The Case for Application-Specific Benchmarking

Margo Seltzer, David Krinsky,
Keith Smith, Xiaolan Zhang

Harvard University

March 1999

No General-Purpose Benchmarks

- **There is no such thing as a general-purpose benchmark.**
- **People care how systems will perform on their workload.**
  - Their application
  - Their access pattern presented to an application
- **Benchmarks that assume any particular workload are, by definition, not generally applicable.**

What is the Right Approach?

- **Application-specific benchmarking.**
- **Benchmark is a framework into which one applies a specific workload.**
- **The hBench Approach.**
  - Characterize both a system and an application
  - Carefully combine characterizations
  - Three methodologies

Vector-Based Methodology

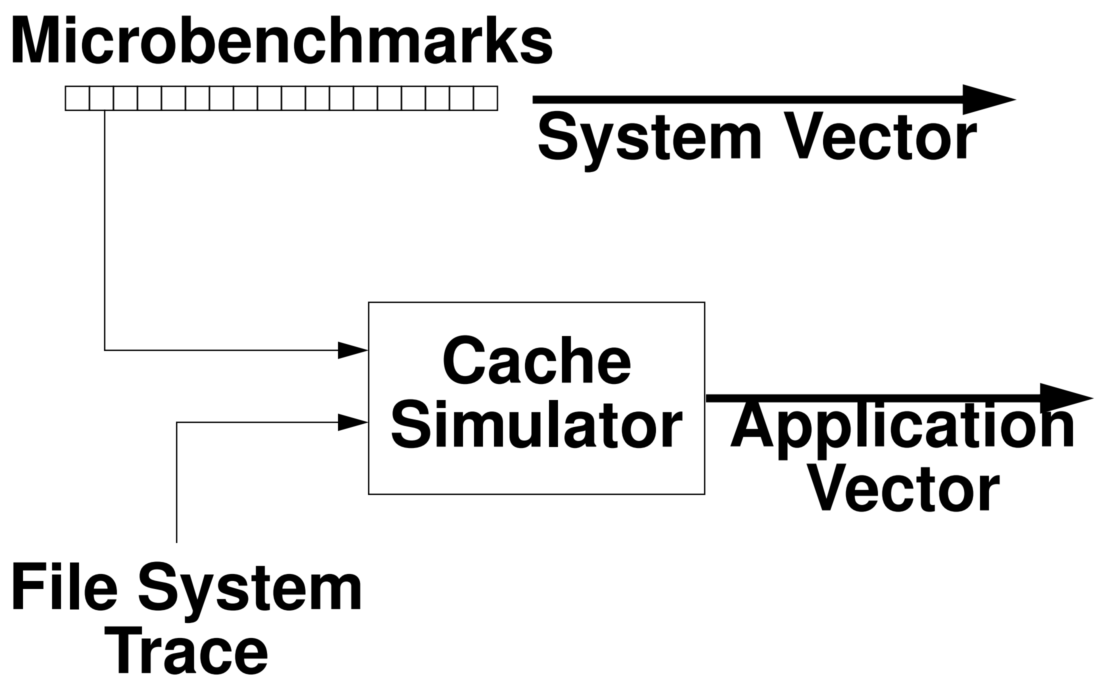
- **Use microbenchmarks of system primitives to form system vector.**
- **Use application profile to form application vector.**
- **Multiply. <u>(possibly transform)</u>**
- **hBench:OS**
  - System vector comprised of microbenchmark results
  - Application vector comprised of system call trace
- **hBench:Java <u>challenge=GC</u>**
  - System vector comprised of microbenchmark results
  - Application vector comprised of language operations

Trace-based Methodology

- **Characterize traces or logs.**
- **Use characterization to generate workload model.**
- **Stochastically generate workload based on model.**
- **hBench:Web, hBench:Proxy**
  - Characterize set of files accessed
  - Characterize user/client access patterns
  - Stochastically generate load representative of logs.
  - Tweak characterization parameters to answer “what if?” questions

Hybrid Methodology

- **hBench:FS**



Conclusions

- General-purpose benchmarks are **not** really general.
- <u>That's OK.</u> Users don't want general-purpose benchmarks.
- The goal of performance work is to make applications run faster.
- **Must not ignore the applications in measuring performance.**