Benchmarking Working Group Status update

D. Giordano (CERN)

hepix-cpu-benchmark@hepix.org
https://twiki.cern.ch/twiki/bin/view/HEPIX/CpuBenchmark

WLCG Workshop 2017
21June 2017

Plan of the Session

Summary of the HEPiX Benchmarking WG activity

- Mandate, recent activities, foreseen plans

Discussion, animated by a panel

- Alessandra Forti (Atlas experiment and site repres.)
- Andrew McNab (LHCb and site repres.)
- Manfred Alef (WG chair and site repres.)
- Pepe Flix (CMS experiment and site repres.)
- Latchezar Betev & Costin Grigoras (ALICE experiment repres.)

Objective

- Discuss the discrepancy among HS06 and HEP workloads
  - Among the studied benchmarks, is there a valid substitute of HS06?
- Clarify the opinion of the Experiments about current fast benchmarks

Performance Measurement

"Performance is a key criterion in the design, procurement, and use of computer systems […] to get the highest performance for a given cost."

"The types of applications of computers are so numerous that it is not possible to have a standard measure of performance […] for all cases."

"The first step in performance evaluation is to select the right measures of performance, the right measurement environments, and the right techniques."

"The process of performance comparison for two or more systems by measurements is called benchmarking, and the workloads used in the measurements are called benchmarks."

Performance Measurement

“Performance is a key criterion in the design, procurement, and use of computer systems […] to get the highest performance for a given cost.”

“The types of applications of computers are so numerous that it is not possible to have a standard measure of performance […] for all cases.”

“The first step in performance evaluation is to select the right measures of performance, the right measurement environments, and the right techniques.”

“The process of performance comparison for two or more systems by measurements is called benchmarking, and the workloads used in the measurements are called benchmarks.”

- From “*Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling*”
  - by **Raj Jain** , Wiley Computer Publishing, John Wiley & Sons, Inc
  - 1992 Computer Press Award Winner

Mandate of the Working Group

Investigate **scaling issues** between **HS06** and CPU intensive HEP workloads (i.e. EvtGen, Simulation)

- HS06 is strictly connected to accounting and pledges of compute resources

Study the **next generation of long-running benchmark**

- successor of HS06

Evaluate **fast benchmarks**

- identify their properties; provide recommendations to the community

Main Subject of the Last 4 Months

Scenarios for fast benchmarks adoption

- 😉 Forecasting job slot duration or checking performance of VM in cloud environments (large consensus)
- 😕 Replacement of HS06 for site pledge and procurement
  - Large divergence of opinions
  - Limited instruction mix ➔ exposed to microarchitecture changes/optimization
  - Risk of missing all implications of that choice on the medium-long term

– Triggered a major effort to study in detail the fast benchmarks

Breaking News: SPEC CPU2017 is Available!

SPEC releases major new CPU benchmark suite

The SPEC CPU2017 benchmark suite features updated and improved workloads, use of OpenMP to accommodate more cores and threads, and optional metric for measuring power consumption

Gainesville, Va., June 20, 2017 -- The Standard Performance Evaluation Corp. (SPEC) today released the SPEC CPU2017 benchmark suite, an all-new version of the non-profit group's software for evaluating compute-intensive performance across a wide range of hardware systems.

The SPEC CPU2017 benchmark suite is the first major update of the worldwide standard CPU performance evaluation software in more than 10 years. The new suite includes updated and improved workloads with increased size and complexity, the use of OpenMP to allow performance measurement for parallelized systems with multiple cores and threads, and an optional metric for measuring power consumption.

Current SPEC CPU subcommittee members include AMD, ARM, Dell, Fujitsu, HPE, IBM, Inspur, Intel, Nvidia and Oracle.

Requirements for HEP Benchmark(s)

Scale, within a given accuracy, with a representative WLCG job mix

Target domains adopted (or to be adopted) in WLCG

- Architectures (x86 Vs ARM, GPU)
- OS (SLC6 -> CentOS7)
- Infrastructures (Grid -> Cloud, HPC)

… Scale, within a given accuracy, with a representative WLCG job mix

'80-'90 CERN Unit

GG CERN-CN 92-13

CERN - Computing and Networks Division
CN/92/13
December 1992

Benchmarking Computers for HEP

Eric McIntosh

CERN, Geneva, Switzerland

Abstract

This report summarises the results of the CERN benchmark tests carried out on a variety of Mainfames and Workstations during the last fifteen years. The tests are a suite of FORTRAN programs used to determine the CPU power of a computer system for running High Energy Physics applications. They are essentially scalar due to the well known difficulties in vectorising this type of application, but a matrix inversion in

convenience and continuity I decided to keep the CPU metric, but reduce the number of production codes. I was able to make GABI (now CRN5), JAN (CRN12), and FOWL (CRN3), reasonably easy to port and I added another modern event generator LUND (CRN4) to give a 50/50 distribution between event generation and reconstruction as that was the workload distribution at the time. I supplemented these codes with several "kernel" type applications to at least get a feel for compilation times (CRN4C), vectorisation

- https://cds.cern.ch/record/245028/files/CM-P00065729.pdf

The current WLCG job mix

Feedback from Exp. Repres.

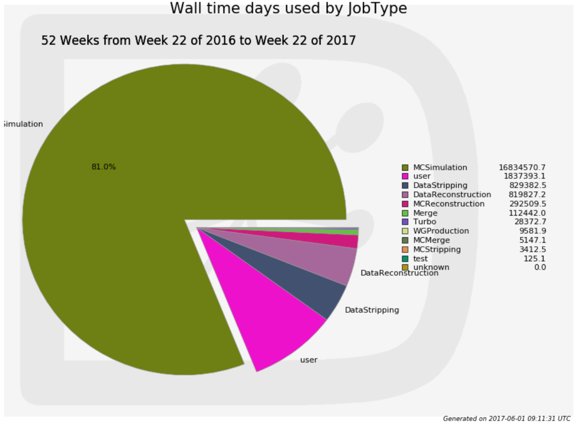
Simulation (Evt Gen + Geant) still dominates,

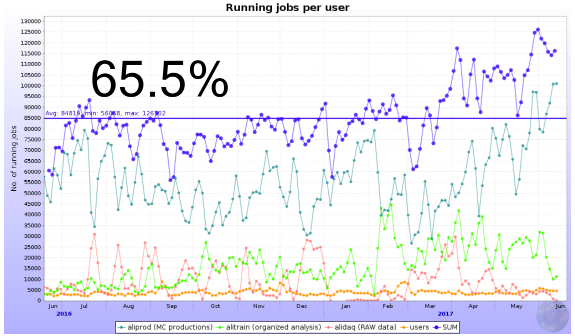
- LHCb (~80%),ALICE (~65%), ATLAS (~52%), CMS (27%)

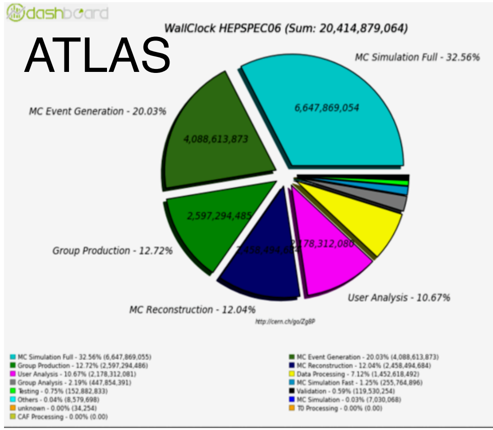
I/O intensive workloads are relevant too

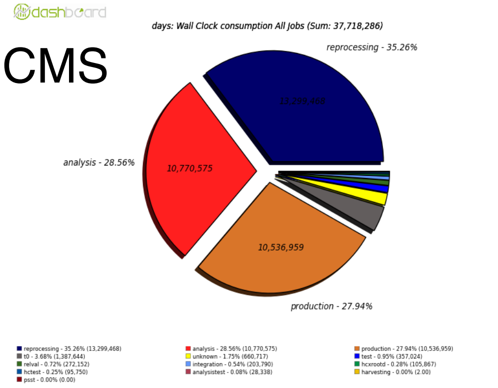
- To be included in the performance measurement

Currently most of the studies are still focusing on CPU-bound workloads









HEP-SPEC06 (HS06): A brief reminder

HS06 benchmarks

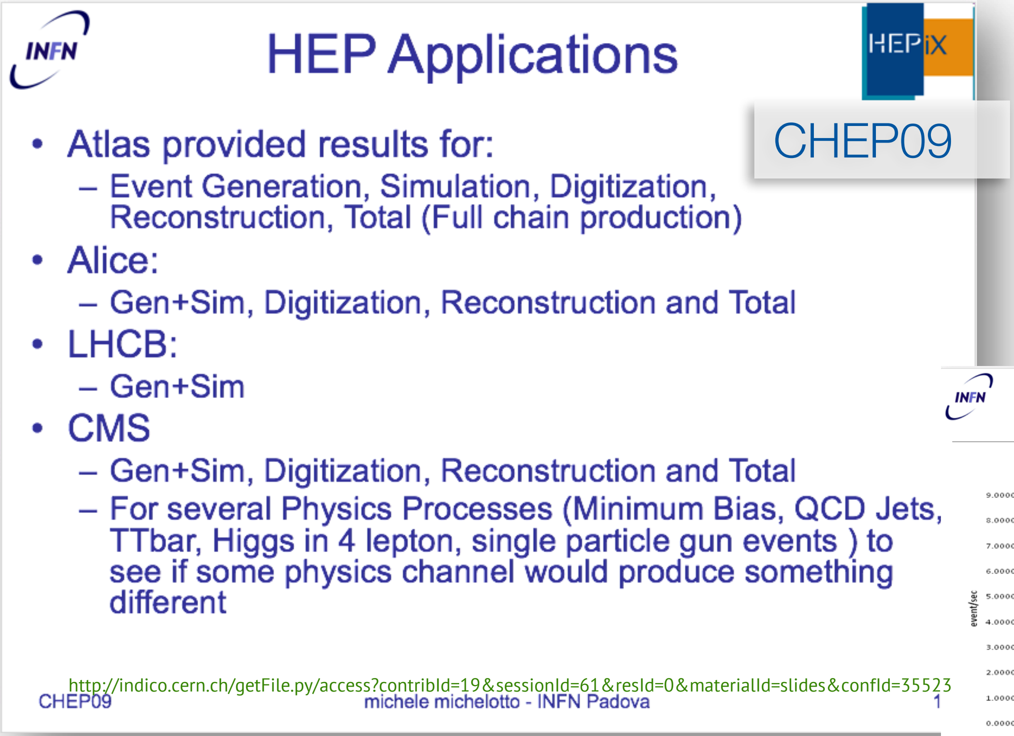
The WLCG CPU benchmarking group

- selected
  - SPEC all_cpp benchmarks
- requires to
  - Run the benchmark in the same OS which is provided by the site
  - Compiler flags

    -O2 -pthread -fPIC **-m32**

Bmk	Int vs Float	Description
444.namd	CF	92224 atom simulation of apolipoprotein A-I
447.dealII	CF	Numerical Solution of Partial Differential Equations using the Adaptive Finite Element Method
450.soplex	CF	Solves a linear program using the Simplex algorithm
453.povray	CF	A ray-tracer. Ray-tracing is a rendering technique that calculates an image of a scene by simulating the way rays of light travel in the real world
471.omnetpp	CINT	Discrete event simulation of a large Ethernet network.
473.astar	CINT	Derived from a portable 2D path-finding library that is used in game's AI
483.xalancbmk	CINT	XSLT processor for transforming XML documents into HTML, text, or other XML document types

Studies for the adoption of HS06



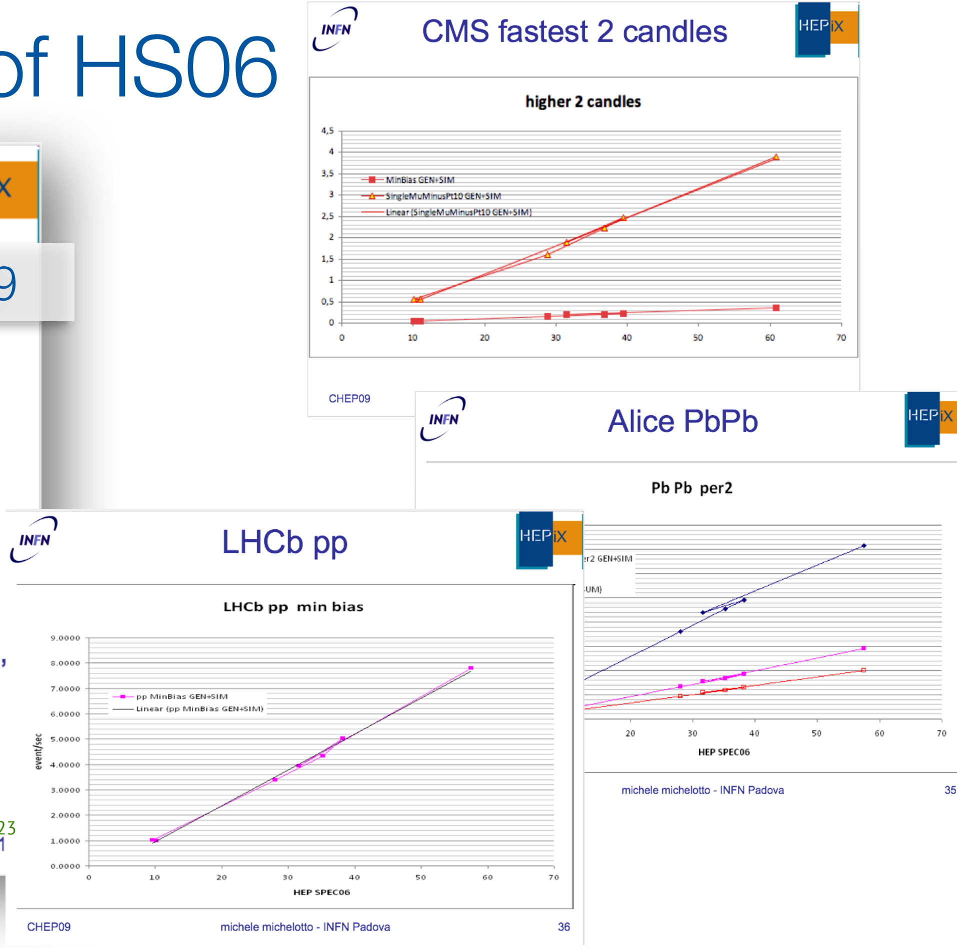
CHEP09

HEP Applications

- Atlas provided results for:
  - Event Generation, Simulation, Digitization, Reconstruction, Total (Full chain production)
- Alice:
  - Gen+Sim, Digitization, Reconstruction and Total
- LHCB:
  - Gen+Sim
- CMS
  - Gen+Sim, Digitization, Reconstruction and Total
  - For several Physics Processes (Minimum Bias, QCD Jets, TTbar, Higgs in 4 lepton, single particle gun events) to see if some physics channel would produce something different

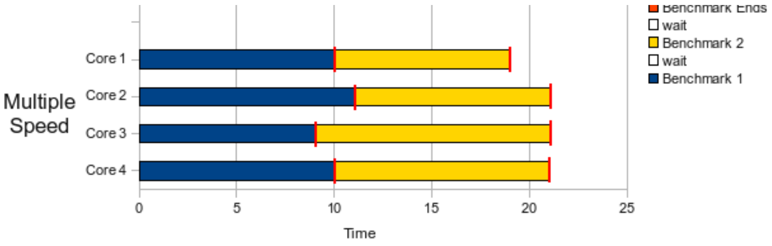
http://indico.cern.ch/getFile.py/access?contribId=19&sessionId=61&resId=0&materialId=slides&confId=35523

CHEP09 michele michelotto - INFN Padova 1



HS06 score computation

- For each core (vCPU) the sequence of benchmarks runs 3 times
  - Each core sequence is independent (potential **time misalignment**)
  - Multiple-Speed approach



- For each core and benchmark, the **median** value of the **3 measurements** is taken, and a ratio respect to a reference value is computed
- Compute the **geometric mean** of the ratio values (per core)
- HS06 score = sum of the geometric means across cores
- Execution time of the full HS06 suite O(4h)

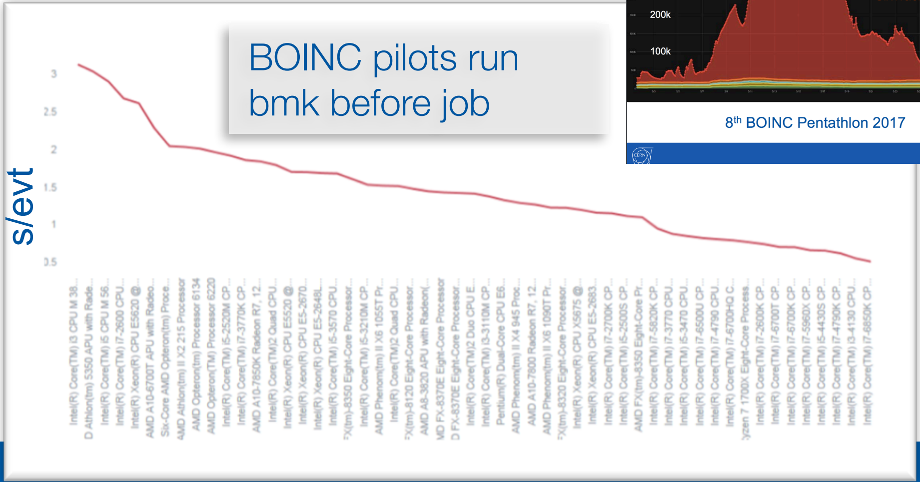
The Age of Fast Benchmarks

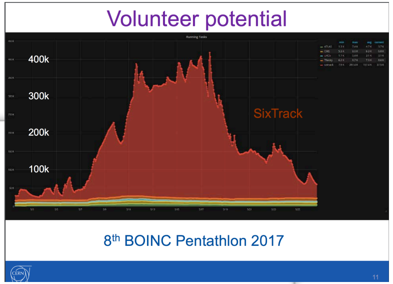
Useful in contexts where changing conditions require prompt feedback (but not necessary high accuracy)

- Example of changing conditions: the load / interference generated by "neighbor applications"
- A long-running benchmark (~4h) wouldn't be effective

Areas of adoption

- Grid pilot jobs
- Commercial Clouds
- Volunteer computing





Fast Benchmarks

Started with 5 candidates

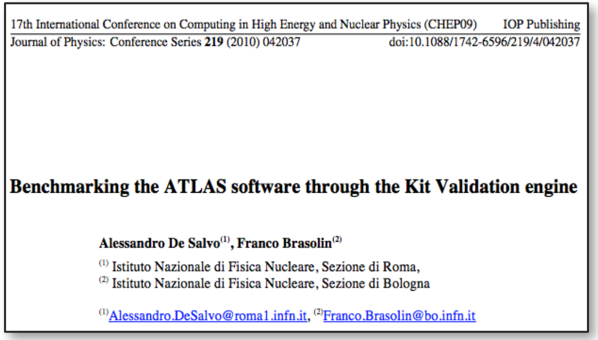
- ATLAS KV (KitValidation)
  - Mainly GEANT4. Default workload: 100 single muon event simulation
- DIRAC Benchmark 2012 (DB12), a.k.a. FastBmk, LHCbMarks
  - Python script: random.normalvariate()
- ROOT Stress test
- Legacy benchmarks: Whetstone, Dhrystone

- Systematic studies converge towards DB12 and Atlas KV
  - 2 options for running the benchmark:
    (a) "in-job" (b) "whole node" performance (a.k.a. "at-boot")

Why KV?

ATLAS Kit Validation tool

- Used in commercial cloud evaluations @ CERN in 2014-2016
  - Build on past experience
    - Comparison with HEP-SPEC06



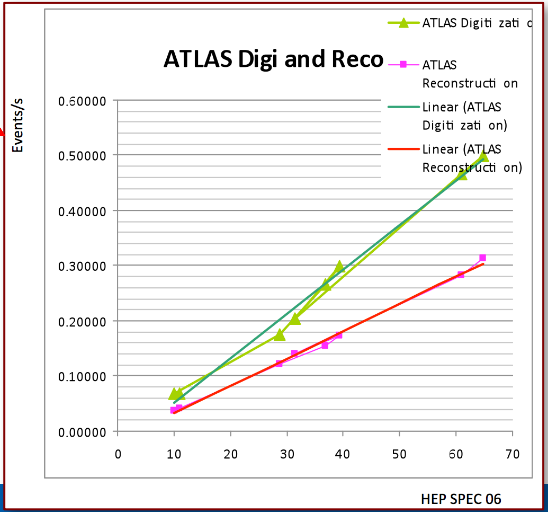
17th International Conference on Computing in High Energy and Nuclear Physics (CHEP09) IOP Publishing
Journal of Physics: Conference Series **219** (2010) 042037 doi:10.1088/1742-6596/219/4/042037

Benchmarking the ATLAS software through the Kit Validation engine

Alessandro De Salvo[1], Franco Brasolin[2]

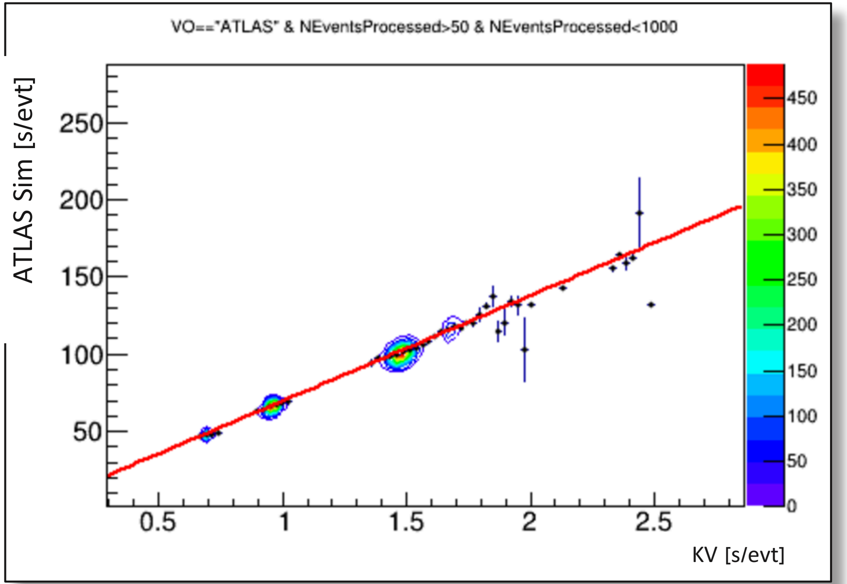
[1] Istituto Nazionale di Fisica Nucleare, Sezione di Roma,
[2] Istituto Nazionale di Fisica Nucleare, Sezione di Bologna

[1]Alessandro.DeSalvo@roma1.infn.it, [2]Franco.Brasolin@bo.infn.it



- ATLAS Kit Validation (KV)
  - Well known tool used by the ATLAS community
  - Framework essentially independent from the underlying tests
    - ATLAS code accessed from CVMFS

KV performance Vs ATLAS Sim job



Azure A3 and D3 series

Running benchmarks and jobs in Commercial Clouds (VM)

ATLAS Sim jobs Vs KV

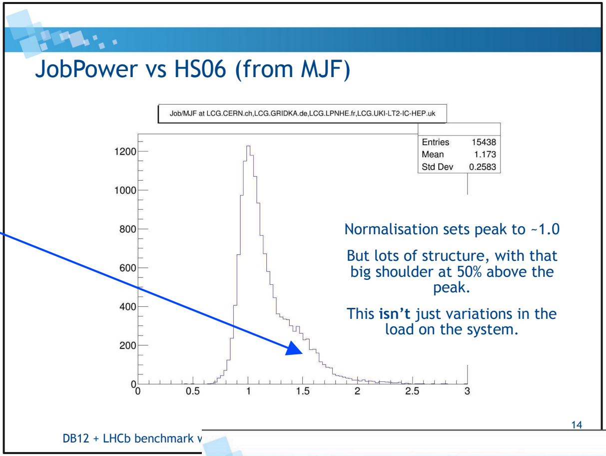
- At first order good linearity proven across different VM (and CPU) models

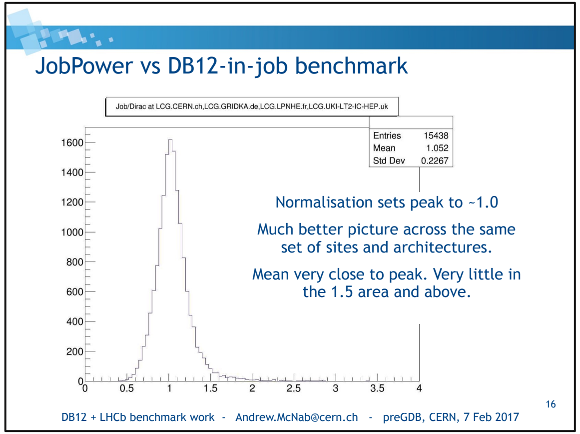
Why DB12?

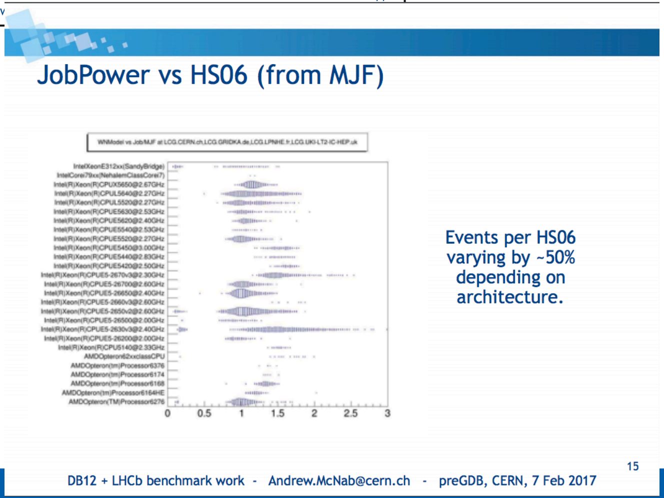
DB12 Vs jobs: LHCb

Andrew McNab
University of Manchester
LHCb and GridPP

- HS06 not a good predictor for MC
  - In particular for Intel Haswell CPUs
  - Not as bad for LHCb reconstruction jobs though
- DB12 is much better

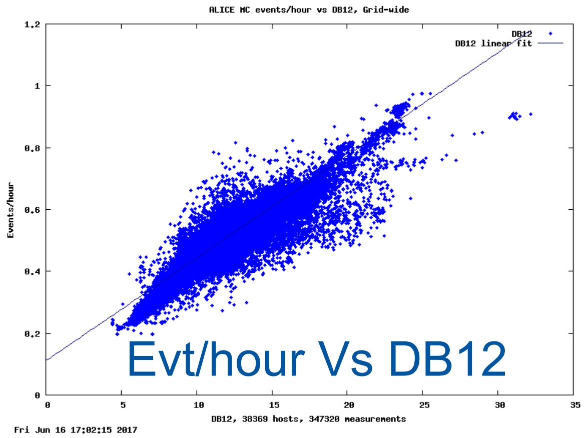


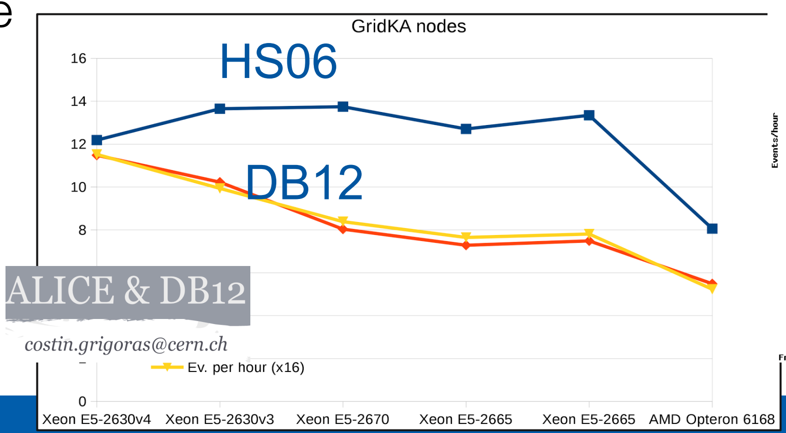


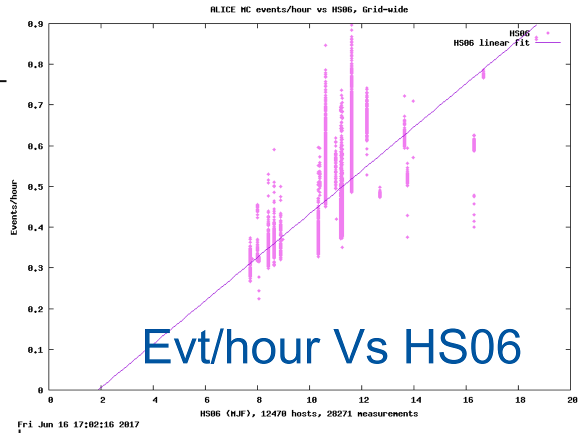


DB12 Vs jobs: ALICE

- Very good correlation of DB12 Vs MC
  - Running DB12 in pilot job
- Large discrepancy respect to HS06 from MJF
  - HS06 measures the pessimistic scenario of full load
  - Indication that the server load is a crucial component to take into account

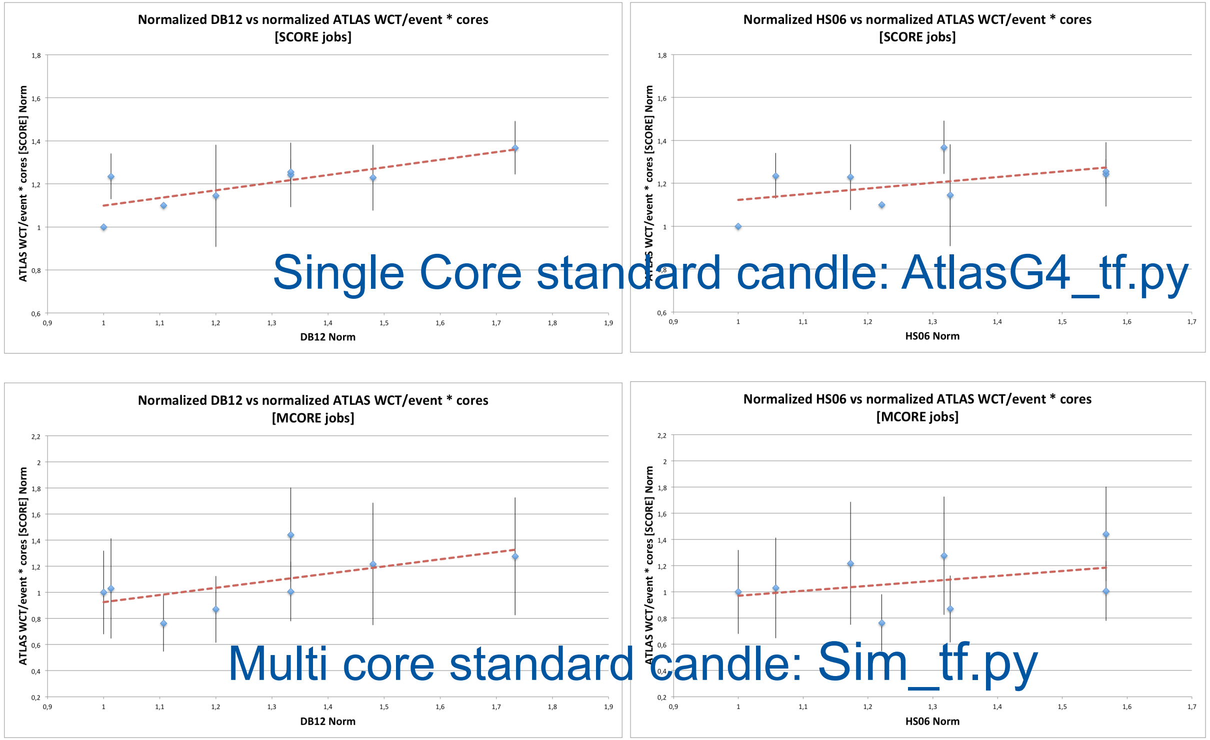






ATLAS

Alessandro De Salvo
on behalf of the ATLAS Distributed Computing group



Single Core standard candle: AtlasG4_tf.py

Multi core standard candle: Sim_tf.py

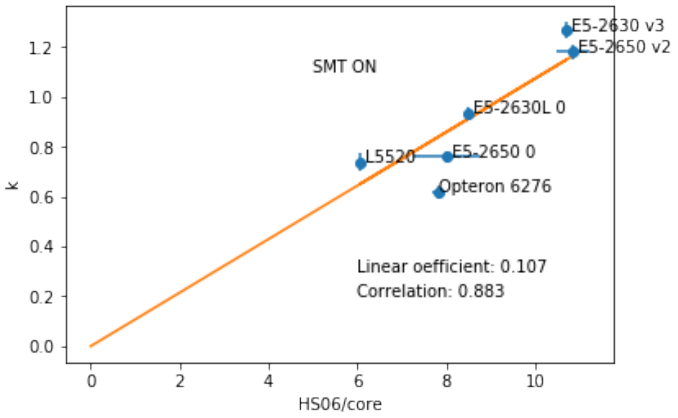
- HammerCloud reference jobs (single and multi core) running on benchmarked resources at GridKa
- Further investigation is needed
  - In particular for multi core jobs

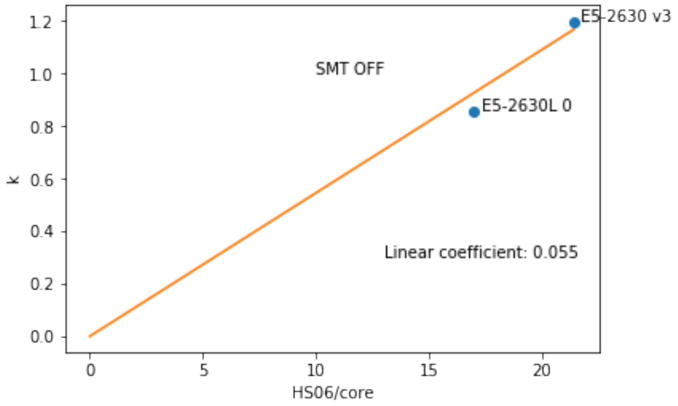
Results from Feb. pre-GDB

Passive Benchmarks of ATLAS Tier-0 CPUS

A. Sciabà

- Use real jobs to measure the relative speeds of different CPU models
  - Description of the analysis in the pre-GDB talk
- The analysis was applied to jobs run at the ATLAS Tier-0
  - Mainly reco jobs
  - Scaling generally good, two exceptions
    - Opteron way off
    - Haswell tends to perform better than what HS06 predicts: +10% with both SMT ON and OFF





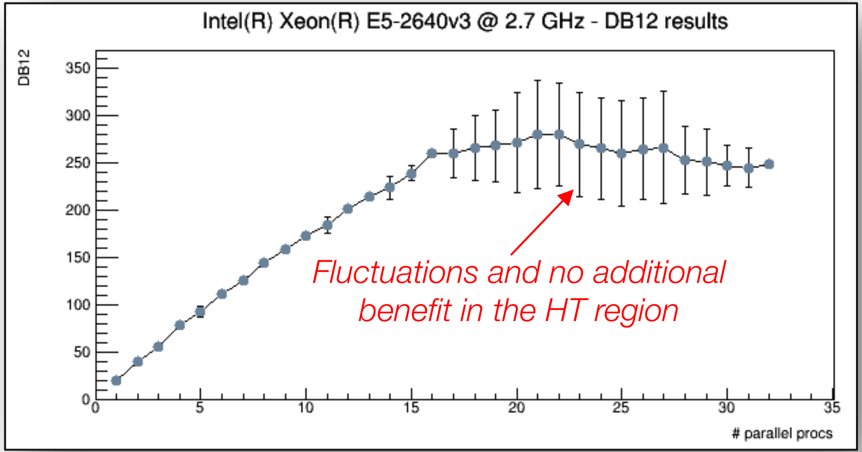
CMS Tests

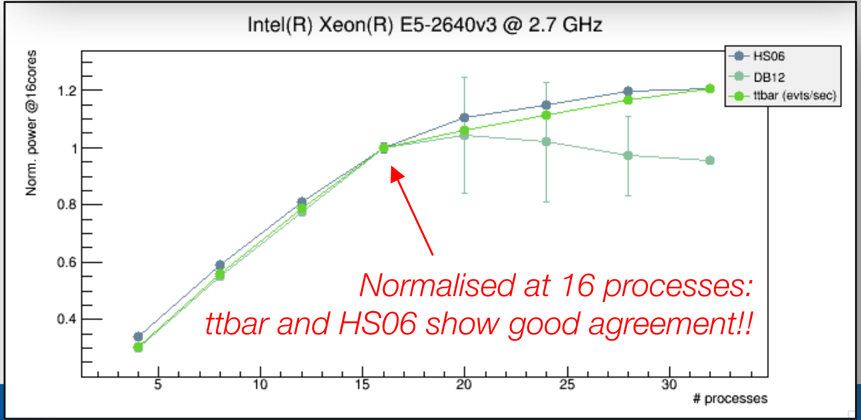
J. Flix https://indico.cern.ch/event/624830/

Comparison of DB12, KV, HS06 Vs CMS jobs

– Several dedicated nodes targeted for benchmarking

- **E5-2640v3 2.60 GHz (td102.pic.es)**
- **E5-2650 2.00 GHz (td713.pic.es)**
- E5-2650v2 2.60 GHz
- **E5645 2.40 GHz (td608.pic.es)**
- **X5650 2.67 GHz (td550.pic.es)**
- E5-2680v4 2.40GHz (new)





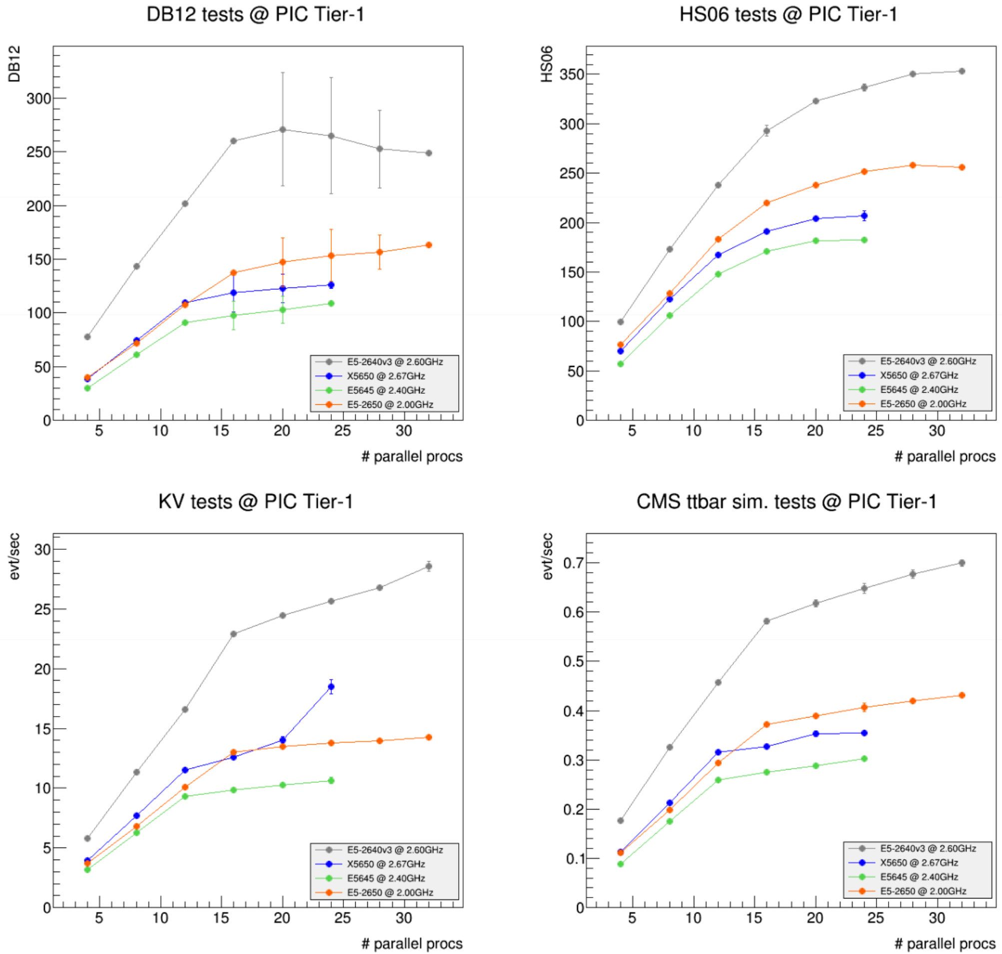
CMS Tests

DB12 fluctuation quite large (5%-25%) in HT enabled region

PIC port d'informació científica

Tests made at PIC so far

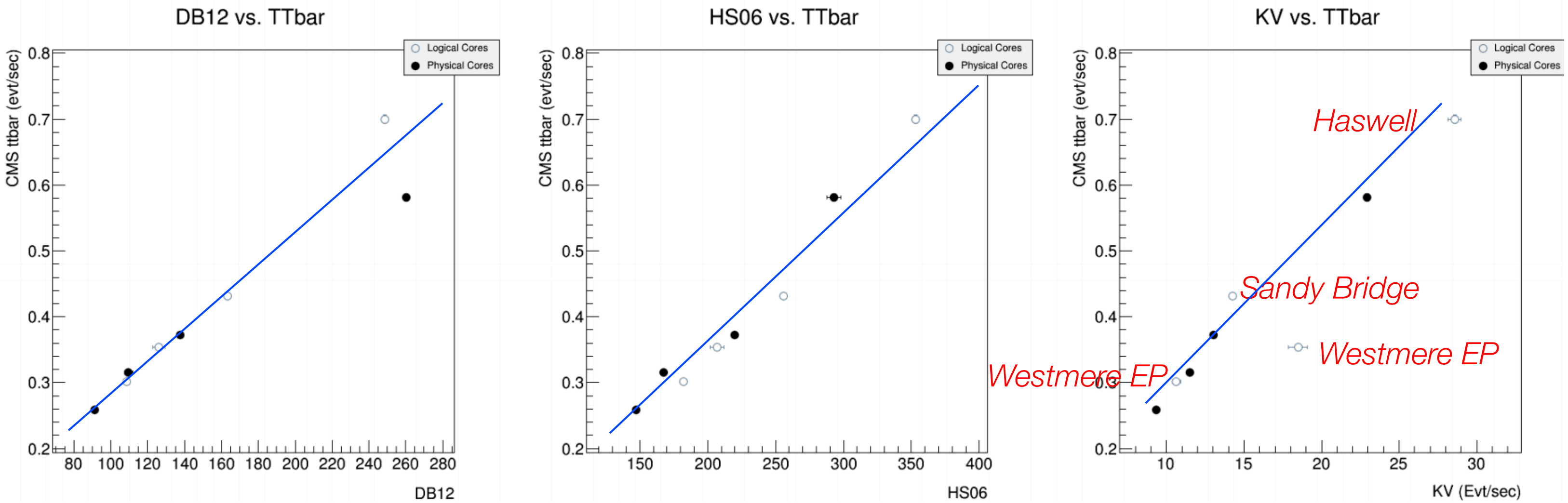
SUMMARY



Notes on stdev

- DB12:
  - → from 1 to Phys. Cores: 1%-5%
  - → from Phys. Cores to Log. Cores: 5%-25%
  - → @ Log. Cores: 1%-2.5%
- HS06: <2%
- KV: <3%
- CMS ttbar sim: <2%

CMS Tests: Summary



No big preference for a benchmark respect to other

– DB12 seems to have larger discrepancy for Haswell CPU model

Study still ongoing

– NB: the blue lines are not fits!!

– Need to add Broadwell, and (a.s.a.p.) Skylake

DB12 studies: Summary

Seen the multiple and somehow contradictory results on DB12 the working group has invested effort in additional studies

– Application profiling

– Effect of different implementations: C++ or using Numpy

– Reproducibility under different Python versions

DB12 studies: Summary

Seen the multiple and somehow contradictory results on DB12 the working group has invested effort in additional studies

– Application profiling

-> **DB12 benefits from better branch prediction** (see next slides)

-> **DB12 doesn't profit from HT enabled**

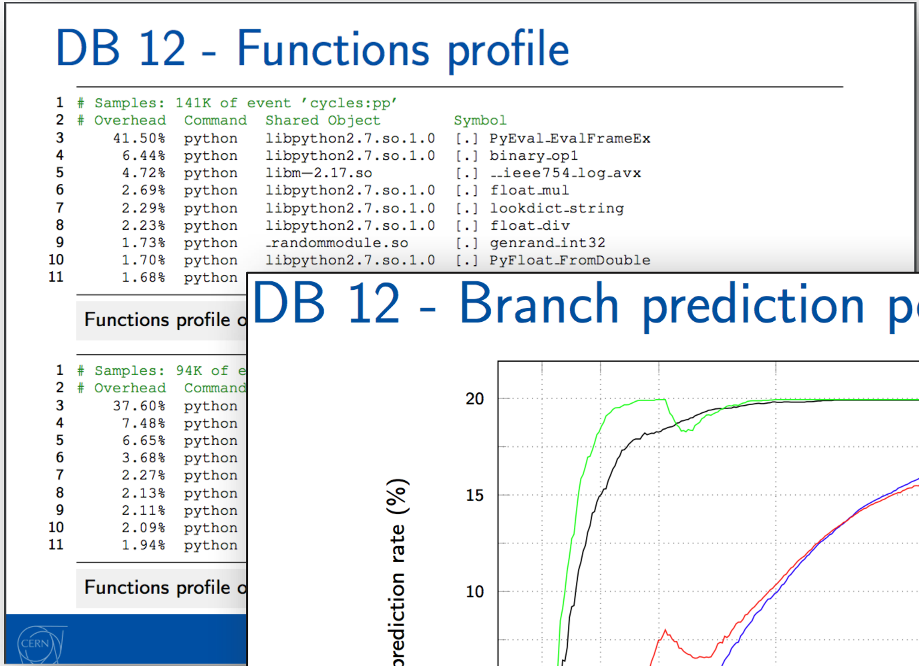
– Effect of different implementations: C++ or using Numpy

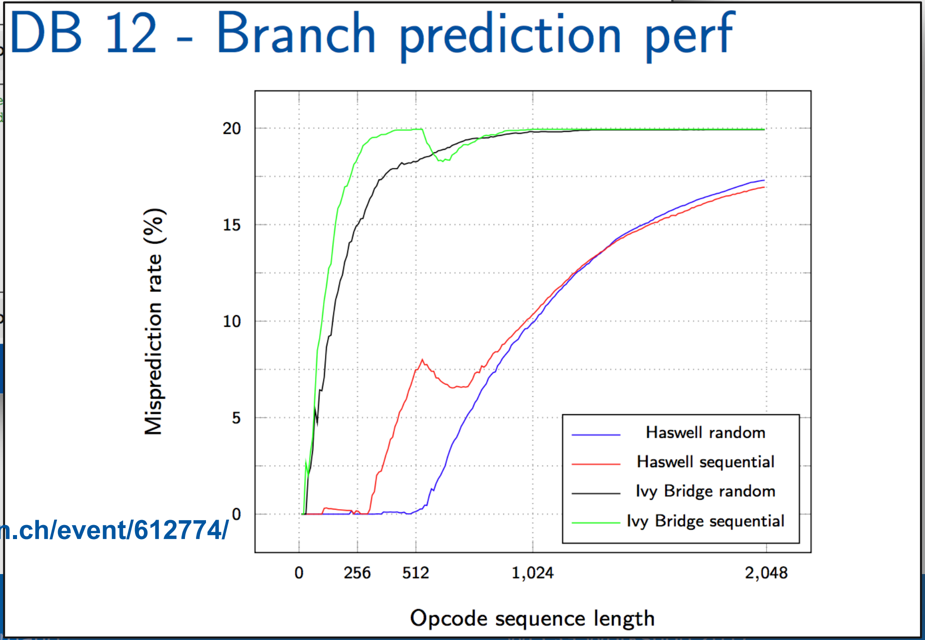
-> **DB12 is not dominated by rand number calls** (backup slides)

– Reproducibility under different Python versions

-> **DB12 suffers of dependency from python versions** (backup slides)

The Branch Prediction due to CPython Module





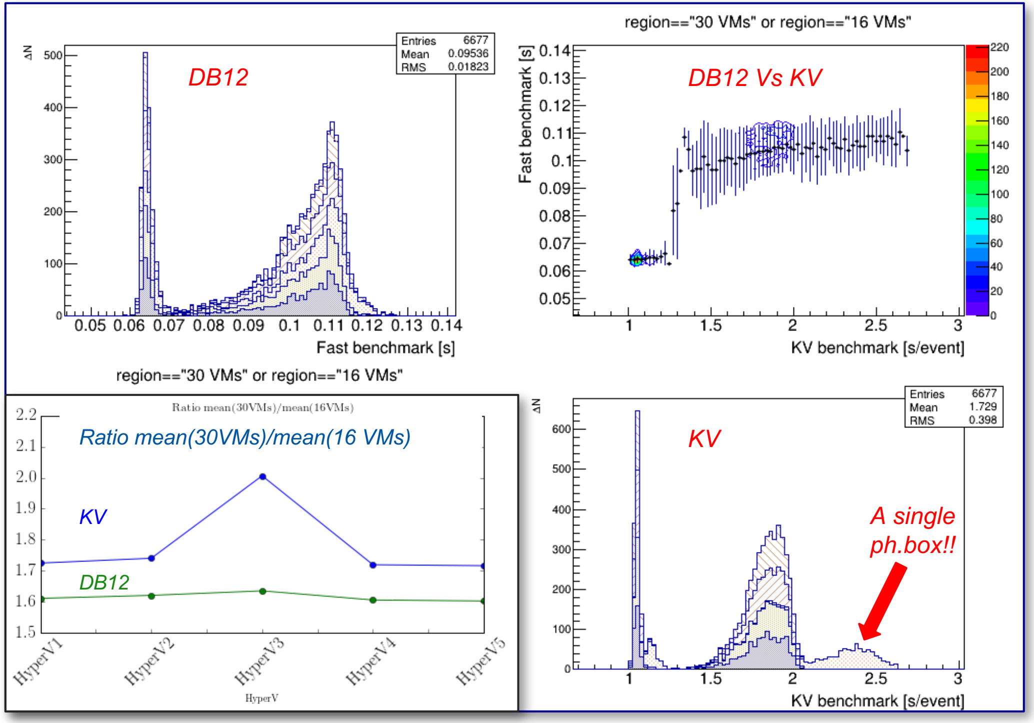
https://indico.cern.ch/event/612774/

- **perf** studies from M. Guerri have shown that the main contributor to DB12 is **`PyEval_EvalFrameEx`**
- Starting from Haswell models, this module benefits from a better branch prediction that boosts the DB12 performance

What can go wrong with DB12

Example:

- In whole-node server test, DB12 can fail in spotting badly behaving servers
- The average performance degradation differs if DB12 or KV are used



HS06 32 bits Vs 64 bits

HS06 32 bits Vs 64 bits

The official HS06 benchmark must be compiled at 32 bits

It has been questioned if compiling it at 64 bits would compensate the discrepancy respect to HEP workloads (that are compiled at 64 bits)

Also this aspect has been investigated

- Tested 4 CPU model: Sandy Bridge, Ivy Bridge (v2), Haswell (v3), Broadwell (v4)

Results (details in backup slides):

- HS06 score would change of ~15% moving from 32 to 64 bits
  - Factor is different for different CPU models, but within 5%
  - **A change of the official procedure is not justified**

Preparation to the next long-running benchmark

In view of a HS06 successor

Prepare the test environment

- Disentangle effects such as
  - Bare-metal server Vs VMs, HT ON/OFF, different OS, load on neighbor slots,…

- Perform reproducible studies
  - **Document procedures is crucial**
    - How to setup the environment, the application parameters, and the proper configuration to run a given workload
    - Experiments: **identify and share representative job types (candles)**
      - e.g. via CVMFS, containers, etc..
    - Have access to monitoring data of production jobs

It was similarly done for HS06!

An option for ”self-contained” candles

Use **Docker containers** to embrace only what needed to run the reference workloads

- Can also include in the container only the **cvmfs** files that are used
  - reduce image size

Advantages of container :

- Easy to use, doesn’t need cvmfs mounted if snapshot used, lighter than a VM
- Possibility to run also with Container Orchestration Engines, and target large clusters

Drawback: needs recent host OS (like CentOS) or kernel version > 3.10

- SLC6 has a too old kernel

HEP-Workload in containers (A Proof of Concept)

Publicly available here, try it!

- https://gitlab.cern.ch/giordano/hep-workloads/tree/master

Started including

- Atlas KV
- CMS TTbar GEN-SIM

Work in progress

- Extend the approach to the other experiments
- Explore alternatives to snapshotting
  - docker-volume-cvmfs mount

```
docker build -t gitlab-registry.cern.ch/giordano/hep-workloads/image .

docker push gitlab-registry.cern.ch/giordano/hep-workloads/image
```

giordano/hep-workloads

giordano/hep-workloads/cms-gen-sim

Tag	Tag ID	Size
latest	f199e4d5c	1.31 GB · 4 layers
test	f199e4d5c	1.31 GB · 4 layers

giordano/hep-workloads/atlas-kv-bmk

Tag	Tag ID	Size
test	c22cd430d	452 MB · 4 layers
latest	c22cd430d	452 MB · 4 layers

Few Conclusions

DB12

DB12 (python version) has been deeply studied

- DB12 “in-job” scales well with ALICE and LHCb MC applications
- BUT:
  - Runtime is dominated by libpython calls.
    - Nothing to do with random number generation!
  - DB12 shows dependency from Python version, and it doesn’t benefit from SMT enabled
  - DB12 “at-boot”: +40% boost (respect to HS06) from Intel Sandy Bridge to Haswell only for ½ loaded servers (SMT enabled)
    - Discrepancy with HS06 goes down when processes running are x2 physical cores
    - The initial boost is due to a better branch prediction in the Haswell CPU frontend

DB12 (cont.)

- DB12 doesn't show the stability and characteristics to probe all components of the CPU potentially used by HEP workloads
  - Limited instruction mix; does not stress the memory subsystem;
  - Adoption for procurement would represent significant risk

- DB12 is still attractive for fast benchmark in jobs

Long-running Benchmark

HS06

- Preliminary study still shows good agreement among HS06 and CMS MC ttbar, when server fully loaded
- Passive benchmarking
  - Discrepancies among HS06 and ATLAS reco jobs are within 10%
- Need to better understand the reasons of the discrepancies for LHCb and ALICE

SPEC2017 is now available: should start testing it

Work in progress to setup a testbed for the HS06 successor

- Support from the Experiments is mandatory here

Follow the Exp. Software Evolution

Crucial condition for the W.G.:

Be aware of the major changes in the Exp. applications

- Changes that wouldn't follow the benchmark scaling factor across CPU models
  - Good motivation for the adoption of new benchmark(s)

Currently we are evaluating benchmark candidates <u>only</u> vs running Exp. applications

- For a future long-term benchmark the next improvements can be as important as the currently running applications

The information & experience should timely reach the WG

- See yesterday discussion on "Efficiency and cost"

CPU Unit:

How to introduce a new benchmark
&
learn from past experience

Status of the CPU Unit proposal

- Presented by Andrew in several meeting
  - MB, GDB, Accounting TF, Benchmarking WG

- From M.B. minutes
  - […] *there are no objections in principle from the MB to the CPU Unit proposal; on the contrary, Andrew McNab should follow this up within the benchmarking working group, to ensure that a document about the CPU Unit proposal is prepared at the same time as the recommendation for a new benchmark, so that the two proposals can be analysed together by the MB.*

Mainly an Accounting Aspect

Context

- The idea of changing the benchmarks we use for accounting and pledges has been raised
- The HEPiX Benchmarking WG has been asked to start looking for candidates
- The WLCG Accounting TF was asked to report on how hard it would be to change APEL, Accounting Portal etc
- This proposal is one way of managing that kind of change, in a way that makes it easier to change in the future
- It's important to stress that we may want to change again if there is a repeat of the Haswell step-change in performance
  - +40% for HEP applications and fast benchmarks; but not for HS06. So improved delivery to experiments is not recognised

"CPU Units" idea

- WLCG adds a "CPU Unit" (CU) in parallel with HS06 in the accounting system (APEL, accounting portal etc.)
- To start with, 1.0 CU = 1.0 HS06
- WLCG can update the definition of CU to reflect changes in the technology (eg the Haswell scenario)
  - It can be a combination of one or more benchmarks
  - New benchmarks can be included; old ones dropped
- Since CU is designed to be updated, we don't have to change the accounting system, pledges etc each time
- But this puts constraints on what revisions can be made to the CU definition

"CPU Units" revision constraints

- CU definition should be based on empirical evidence about experiment software performance across relevant hardware
- Avoid penalising sites for good faith decisions in the past
  - So sites may choose to continue to publish previously published CU values after a revision
  - Guarantees that their ability to pledge won't go down
  - But prevents them using an old definition of CU on new hardware
- Weights/scale used within CU should be chosen to ensure that on older (oldest?) hardware:

  previous CU value = new CU value

"CPU Units" revision consequences

- On newer hardware if the new definition is sensitive to improvements in technology, then new CU value may go up
  - This is a Good Thing: it gives credit for hardware which is doing more work for experiments than we thought
  - Motivates sites to buy hardware which is better for the experiments
- WLCG has the choice about whether to stay with the same CU definition for a decade or change next year
  - Don't have to worry about cost of changing APEL etc
  - Don't get paralysed by the thought that we might make the wrong decision and be stuck with it for ten years
  - Can respond to evolving experiment code

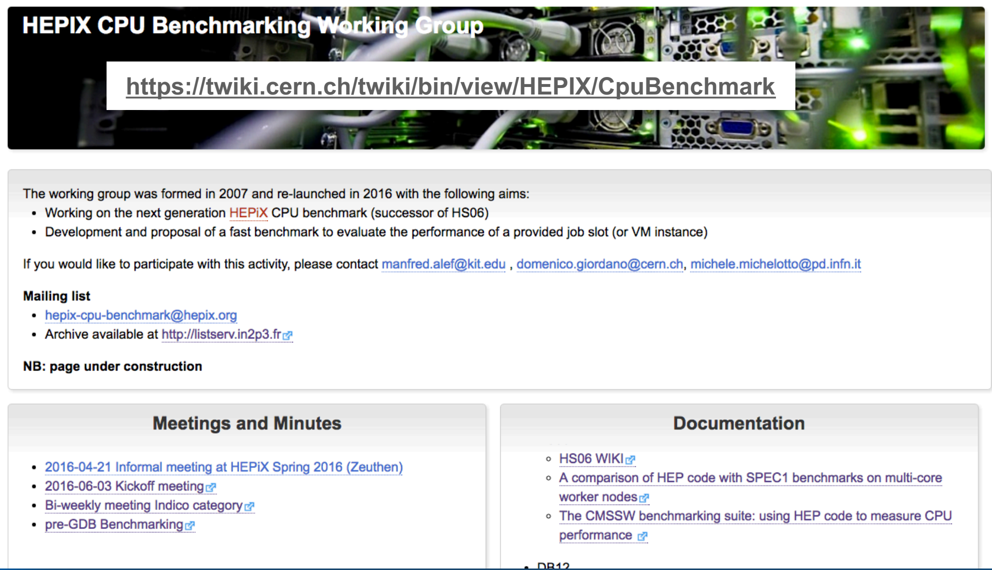
CERN

Working Group members

O(50) members subscribed to the list

Bi-weekly meetings (Friday 14:00) restarted since ~1 year

- https://indico.cern.ch/category/1806/
- O(10) people attending, mainly remotely. Increasing participation in the last 6 months
  - Typically 1 repres. from each Exp.
  - + people involved in performance studies (CERN-IT UP & Procurement)



HEPIX CPU Benchmarking Working Group

https://twiki.cern.ch/twiki/bin/view/HEPIX/CpuBenchmark

The working group was formed in 2007 and re-launched in 2016 with the following aims:

- Working on the next generation HEPiX CPU benchmark (successor of HS06)
- Development and proposal of a fast benchmark to evaluate the performance of a provided job slot (or VM instance)

If you would like to participate with this activity, please contact manfred.alef@kit.edu , domenico.giordano@cern.ch, michele.michelotto@pd.infn.it

Mailing list

- hepix-cpu-benchmark@hepix.org
- Archive available at http://listserv.in2p3.fr

NB: page under construction

Meetings and Minutes

- 2016-04-21 Informal meeting at HEPiX Spring 2016 (Zeuthen)
- 2016-06-03 Kickoff meeting
- Bi-weekly meeting Indico category
- pre-GDB Benchmarking

Documentation

- HS06 WIKI
- A comparison of HEP code with SPEC1 benchmarks on multi-core worker nodes
- The CMSSW benchmarking suite: using HEP code to measure CPU performance
- DB12

Other Implementations of DB12

- DB12 Optimised Python version with Numpy (V. Innocente) : x10 faster than standard DB12
- DB12 C++ implementation (D. Giordano): x10 faster than standard DB12
  - Multi-processing using `fork`, `pipe`, same `normalvariate` algorithm
- Comparison of the **perf profile** for the three implementations shows that, contrary to the standard DB12 code, the Numpy and C++ versions are dominated by calls to math and rand libs

DB12 standard			DB12 numpy			DB12 C++		
Shared Object	**Overhead [%]**	**Num of Symbols**	**Shared Object**	**Overhead [%]**	**Num of Symbols**	**Shared Object**	**Overhead [%]**	**Num of Symbols**
libpython2.7.so.1.0	86.64	478	mtrand.so	37.82	36	DB12.exe	39.85	4
libm-2.17.so	5.19	4	libm-2.17.so	23.78	4	libm-2.17.so	39.75	3
_randommodule.so	4.06	63	libpython2.7.so.1.0	16.7	364	libc-2.17.so	19.8	3
math.so	2.63	73	umath.so	7.16	286	kernel.kallsyms]	0.48	12
kernel.kallsyms]	0.5	13	multiarray.so	5.62	318			
libpthread-2.17.so	0.44	3	libc-2.17.so	2.22	16			
libc-2.17.so	0.2	2	kernel.kallsyms]	0.97	43			
			libpthread-2.17.so	0.36	4			
Total	**99.66**			**94.63**			**99.88**	

- Documented @ http://cern.ch/go/Pq9T

DB12 Vs OS (and python) Versions

- DB12 scores are affected by changing python version
  - Variation of 10%-18%
- Ratio **DB12(32 cores) / DB12 (16 cores) = ~1**
  => no gain in SMT=ON
- As reference:
  - the scores of the C++ version are less affected by the different OS (~5%)
  - Ratio **DB12(32 cores) / DB12 (16 cores) = ~1.5**
    => 50% gain with SMT=ON
- **NB:** *This is **not** a suggestion to migrate DB12 to a C++ version, but just an example to highlight potential issues and discrepancies among implementations*
- Documented @ http://cern.ch/go/Pq9T

DB12 python

OS	version	32 procs ratio_to_python2.6	ratio 32/16
slc6-base	Python 2.6.6	1	1.00
CC7	Python 2.7.5	1.09	1.02
cc7-base	Python 2.7.5	1.09	1.04
python:2.7	Python 2.7.13	1.18	1.01
python:3	Python 3.6.0	1.05	0.98

DB12 C++

OS	version	32 procs ratio_to_python2.6	ratio 32/16
slc6-base	Python 2.6.6	1	1.49
CC7	Python 2.7.5	1.04	1.47
cc7-base	Python 2.7.5	1.03	1.46
(Debian 8.7) python:2.7	Python 2.7.13	1.05	1.49
(Debian 8.7) python:3	Python 3.6.0	1.05	1.48

Benchmark Comparison on Grid Nodes

Comparison on different HW models (M. Alef)

- Better scaling of DB12-cpp and DB12-np with HS06 than initial DB12 Python script
- The **+45%** boost appears **only** when running DB12 on

  # of slots =< # physical cores

  and goes down when SMT is enabled
  - Another effect of the lack of gain of DB12 with SMT enabled
  - NB: In SB also DB12 benefits of the 20% gain with SMT=ON

Double ratio respect to a given ref machine **(E5-2665 32 copies)** to the number of copies and to the HS06 of each machine (to remove frequency effect)

(bmk_X_CPU_Y_copies_N/bmk_HS06_CPU_Y_copies_N) / (bmk_X_CPU_REF_copies_M/bmk_HS06_CPU_REF_copies_M)

Hardware model		#copies	ratio	DB12/ HS06	DB12-cpp/ HS06	DB12-np /HS06
E5-2630v4	Broadwell	20	1	1.38	0.94	1.08
E5-2630v4		32	1.6	1.23	1.03	1.10
E5-2630v4		40	2	1.15	1.11	1.15
E5-2630v3	Haswell	16	1	1.44	0.91	1.05
E5-2630v3		24	1.5	1.24	0.95	1.05
E5-2630v3		32	2	1.08	1.05	1.07
E5-2660v3		20	1	1.48	0.94	1.05
E5-2660v3		32	1.6	1.24	1.04	1.10
E5-2660v3		40	2	1.17	1.12	1.14
E5-2665	Sandy B.	16	1	0.99	0.71	0.79
E5-2665		24	1.5	0.94	0.88	0.91
E5-2665		**32**	**2**	**1.00**	**1.00**	**1.00**

HS06 32 bits Vs 64 bits: Approach

Compare HS06 scores on different HW models and also different OS (SLC6 and CC7)

Each server is benchmarked fully loading the available cores

- SMT enable

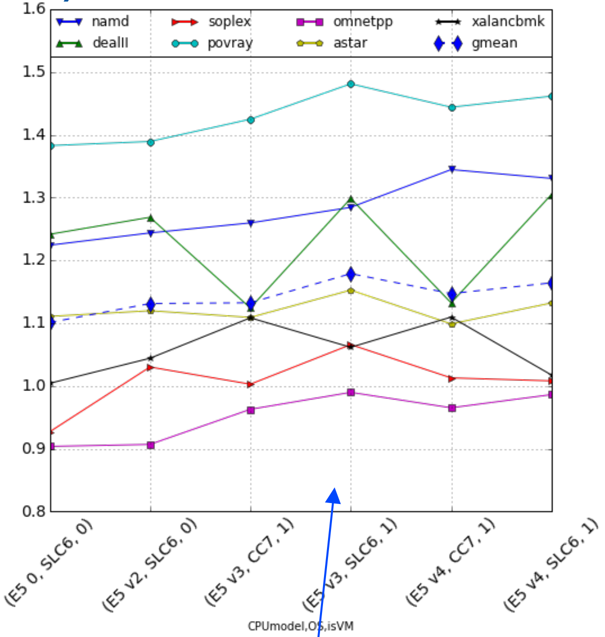
Compare scale factors respect to other benchmarks (DB12, KV)

Abbr.	**Family**	**Model**	**nodes**	**OS**
E5 0	Sandy Bridge	E5-2690 0 @ 2.90GHz	1 ph. node	SLC6
E5 v2	Ivy Bridge	E5-2650 v2 @ 2.60GHz	2 ph. nodes	SLC6
E5 v3	Haswell	E5-2630 v3 @ 2.40GHz	VMs	SLC6 , CC7
E5 v4	Broadw ell	E5-2630 v4 @ 2.20GHz	VMs	SLC6 , CC7

Ratio HS06 (score @ 64)/(score @32)

			bits	32	64	ratio
CPUmodel	OS	isVM	ncores	HS06 score		
E5 0	SLC6	0	32	334.443039	368.200709	1.10
E5 v2	SLC6	0	32	339.945662	384.547059	1.13
E5 v3	CC7	1	16	167.027239	190.819932	1.14
			32	336.416957	380.143415	1.13
	SLC6	1	16	167.284242	198.056291	1.18
			32	330.745914	389.659848	1.18
E5 v4	CC7	1	20	204.118525	234.214985	1.15
			40	404.375505	463.695030	1.15
	SLC6	1	40	398.016795	463.375925	1.16

Abbr.	Family	Model
E5 0	Sandy Bridge	E5-2690 0 @ 2.90GHz
E5 v2	Ivy Bridge	E5-2650 v2 @ 2.60GHz
E5 v3	Haswell	E5-2630 v3 @ 2.40GHz
E5 v4	Broadwell	E5-2630 v4 @ 2.20GHz



Ratio <HS06$_{64bits}$>/<HS06$_{32bits}$> (average per CPU model, OS, cores)

- Ranges from 10% (S.B.) to 18% (Haswell)
- Is consistent for VM 16 and 32 on same ph. node
- Each individual benchmark in HS06 suite has completely different ratio values (remember: geom. mean)

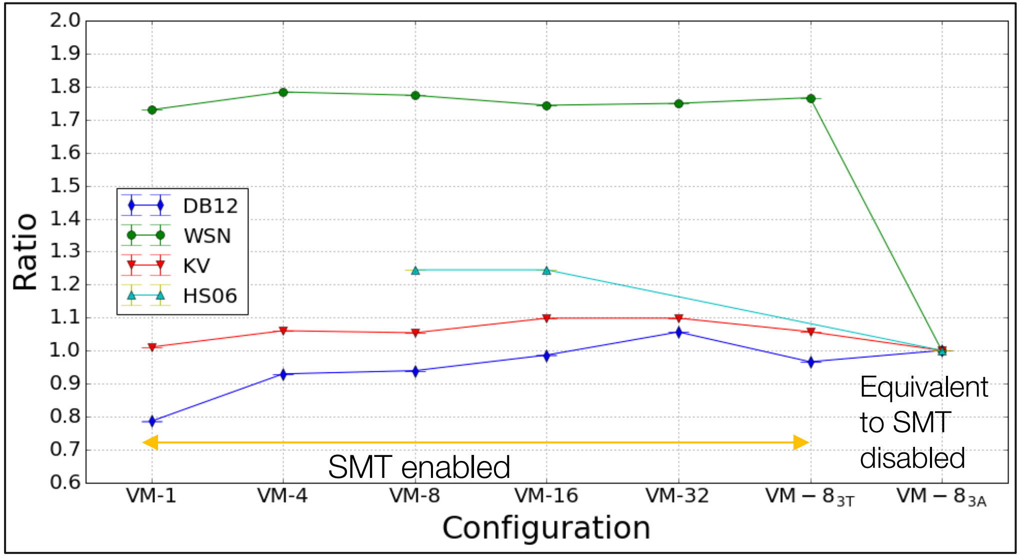
Scaling factors across VM configurations

- Benchmarking of Haswell servers (E5-2630 v3 CPUs) in virtual environment

- Partition the available compute resources in a number of VMs of same size
  - => Fully load the servers as done for HS06

- Study the ratio of performance among different configurations
  - VM sizes and SMT enabled/disabled

- The +20% gain in performance seen in HS06 with SMT=ON is not reproduced by DB12 and KV
  - Bigger discrepancy from DB12



Dissecting Benchmarks with Perf

M. Guerri

ITTF: https://indico.cern.ch/event/612774/
Internal Note http://cds.cern.ch/record/2257973?ln=en

Open Question

DB12 does not profit from SMT on Haswell/Broadwell architecture. KV apparently does the same. What is the reason behind this slowdown? Is it the same for both benchmarks? e.g. (hypothesis, not validated yet):

- DB12 is heavy on branches/jump. On Haswell, there are two execution ports that can execute branch instruction (compared to only one on IvyBridge. These are not ports exclusively for branch instructions). Validation of speculative fetch happen very fast keeping the pipeline always very busy.
- KV heavily profits from iTLB. When running two hardware threads, iTLB entries are thread specific. High number of conflicts.

17th February 2017 Benchmarking Working Group 1

Conclusions 1

- Simulation (Geant4) (and python?) shows a high degree of code non-locality
  - Any different behavior in instruction pre-fetching will affect it much more than other benchmarks
- HS06 is memory greedy (at least compared to cms simulation& reconstruction)
  - Multiple instances running against common resources will scale worse than CMS Sim&Reco
  - It surely depends on the details of the cache hierarchy (arm, atom, knl, skylake, amd)

6/4/17 VI benchmark 9

V. Innocente

https://indico.cern.ch/event/624828/contributions/2547881/attachments/1441812/2220330/Benchmarking.pdf

Example: slim KV benchmark

Create a Docker image based on slc6-base and that contains

- Only libraries needed to run athena
  - Limited set of files from cvmfs (**624MB**)
    - **atlas-condb.cern.ch atlas-nightlies.cern.ch atlas.cern.ch sft.cern.ch**
  - A number of standard applications
- Slim Conditions sqlite file (thanks to L. Rinaldi)
  - ALLP200-OFLCOND-SDR-BS7T-04-03.slim.sqlite (**490KB**)
- Total size of the container **1.16 GB**

```
FROM gitlab-registry.cern.ch/linuxsupport/slc6-base:latest

MAINTAINER Domenico Giordano <domenico.giordano@cern.ch>

RUN yum install -y \
    which \
    man \
    file \
    util-linux \
    gcc \
    wget \
    tar \
    perl ; yum clean all
#workaround https://github.com/CentOS/sig-cloud-instance-images/issues/15

RUN wget https://test-giordano.web.cern.ch/test-giordano/Benchmarking/cvmfs_kv-bmk-v17.8.0.9.tgz; tar -xvzf cvmfs_kv-bmk-v17.8.0.9.tgz; rm cvmfs_kv-bmk-v17.8.0.9.tgz

COPY ./kv-bmk /kv-bmk

ENTRYPOINT ["/kv-bmk/kv-bmk.sh"]
CMD ["-n0"]
```

Example: slim KV benchmark (II)

How to run a candle in a container

- `docker run -it --rm gitlab-registry.cern.ch/giordano/hep-workloads:atlas-kv-bmk-v17.8.0.9`
  - Automatically detects number of available CPUs
  - Output printed out, in evt/sec, with average, median, min, max
    - json output soon available (compatible with the benchmark suite format)

```
[root@bmk16-slc-e839aa4f-ca2b-4b3c-be68-dc6271caaf30 ~]# time docker run -it --rm gitlab-registry.cern.ch/giordano/hep-workloads:atlas-kv-bmk-v17.8.0.9
ncopies= 16
Using AtlasProduction/17.8.0.9 [cmt] with platform x86_64-slc6-gcc47-opt
        at /cvmfs/atlas.cern.ch/repo/sw/software/x86_64-slc6-gcc47-opt/17.8.0
************************************************************************************************************************
KV cpu performance [evt/sec]: avg  0.758 over 16 threads. Median 0.758 Min Value 0.712 Max Value 0.793
************************************************************************************************************************
```

- Possibility to pin a subset of cores using Docker functionalities for more detailed studies,
  - `docker run -it --rm --cpuset-cpus=XXX gitlab-registry.cern.ch/giordano/hep-workloads:atlas-kv-bmk-v17.8.0.9`

NB: the duration is, as before, dominated by the application initialization phase

- Could benefit from the work ongoing in the experiments to snapshot the initialization phase