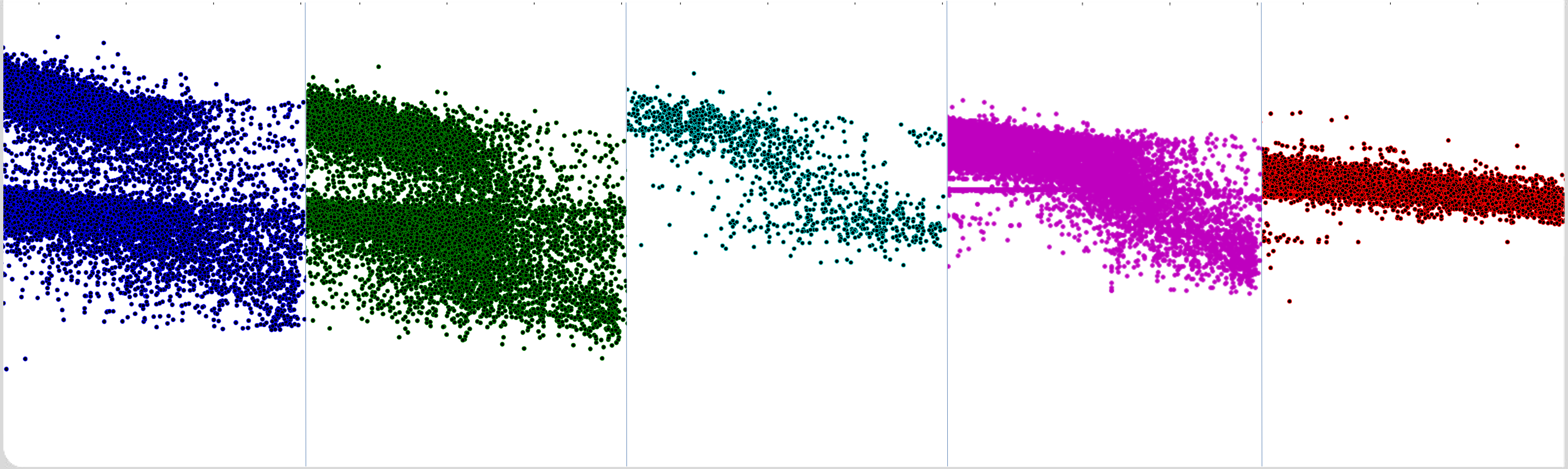
Karlsruhe Institute of Technology

Fast Benchmark Candidates
HEPiX Benchmarking Working Group 2016-06-17

Manfred Alef

STEINBUCH CENTRE FOR COMPUTING (SCC)



Use Cases and Requirements

Use Cases and Requirements

- Use cases:
  - → Estimate the performance of the provided job slots (or WM) in case of anonymous hardware (e.g. in commercial clouds)
    - Job matching / masonry
      (e.g. “can a pilot run another payload with the resources left?”)
    - Accounting if HS06 score not available
    - ...

Use Cases and Requirements

- Requirements:
  - → As fast as possible
    - ~1 min runtime?
  - → As exact as possible
    - Low spread of results
    - Be aware: the shorter the runtime of a fast benchmark the higher the probable inaccuracies
      - What are the neighbors doing?
      - Regular reassessments to iterate
  - → Freeware (e.g. GPL), no licensing costs

Benchmark Candidates

- 5 fast benchmarks related to WLCG use cases

Benchmark Candidates

- Analysis:
  - ~20,000 single-core batch jobs at GridKa
    - System load at job start time
    - HS06 score of the provided job slot (from MJF)
    - Results of 4 fast benchmark candidates
      (+ ~2,500 jobs running candidate #5)
  - Time period: 2 weeks
    - GridKa farm utilization level varying between 50 and 100%
  - How to read the diagrams:
    - X-axis: normalized system load (1-min load / number of slots)
    - Y-axis: normalized benchmark score (per HS06)
      (flat line ⇔ perfect scaling)

Benchmark Candidates

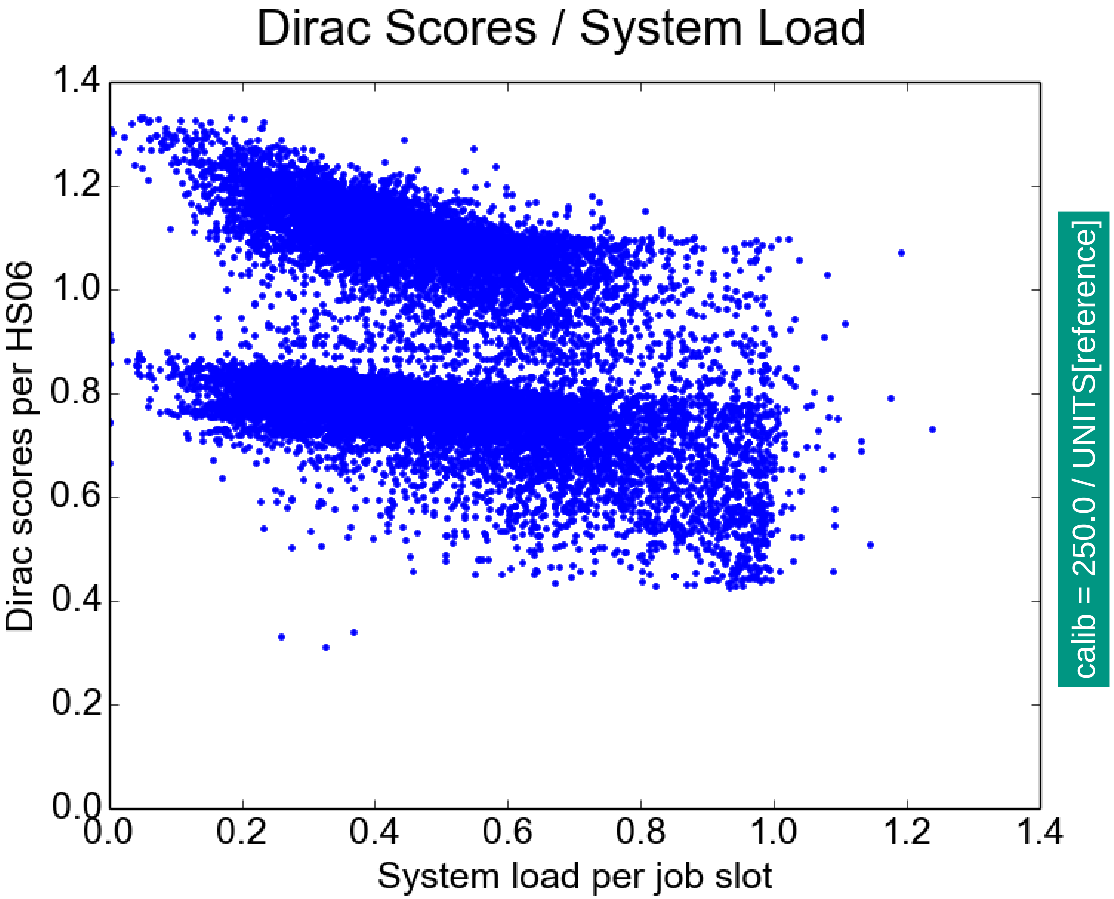
- Dirac fast benchmark („fastBmk”, „LHCb fast benchmark”, ...)
  - Used by:
    - LHCb
    - Belle II, ...
  - Python script
    - Compiler flags (Python package) are hidden to the user
  - License (Dirac framework):
    - Mozilla Public License 1.1 (MPL)
  - Be aware of (at least) 2 versions of this benchmark:
    - Single-core version (calib = 250.0 / UNITS[reference])
    - Multi-core version (calib = 360.0 / UNITS[reference])
      e.g. used in CERN Cloud Benchmark Suite

Benchmark Candidates

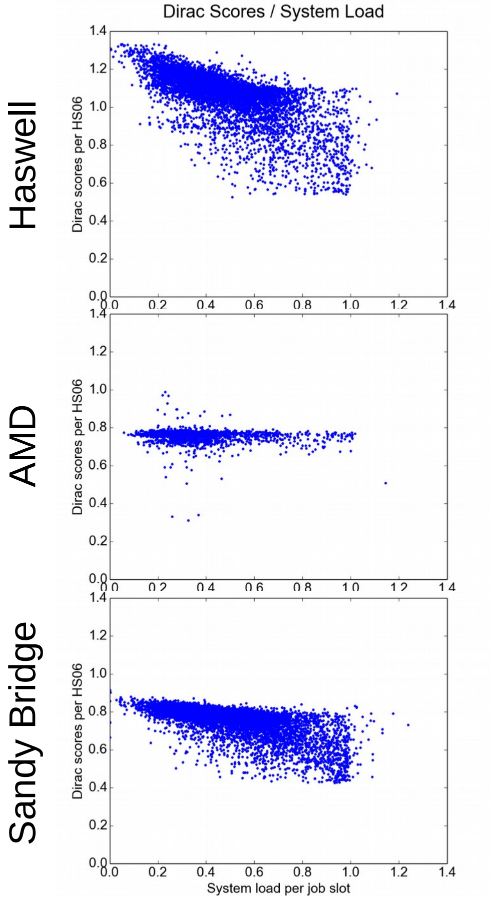
- Dirac fast benchmark („fastBmk”, „LHCb benchmark”, ...)
  - → Fast
    - Runtime: 0:30 ... 2:30 minutes
  - → Correlations with job performance and with HS06:
    - Well scaling with LHCb workload [1,2]
    - Mismatch with HS06 of up to ~40% [3,4]

Benchmark Candidates

- Dirac fast benchmark („fastBmk”, „LHCb benchmark”, ...)



calib = 250.0 / UNITS[reference]



Benchmark Candidates

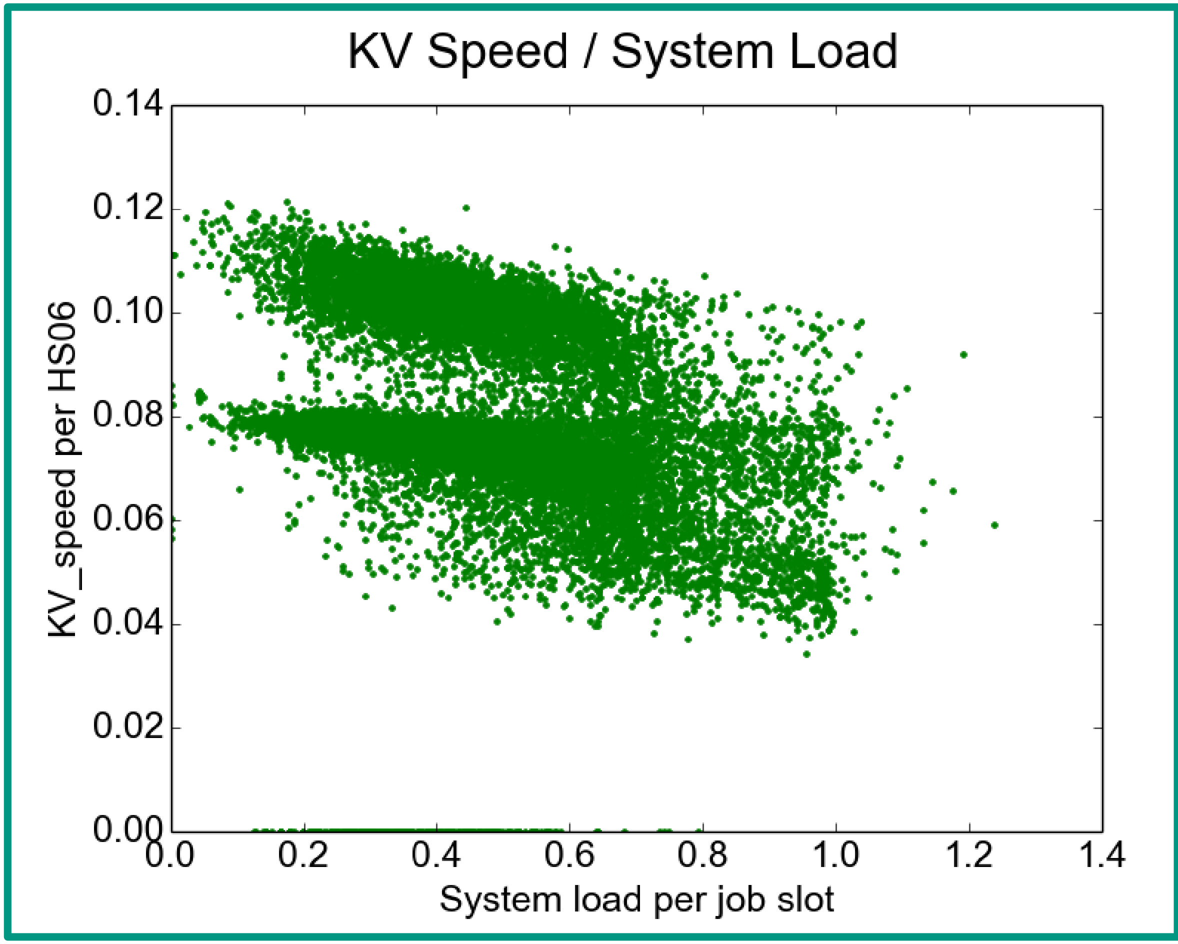
- Geant4 simulation (100 Single Muon Events) via Atlas KV toolkit
  - Used by:
    - Atlas
  - Framework for benchmarking, various benchmarks can plug-in
    - Wrapper script "kv-script.sh" downloaded from [5]
      - Somewhat modified to run in batch environment instead of VM
    - Default workload (Geant4: 100 Single Muon Events)
    - Compiler flags are hidden to the user (but not frozen)
  - License:
    - ???

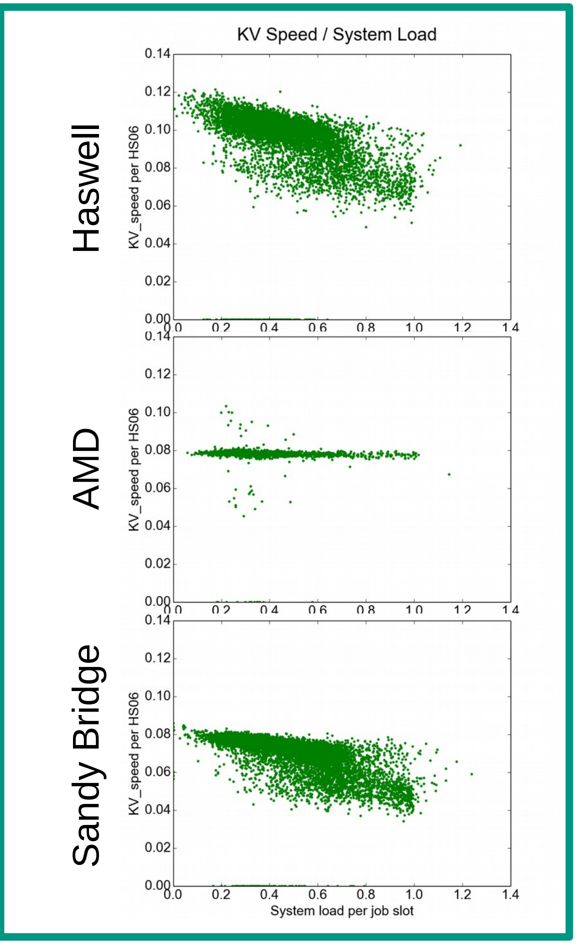
Benchmark Candidates

- Geant4 simulation (100 Single Muon Events) via Atlas KV toolkit
  - Slow
    - Runtime: around 4 minutes
  - Correlations with job performance and with HS06:
    - Scaling well with Atlas simulation jobs [6]
    - Probable mismatch with HS06 of up to 40%?
  - Units
    - KV score is in units of seconds/event, in this talk: converting to inverse results ("KV speed")

Benchmark Candidates

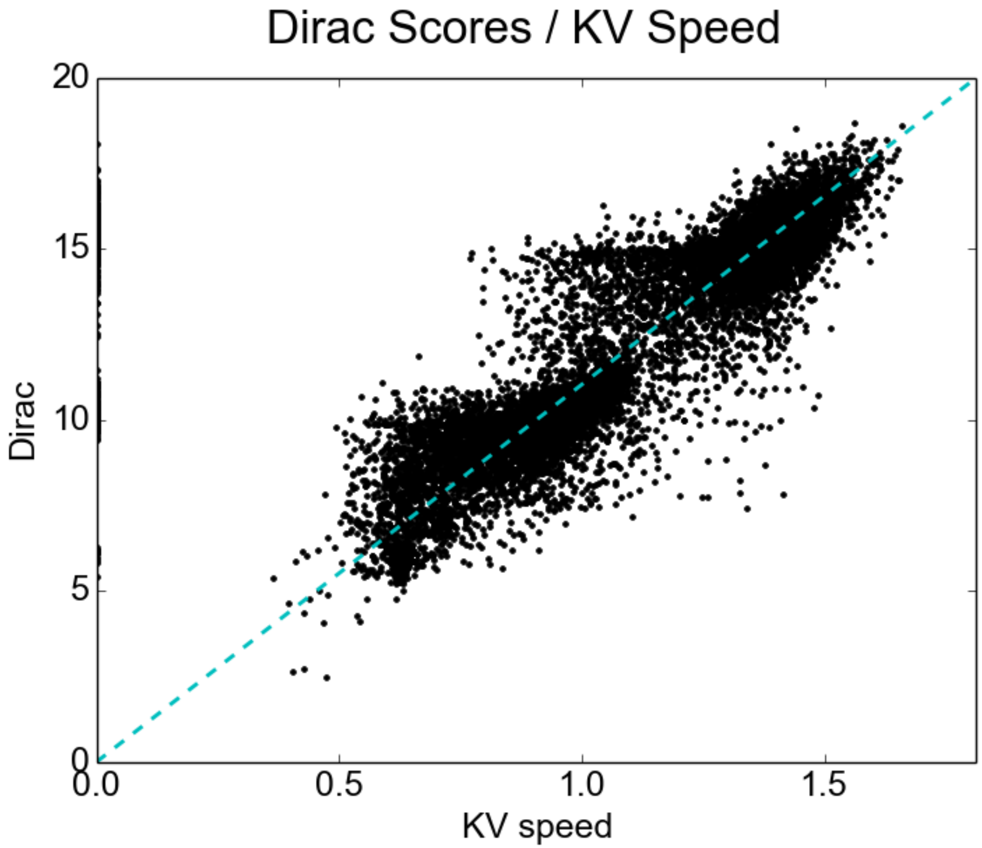
- Geant4 simulation (100 Single Muon Events) via Atlas KV toolkit





Benchmark Candidates

- Good correlation between Dirac fast benchmark and KV speed (see also talk by Domenico Giordano [9])



Benchmark Candidates

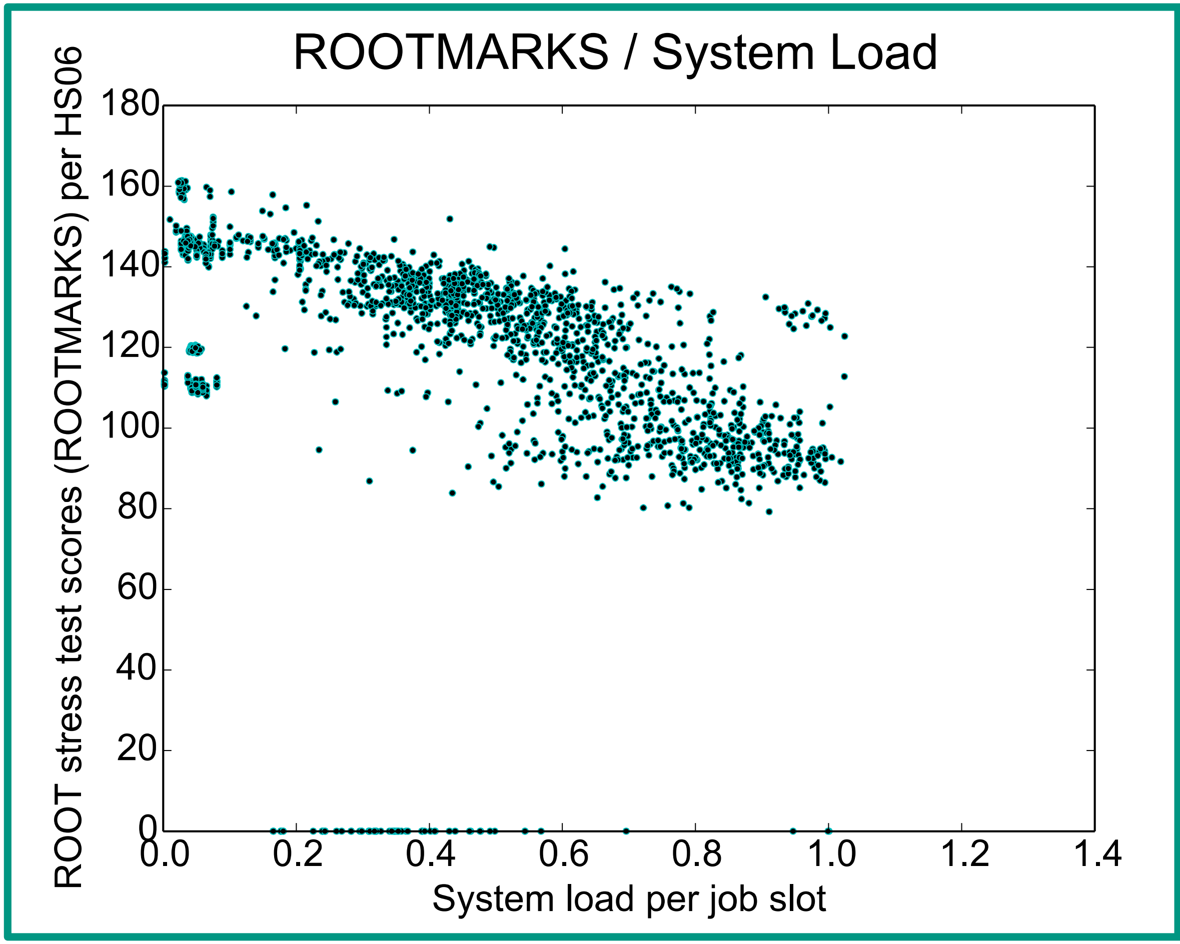
- ROOT stress test
  - Used by:
    - Alice
  - By default using pre-compiled binary
    - Tricky to build from source
    - Very low (no) dependency on specific compiler flags found
  - License:
    - GNU Lesser General Public License 2.1

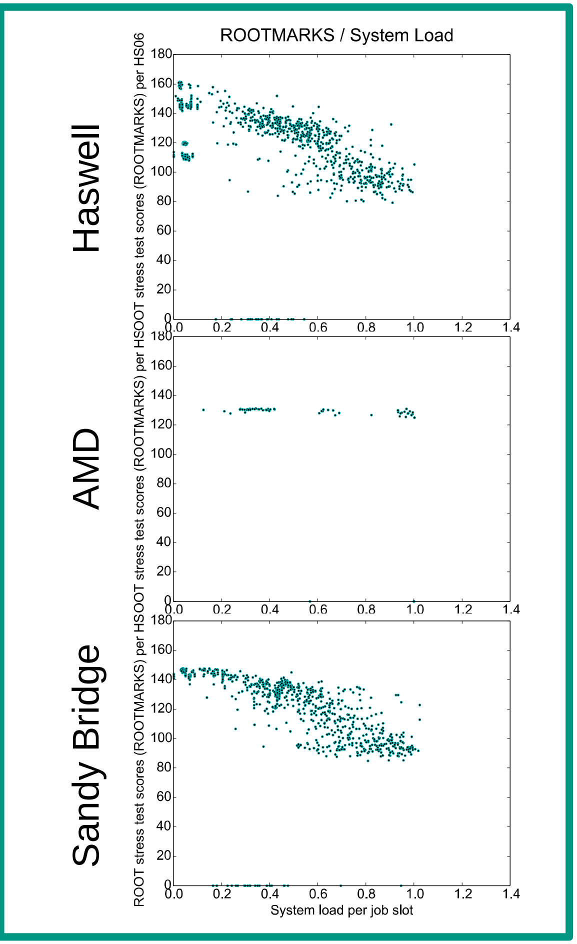
Benchmark Candidates

- ROOT stress test
  - → Very fast
    - Runtime: 0:30 minutes (1,000 iterations)
    - Suggestion: run 2,000 ... 2,500 iterations (default: 1,000)
  - → Correlations with job performance and with HS06:
    - Scaling linearly with Alice simulation efficiency [7]
    - Better scaling with HS06 than Dirac and KV

Benchmark Candidates

- ROOT stress test





Benchmark Candidates

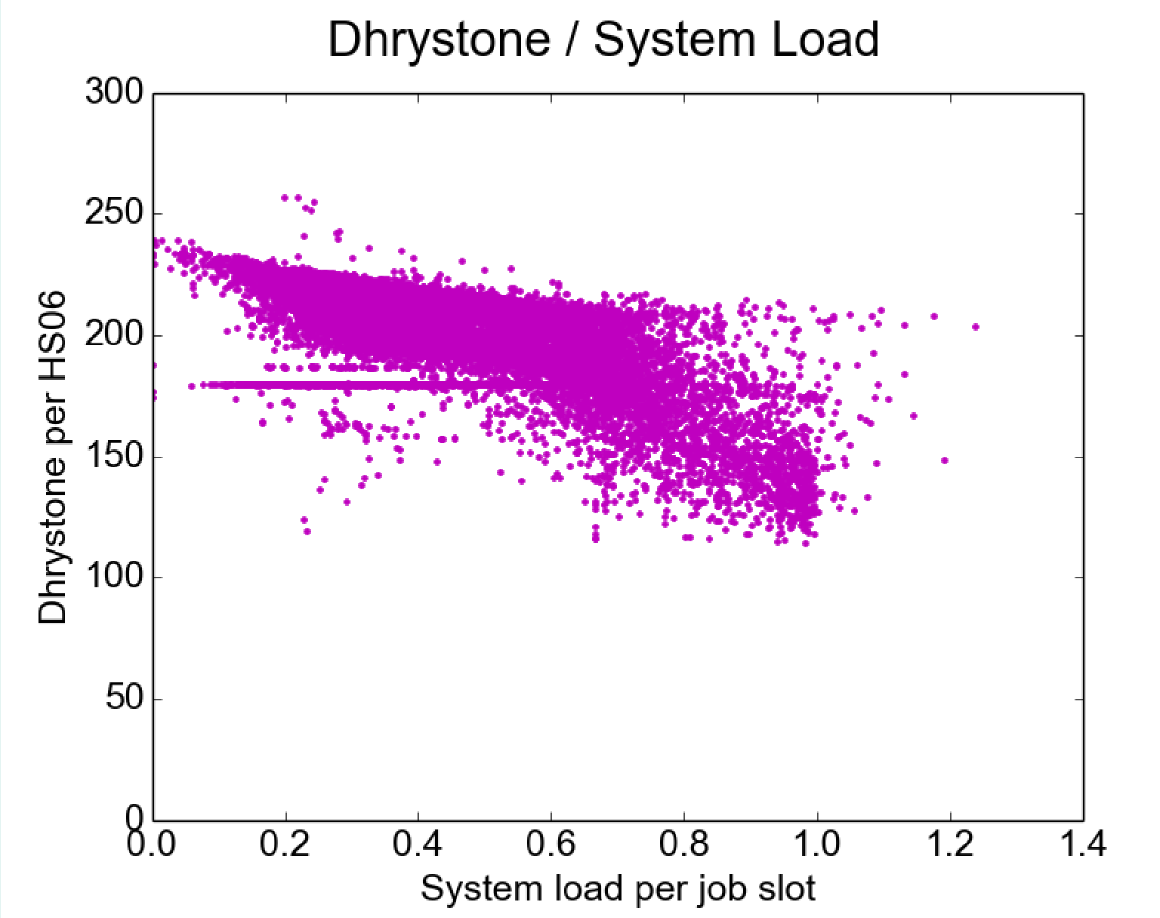
- Dhrystone, Whetstone
  - Used by:
    - Alice, Atlas, ...
    - HTCondor (condor_kflops, condor_mips), Boinc, ...
  - I have used the unmodified Makefile (from UnixBench package [8])
  - License:
    - GNU General Public License 2
  - Very fast
    - Runtime: 2:00 ... 3.00 minutes per benchmark
      for 10 iterations (default in UnixBench package)
      - Can run less than 10 iterations
  - Very small memory footprint (less than L3 cache size)

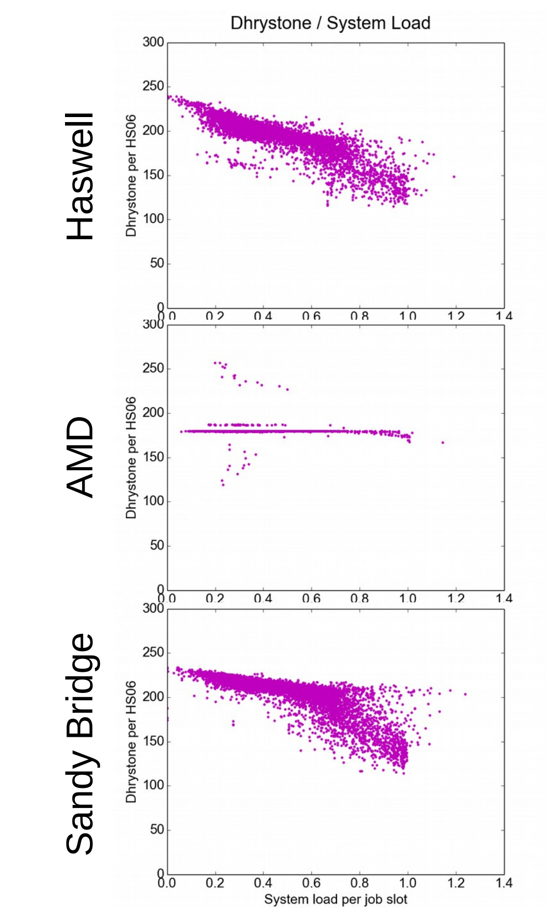
Benchmark Candidates

- Dhrystone
  - ➔ Benchmark results very similar to ROOTMARKS

Benchmark Candidates

- Dhrystone



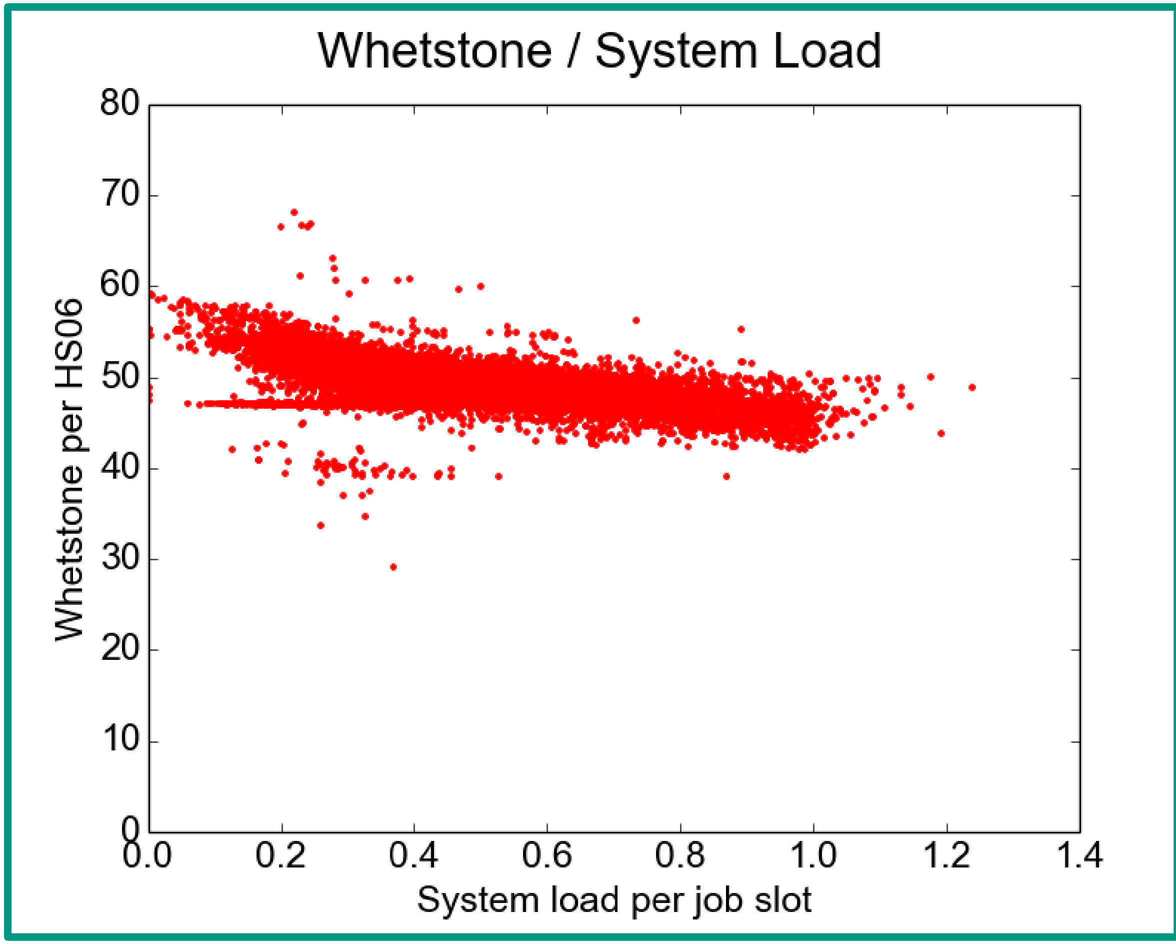


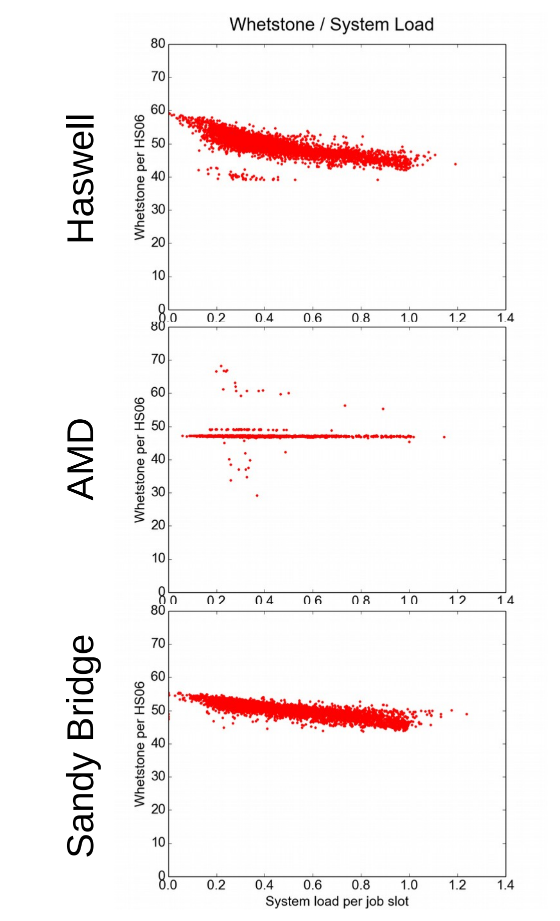
Benchmark Candidates

- Whetstone
  - → Correlations with job performance and with HS06:
    - Scaling very well with HS06 [3,4]
    - Probable mismatch with WLCG application performance
  - → Not only the dispersion of the HS06-normalized results, but also their dependency on the system load, are very low
    - Most suitable candidate to estimate HS06 in cloud VM’s?

Benchmark Candidates

- Whetstone





Suggestions

Suggestions

- Benchmark names
  - ➔ Can we agree on standardized benchmark names?
    - Avoid experiment specific names
    - Proposal:
      - Dirac Fast Benchmark
      - KV Single Muon Simulation
      - ROOT stress test
      - Dhrystone, Whetstone

Suggestions

- Benchmark versions
  - ➔ Can we agree on frozen benchmark versions?
    - Compiler flags
      - Dirac: run with default Python
      - KV: need more transparent compiler version and flags
      - ROOT: very tricky to compile;
use binary which is used at GridKa?
      - Dhrystone, Whetstone: use default Makefile

Suggestions

- Calibration and units
  - → Can we agree on calibration factors?
    - Dirac: 250 or 360?
  - → Benchmarks measure the speed of computers:
    the higher the benchmark score the faster the system
    - KV scores are in units of events/second not seconds/event
      - Compute the inverse ('KV speed')

Suggestions

- Benchmark harness
  - ➔ CERN Cloud Benchmark Suite (see talk by Domenico Giordano [9])
    - Run several types of benchmarks
    - Collect results at single place for final analysis
    - Definitely the right direction ...
    - ... but installation requires root permission
      - E.g. to run 'chmod 666 /dev/fuse' (which is already the default mode when CVMFS is available on the host)
      - Currently no public repository because of license issue
      - Not recommended to run from grid pilot jobs
    - Current release provides only 3 of the proposed benchmarks: Dirac (multiprocessor version, calib=360), KV, and Whetstone

Suggestions

- Benchmark harness
  - Direct use of scripts/binaries in batch jobs
    - Dirac: Python script(s), no problem
    - KV: ?
    - ROOT: binary used at GridKa
    - Dhrystone, Whetstone:
      - Download:
        git clone https://github.com/kdlucas/byte-unixbench.git
      - Run script to start benchmarks
        - Increase maxCopies (default = 16) to run full load
    - Upload required files to twiki

References

[1] https://indico.cern.ch/event/319751/session/0/contribution/6/attachments/1153280/1656518/150909-MJFandBenchmarking-LHCb.pdf

[2] https://indico.cern.ch/event/319754/session/0/contribution/8/attachments/1202029/1749779/151209-MJFUpdate-LHCb.pdf

[3] https://indico.cern.ch/event/319754/session/0/contribution/9/attachments/1202373/1750492/Results_of_HS06_Scaling_Studies_at_KIT_2015-12-09.pdf

{4] https://indico.cern.ch/event/433164/session/2/contribution/9/attachments/1220374/1783838/Results_of_HS06_Scaling_Studies_at_GridKa_2016-02-01.pdf

[5] https://sdccloud.web.cern.ch/sdccloud/KV/DO-29401/index.html

[6] https://indico.cern.ch/event/319751/session/0/contribution/8/attachments/1151865/1653919/gdb-20150910.pdf

[7] https://indico.cern.ch/event/319751/session/0/contribution/9/attachments/1151876/1654130/ALICE_benchmarks.pdf

[8] https://github.com/cloudharmony/unixbench

[9] https://indico.cern.ch/event/535458/contributions/2176092/attachments/1284582/1909948/CERNCloudBenchmarkSuite_HEPiXBmkWG_giordano.pdf