EPJ Web of Conferences **214**, 08011 (2019) https://doi.org/10.1051/epjconf/201921408011
CHEP 2018

Next Generation of HEP CPU Benchmarks

Domenico Giordano[1,4,*], *Manfred* Alef[2,4], and *Michele* Michelotto[3,4]

[1]CERN
[2]Karlsruhe Institute of Technology (KIT)
[3]INFN Padova
[4]on behalf of the HEPiX Benchmarking Working Group

Abstract.
As of 2009, HEP-SPEC06 (HS06) is the benchmark adopted by the WLCG community to describe the computing requirements of the LHC experiments, to assess the computing capacity of the WLCG data centres and to procure new hardware. In the recent years, following the evolution of CPU architectures and the adoption of new programming paradigms, such as multi-threading and vectorization, it has turned out that HS06 is less representative of the relevant applications running on the WLCG infrastructure. Meanwhile, in 2017 a new SPEC generation of benchmarks for CPU intensive workloads has been released: SPEC CPU 2017. This report summarises the findings of the HEPiX Benchmarking Working Group in comparing SPEC CPU 2017 and other HEP benchmarks with the typical WLCG workloads mixes.

1 Introduction

Since its foundation in 1991, the HEPiX forum brings togethers worldwide IT professionals from the HEP laboratories and institutes [1], with the purpose of sharing experience on their common computing challenges. In 2006 a dedicated working group, the HEPiX Benchmarking Working Group (here BWG), was established to cover the CPU benchmarking topic and, in particular, to address the identified discrepancies between the performance of the HEP applications and SI2K [2], the standard benchmark used at the time to measure the CPU performance. In 2009 the BWG suggested the adoption of a new HEP specific benchmark suite, HEP-SPEC06 (HS06) [3], based on a subset of the industry standard benchmark SPEC CPU2006 [4]. Since then HS06 has been used to describe experiment requirements, lab commitments, existing compute resources and specifications to procure new hardware. Therefore HS06 measurements of any CPU and motherboard models deployed in any WLCG [5] site has been collected [6].

Almost ten years after the review of SI2K, in 2016 evidences of discrepancies between HS06 and the HEP workloads were reported by some of the LHC collaborations [7]. In addition the adoption of public and private cloud resources generated interest in a family of benchmarks that could run much faster than HS06 while keeping adequate correlation with the HEP workloads. This situation revived the activity of the BWG.

The next sections describe the state of the art of the BWG's activities, the achieved results, and the future plans.

*e-mail: domenico.giordano@cern.ch

2 HS06 overview

HS06 is composed by the SPEC CPU2006 benchmarks based on C++ code, named *cpp_all* benchmark set. This set showed a good correlation with all the major HEP experiments' applications considered ten years ago [3]. The main explanation was that it has a mixture of integer and floating point instructions similar to the patterns observed for the experiments' applications. Yet some noticeable differences exists: with respect to SPEC CPU 2016 the running mode is not one of the two proposed defaults (namely *rate* and *speed*). The HS06 running mode is an adaptation of the *speed* mode, modified to run on as many parallel and independent instances as are the available threads of the CPU. For this reason it is called *multiple speed*. Other differences with respect to the original benchmark reside on the compiler optimization flags, as the binaries are 32-bits.

In recent years, with the broad adoption of Intel Haswell and Broadwell CPU models, as well with the adoption by all the LHC experiments of 64-bits compiled applications, discrepancies have been highlighted between the HS06 scores and the average normalised time spent by the applications [7]. The discrepancies being more accentuated for the LHCb simulation applications. At the same time analysis on ATLAS and CMS WLCG jobs still proved that simulation applications were in good agreement with HS06 within 10% of resolution [8].

Moreover, with the advent of Simultaneous MultiThreading (SMT), the WLCG sites have configured their computing infrastructure to run a number of simultaneous applications per physical core spanning from one to two, driven by the WLCG applications' memory requirement of at least 2 GB of memory available per processor. The BWG's studies have highlighted the lack of a common scaling factor across WLCG applications and HS06, when comparing the delivered performance of a different number of simultaneously running applications (fig. 1).

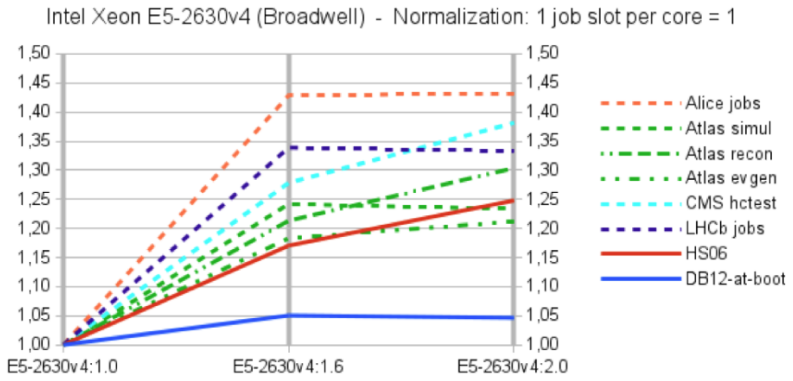


Figure 1. HS06 score and typical WLCG applications' performance reported by Alice, ATLAS, CMS, and LHCb for three values (1, 1.6, 2) of simultaneously running applications per core. The benchmarked platform consists of several servers with Intel Xeon E5-2630 v4 processors at GridKa Tier-1 (KIT).

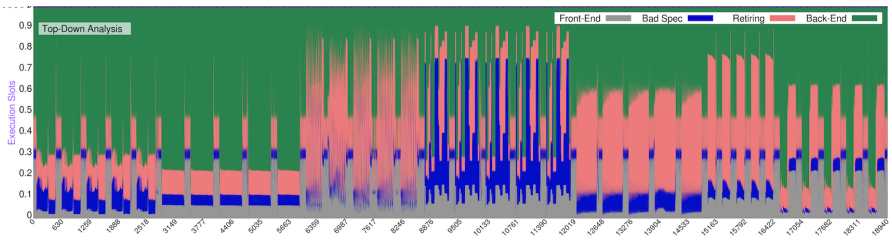
From the initial round of discussions and results, two aspects became evident, both requiring the adoption of new tools. Firstly it was necessary to extend the studies to the CPU microinstructions, in order to compare at a fine-grained level the difference between the HEP workloads and HS06 (see Sec. 2.1). Second there was a need to perform the benchmark studies in a more reproducible manner, to avoid some of the common pitfalls that are also faced by other communities [9] (see Sec. 2.2).

2.1 Trident analysis

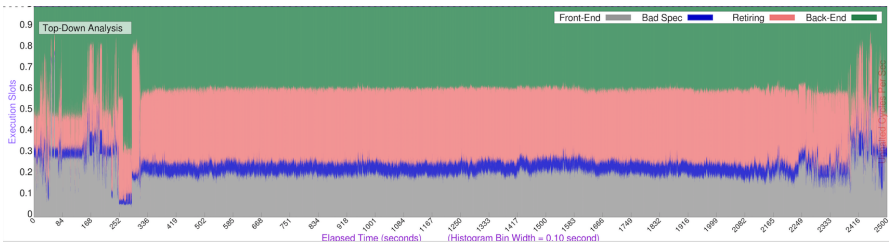
The Trident tool [10] collects and analyses hardware and software counters while a given workload is running. It leverages the performance monitoring unit (PMUs) of modern processors to quantify the usage of the processing units, memory and IO subsystems.

Trident enables the unveiling of differences between the HEP workloads and HS06. For example Trident can show the classification of notional execution slots between used slots and unused slots, i.e. when execution opportunity is lost. Losses can happen within the front-end or back-end portion of the instruction pipeline due to availability or limits in resources, such instruction decodes latency or throughput, execution port availability and latency of memory access. In addition a loss may be classified as bad speculation when due to branch prediction.

Figure 2 shows this comparison between HS06 and the ATLAS digitization and reconstruction sequence. In the case of HS06, the transitions of patterns correspond to the transition between two *cpp_all* benchmarks running in that particular phase of the suite (fig. 2(a)). Similar remarks apply to the ATLAS case (fig. 2(b)). Differences are also found in the amount of memory transactions and bandwidth usage. A detailed report about the Trident tool and the peculiarities of the HEP workloads is available [11].



(a)



(b)

Figure 2. Fraction of micro-operations per cycle that are usefully executed (*retiring*) or lost due to reasons belonging to the *front-end* or the *back-end* portion of the instruction pipeline or lost due to *bad spaculation*. Two different applications are compared: the HS06 suite (a) and the ATLAS digitization and reconstruction workloads (b).

2.2 Benchmarking suite

One of the efforts of the BWG has been to improve the measurement's reproducibility and to simplify the process of running benchmarks and collecting results. This goal has been achieved by developing a toolkit, named *CERN benchmark suite* [12], that allows to plug-in any desired benchmark and to run a configurable sequence of benchmarks on a given computing resource. The current list of supported benchmarks includes HS06, SPEC CPU 2017 [13], KV [14], Dirac Benchmark 2012 (DB12) [15] and Whetstone benchmark (WSN) [16] . Another benefit of the suite is the unified report generated after it runs. The report, in *JSON* format, consists of a metadata section, with information about the server benchmarked, the running conditions, and of a data section with a hierarchical structure of benchmark results (fig. 3).

The suite is used at CERN for continuously benchmarking each CPU model running in the production infrastructure. It has speedup the measurement collection and analysis process. This is typically useful when prompt and accurate measurements are expected as, for example, the performance assessment for the mitigation measures related to the Spectre [17], Meltdown [18] and Foreshadow [19] vulnerabilities. The benchmarking suite is also distributed via *docker* containers to even simplify further installation and running processes. Recently it has been used by other site managers in WLCG.

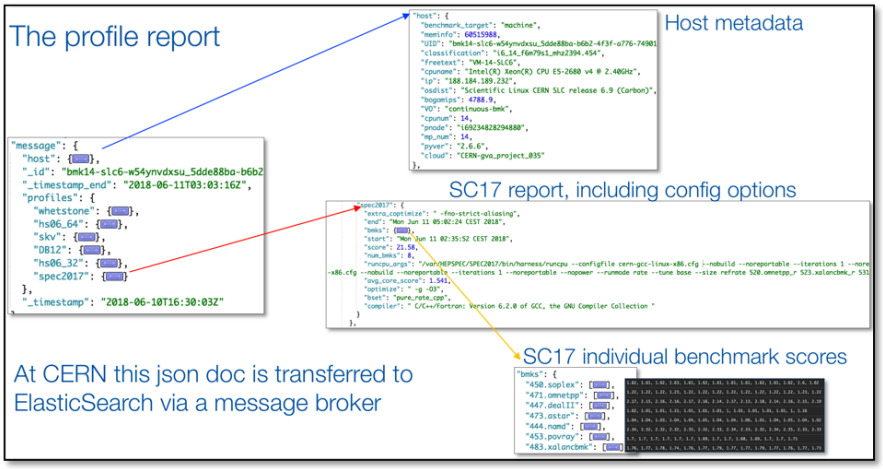


Figure 3. Structure of the profile report produced by the Benchmarking suite, here in the case of SPEC CPU 2017 (SC17).

3 Fast benchmarks

Fast benchmarks require a few minutes to run instead of the several hours required by HS06. They are useful when the environment dynamic and the relatively short availability of computing resources make a long-running benchmark as HS06 impractical. Fast benchmarks are useful to estimate the delivered performance in commercial cloud and HPC opportunistic resources, and are helpful to forecast the duration of a given grid job in a WLCG worker node.

Among several applications studied by the BWG, the ATLAS KV and the DB12 are appreciated by the WLCG collaborations. KV tends to agree well with ATLAS and CMS simulation workloads, DB12 (and the variant DB16) is in agreement with the Alice and LHCb job performance when DB12 runs inside the job. More discrepancies are seen when the DB12 measurement is performed following the same procedure as other benchmarks, i.e. when running as many parallel copies as there are hardware threads [20]. Evidence has been collected that shows short benchmarks are not robust enough to replace long-running benchmarks when the scope of the measurement is to evaluate effects of the branch misprediction rate or the kernel patches for the Meltdown and Spectre vulnerabilities.

4 SPEC CPU 2017

The SPEC CPU 2017 (SC17) [13] benchmark suite has been released on June 2017 by the SPEC organisation, and introduced this new suite as the replacement of SPEC CPU 2006. SC17 is larger and has a more complex codebase, with respect to its predecessor. It is shaped for multi-core and multi-threads applications. SC17 includes 43 individual benchmarks organised into four suites, obtained combining integer and floating point applications with two distinct running modes, *speed* and *rate*.

As done for HS06 ten years ago, the BWG is performing a detailed investigation of SC17, starting from the comparison with HS06. For this purpose only comparable configurations have been considered. The benchmarks under examination have been restricted to the C++ [1], and the selected running mode is *rate*, but with a number of spawn copies fixed to one. In order to mimic the *multiple speed* running mode of HS06, parallel copies of SC17 are spawned by an orchestrator script. These choices [21] allow to have the largest overlap with the application area of the HS06 benchmarks. For the same reasons the HS06 compiler optimisation flags are retained [2].

On this basis, the first objective has been to quantify how much SC17 differs from HS06 on the same hardware. Next objective has been to evaluate if all the individual benchmarks in the SC17 are independent or if, on the contrary, a subset of the benchmarks can be used to build a representative benchmark mix.

The measurements have been performed on several servers, covering seven different Intel CPU models, from four different product families (Ivy Bridge, Haswell, Broadwell, Skylake), and two AMD models (Opteron and Ryzen). The SMT was enabled in all the servers, and the tests have been performed to profile the full physical node. Virtual Machines of different CPU size have been used, running synchronously the benchmarks, so that the server load would be analogous to what is achieved running on a physical server with either the SMT enabled or disabled.

The following benchmarks have been collected for each server: SC17 and HS06 compiled at 64 bits, as well as HS06 compiled at 32 bits, as this the standard benchmark for WLCG.

A high linear correlation (0.975) between SC17 and HS06 has been found (fig. 4), with a scale factor of 0.12 to convert the $HS06_{64bits}$ score to the SC17 score. The residual ratio of the linear fit shows that the extrapolation is within 5% of the measured value, i.e. $|\frac{SC17_{extrapolated}}{SC17_{measured}} - 1| < 5\%$.

This result highlights that the adopted SC17 benchmark set, *pure_rate_cpp*, does not provide more information than HS06, at least for the range of CPU models currently adopted

[1] A dedicated benchmark set has been defined (*pure_rate_cpp*), consisting of the following benchmarks: *508.namd_r, 510.parest_r, 511.povray_r, 520.omnetpp_r, 523.xalancbmk_r, 526.blender_r, 531.deepsjeng_r, 541.leela_r*.

[2] gcc compiler flags: *-O3 -fPIC -pthread*

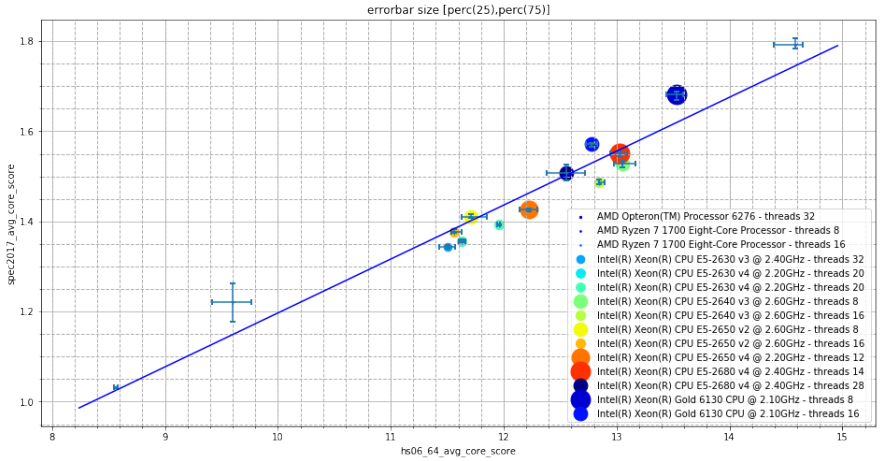
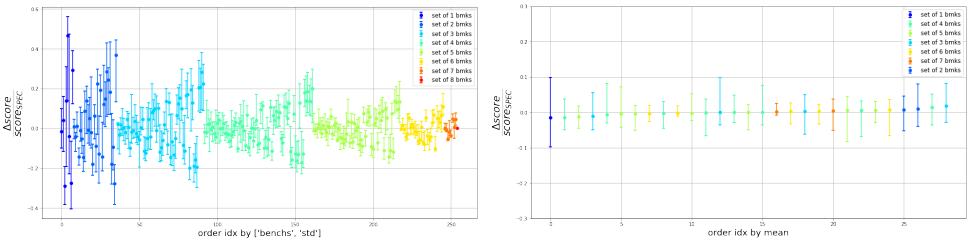


Figure 4. SC17 score vs $HS06_{64bits}$ score, both normalised to the core count. The error bars represent the 5% and 95% of the distribution for each given CPU model and VM core count. The marker size is proportional to the amount of data collected.

in WLCG. Therefore different scaling factors with respect to the HEP workloads are not expected. We stress here that the initial focus has been on running conditions that remain similar to HS06 and to the current HEP workloads.

The second objective has been to evaluate if all the individual benchmarks in the SC17 *pure_rate_cpp* are independent, or if there is at least one subset of this set that is as well representative of the performance of the full set. All the combinations of the eight benchmarks have been built, for a total of $\sum_{k=1}^{7}\binom{8}{k} = 254$ combinations.

For each combination the SC17 score has been re-evaluated using only the individual scores of that combination. This has been possible thanks to the benchmark suite that stores



(a) (b)

Figure 5. Residual ratio $\frac{\Delta_{\text{score}}}{\text{SC17}_{\text{score}}}$ with respect to the original SC17 *pure_rate_cpp* for each of the subsets built from the eight *pure_rate_cpp* benchmarks' scores (a) and for only the subsets compatible with the original SC17 within three sigmas (b). Error bars represent the full range [min,max]. The x value is the subset identifier.

the individual results of each benchmark and thread. From the sample of new scores the weighted average and standard deviation have been determined for each given CPU model and running condition. Afterword the residual ratio $\frac{\Delta_{\text{score}}}{\text{SC17}_{\text{score}}}$ with respect to the initial SC17 *pure_rate_cpp* has been computed. The new scores have a residual ratio that span from less than 1% to more than 50% (fig. 5). Therefore it is possible to identify more than one subset being well in agreement with the original SC17 score, using just two or three benchmarks (fig. 5(b)). As an example, the subset *508.namd_r, 510.parest_r, 531.deepsjeng_r* is compatible with the original SC17, being the residual ratio within the range [-0.035, 0.017] with average of -0.001 ± 0.005.

Having shown that the individual SC17 *pure_rate_cpp* benchmarks are not fully independent, it is possible to use one of those subsets of benchmarks that give the same score ratio, and consequently it is possible to reduce the running time of the suite without loosing in sensitivity. The current running time is on average 2.5 hours/iteration, and scale inversely with the score of a given CPU model. The next phase of the study will evaluate how each of those subsets correlates with the typical HEP job mix.

5 Requirements for a benchmark suite

The software and computing challenges for the HEP field, in particular due to the HL-LHC program [22], have implications on the benchmarking side. The selection of future CPU benchmarks shall take into account the foreseen evolution of the experiment software in order to better use the CPU resources. This implies that new HEP applications could and should in the future perform differently in distinct CPU models with respect to the past applications, and consequently should break the current linearity with HS06.

In this scenario, instead of looking for a third party benchmark suite that must represent the HEP applications, a suite of benchmarks built from the HEP applications themselves can represent the appropriate solution to guarantee the required correlation with the performance of the applications' mix running in WLCG.

In order to build such suite, some requirements need to be fulfilled. First of all the suite should be easy to run, stable and accessible also by professionals external to the WLCG collaborations, such as hardware vendors and site procurement teams. This implies that no assumption should be made on the remote accessibility of the input data and the condition databases. In addition the long-term support of the suite is required, including its evolution to cope with major changes in the experiments' software. Conditions such as strict version control, tamper-proof repository, checksums and clear license statement should be guaranteed. This is feasible with the technologies and software practise already adopted, for example with cvmfs for library access, containers for isolation of a workload, and with continuous integration procedures. But it is only possible if the WLCG community ensures the long term support.

6 Conclusions

HS06 is a decade old suite used to benchmark CPU resources for WLCG. Its adoption spans from the hardware vendors, to the site managers, funding agencies and software experts. It is stable, reproducible, accurate, however it is reaching the end of its life. Initial hints of lack of correlations with the HEP applications have been collected. Looking for suitable alternatives the HEPiX Benchmarking Working Group has evaluated SPEC CPU 2017 and a number of fast benchmarks.

The studies done so far do not show major advantage in adopting SPEC CPU 2017 with respect to HS06. Indeed the suite of C++ benchmarks reports scores highly correlated with

the HS06 scores. Fast benchmarks can play a role in cloud contexts, where re-benchmarking is required, but the current proposed fast benchmarks cannot replace the accuracy of HS06, in particular in the procurement and accounting tasks.

A suite based on the workloads that HEP experiments run can be an alternative to industrial standard benchmarks. The adoption by the experiments of modern software development techniques simplifies the ability to package, distribute and maintain a field specific benchmark suite. The HEPiX Benchmarking Working Group is actively working to make this possible.

References

[1] *HEPiX forum*, www.hepix.org
[2] *SPEC CINT 2000*, https://www.spec.org/cpu2000/CINT2000/
[3] M. Michelotto et al., J. Phys. Conf. Ser. **219**, 052009 (2010)
[4] J.L. Henning, SIGARCH Computer Architecture News **34**, 1 (2006)
[5] *Welcome to the Worldwide LHC Computing Grid*, http://wlcg.web.cern.ch/
[6] *Benchmarking Working Ggroup*, https://w3.hepix.org/benchmarking.html
[7] P. Charpentier, Journal of Physics: Conference Series **898**, 082011 (2017)
[8] *Cpu benchmarking with production jobs*, https://indico.cern.ch/event/651342/contributions/3024516/subcontributions/256445
[9] G. Shapira, Y. Chen, IEEE Transactions on Services Computing **9**, 152 (2016)
[10] *Trident gitlab repository*, https://gitlab.cern.ch/UP/Trident
[11] S. Muralidharan et al., to appear in CHEP 2018 proceedings, EPJ Web of Conferences (2019)
[12] *Cern Benchmark suite: gitlab repository*, http://cern.ch/go/77JV
[13] *SPEC CPU 2017*, https://www.spec.org/cpu2017/
[14] A.D. Salvo, F. Brasolin, Journal of Physics: Conference Series **219**, 042037 (2010)
[15] *Dirac benchmark 2012*, gitlab.cern.ch/mcnab/dirac-benchmark/tree/master
[16] H. Curnow, B. Wichman, Computer Journal **19**, 43 (1976)
[17] P. Kocher et al., *Spectre Attacks: Exploiting Speculative Execution*, in *40th IEEE Symposium on Security and Privacy (S&P'19)* (2019)
[18] M. Lipp et al., *Meltdown: Reading Kernel Memory from User Space*, in *27th USENIX Security Symposium (USENIX Security 18)* (2018)
[19] J. Van Bulck et al., *Foreshadow: Extracting the Keys to the Intel SGX Kingdom with Transient Out-of-Order Execution*, in *27th USENIX Security Symposium (USENIX Security 18)* (Baltimore, MD, 2018), p. 991–1008
[20] *Status report from Benchmarking Working Group*, https://indico.cern.ch/event/609911/contributions/2620190/
[21] *Cern Benchmark suite: gitlab repository*, http://cern.ch/go/tK6D
[22] A.A. Alves, Jr et al. (2017), 1712.06982