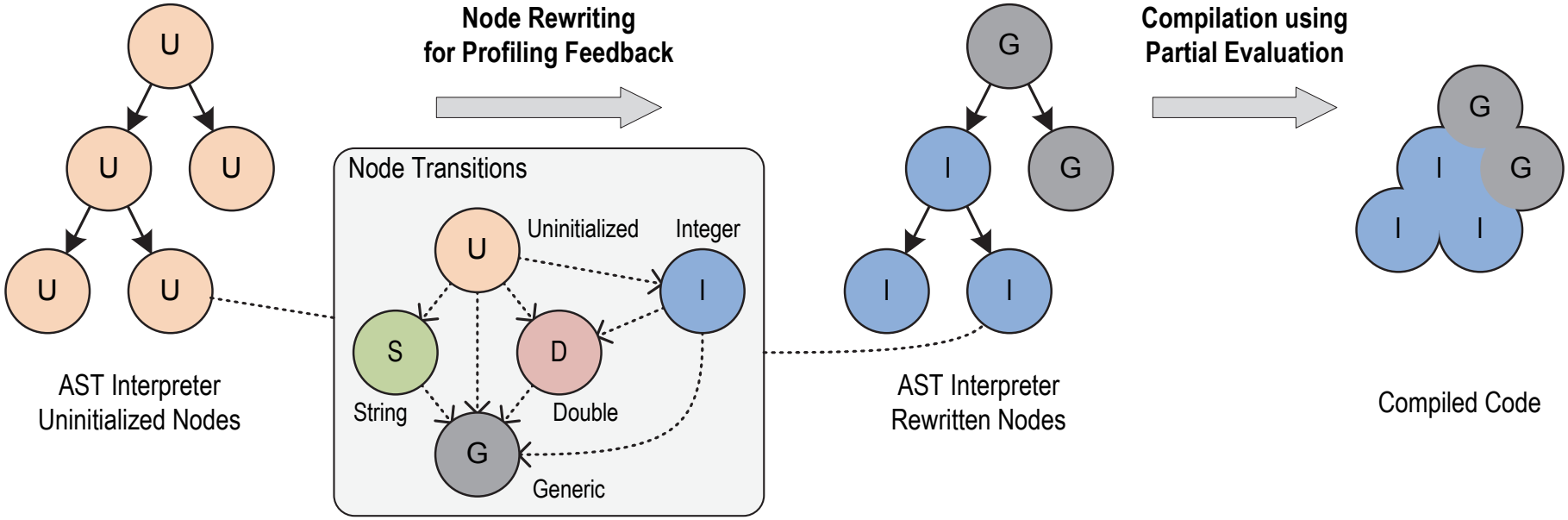
ORACLE

Truffle

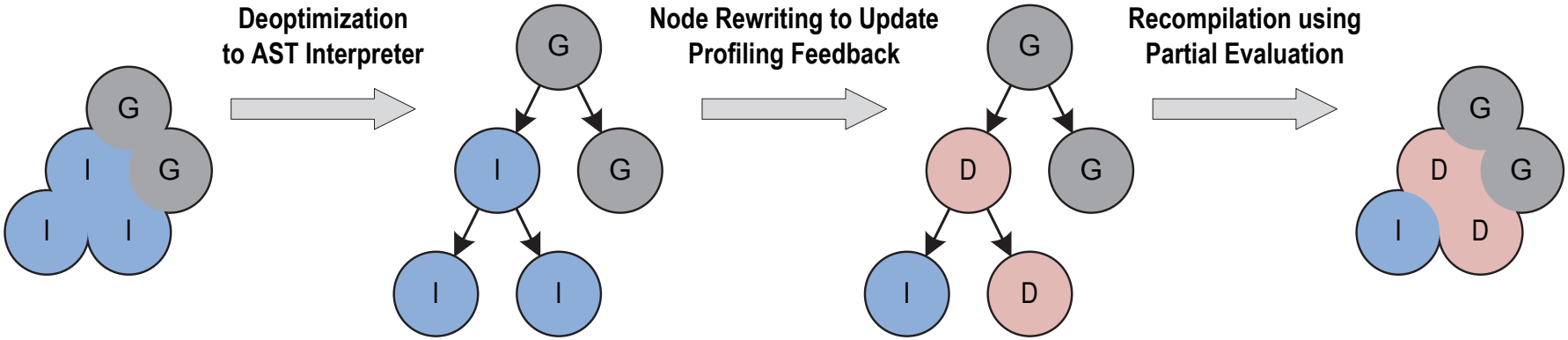
- Automatic application of the first Futamura projection for AST interpreters.
- Extended with the concept of a profiling interpreter and the ability to speculate and deoptimize the generated code.
- Interpreter specified in pure Java code.
- Annotation-based Truffle DSL for simplified specialization definition.
- Enables simple and single definition of semantics and reuse of the same compiler infrastructure for a wide variety of languages without compromising performance.
- Main differentiators to LLVM:
  - No code generation necessary.
  - Built around and optimized for profiling and speculative execution in a managed environment.

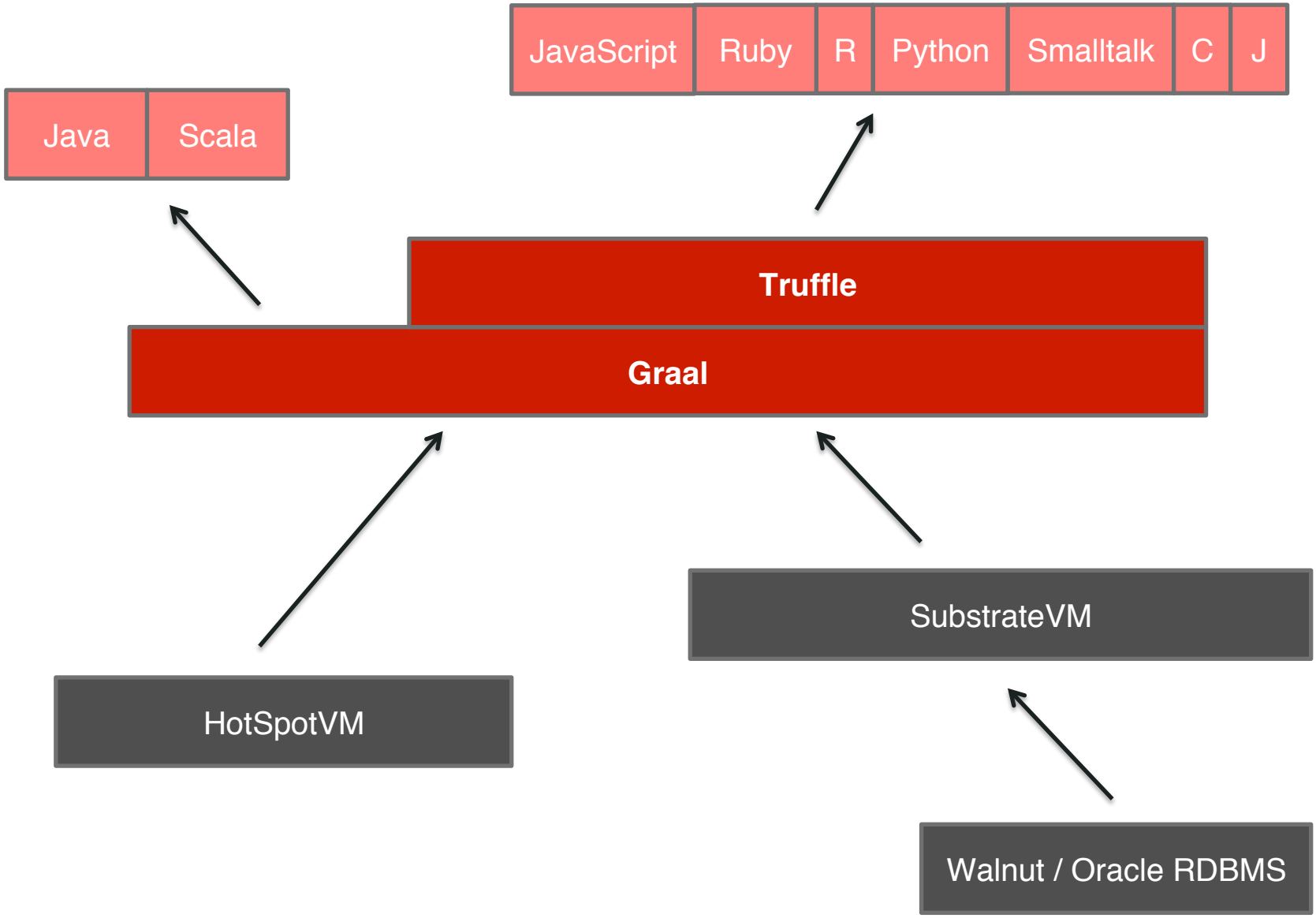
ORACLE

Speculate and Optimize …



… and Deoptimize and Reoptimize!





Substrate VM: Execution Model

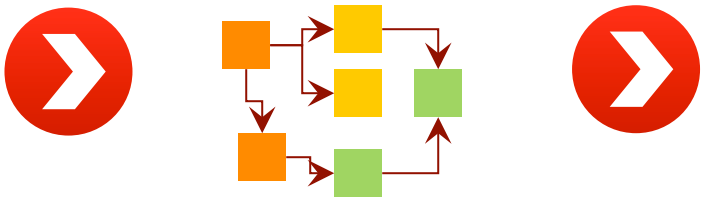
Static Analysis

Ahead-of-Time Compilation

Truffle Language

JDK

Substrate VM



Machine Code

Initial Heap

DWARF Info

ELF / MachO Binary

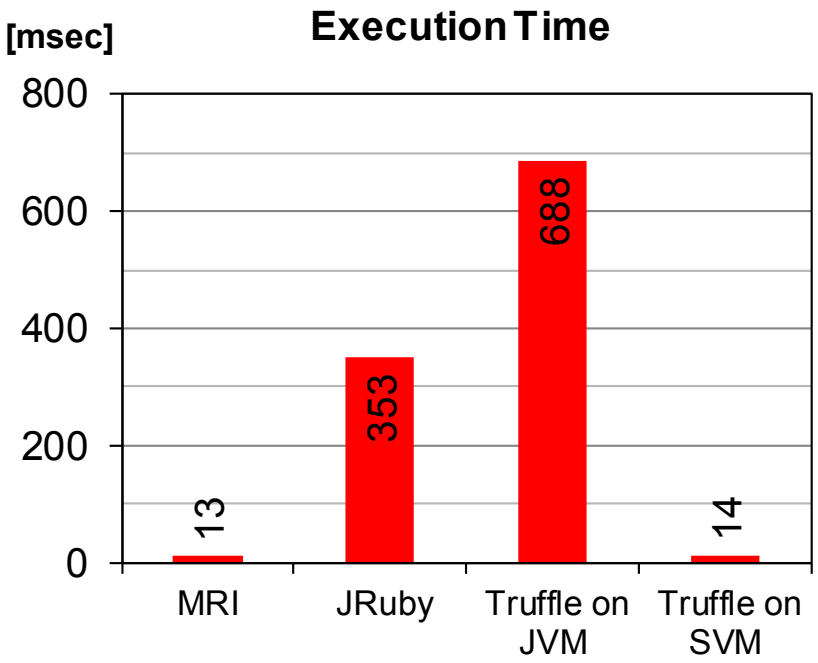
All Java classes from Truffle language (or any application), JDK, and Substrate VM

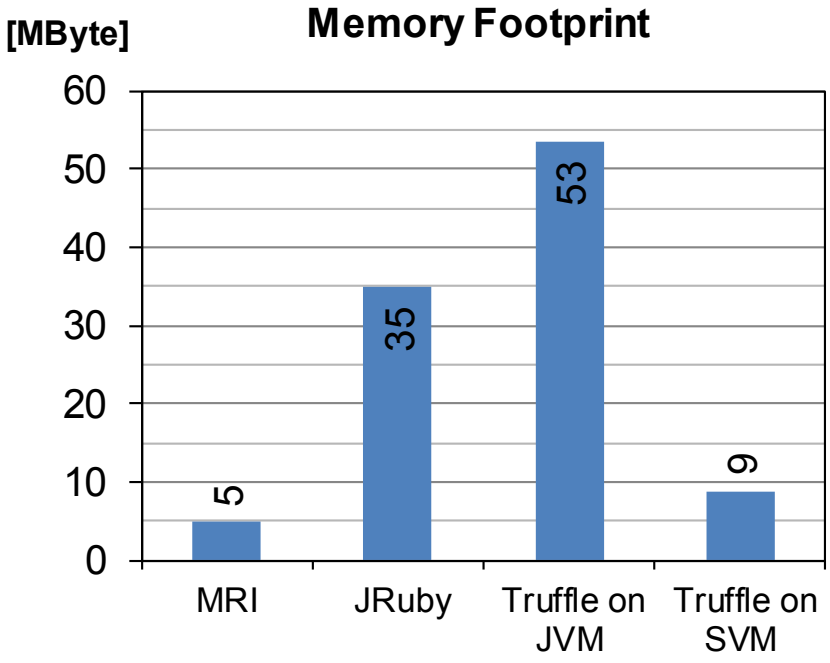
Reachable methods, fields, and classes

Application running without dependency on JDK and without Java class loading

Substrate VM: Startup Performance

- Running Ruby “Hello World”





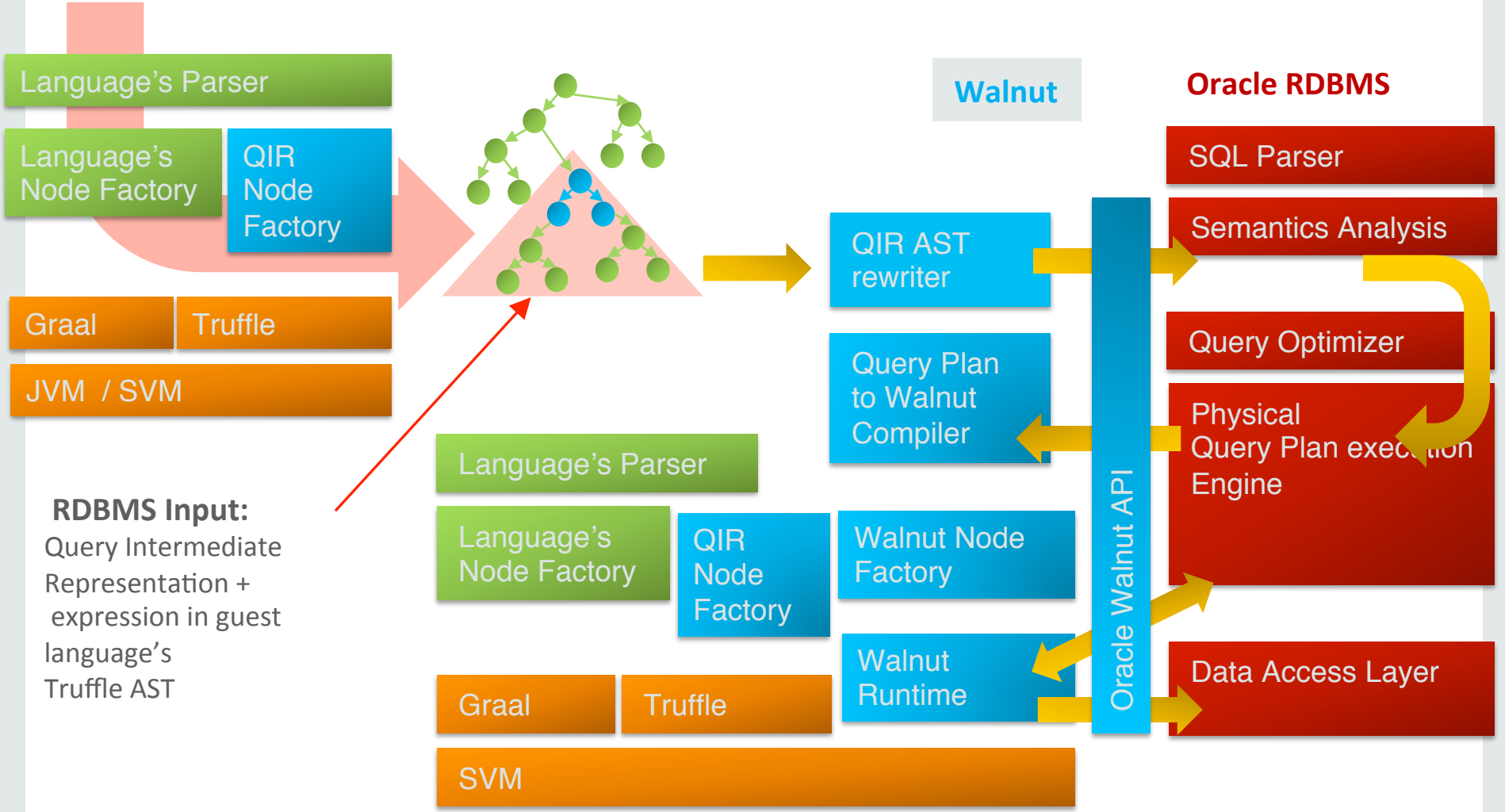
Execution time: `time -f "%e"`

Memory footprint: `time -f "%M"`

Project Walnut

- SVM + Graal enables embedding of language implemented with Truffle in RDBMS
  - Enables unfenced execution of language function/expression
- Language expression is inlined with Truffle-based Query Plans
- Deeper embedding possible if RDBMS provides a logical Query IR interface to Query Compiler
- We are looking for new hires and interns with DB and compiler expertise!

Walnut System Architecture



Acknowledgements

Oracle Labs
Danilo Ansaloni
Daniele Bonetta
Laurent Daynès
Michael Haupt
Peter Kessler
David Leibs
Tom Rodriguez
Roland Schatz
Chris Seaton
Doug Simon
Michael Van De Vanter
Christian Wimmer
Christian Wirth
Mario Wolczko
Thomas Wuerthinger

Interns
Shams Imam
Stephen Kell
Gregor Richards
Rifat Shariyar
Julian Lettner
Stefan Anzinger

JKU Linz
Prof. Hanspeter Mössenböck
Gilles Duboscq
Josef Eisl
Thomas Feichtinger
Matthias Grimmer
Christian Häubl
Josef Haider
Christian Humer
Christian Huber
David Leopoltseder
Manuel Rigger
Lukas Stadler
Bernhard Urban
Andreas Wöß

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

LaBRI
Floréal Morandat

University of California, Irvine
Prof. Michael Franz
Codrut Stancu
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

Inria France, Lille
Stefan Marr

Open Source Links

- Graal and Truffle API
  http://openjdk.java.net/projects/graal/
- R on Truffle
  https://bitbucket.org/allr/fastr
- Ruby on Truffle
  https://github.com/jruby/jruby/wiki/Truffle
- Python on Truffle
  https://bitbucket.org/ssllab/zippy
- Smalltalk on Truffle
  https://github.com/smarr/TruffleSOM

ORACLE

Hardware and Software
Engineered to Work Together


ORACLE®