Adaptive Inlining and On-Stack Replacement in the CACAO Virtual Machine

Institut für Computersprachen
Technische Universtät Wien
Austria

Edwin Steiner Andreas Krall Christian Thalinger

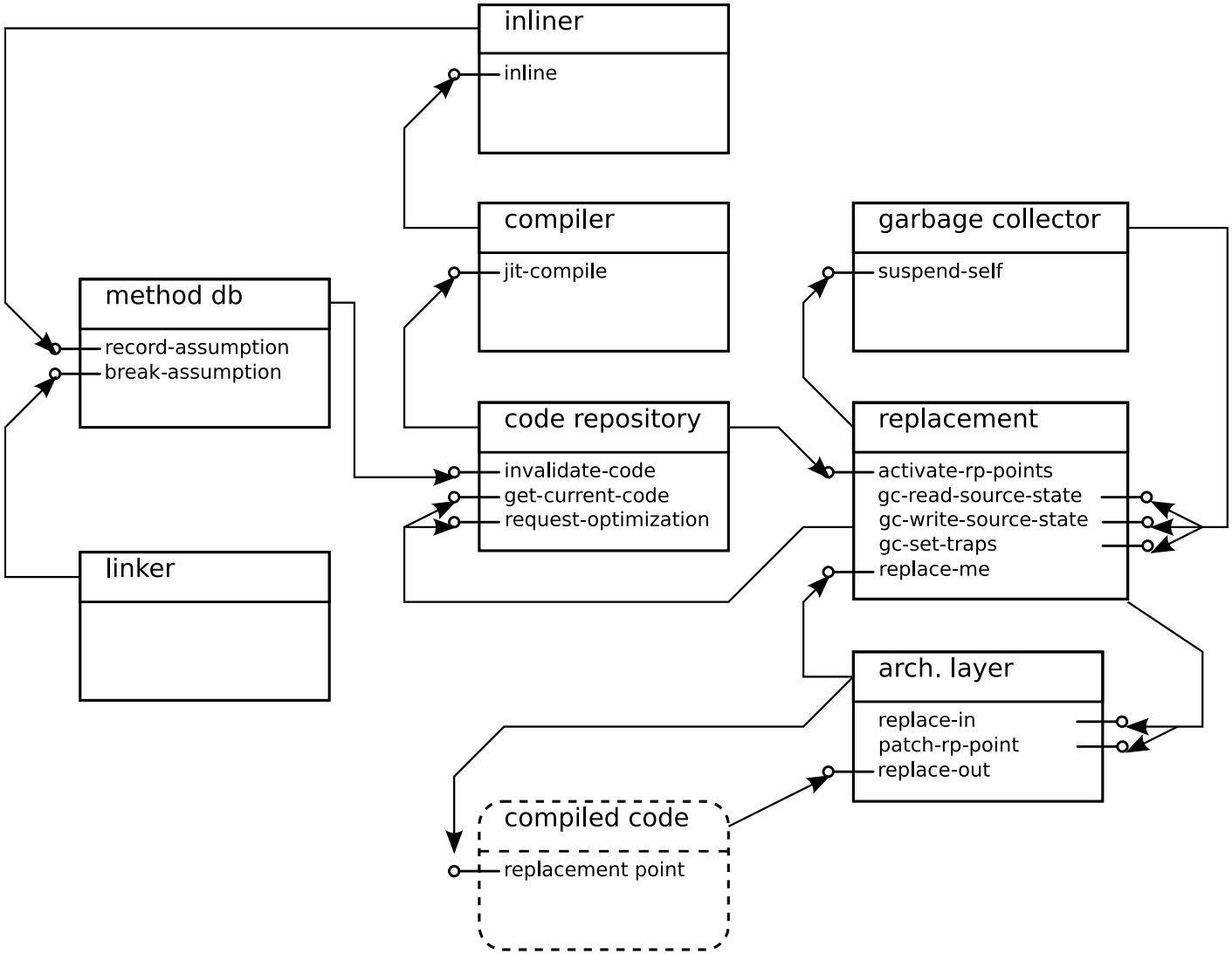
Overview

1. Introduction
2. Adaptive Optimization Framework
   - Modules of the Framework
   - Adaptive Recompilation
   - Inlining
3. On-Stack Replacement
   - Execution/Source State
   - Replacement Points
4. Empirical Evaluation
5. Conclusion and Further Work

Introduction

- object oriented programming style
- optimize program by method inlining
- virtual (synchronized) methods, exceptions
- dynamic class loading (undo)
- adaptive optimization
- CACAO - JIT for multiple architectures

Modules of the Adaptive Optimization Framework



Adaptive Recompilation

- baseline compiler (countdown traps)
- recompilation with instrumentation
- recompilation with optimizations
- deoptimization when assumptions become invalid
- on-stack replacement

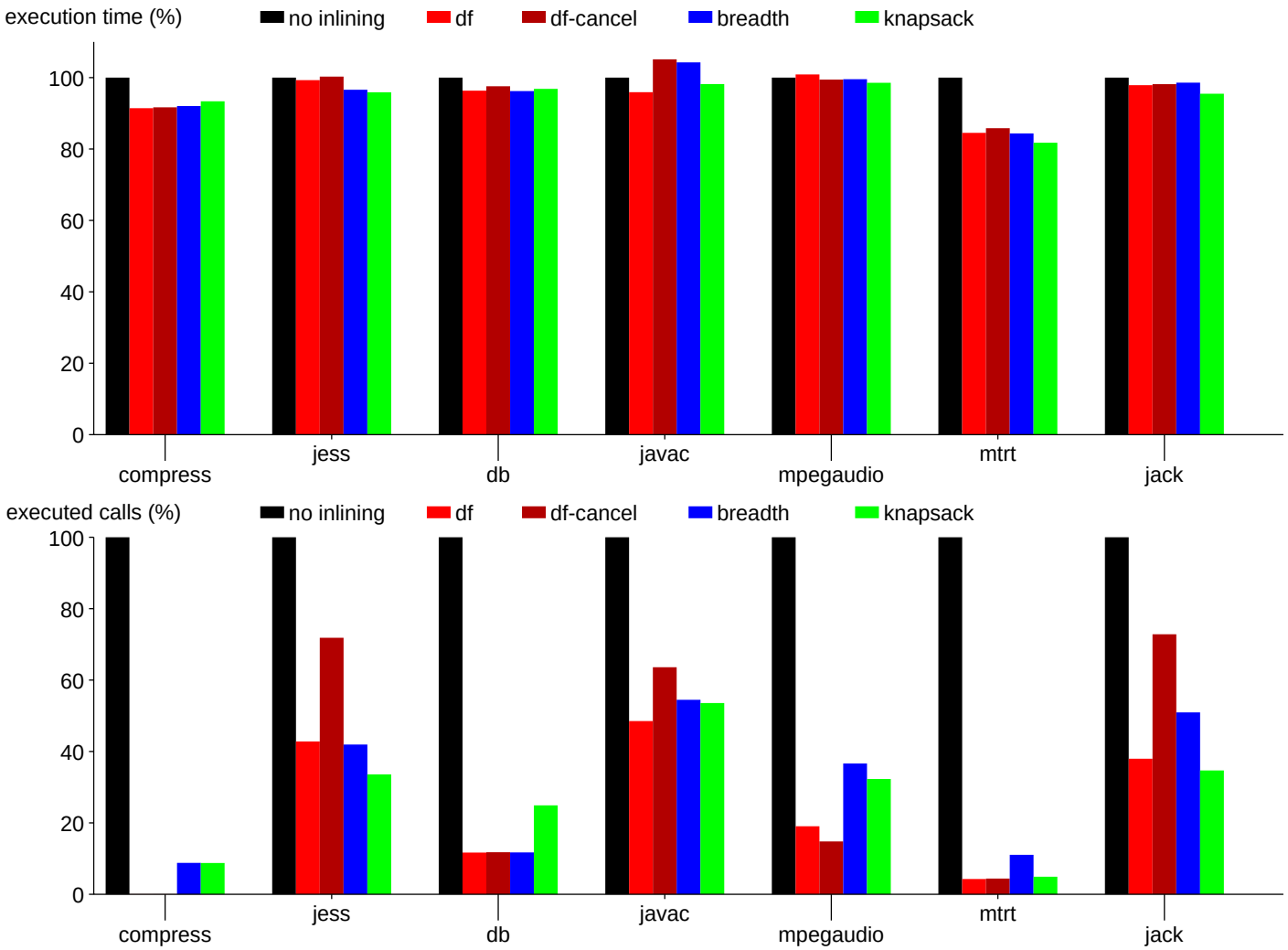
Inlining

- inlining works on intermediate representation
- profile guided (caller of hot methods)
- inlining heuristics
  - aggressive depth-first
  - aggressive breadth-first
  - knapsack

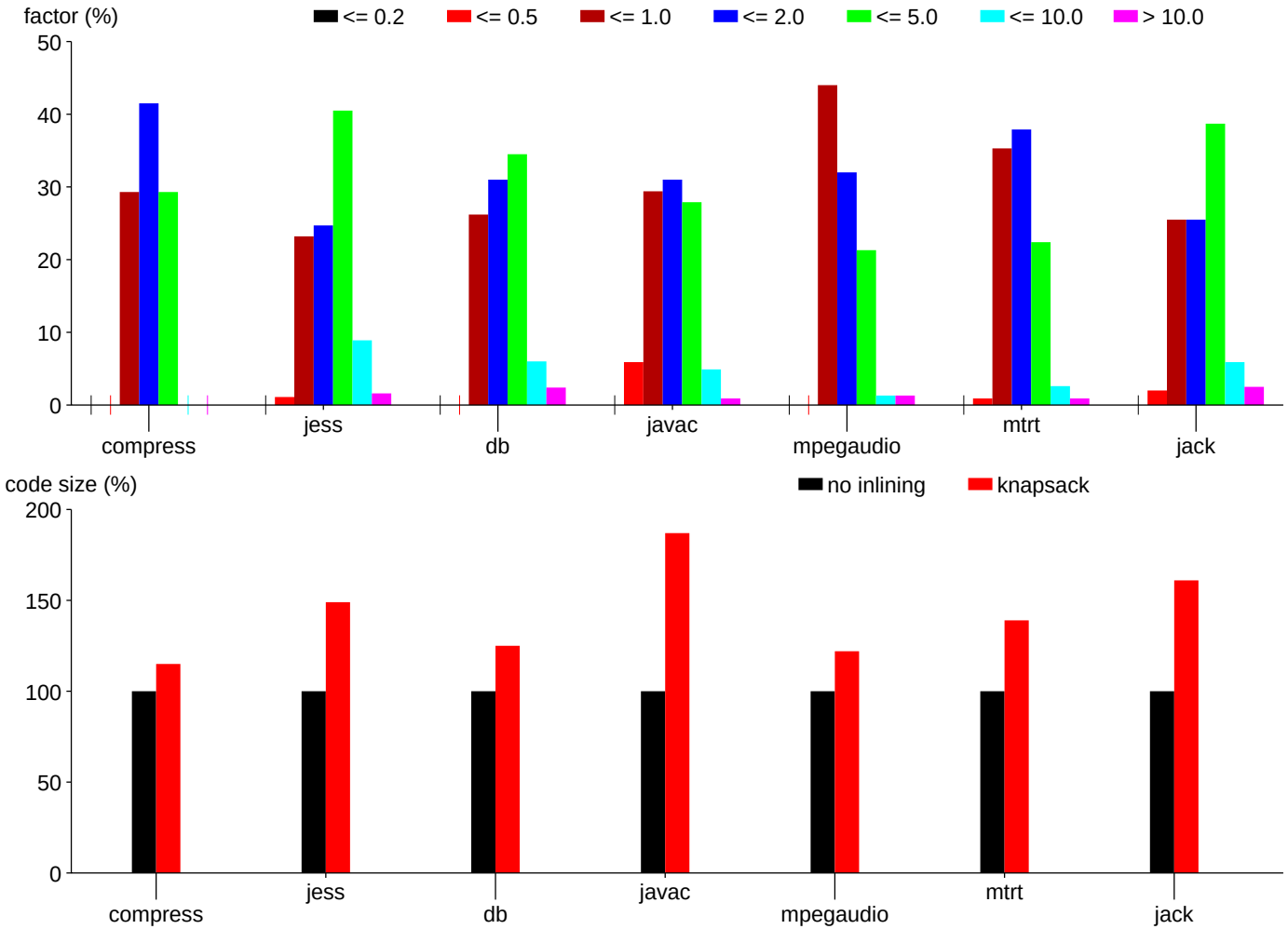
On-Stack Replacement

- execution state
  - snapshot of registers and machine stack
- source state
  - values of Java variables and Java operand stack
- replacement points
  - allocation of data
  - traps

Execution Times and Number of Executed Calls



Code Size Changes (Knapsack Heuristics)



Conclusion

- adaptive compilation framework for CACAO
- different inlining heuristics evaluated
- up to 99% of calls eliminated
- up to 18% speedup
- further improvements by linear scan register allocator expected
- www.cacaojvm.org