VM Performance Evaluation with Functional Models

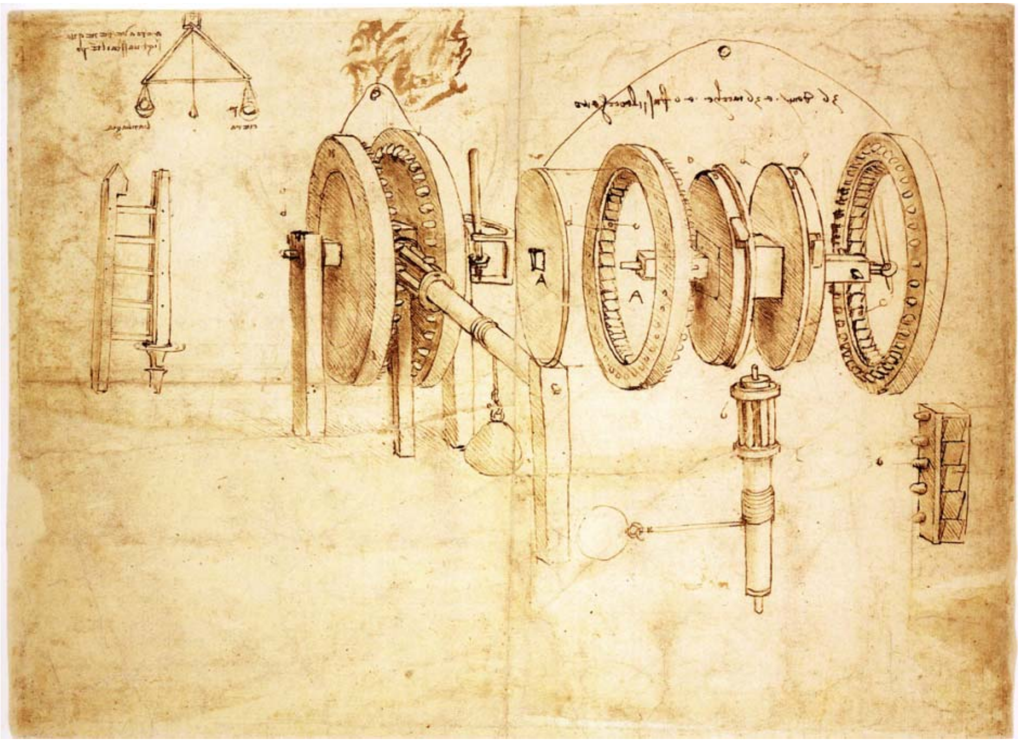
An Optimist's Outlook

TECHNISCHE UNIVERSITÄT DARMSTADT

Jan Sinschek

3rd International Workshop on Virtual Machines and Intermediate Languages
Oct 25th 2009

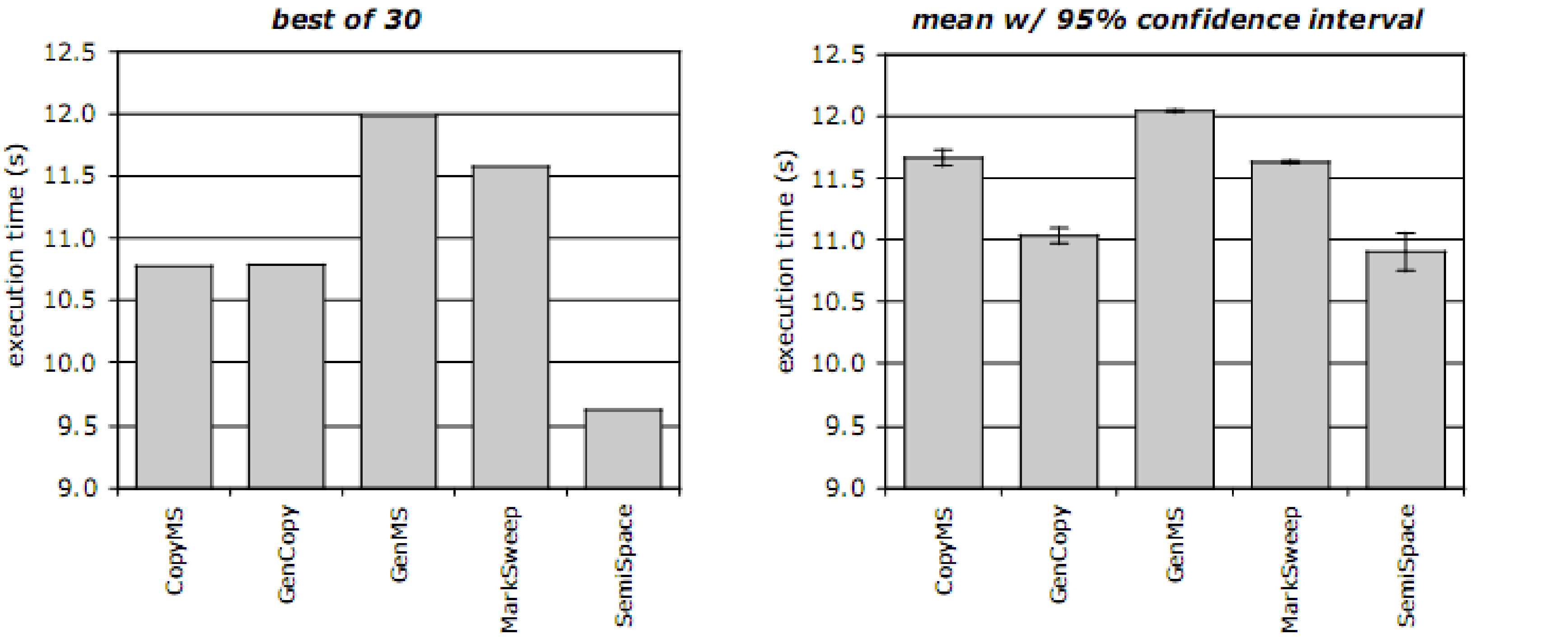
The Status of VM Performance Assessment



- Cascading effects
- Chaotic Behaviour

- Some nondeterminism is accepted
  - Noise
  - ...which is where bias hides

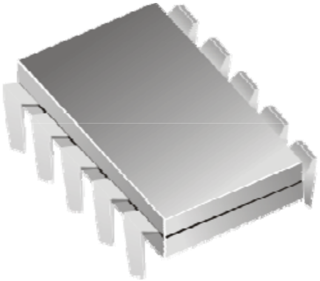
"Statistically Rigorous Java Performance Analysis"



“Wake up and smell the coffee”

- Advocates diverse Benchmarks

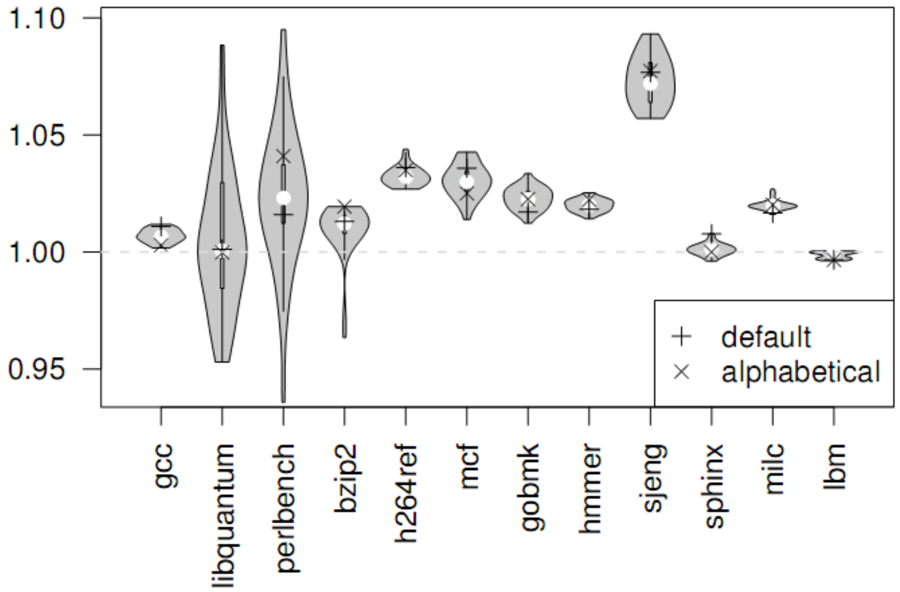
the DaCapo benchmark suite



- Proposes ways to control nondeterminism

- Demands meaningful baseline and publishing the implementation

„Producing Wrong Data Without Doing Anything Obviously Wrong“

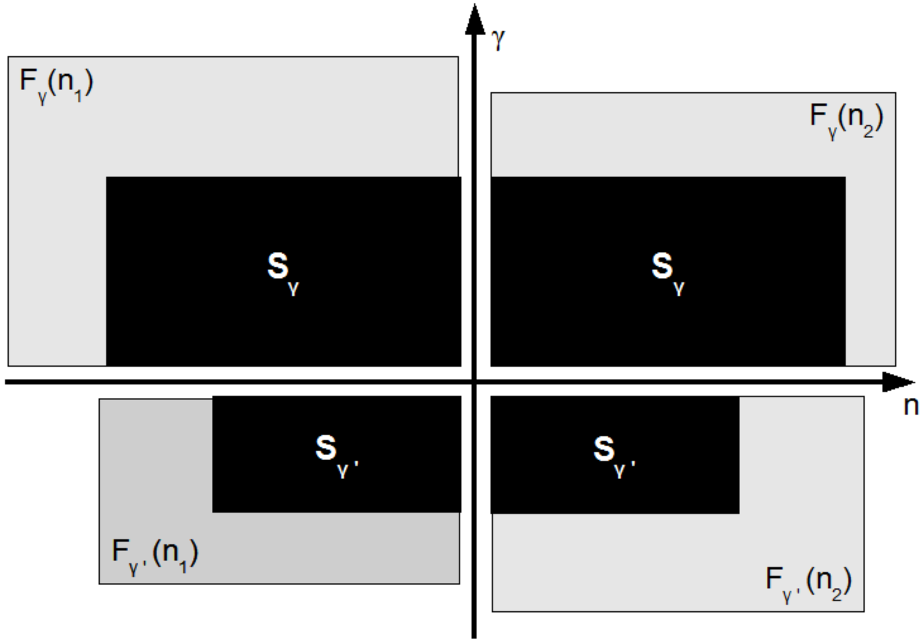


Suggestions for dealing with bias

- Setup randomization
- Causal analysis of an intervention

Functional Performance Models

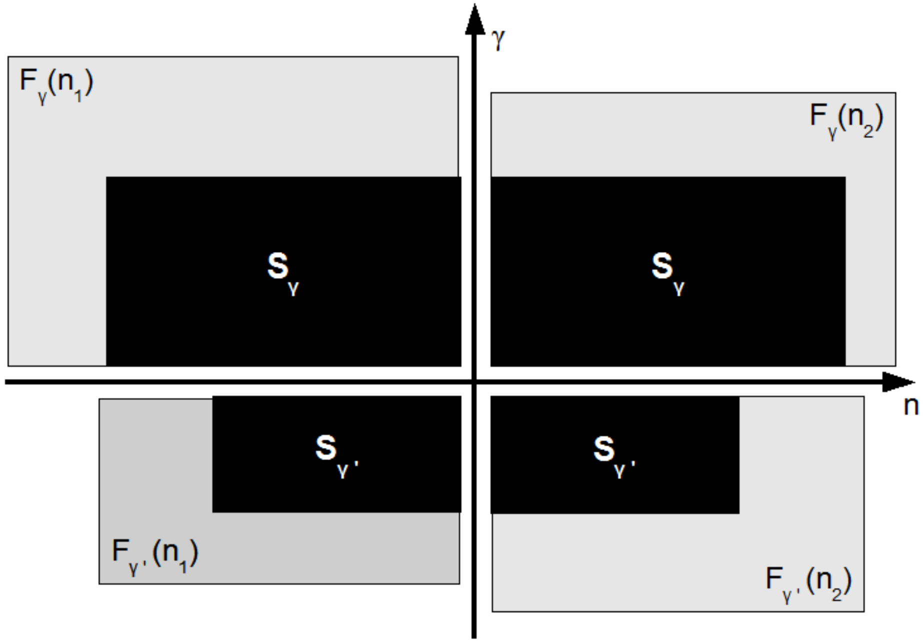
$$P_\gamma(n) = S_\gamma + F_\gamma(n)$$



Functional Performance Models

$$P_\gamma(n) = S_\gamma + F_\gamma(n)$$

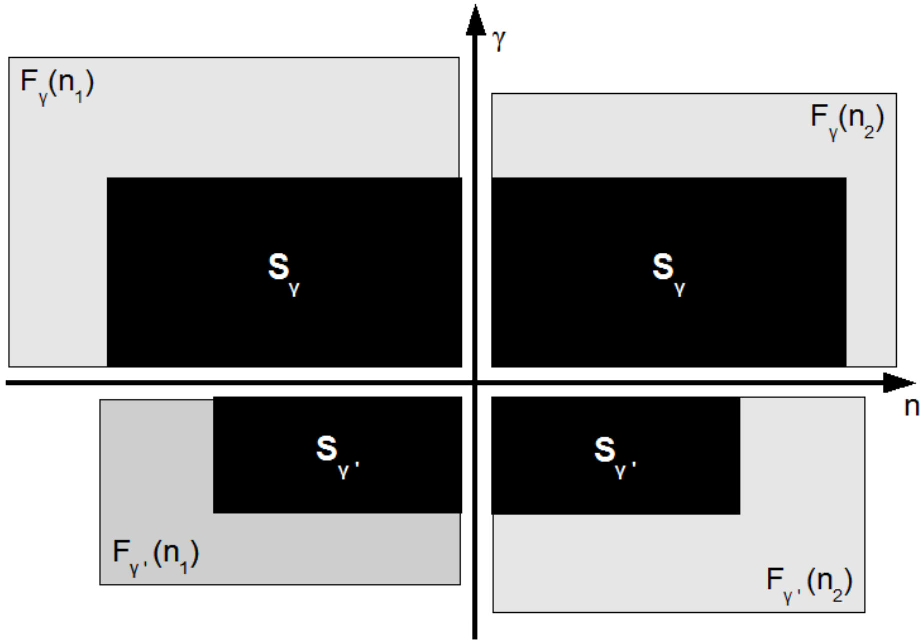
Mean Value



Functional Performance Models

$$P_\gamma(n) = S_\gamma + F_\gamma(n)$$

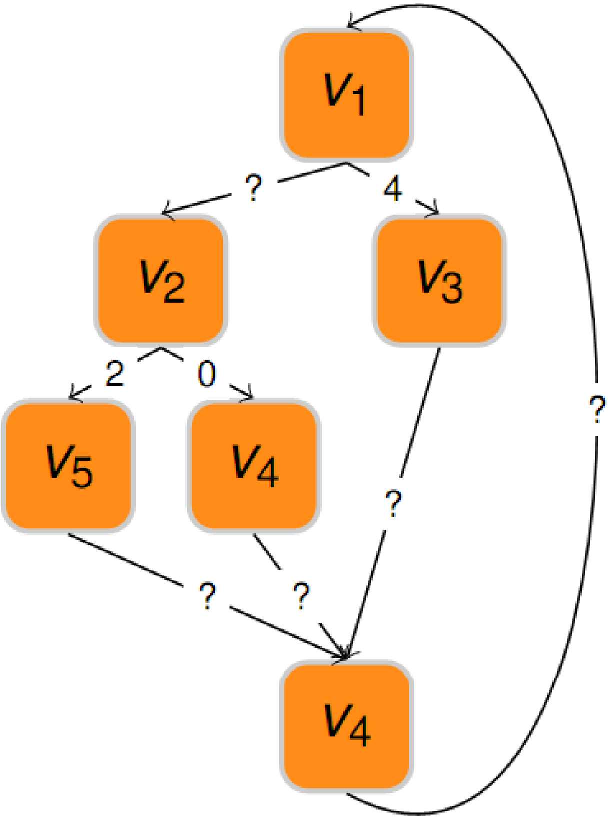
Confidence Interval



A Case Study Proposal

Scenario: Edge Profiling in Jikes VM

- Some counters are redundant
  - JIT compilation could lessen it
  - Values can be reconstructed afterwards
- This might we worthwhile
  - But it would cause overhead
  - We want to determine what strategy to use for overhead reduction

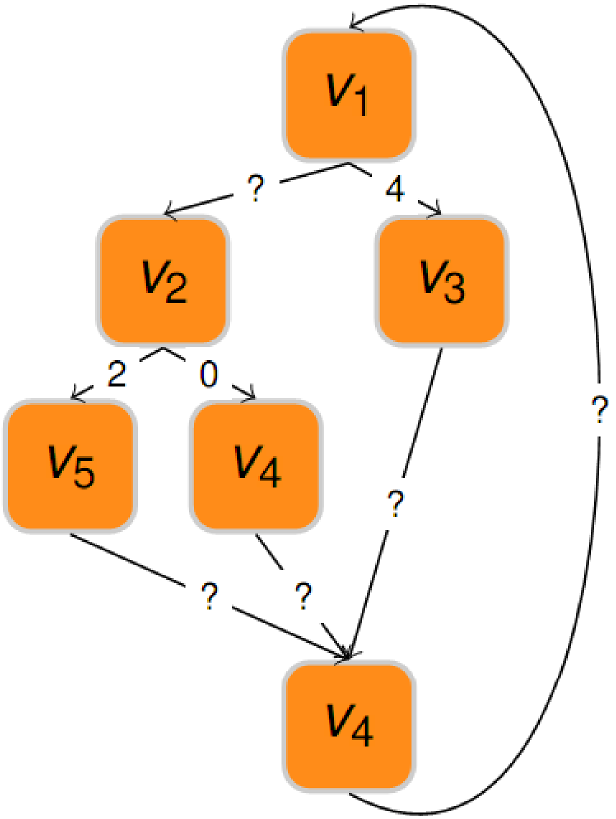


The Model for Counter Cost

- What to model
  - Counters necessary and safeable
  - Their respective performance contributions

- Specific Model:
  - Cost of counters in baseline code (γ)
  - Counter frequency (m_i)

$$P(n) = P(0) + \gamma \sum_{i \in n} placed?_i * m_i$$



A More Realistic Model

- Put counters into a broader model context
  - Based on recompilation plans, with r_0 opt-compiling nothing

$$P_{steady}(r,n) = P_{steady}(r_0,n_0) - \gamma \sum_{i \in n} m_i$$

  - The approach works as part of a rigorous recompilation setup

- Benefits to expect
  - Increased confidence in evaluation
  - Knowledge about the behaviour of profiling counters and the benchmarks
    - Helps to predict what optimization can pay off

What are Feasible Applications?

- Code Instrumentation
- Security Label Checking
- Execution Sampling
- GC Barrier Instrumentation

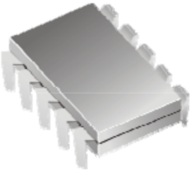
- Adaptive Compilation
- Pretenuring
- Thread Interactions
- Inlining

Summary

- Functional Models geared towards transferable research results

$$P(n) = P(0) + \gamma \sum_{i \in n} placed_i * m_i$$

- The discovery of bias still depends on the variety in the benchmarking environment



- Scope needs to be explored

- Applies to new functionality as well

?