Benchmarks Can Make Sense

Anton Burtsev, Eric Eide, Mike Hibler, John Regehr
Flux Research Group, School of Computing

emulab

THE UNIVERSITY OF UTAH

Motivation

Benchmarks are designed to compare systems
- Return a single number (e.g. throughput, average delay, completion time, CPU utilization, transactions per second)

No way to identify performance bottlenecks
- A simple configuration error can invalidate all results
- Network file system (NFS) benchmark below:
  - No disk write buffering (50%)
  - Synchronous NFS mount (60%)
  - Default NFS request buffer (30%)
  - No RAID buffering (50%)
  - Bandwidth delay product (20%)

No way to debug the benchmarked system

Goal

Benchmarks should make more sense
- Verify benchmark setup
- Aid performance analysis through "verbose" performance model

Idea

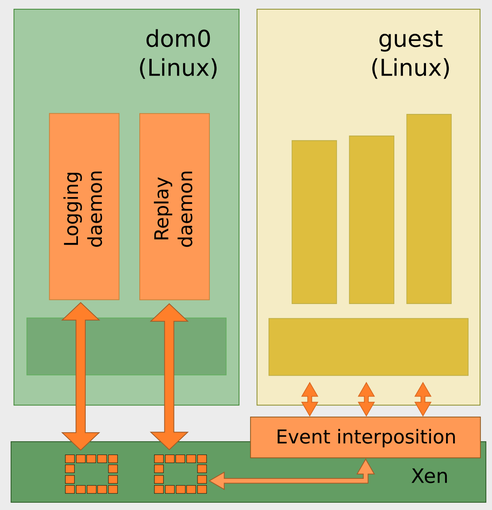
Full-system deterministic replay
- Lightweight non-intrusive way to record and replay execution

Replay is a mechanism allowing us to combine benchmarks and performance analysis
- Realistic execution
- Enables sophisticated computation-heavy analysis

Analysis runs off-line on a cloned copy of execution
- Global comprehension: no irreproducibility/observability problems
- No restrictions on complexity of analysis (no probe effect)
- Embarrasingly parallel

Implementation

Full-system replay:



Xen virtual machine monitor
- Full-featured virtualization platform
- Support for production workloads

We extend it with low-overhead recording facility
- Cooperative logging for network of machines
- Versioning storage for deterministic disk communication

Develop tools for execution comparison and automatic detection of non-determinism

Performance model:
Do performance measures stay sound during replay?

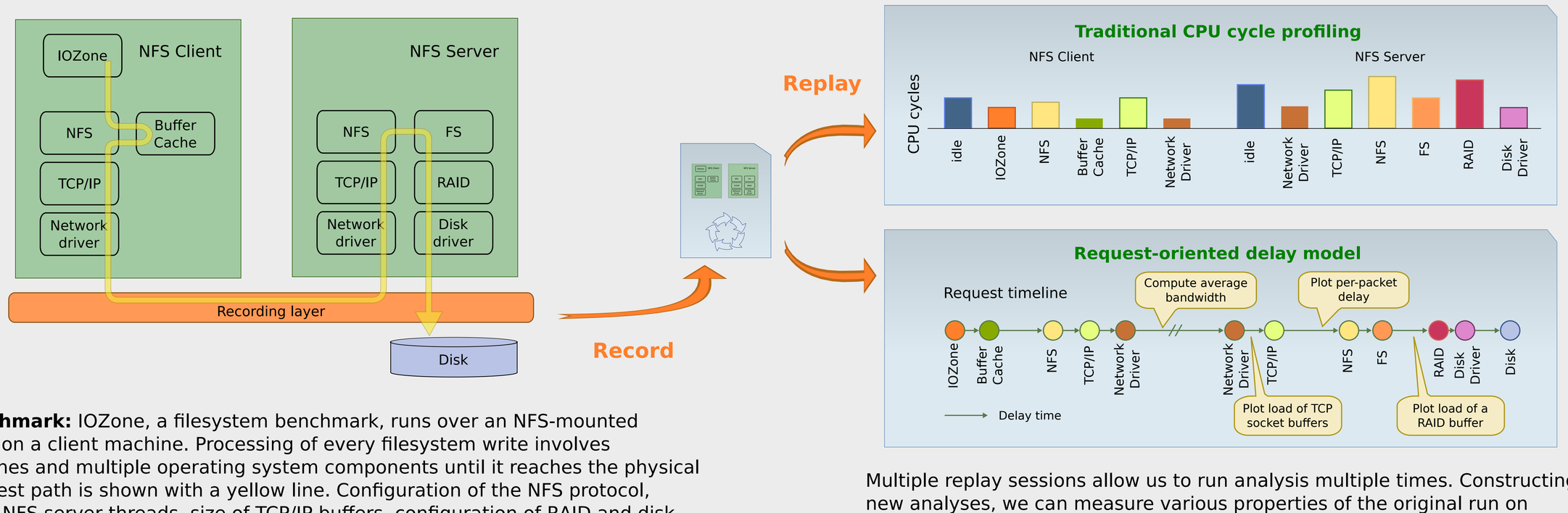
Replay
1) Set branch counters to overflow (cause exception)
2) Iterate in a single-step CPU mode to a target IP
3) Inject external event

Hardware model during replay
- Multiple exceptions (flushes CPU pipeline)
- Keeps caches almost warm

Analyze ILP during original and replay run

Analysis interface:
Additional research needed to understand which information can actually help performance analysis
- Performance metrics
- Functional properties of the system

Big Picture



NFS benchmark: IOZone, a filesystem benchmark, runs over an NFS-mounted filesystem on a client machine. Processing of every filesystem write involves two machines and multiple operating system components until it reaches the physical disk. Request path is shown with a yellow line. Configuration of the NFS protocol, number of NFS server threads, size of TCP/IP buffers, configuration of RAID and disk write buffering, and many ofther factors affect performance of the system.

Multiple replay sessions allow us to run analysis multiple times. Constructing new analyses, we can measure various properties of the original run on demand.

Challenges

Efficient full-system replay

Faithful performance model
- Replay may interfere with the execution of a system

Debugging and analysis interface
- Non-intrusive probes
- DTrace-like language interface to collect information about execution

This material is based upon work supported by the National Science Foundation under Grant No. 0524096. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation

Status

Implemented a basic deterministic logging and replay infrastructure
- Can replay beginning of the Linux boot (650K instructions)
- Replay mechanisms are designed to treat the state of a guest system as a set of memory pages
- Right choice to support heterogenity of replay in the future

Support most non-deterministic events
- Lack support for logging device driver communication