Speeding up CMS simulations, reconstruction and HLT code using advanced compiler options

Niccolò Forzano[1,2],

Danilo Piparo[1], Malik Shahzad Muzaffar[1], Vincenzo Innocente[1] and Vladimir Ivantchenko[1,3]

[1] CERN, Geneva, Switzerland
[2] Università di Milano-Bicocca, Italy
[3] Princeton University, New Jersey, U.S.A.

Abstract. The CMS simulation, reconstruction, and HLT code have been used to deliver an enormous number of events for analysis during Runs 1 and 2 of the LHC at CERN. In fact, these techniques have been regarded as of fundamental importance for the CMS experiment. In this paper several ways of improving the efficiency of these procedures will be described and it will be displayed how no particular conceptual or technical blocker has been identified in their implementation. In this framework, particular attention will be devoted to highlight how CMS simulation, reconstruction and HLT will gain a considerable increase in speed recompiling several CMS sub-libraries using advanced compiler options. In fact, using this methodology,the compiler will be leveraged to obtain an up to 10% speedup. As will be shown, the focus of the reasonings reported will be on the LTO (Link Time Optimization) and PGO (Profile Guided Optimization) approaches: using these advanced tools, several results will be seen about improving the event loop time and event throughput and the differences between the profiles of the processes will be shown. Moreover, an important feature of the PGO approach will be considered: profiles obtained running events based on one process will be enough to speedup many other ones (and a profile obtained with the Phase 1 detector configuration will manage to give an improvement for Phase 2 processes too).

1. Summary of CMS data processing steps

This section is dedicated to a short description of the CMS data processing steps. The chain of steps starts with event generation, which consists in the creation of events of a certain type from a Monte Carlo Event Generator (e.g. Pythia [1], MadGraph [2], Sherpa [3]). Simulation follows and it transforms generated particles into simulated hits in the CMS detector. The tool used is Geant 4 [3], in connection with some detector specific fast simulation techniques (e.g. Russian Roulette [4] for neutrons, parametrized forward showers [5]). The next step is the transformation of sim hits into the response the detector would have had in presence of such energy depositions. In simulation, this step runs virtually always coupled to pileup mixing, which consists in the overlay a "pileup only" event to the hard scatter, reading it from a "Pileup library" which is a CMS dataset (typically placed at FNAL or CERN, accessing it through XRootD [6] remote reads). The event filtering, or High Level Trigger (HLT) step runs on data at Point 5 and can be also run offline. A differently configured, extremely fast reconstruction to decide what events are interesting and why (i.e. according to which "trigger path"). Reconstruction follows, and, as HLT, is common to simulation and data. This steop outputs collections of high level quantities,

e.g. particles like photons, electrons, muons, or jets. In order to carry out analysis, the creation of MiniAOD [7] and NanoAOD [8] samples are created after reconstruction

CMS software can go from generation to reconstruction in one single step, called "Fast Monte Carlo Chain" (or in CMS jeargon "fastsim" [9]), which is much faster than the high fidelity Geant 4 based simulation and reconstruction.

2. A quest for more throughput

The Offline Software and Computing team in CMS has the strategic goal of ensuring that there are appropriate levels of computing resources available to enable the physics program of the CMS experiment now and in the future, while using those resources efficiently with highly-performant software applications and minimizing operational effort, and completing computing requests in short, predictable amounts of time.

A computing model in which we can utilize processing (including accelerators and heterogeneous architectures, HPCs), disk and tape storage, and network in a flexible manner, all while having a unified code base, best fulfills the above goals.

Achieving more throughput in terms of events delivered per unit time delivered to the CMS analysis community is a way in which the aforementioned strategic goals are honored. Throughput can be increased with the speedup of the CMS applications, and one powerful way of increasing the runtime performance of our software is through so-called "technical optimizations". These optimizations are the ones that rely on faster mathematical libraries or compilation flags.

3. Link Time Optimization and Profile Guided Optimization

Link Time Optimization (LTO) and Profile Guided Optimization (PGO) are two examples of technical optimizations. LTO consists in letting the compiler instrument compilation units with metadata which is then consulted to optimize the building of shared objects. LTO expands the scope of inter-procedural optimizations to encompass everything that is visible at link time. PGO, unlike LTO, implies two compilation passes and one execution of the application being compiled. At first, instrumented binaries are built in order to be able to produce a profile of the application. Then, guided by the information in the profile, a re-build of the code is performed. The generated binary greatly benefits from optimizations such as inlining, block ordering, register allocation, conditional branch optimization and virtual call speculation.

In the past, techniques like PGO were studied with the objective of accelerating HEP applications [10], in this work we systematically explored the technique profiting from years of improvements of the GCC compiler and we complemented the study with measurements of the LTO otpimization.

4. Testbed machines and details about the performance measurements

To perform these studies, the main machine used was equipped with an *AMD EPYC 7302 16-Core* CPU. The compiler was GCC 10.3.0, in combination with the CMS software for data taking for the year 2022: the release *CMSSW 12_0*. The Geant 4 version was 10.7.2. All timings cited in this work are relative to the event loop, that considered several thousands of events. The pileup conditions considered were the most recent known at the time, e.g. the 2018 ones. For tests involving the simulation workflows, both Geant 4 and VecGeom [11] were rebuilt to profit from LTO and PGO. Performance improvements were measured to be identical in sequential and multithreading mode.

5. Results for the Geant 4 based simulation

For LTO, we chose the simulation of the process leading to the creation of a pair of top-anti top quarks in order to use the largest number of Geant 4 code paths possible, given the ample range

of decay modes of those heavy particles. The optimization led to a runtime reduction of 3.2 %.

For PGO, we measured that a profile generated through a certain kind of physics process could not only optimize that process but also others. We chose a representative set of standard candles, namely top-antitop pairs creation (both considering the Run 3 and HL-LHC CMS detector), minimum bias events, Z boson decaying into two muons and single electron generation. Table 1 shows a matrix with the optimization for a given profile in each row applied to all profiles in the various columns.

Table 1. Summary of the percent speedup relative to the non optimized binaries obtained with LTO and PGO combined. Processes labeling columns are optimized through the PGO profiles obtained with the process labeling the row. The number of events used to obtain the profile and verify the speedup are also cited. The highest speedups are singled out in bold.

# Events	**Running →**	**150**	**384**	**384**	**384**	**150**
Profiling ↓	Processes	TTBar	MinBias	$Z \to \mu\mu$	Single e	Phase-2 TTBar
25	TTBar	**10.7**	10.2	10.4	16.0	9.2
64	MinBias	8.9	9.5	10.8	**11.9**	9
64	$Z \to \mu\mu$	9.5	11.0	9.5	**12.0**	8.2
64	Single e	6.8	7.7	6.9	**12.6**	6.6
25	Phase-2 TTBar	7.6	8.4	7.0	8.8	**12.1**

The profile generated with the TTBar process consideting the Run 3 detector is enough to optimize for all processes, even the simulation that considers the HL-LHC CMS detector. The combined effect of LTO and PGO is quantifiable in a speedup of 10%.

6. Offline and online reconstruction

In CMS HLT and offline reconstruction share the same code base, configured differently. For example, tighter cuts are used or some algorithms are used exclusively in only one of the two reconstruction sequences. A very different set of modules is used for for Run 3 and HL-LHC processing: this reflects the upgrade of the detector. For the measurement of the performance improvement, we chose an approach inspired from simulation. We selected primary datasets (events selected by the same set of triggers) instead of a set of generator processes. In particular, we focussed on *JetHT*, containing events with high transversal hadronic activity, *MET* containing events with considerable missing transverse energy, *Zerobias* containing low occupancy events, and, in absence of HL-LHC real data, a top-antitop sample with an average pileup of 200 was considered. For the HLT emulation, the standard set of unbiased events used to profile online reconstruction were used.

The runtime reduction achieved with LTO was 1.7% for offline reconstruction and 2.7% for HLT. The combined effect of PGO and LTO sped up by 9.4% offline reconstruction and by 10.5% HLT. As observed for simulation, one flavor of events is enough to produce profile that allows to speed up the processing of all kind of physics signatures. See table 2.

6.1. Number of events necessary to build a useful PGO report

As part of this study, and with a potential future deployment of PGO in the CMS build infrastructure in mind, we studied the dependency of the application speedup on the number of events used for the profile creation. We could verify that running on a few tens of events is enough to obtain the maximum speedup achievable. Figure 6.1 quantifies this behavior for the reconstruction of the JetHT primary dataset.

Table 2. Summary of the percent speedup relative to the non optimized binaries obtained with LTO and PGO combined. Event types labeling columns are optimized through the PGO profiles obtained with the type labeling the row. The number of events used to obtain the profile and verify the speedup are also cited. The highest speedups are singled out in bold.

# Events	**Running →**	**200**	**200**	**200**	**200**	**100**	**4313**
Profiling ↓	PD's	JetHT	MET	Single μ	ZeroBias	Phase-2 TTbar	HLT
70	JetHT	**9.4**	**9.4**	8.9	8.8	6.4	4.2
70	MET	9.2	**9.3**	7.6	7.7	6.1	3.9
70	Single μ	**9.3**	8.3	8.1	7.5	5.2	4.1
70	ZeroBias	**9.3**	**9.3**	8.1	8.1	5.7	2.3
30	Phase-2 TTBar	2.1	2.0	2.1	1.9	**10.9**	-
300	HLT	-	-	-	-	-	**10.5**

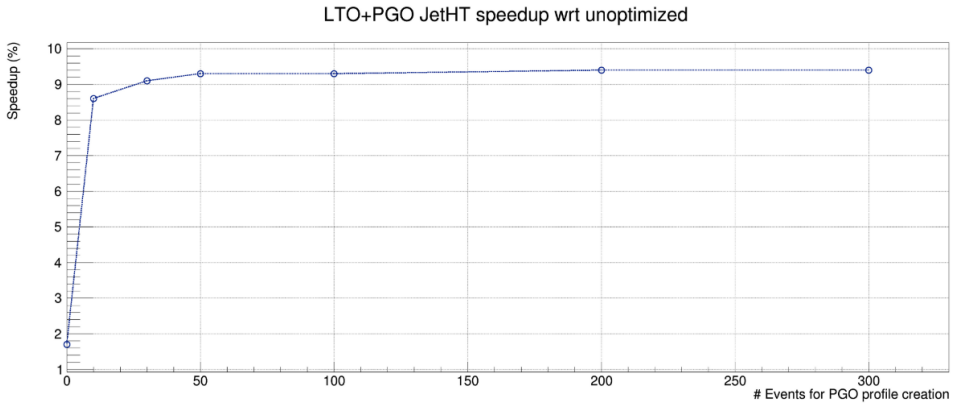

Figure 1. Dependency of the speedup achieved with LTO and PGO as a function of the number of events processed to produce the profile. After just a few events, the maximum level of optimization is reached. The point at zero event shows the effect of LTO without PGO for completeness.

7. Merging profiles

We verified that profiles obtained via the execution of different applications, such as reconstruction or simulation or considering different event types, could be merged easily. This was done with the `gcov-tool`, a utility to test coverage and manage performance profiles usable in conjunction with GCC. The GCC compiler could be easily instructed to consider the new profile for optimizing binaries.

Table 3 shows the overall speedups obtained by merging together the profiles obtained with the processes described in sections 5 and 6.

8. Conclusions and future work

We investigated the effect of the LTO and PGO techniques for the CMS reconstruction, HLT and simulation production codes, considering both the Run 3 and HL-LHC detector. We measured the time per event restricting ourselves to the event loop. Significant speedups could be measured: up to 3% for LTO alone and about 10% when LTO and PGO were used together. Such improvements are significant, in particular because no algorithmic change was necessary

Table 3. Effect of PGO and LTO after merging all the profiles into one.

Simulation						
Process	TTBar	MinBias	$Z \to \mu\mu$	Single e	Phase-2 TTBar	
Speedup	10.5	10.3	10.6	15.9	11.9	
Reconstruction						
Process	JetHT	MET	Single μ	ZeroBias	Phase-2 TTBar	HLT
Speedup	9.1	8.9	9.0	8.6	11.3	11.3

and only different compilation flags had to be chosen. During 2023, CMS plans to deploy LTO builds and to elaborate a strategy for streamlining the production of of PGO optimized builds.

References

[1] Sjöstrand T, Mrenna S and Skands P 2006 *Journal of High Energy Physics* **2006** 026–026 URL https://doi.org/10.1088%2F1126-6708%2F2006%2F05%2F026

[2] Nason P, Oleari C, Rocco M and Zaro M 2020 *The European Physical Journal C* **80** 985 ISSN 1434-6052 URL https://doi.org/10.1140/epjc/s10052-020-08559-7

[3] Bothmann E, Singh Chahal G, Höche S, Krause J, Krauss F, Kuttimalai S, Liebschner S, Napoletano D, Schönherr M, Schulz H and et al 2019 *SciPost Physics* **7** ISSN 2542-4653 URL http://dx.doi.org/10.21468/SciPostPhys.7.3.034

[4] Pedro K (CMS Collaboration) 2020 Integration and Performance of New Technologies in the CMS Simulation Tech. rep. cHEP2019 proceedings, submitted to Eur. Phys. J. Web Conf (*Preprint* 2004.02327) URL https://cds.cern.ch/record/2715335

[5] Lange D, Hildreth M, Ivantchenko V and Osborne I 2015 *Journal of Physics: Conference Series* **608** 012056 URL https://doi.org/10.1088%2F1742-6596%2F608%2F1%2F012056

[6] Bauerdick L A T, Bloom K, Bockelman B, Bradley D C, Dasu S, Dost J M, Sfiligoi I, Tadel A, Tadel M, Wuerthwein F and and A Y 2014 *Journal of Physics: Conference Series* **513** 042044 URL https://doi.org/10.1088/1742-6596/513/4/042044

[7] Rizzi A, Petrucciani G and Peruzzi M (CMS) 2019 *EPJ Web Conf.* **214** 06021

[8] Petrucciani G, Rizzi A and Vuosalo C (CMS) 2015 *J. Phys. Conf. Ser.* **664** 7 (*Preprint* 1702.04685)

[9] Giammanco A 2014 *Journal of Physics: Conference Series* **513** 022012 URL https://doi.org/10.1088%2F1742-6596%2F513%2F2%2F022012

[10] Rauschmayr N *Testing AutoFDO for Geant4* available https://indico.cern.ch/event/587970/contributions/2369824/attachments/1374948/2087355/slides.pdf

[11] Apostolakis J, Cosmo G, Gheata A, Gheata M, Sehgal R and Wenzel S (VecGeomteam) 2019 *EPJ Web Conf.* **214** 02025 URL https://cds.cern.ch/record/2701783