URSI GASS 2021, Rome, Italy, 28 August - 4 September 2021

QuartiCal - distributed calibration using Numba and Dask

Jonathan S. Kenyon*(1)(2), Simon J. Perkins(1), and Oleg M. Smirnov(1)(2)
(1) Rhodes University, Grahamstown, Eastern Cape, South Africa
(2) SARAO, Cape Town, Western Cape, South Africa

Calibration is one of the many crucial steps in the reduction of radio interferometric data. The processing requirements of instruments such as the MeerKAT and LOFAR are already pushing the limits of what can be accomplished using existing calibration software. With both the Square Kilometer Array (Phase 2) and the Deep Synoptic Array on the horizon, the need for parallel and scalable calibration algorithms and implementations is growing increasingly urgent.

QuartiCal is a Python package that aims to address this need. As the spiritual successor to CubiCal [1], QuartiCal makes similar use of complex optimization to perform fast and approximate gain calibration in an embarrassingly parallel fashion. Like CubiCal, QuartiCal supports solving for chains of Jones terms (gains) and this functionality has been improved to support the inclusion of parameterised terms. The flexibility this provides is unparalleled.

Where QuartiCal truly distinguishes itself from its predecessor is in its implementation. QuartiCal makes extensive use of Dask [2], a Python package implementing dynamic task scheduling and parallel "Big Data" collections. Dask allows appropriately written code to scale almost effortlessly from running serially on a single core to running in parallel on a compute cluster. The embarrassingly parallel nature of QuartiCal's gain updates is well-matched to this design.

Dask is complemented by the use of Numba [3], a just-in-time (JIT) compiler for a subset of Python and NumPy code. Numba makes use of the LLVM compiler library to produce highly optimised machine code at runtime. This allows for the retention of Python's simplicity and syntax whilst simultaneously providing C-like performance. In the context of QuartiCal, Numba is used the optimise the most computationally demanding operations performed on Dask's parallel data structures.

QuartiCal is currently in early alpha testing. It has been deployed successfully on a ten node cluster and preliminary results on real MeerKAT data have been promising. QuartiCal has been shown to be faster than CubiCal in addition to reducing memory usage by more than an order of magnitude. This substantial improvement can be attributed to QuartiCal's functional design and emphasis on low-memory data structures.

References

[1] J. S. Kenyon, O. M. Smirnov, T. L. Grobler, and S. J. Perkins,"CUBICAL - fast radio interferometric calibration suite exploiting complex optimization", *Monthly Notices of the Royal Astronomical Society*, **478**, 2, August 2018, pp. 2399-2415, doi:10.1093/mnras/sty1221.

[2] M. Rocklin,"Dask: Parallel Computation with Blocked algorithms and Task Scheduling" in *Proceedings of the 14th Python in Science Conference*, 2015, pp. 126-132, doi: 10.25080/Majora-7b98e3ed-013.

[3] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: A llvm-based python jit compiler" in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 2015, pp. 1-6, doi: 10.1145/2833157.2833162.