HEP Benchmarks and HEPSCORE Deployment Task Force

Michele Michelotto INFN Padova

Slides material thanks to Domenico Giordano (CERN/IT) and Miguel Medeiros (CERN/IT)

Pre-GDB “Worker Nodes” 13 July 2021

Intro

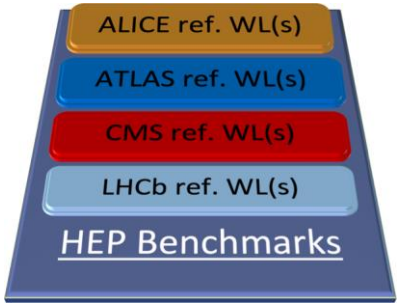
- This report focuses on the HEP Benchmarks project https://gitlab.cern.ch/hep-benchmarks
  - Main activity of the WG in the last year

Scenarios	HS06	HEPscore
x86 CPUs (y. 2010-2020)	✓	✓
New CPUs models and/or arch	?	✓
New Exp Sw	?	✓(w/ new reference WLs)
CPU + GPU/FPGA/…	✗	✓(same speed definition: event/s)

- In short

  WLCG has to change the benchmark HS06 sooner or later
  - Motivations extensively presented at the last HEPiX Workshop ‘19
  - Briefly: HS06 end of technical support (2017), targets only CPUs, we don’t know if will continue to scale well w.r.t. new HEP sw

- Field-specific (HEP) workloads guarantee by construction
  - A score with high correlation to the throughput of HEP workloads
  - A usage pattern that is similar to that of HEP workloads



Current WLCG benchmark: *HEP-SPEC06 (HS06)*

- Based on SPEC CPU2006
  - Standard Performance Evaluation Corporation was founded in 1988
  - SPEC CPU2006: Industry-standard, CPU-intensive, benchmark suite
  - Current SPEC CPU subcommittee members include AMD, ARM, Dell, Fujitsu, HPE, IBM, Inspur, Intel, Nvidia and Oracle [*]

- HS06 is a subset of SPEC CPU® 2006 **benchmark**, tuned for HEP
  - 7 C++ benchmarks recompiled with gcc optimizer switches of LHC experiments' software
  - In 2009, proven high correlation with HEP workloads

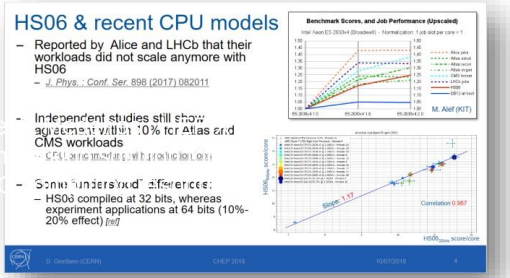
Bmk	Int vs Float	Description
444.namd	CF	92224 atom simulation of apolipoprotein A-I
447.dealII	CF	Numerical Solution of Partial Differential Equations using the Adaptive Finite Element Method
450.soplex	CF	Solves a linear program using the Simplex algorithm
453.povray	CF	A ray-tracer. Ray-tracing is 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

The 7 C++ HS06 benchmarks

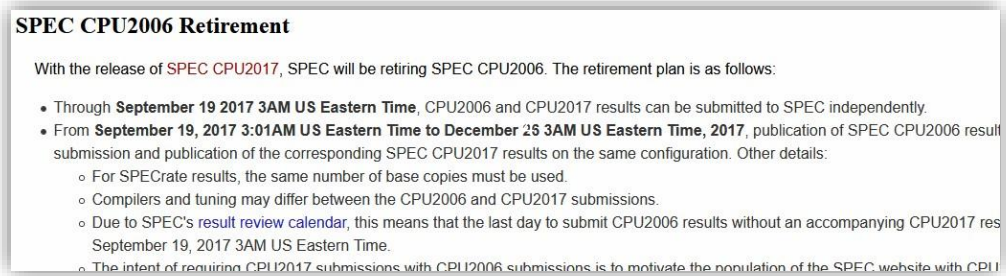
Benchmarking: challenges

Concerns with HS06

- Becoming “old”: the SPEC CPU 2006 is already retired.
- License is required by the *Standard Performance Evaluation Corporation (SPEC)*.
- **The CPU architecture landscape evolved greatly in the past 20 years.**
  - 64-bit compilations, multi-thread/multi-process, vectorized instructions (AVX), etc.
- **Reported deviations between HS06 score and actual system performance.**
  - Not representative of full machine potential (and of improved experiment workloads).
- Does not work “*out of the box*” on heterogeneous environments.



D. Giordano, CHEP2018



SPEC CPU2006 Retirement

With the release of SPEC CPU2017, SPEC will be retiring SPEC CPU2006. The retirement plan is as follows:

- Through **September 19 2017 3AM US Eastern Time**, CPU2006 and CPU2017 results can be submitted to SPEC independently.
- From **September 19, 2017 3:01AM US Eastern Time to December 25 3AM US Eastern Time, 2017**, publication of SPEC CPU2006 result submission and publication of the corresponding SPEC CPU2017 results on the same configuration. Other details:
  - For SPECrate results, the same number of base copies must be used.
  - Compilers and tuning may differ between the CPU2006 and CPU2017 submissions.
  - Due to SPEC's result review calendar, this means that the last day to submit CPU2006 results without an accompanying CPU2017 res September 19, 2017 3AM US Eastern Time.
  - The intent of requiring CPU2017 submissions with CPU2006 submissions is to motivate the population of the SPEC website with CPU

https://www.spec.org/cpu2006/

Benchmarking: challenges

Heterogeneous environments:

- *Increasing GPU landscape* → *How we benchmark and procure these resources (with HS06)?*
- *ARM based architectures* → *Can we rely on the result on this architecture? (HS06 not natively supported)*
- *HPC sites, Commercial Clouds* → *Can we benchmark these resources without license constraint*
- *Complex systems* → *Can we benchmark the full system (CPU + GPU?*

In short…

Scenarios	HS06
x86 CPUs (2010-2021)	✓
New CPU models and/or architectures	?
New Experiment SW	?
CPU + GPU/FPGA/Other	X
License	Needed by SPEC

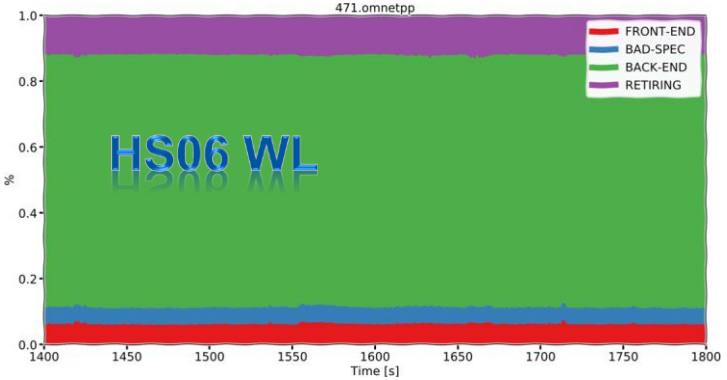
Quantitative comparison with WLCG workloads

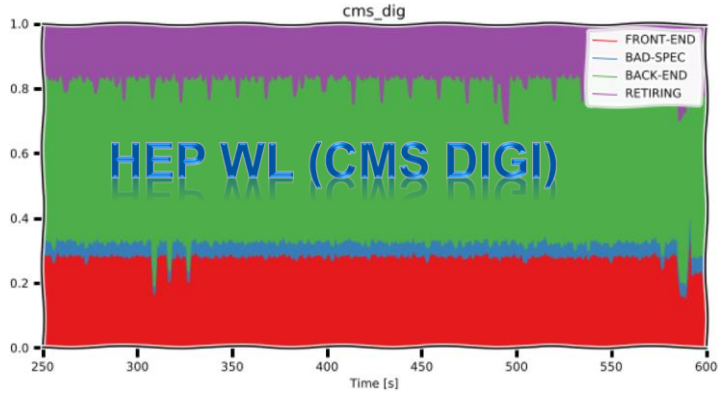
- Unveil the **dissimilarities** between HEP workloads and the SPEC CPU benchmarks
  - Using the **Trident** toolkit
    - analysis of the hardware **performance counters**

Characterization of the resources utilised by a given workload

Percentage of time spent in

- **Front-End** – fetch and decode program code
- **Back-End** – monitor and execution of uOP
- **Retiring** – Completion of the uOP
- **Bad speculation** – uOPs that are cancelled before retirement due to branch misprediction



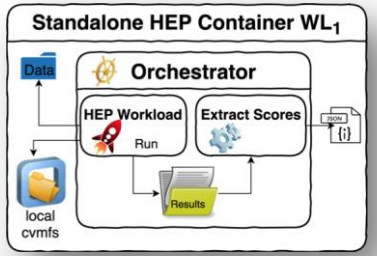


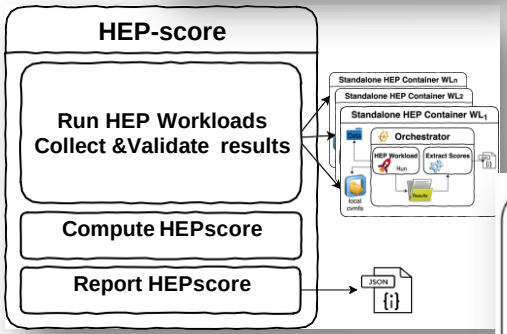
HEP Benchmarks project

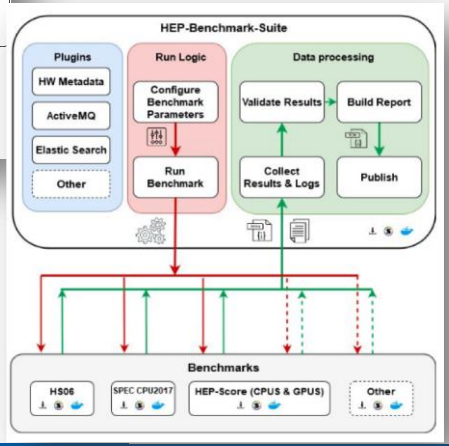
Three main components being developed since ~2 years

!!! Released under GPLv3 licence !!!

- *HEP Workloads*
  - Individual reference HEP workloads
  - Common build infrastructure
- *HEP Score*
  - Orchestrate the run of a series of HEP workloads
  - Compute the HEPscore value
  - Report whole set of WL results
- *HEP Benchmark Suite*
  - Meta-orchestrator of multiple benchmark suites
    - HEPscore, HS06, SPEC CPU2017…







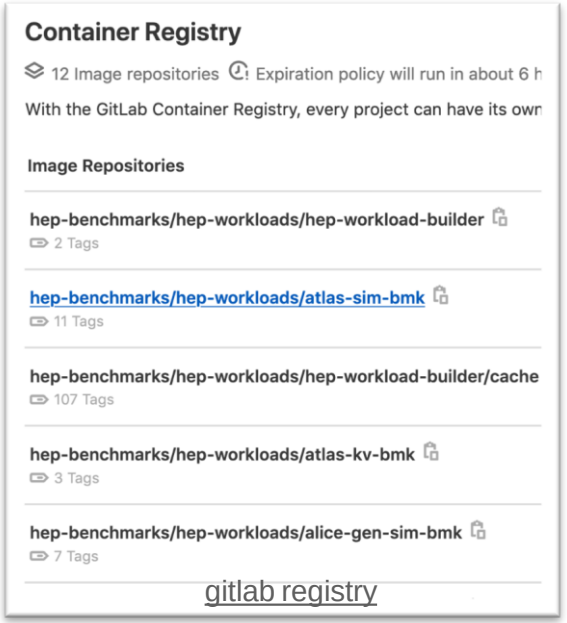
HEP Workloads

- Standalone containers encapsulating all and only the dependencies needed to run each workload as a benchmark
  - Gen, Sim, Digi, Reco workloads are available as container images
  - Report results in a structured json format with rich metadata set (see next slides)

- Standalone docker containers available in gitlab registry and
  - *!!NEW!!* distributed via CVMFS
    /cvmfs/unpacked.cern.ch/gitlab-registry.cern.ch/hep-benchmarks/hep-workloads
  - Run a given workload via a single command line:

    >singularity run $IMAGE_PATH <args>

    >docker run $IMAGE_PATH <args>



gitlab registry

HEP Workloads: Status

- Included new CPU Experiments workloads
  - BelleII (done), Atlas sim MT (in progress)
  - To come: Dune, gravitational waves, WeNMR, …
- First GPU workload containerized
  - SimpleTrack (LHC simulation)
  - Multi-GPU container workloads (Nvidia, AMD, Intel...)
- Plan:
  - Include the Run-3 CPU workloads proposed in the WLCG HEP Score Task Force
    - Support multiple architectures (e.g. ARM) as long as Experiments' software has been ported
  - Integrate other GPU workloads: next CMS Patatrack (HLT Track reco), MC Madgraph

Summary of currently supported HEP workloads

Experiment	Name	Description	Experiment license	Latest Container	Readiness	Pipeline status
Alice	gen-sim	link	GNU GPL v3	docker	w.i.p.	pipeline passed
Atlas	gen	link	Apache v2	docker	Y	pipeline passed
Atlas	sim	link	Apache v2	docker	Y	pipeline passed
Atlas	digi-reco	link	Apache v2	docker	w.i.p.	pipeline passed
CMS	gen-sim	link	Apache v2	docker	Y	pipeline passed
CMS	digi	link	Apache v2	docker	Y	pipeline passed
CMS	reco	link	Apache v2	docker	Y	pipeline passed
LHCb	gen-sim	link	GNU GPL v3	docker	Y	pipeline passed
Belle2	gen-sim-reco	link	GNU GPL v3	docker	Y	pipeline passed

https://gitlab.cern.ch/hep-benchmarks/hep-workloads

HEP Score v1.2 : The software framework

- Several new features included in this new release
  - Singularity and Docker engines are both supported, forced user namespace too
  - Access of cvmfs unpacked images
  - Better handling of disk space, configurable cleanup of the working directory
  - Optimised the report structure, allows retries in case of run failures
  - Improved CI tests
  - Configurable weighted geometric mean for the HEP workloads
  - Python wheels available: useful for installations in sites with limited external connectivity
- To install & Run: documentation at https://gitlab.cern.ch/hep-benchmarks/hep-score
- Main developers: C. Hollowell, C. van der Laan, D. Southwick

HEPscore20POC:

- HEP Score tool is configurable
  - Config: list of workloads to run, reference scores & weights, settings
- HEP Score pkg distributed with a default config file:
  - POC → Proof of concept
  - Includes the most stable workload tested on CPU up to 256 core
- **We use this to make comparison with other benchmark or with the individual workloads**
- Other config files can be used and passed to the tool
  - Convenient to include new workloads and perform studies
  - Each config is associated to a unique ID in the final report
- The **official** config will be defined by the WLCG HEP Score Task Force
  - After inclusion of Run3 workloads and performance study

hepscore2X_0.8.yaml 1.45 KB

```yaml
hepscore_benchmark:
  benchmarks:
    atlas-gen-bmk:
      results_file: atlas-gen_summary.json
      ref_scores:
        gen: 384
      weight: 1.0
      version: v2.1
      args:
        threads: 1
        events: 200
    belle2-gen-sim-reco-bmk:
      results_file: belle2-gen-sim-reco_summary.json
      ref_scores:
        gen-sim-reco: 5.44
      weight: 1.0
      version: v2.2
      args:
        threads: 1
        events: 50
    cms-gen-sim-bmk:
      results_file: cms-gen-sim_summary.json
      ref_scores:
        gen-sim: 0.726
      weight: 1.0
      version: v2.1
      args:
        threads: 4
        events: 20
    cms-digi-bmk:
      results_file: cms-digi_summary.json
      ref_scores:
        digi: 3.58
      weight: 1.0
      version: v2.1
      args:
        threads: 4
        events: 50
    cms-reco-bmk:
      results_file: cms-reco_summary.json
      ref_scores:
        reco: 2.196
      weight: 1.0
      version: v2.1
      args:
        threads: 4
        events: 50
    lhcb-gen-sim-bmk:
      results_file: lhcb-gen-sim_summary.json
      ref_scores:
        gen-sim: 90.29
      weight: 1.0
      version: v2.1
      args:
        threads: 1
        events: 5
  settings:
    name: HEPscore2X
    reference_machine: "CPU Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz"
    registry: docker://gitlab-registry.cern.ch/hep-benchmarks/hep-workloads
    method: geometric_mean
    repetitions: 3
    retries: 1
    scaling: 355
    container_exec: singularity
```

HEPscore2X

- **The "X" will mark the year of its release.**

- **A given configuration has the workloads "frozen" in time.**

- **A WLCG Task Force was constituted.**
  - Perform benchmark studies.
  - Study the mix of workloads that will constitute the metric.

- **Goal is to ensure a smooth transition from HS06 to HEPscore2X**
  - From operators to accounting teams.

WLCG HEP-SCORE Deployment Task Force

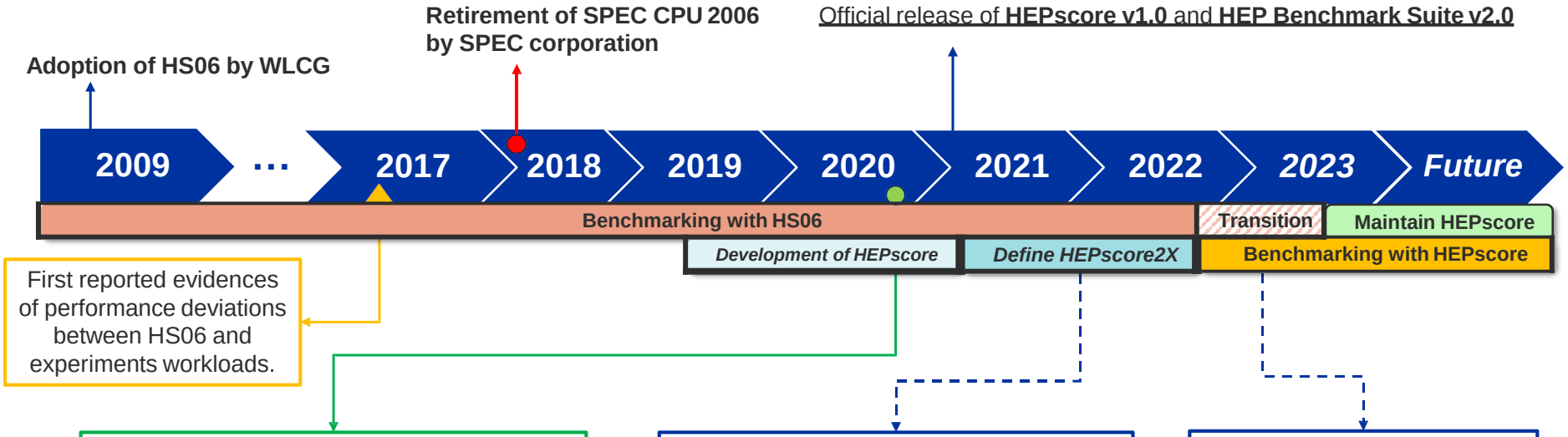
Helge Meinhard / CERN-IT
04 November 2020

https://indico.cern.ch/event/969947/

Scenarios	HS06	HEPscore
x86 CPUs (2010-2020)	✓	✓
New CPU models and/or architectures	?	✓
New Experiment SW	?	✓ (with new reference WLs)
CPU + GPU/FPGA/Other	X	(✓) (same speed definition: events/s)
License	Needed by SPEC	GPLv3

Next steps

Disclaimer: Future timeframes dates are preliminary.



Creation of WLCG HEP-SCORE Deployment Task Force

- Evaluate feasibility of replacing HS06 with HEPscore.
- Define composition of workloads of HEPscore2x.

Performance studies

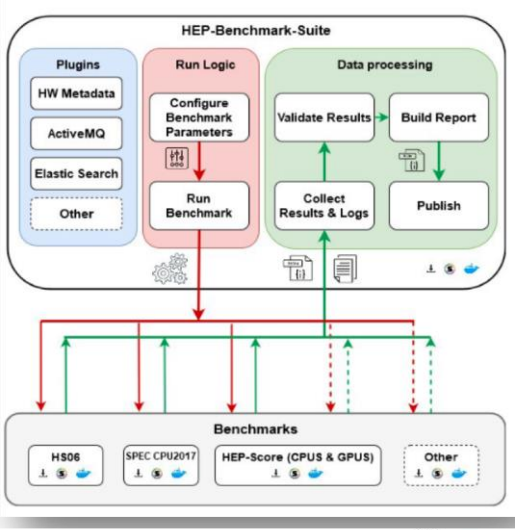
- Compare HS06, SPEC2107, SPECINT and HEPscore
- Test on various computing sites and CPU models.

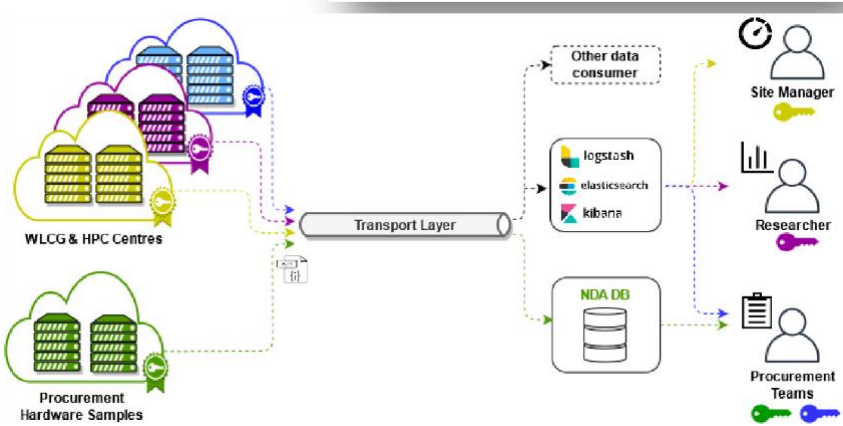
WLCG HEP-SCORE Task Force Decision

- Accounting transition

HEP Benchmark Suite v2.1

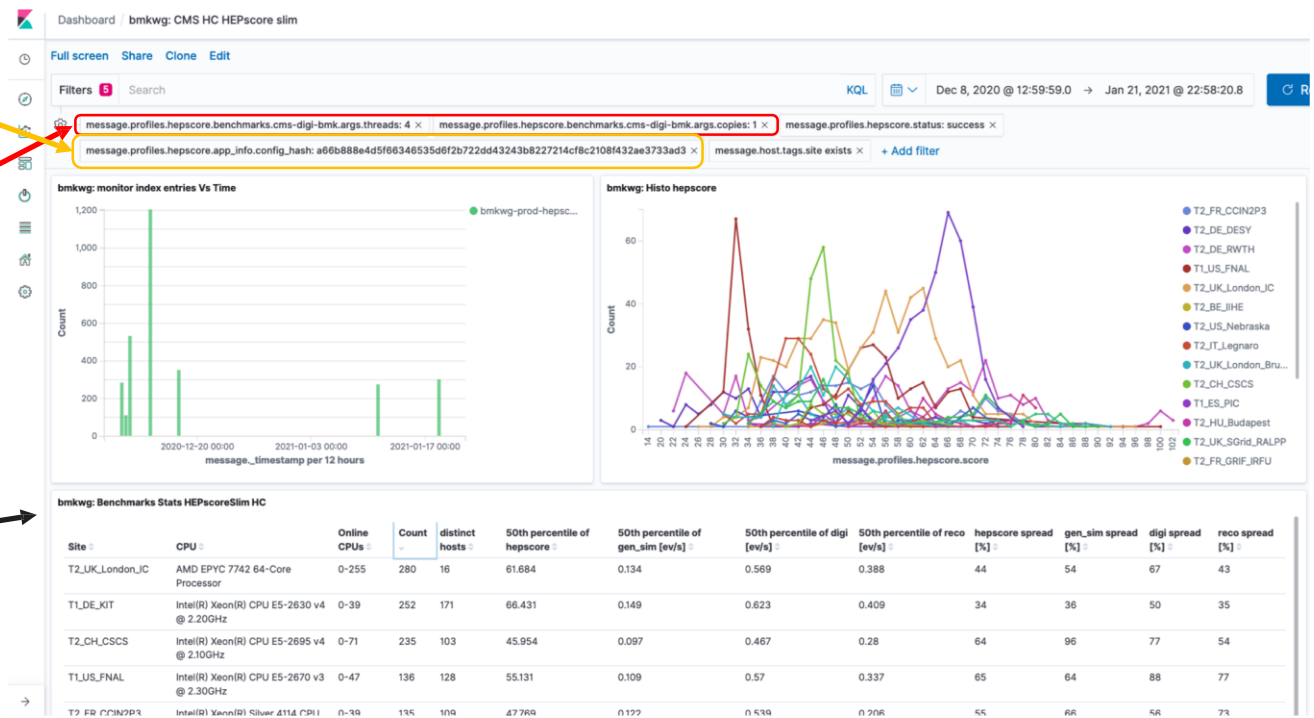
- Meta-orchestrator for the execution of several benchmarks and the publication of the suite's report
  - HS06, HEP Score, SPEC CPU 2017, …
- Features of this new version (2.0 and following)
  - Modular design, fully rewritten in python3.6+
  - Distributed via pip install, python wheels available
  - Metadata section with detailed HW information
  - Install as unprivileged user
    - HPC compatible: example of SLURM submission
    - Run also on Grid pilot jobs: see next slide
- Main developers: M. Fontes Medeiros, D. Southwick





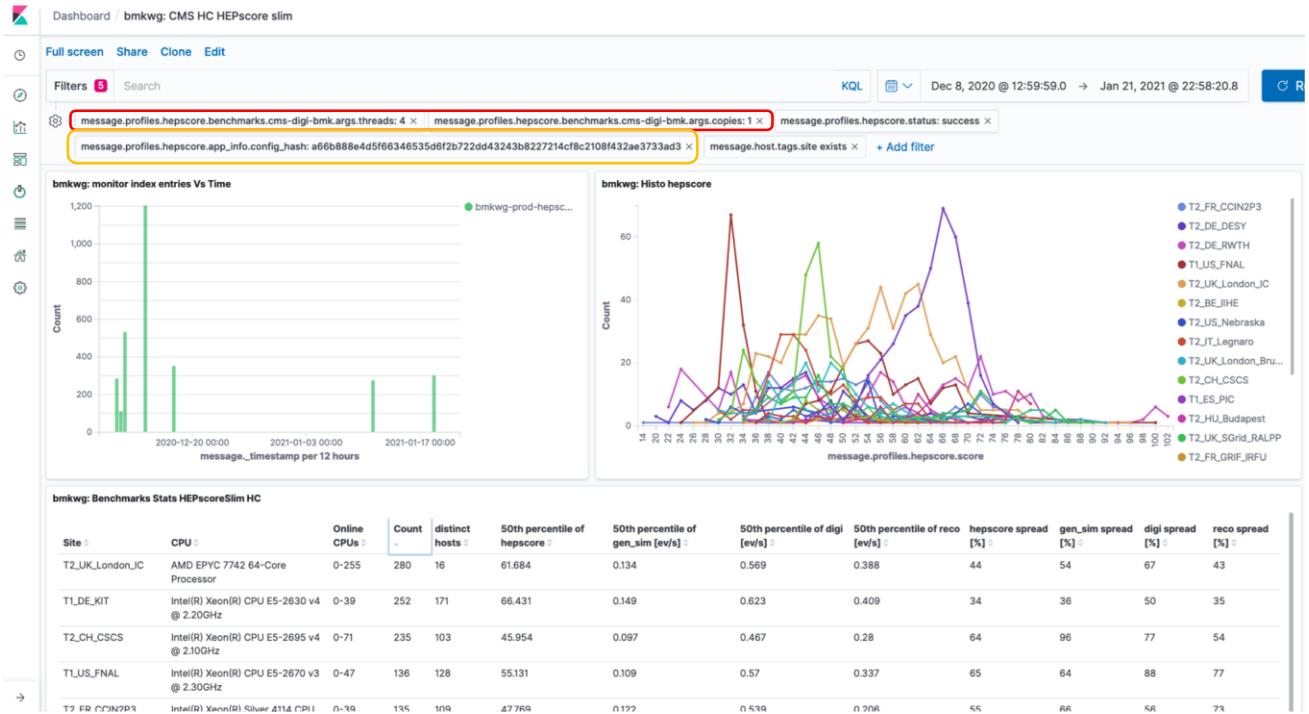
HEPscore “slim” config on grid sites

- A “slim” version of the HEPscore2X config, including only CMS gen-sim, digi, reco
- Running on 4-cores job slots
- **User job submission, glide-in singularity pilot runs**
  - Requires singularity-in-singularity (i.e. user namespaces) enabled on the grid site
- Results collected in central DB (Elasticserach) and monitored via Kibana dashboards



HEPscore “slim” config on grid sites

- We can use a “fast” version of HEPscore to benchmark the Worker Nodes
- Atlas have done something similar but we don’t have access to their results at the moment



HS06 and SPEC CPU 2017

- Make sure that HS06 and SPEC CPU 2017 can run via the Suite
  - Orchestrator scripts and libraries available in a container image, built at https://gitlab.cern.ch/hep-benchmarks/hep-spec/
- HS06
  - Config is linux_gcc_cern.cfg used by HS06 in the last decade, with few adaptations
- SPEC CPU 2017
  - Default “HEP” benchmark set to mimic HS06: Cpp-rate set of benchmarks
    - Benchmark set can be reconfigured. Eg. –b intrate will run SPEC INT 2017 rate
    - NB: All configuration changes are tracked in the reported results
  - Config similar to linux_gcc_cern.cfg, distinct for x86 and ARM

Credits

- Collective effort of several member of the HEPiX Benchmarking WG
- Weekly meeting of the HEPiX Working Group or Jira Sprint Meeting
- CHEP21 abstract
  - Plenary talk
  - to be pubblished on Computing and Software for Big Science (Springer)

HEPiX benchmarking solution for WLCG computing resources

Miguel F. Medeiros[1,*], *Manfred* Alef[2], *Luca* Atzori[1], *Jean-Michel* Barbet[3], *Ingvild* Brevik Høgstøyl[4], *Olga* Datskova[1], *Riccardo* De Maria[1], *Domenico* Giordano[1], *Maria* Girone[1], *Christopher* Hollowell[5], *Michele* Michelotto[6], *Andrea* Sciabà[1], *Tristan* Sullivan[7], *Randal* Sobie[7], *David* Southwick[1,8], and *Andrea* Valassi[1]
from HEPiX Benchmarking Working Group

[1]CERN, Geneva, Switzerland
[2]KIT, Karlsruhe, Germany
[3]Laboratoire SUBATECH, CNRS-IN2P3, Nantes, France
[4]Norwegian University of Science and Technology, Norway
[5]Brookhaven National Laboratory, USA
[6]INFN, Istituto Nazionale di Fisica Nucleare, Padova, Italy
[7]University of Victoria, Canada
[8]University of Iowa, USA

Abstract. The HEPiX Benchmarking Working Group has been developing a

Conclusions

- HEP Benchmark Suite and HEP Score are ready to be tested by our community
- New HEP Workloads (mainly LHC Run3) will be made available during 2021
- Need volunteer sites to run the Suite and benchmark several CPU models
  - This will permit the studies recommended by the WLCG HEP Score Task Force
- Feedback and support questions are welcome in the HEP Benchmarks Project Discourse Forum


INFN

Backup Slides

HS06 for ARM CPUs

```
We do not appear to have working vendor-supplied binaries for your
architecture.  You will have to compile the tool binaries by
yourself.  Please read the file
   SPEC_CPU2006_v1.2/Docs/tools-build.html
for instructions on how you might be able to build them.
Please only attempt this as a last resort.
```

- Enable support for ARM
  - Multi-architecture container
    - gitlab-registry.cern.ch/hep-benchmarks/hep-spec/hepspec-cc7-multiarch:v2.0
  - SPEC CPU 2017 already supports ARM cpus. Only CPU model needs to be changed when running on ARM
  - HS06 too old to support natively ARM cpus: SPEC 2006 tooklit needed to be built to work
    - Build the toolkit following instructions https://www.spec.org/cpu2006/Docs/tools-build.html after patching some old code
      - Patch procedure available https://gitlab.cern.ch/hep-benchmarks/hep-spec/-/tree/master/patch_SPEC2006

- NB: patching the toolkit is one time operation
  - The toolkit is then included in the *tool/bin* area
    Re-creating an archive for SPEC CPU 2006 allows to use it in any other aarch64 machine

- NB: HS06 for ARM only supported at 64 bits

Running HS06 on ARM

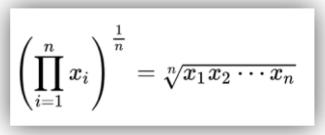
AWS Graviton2 bare-metal server benchmarked using the hepspec multi-architecture container (see previous slide)

```
$ echo ${SECRET_URL} | /hep-spec/scripts/hep-spec.sh -w $BMK_RUNDIR -b $BMK -m $BMK_OPTION -p ${SPEC_DIR} -u -
############################
         CERN HEPSPEC
Sat Feb 27 17:16:00 UTC 2021
############################
2021-02-27T17:16:00 [/hep-spec/scripts/hep-spec.sh] Variable values:
       HEPSPEC_SOURCEDIR=/hep-spec/scripts
       HEPSPEC_BMK=hs06
       HEPSPEC_NUMPROC=64
       HEPSPEC_PATH=/scratch/HEPSPEC/CI_hs06_ext_g2_bare_12384842
       HEPSPEC_SET=all_cpp
       HEPSPEC_MACHINE_OPTION=default
       HEPSPEC_ITERATIONS=3
       HEPSPEC_WORKDIR=/scratch/jobs/hs06_ext_g2_bare_12384842/hep-spec
       HEPSPEC_DEBUG=0
```

```
$ lscpu
Architecture:                    aarch64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
CPU(s):                          64
On-line CPU(s) list:             0-63
Thread(s) per core:              1
Core(s) per socket:              64
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       ARM
Model:                           1
Model name:                      Neoverse-N1
Stepping:                        r3p1
BogoMIPS:                        243.75
L1d cache:                       4 MiB
L1i cache:                       4 MiB
L2 cache:                        64 MiB
L3 cache:                        32 MiB
NUMA node0 CPU(s):               0-63
```


```
{"hs06":{"start":"Sat Feb 27 17:19:26 UTC 2021", "end":"Sat Feb 27 20:10:46 UTC 2021", "copies":64,
            "runcpu_args":"1 runspec: runspec --define machine_option:64 --config=linux_gcc_cern.cfg --action=build all_cpp;64 runspec: runspec --define machine_option:64 --config=linux_gcc_cern.cfg --nobuild --noreportable --iterations=3 all_cpp", "bset":"all_cpp", "LINK":"  6  g++ -O2 -fPIC -pthread  -DSPEC_CPU_LP64    <objects>     -o options;  1  g++ -O2 -fPIC -pthread  -DSPEC_CPU_LP64 -DSPEC_CPU_LINUX   <objects>     -o options;", "hash":"7b84bb375cee11731a958a26d6fc155d",
          "score":1170.998, "avg_core_score" : 18.296, "num_bmks":7 ,"bmks":{ "444.namd":[  23.5, 23.5, 23.5, 23.5, 23.5, 23.5, 2
```

Benchmark comparing “speed factors”



https://en.wikipedia.org/wiki/Geometric_mean

- In order to compare servers HS06 and HEP-Score implement the geometric mean approach. Needs:
  - a set of reference workloads (WLs)
  - a measure of performance per WL (m_i), that typically goes as [1/s] (eg. can be the event throughput)
  - a reference machine
- The score S of a server (srv) is defined as the geometric mean of the speed factors $x_i(srv,ref) = m_i(srv)/m_i(ref)$ respect to the reference machine (ref)
  - i.e. “speed” is *normalised* respect to the reference machine “speed”
- The relative score between srv_A and srv_B is the ratio of the scores S(srv,ref) , this is still a geometric mean of speed factors

	WL_1		WL_2		WL_n		Score	S(A,B)
Ref. Srv	m_1(ref)	1 (by def)	m_2(ref)	1 (by def)	m_n(ref)	1 (by def)	$\left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}}$	
Srv A	m_1(A)	x_1(A,ref)	m_2(A)	x_2(A,ref)	m_n(A)	x_n(A,ref)	S(A,ref)	!(#,%&')
Srv B	m_1(B)	x_1(B,ref)	m_2(B)	x_2(B,ref)	m_n(B)	x_n(B,ref)	S(B,ref)	!(),%&')