



Contribution ID: 161

Type: poster

Distributing the Simulated Annealing workload for Quantum Unfolding in HEP

Every High-Energy Physics (HEP) experiment exhibits a unique signature in terms of detector efficiency, geometric acceptance, and software reconstruction. The resulting effects alter the original observable distribution (given by nature or simulated at parton level) by adding smearing and biasing stochastic terms. Unfolding is the statistical technique devoted to the retrieval of the original distribution, given the measured quantities at detector level.

Thanks to this process, it is possible to fill the gap between different experimental measurements and the corresponding theoretical predictions.

The emerging technology of Quantum Computing represents an enticing opportunity to enhance the unfolding performance - tackling such a complex computational problem - and potentially yielding more accurate results. To accomplish this task, a simple Python module named QUnfold has been designed and developed, addressing the unfolding challenge by means of the quantum annealing optimization process. In such a context, the regularized log-likelihood minimization formulation - required by the unfolding problem - is translated into a Quantum Unconstrained Binary Optimization (QUBO) model, which can be solved via quantum annealing systems.

Despite being a promising approach to tackle the unfolding problem, a potential bottleneck resides in the computational complexity of the Simulated Annealing algorithm. This, in fact, is deeply connected with the size of the QUBO problem which is increasing with the total amount of physical events being processed. This issue will become even more relevant in the next phase of high-luminosity at the LHC, when the throughput will rise up to hundreds of petabytes per year.

This contribution will cover the effort of porting the QUnfold library in a new interactive high throughput platform, based on a parallel and geographically distributed back-end, leveraging open-source industry standards like Jupyter, Dask and HTCondor. In this way, it is possible to offer users a more flexible and dynamic data access, as well as speeding up the overall execution time by distributing the workload.

Thanks to the inherently iterativeness of the simulation process, the task scheduling on such a distributed environment is the perfect testbed for the usage of this kind of computing platform. The approach is validated on Monte Carlo samples from the CMS Collaboration simulated at generator level (thus containing the parton level observables) and reconstructed with the full pipeline used in data taking conditions.

A comparison between the current implementation of QUnfold - running serially on a local machine - and the distributed implementation using Dask is provided, highlighting the speedup in terms of the number of worker nodes used for the computation.

Email Address of submitter

tommaso.diotalevi@cern.ch

Short summary

Authors: LORUSSO, Marco (Universita e INFN, Bologna (IT)); GASPERINI, Simone (Universita e INFN, Bologna (IT)); DIOTALEVI, Tommaso (Universita e INFN, Bologna (IT))

Presenter: DIOTALEVI, Tommaso (Universita e INFN, Bologna (IT))

Session Classification: Networking cocktail and Poster Session