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

A Dask Distributed Radio Astronomy Reduction Framework

Simon J. Perkins*(1), Hertzog L. Bester(1), Benjamin Hugo (1), Jonathan S. Kenyon(2), and Oleg M. Smirnov(1)(2)
(1) SARAO, Cape Town, Western Cape, South Africa, https://sarao.ac.za
(2) Rhodes University, Makhanda, Eastern Cape, South Africa, https://ru.ac.za

Abstract

Interferometric coherency measurements scale quadratically with the number of stations in the interferometer. This, combined with high spectro temporal resolution of the data necessitates the use of modern computing strategies such as MAPREDUCE, and cluster computing frameworks to reduce data in tractable amounts of time. Frameworks such as SPARK and DASK [1] lean towards a streaming, chunked, functional programming style with minimal shared state. Individual tasks processing chunks of data are flexibly scheduled on multiple cores and nodes. To process the quantities of data produced by contemporary radio telescopes such as MeerKAT, and future telescopes such as the SKA, radio astronomy codes must adapt to these paradigms. In what follows, we describe two Python libraries, DASK-MS and CODEX-AFRICANUS, which enable the development of distributed High-Performance Radio Astronomy code.

1 dask

DASK [1] is a Python Parallel Programming Framework that expresses parallel, distributed programs as graphs, represented by Python dictionaries where each key-value pair { id: (function, arg1, ..., argn) } is a node identifier and task definition, respectively. The simplicity of this definition allows strong integration with other PYDATA ecosystem packages, such as NUMPY, SCIPY and PANDAS. DASK has first-class support for this integration: DASK arrays and dataframes have similar semantics to their NUMPY and PANDAS counterparts, presenting the scientific programmer with a familiar interface for developing parallel and distributed applications.

Instead of strictly representing data, these structures encapsulate lightweight metadata and an associated computational graph. Encoded in this metadata are *chunks*: a list of heterogenous sizes that subdivide each Array dimension. A task is associated with each chunk, providing the primary mechanism for expressing *data parallelism* within DASK.

DASK arrays are *lazily-evaluated*: The scientific programmer can create complex expressions involving DASK arrays but no computation occurs until explicitly requested and tasks unrelated to the final result are not executed. Thus, DASK array expressions are created with NUMPY semantics, but in practice create a graph, or program. This means that DASK arrays are both *programs* and *interfaces* to other programs, supporting flexible composition of codes.

DASK graphs can be scheduled using multi-threaded, multi-process and distributed schedulers. The first two target Python threads and processes, depending on the numeric intensity of the executed functions. The distributed scheduler is more sophisticated and schedules tasks on multiple cluster nodes, enabling *horizontal scaling* of applications.

2 dask-ms

DASK-MS [1] is a Python package that exposes data store and load operations, to and from *Columnar Storage Formats*, as DASK arrays. Currently, DASK-MS supports CASA Tables [3] (especially the Measurement Set), ZARR [2] and PARQUET (via APACHE ARROW [3]) formats. The CASA Table format offers high interoperability with current Radio Astronomy software, while ZARR and PARQUET are designed for performance on distributed and cloud filesystems. In particular the Apache Arrow format is a widely adopted, language-independent, Columnar format. This adoption means that data stored in an Arrow compatible is available to a wide range of software, ranging from the Apache Hive database [4] to the DataFusion [5] analytic query engine.

DASK-MS attempts to faithfully represent CASA Tables and columns as XARRAY [2] datasets and DASK arrays respectively. A dataset logically groups arrays together and aligns them with coordinates along mutually shared dimensions. One case where this fails is that of variably shaped CASA Table Columns: DASK arrays require a fixed shape. To work around this, the XARRAY datasets are partitioned by the DATA_DESC_ID column.

This partitioning is sufficiently coarse for DASK-MS to provide seamless conversion between the aforementioned formats. Furthermore, as a *Data Access Layer*, DASK-MS supplies DASK arrays for ingest and transform by Radio As-

[1] https://github.com/ska-sa/dask-ms
[2] https://zarr.readthedocs.io
[3] https://arrow.apache.org
[4] http://hive.apache.org/
[5] https://github.com/apache/arrow/tree/master/rust/datafusion

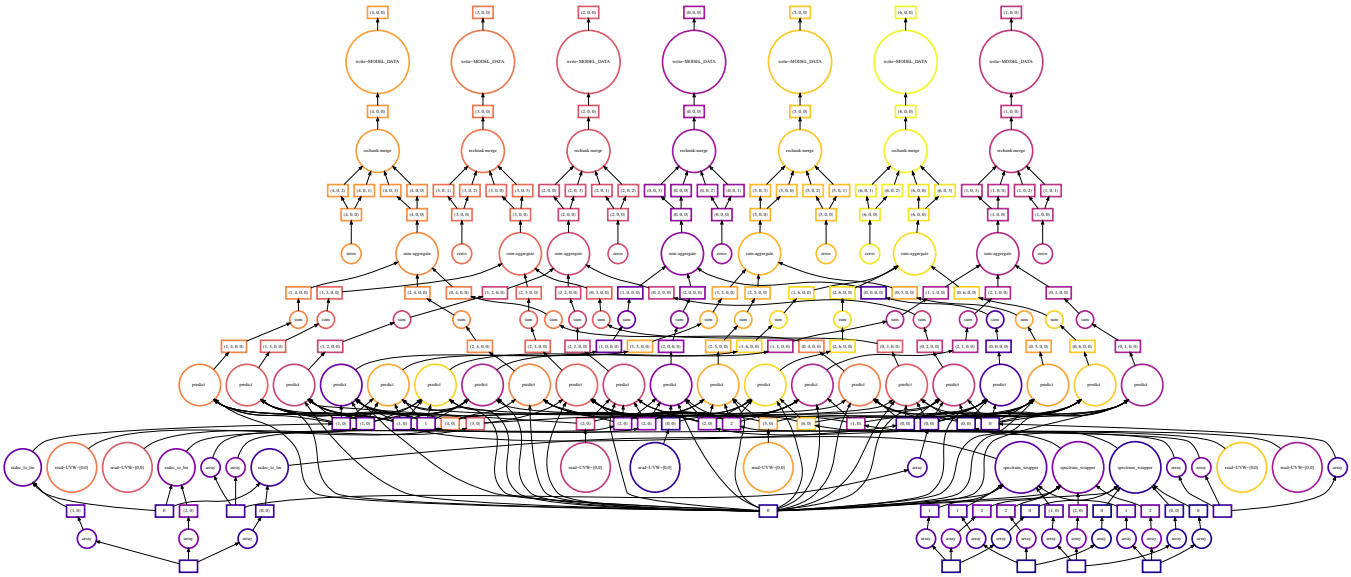

Figure 1. A Dask graph representing the prediction of model visibilities. Nodes with a "read" suffix represent reads of Measurement Set column chunks, exposed by dask-ms, while nodes with a "write" suffix represent writes to the Measurement Set model data column. The relative priority of each node is represented with a dark to light colour-scale where dark nodes have higher priority than lighter nodes.

tronomy algorithms, and receives DASK arrays produced by these algorithms for storage in the given formats.

3 codex-africanus

CODEX-AFRICANUS [6] is a Python *Applications Progamming Interface* (API) that exposes Radio Astronomy algorithms as DASK arrays via a multi-layered approach. For each algorithm, the core layer consists of a function which solves a chunk of the algorithm by ingesting and outputting NUMPY arrays. Frequently, the algorithm is implemented in NUMBA [11], a just-in-time compiler that compiles a subset of Python into highly efficient machine code with performance characteristics similar to C++ and FORTRAN. Then, an outer layer maps the core function over the chunks of a DASK array to produce an output DASK array.

CODEX-AFRICANUS contains state of the art of algorithms, exposed in dask. These include: (1) The Radio Interferometer Measurement Equation (RIME) [12], (2) A highly accurate W-stacking gridder [13], (3) A Polyhedron Faceting gridder [14] (4) Baseline-Dependant Time-and-Channel Averaging [15]

4 A Radio Astronomy Ecosystem

There two packages support the creation of DASK graphs, or expressions, that describe distributed radio astronomy applications in which data is loaded from Columnar Storage, transformed via functions and stored. The use of DASK arrays as an *interface* allows flexible composition of algorithms and data sources/sinks. This framework provides a rich ecosystem for development of distributed Radio Astronomy Applications.

CRYSTALBALL [9] (Figure 1) implements prediction of model visibilities using the Direct Fourier Transform (DFT). As the DFT is computationally expensive, it is well-suited to a NUMBA implementation parallelised on multiple cores with DASK. The expense of the DFT is warranted in certain science cases such as producing spectral-line data products from recent MeerKAT data[17].

QUARTICAL [4] is a calibration suite which reformulates radio interferometric gain calibration using complex optimisation. Quartical can solve chains of gain terms, allowing users to separate sources of error from each other. Additionally, complex gains gain be differentiated with respect to parameters supporting the development of specialised solvers. Quartical solvers are implemented in NUMBA and ingest data from DASK-MS datasets. Quartical leverages dask's distributed scheduler to distribute work over Compute Clusters to enable horizontal scaling.

TRICOLOUR [5] is an accelerated flagging application for flagging Radio Frequency Interference (RFI) implemented in Numba and Dask. Scans of data represented as DASK-MS datasets are transposed into (baseline, correlation, time, channel) ordering and flagged with a variant of the Sum Threshold algorithm [16] implemented in NUMBA.

PFB-CLEAN [8] is a widefield imager implementing novel deconvolution algorithms, most notably a pre-conditioned forward backward algorithm which greatly reduces the number of minor cycles required to produce a final image. The CODEX-AFRICANUS dask wrappers of the W-stacking

[6] https://github.com/ska-sa/codex-africanus

gridder [13] are used for gridding and degridding.

SHADEMS [6] is a rapid Measurement Set (MS) plotting tool providing the ability to plot almost any MS column against each other. Data from DASK-MS datasets are transformed and fed into the DATASHADER [7] for rasterisation. SHADEMS colour scales are based on "principle of maximum beigeness" – data falling in accepted limits are coloured in dull uninteresting colours, while outliers are vividly coloured to attract the viewers interest. Similarly, RAGAVI [10] uses DASK-MS and DATASHADER to visualise Gains and Complex Visibility data.

XOVA [7] is a Measurement Set averaging tool implementing Baseline-Dependent Time and Channel Averaging. The UV plane is subdivided into time and channel bins into which visibility samples are grouped. Greater number of times and channels can be averaged together close to the phase centre – this results in irregular channelisation which is represented with multiple Spectral Windows with the same total bandwidth, but differing numbers of channels and channel widths. On MeerKAT data, compression factors of 37 $\times$ are achieved given for up to 5% loss of flux in a 1.5 degree field of view.

5 Conclusion

This work introduces DASK-MS and CODEX-AFRICANUS, two packages which enable parallel distributed Radio Astronomy application development. Both packages are designed around DASK, a Python parallel and distributed programming framework which allows arbitrary composition of data sources, algorithms and data sinks to create highly flexible applications. At the same time, high performance is obtained via the use of NUMBA to implement algorithms on single cores with machine code performance while using DASK to obtain parallelism on a single node as well as horizontal scaling on a cluster.

References

[1] M. Rocklin, "Dask: Parallel Computation with Blocked algorithms and Task Scheduling", *Proceedings of the 14th Python in Science Conference*, 2015, pp. 130 – 136.

[2] S. Hoyer and J. Hamman, "xarray: N-D labeled arrays and datasets in Python", *Journal of Open Research Software*, **5**, 1, 2017.

[3] G. Diepen, "Casacore Table Data System and its use in the MeasurementSet", *Astronomy and Computing*, **2**, June, 2016.

[4] J.S. Kenyon et. al. "QuartiCal - Accelerating calibration using Numba and Dask", ADASS XXX, *ASP Conf. Ser.*, 2021

[5] B. Hugo et. al. "Tricolour: an optimized parallel sum threshold flagger for MeerKAT", ADASS XXX, *ASP Conf. Ser.*, 2021

[6] O.M. Smirnov et. al. "shadeMS: rapid plotting of Big radio interferometry Data", ADASS XXX, *ASP Conf. Ser.*, 2021

[7] M. Atkemkeng et. al. "xova - a Baseline-Dependent Time and Channel Averaging Implementation", ADASS XXX, *ASP Conf. Ser.*, 2021

[8] H.L. Bester et. al., "A practical pre-conditioner for wide-field continuum imaging of radio interferometric data", ADASS XXX, *ASP Conf. Ser.*, 2021

[9] P. Serra et. al., "Crystalball: a Dask and Numba accelerated DFT Model Predict", ADASS XXX, *ASP Conf. Ser.*, 2021

[10] L.Andati et. al., "RaGaVi: A Radio Astronomy Gains and Visibilities Inspector", ADASS XXX, *ASP Conf. Ser.*, 2021

[11] S.K. Lam, A. Pitrou and S. Seibert, "Numba: A LLVM-Based Python JIT Compiler", LLVM '15, *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, **7**, 2015.

[12] O.M. Smirnov, "Revisiting the radio interferometer measurement equation", *Astronomy & Astrophysics*, **527**, Feb 2011.

[13] P. Arras, M. Reinecke, R. Westermann, T.A. Enßin, "Efficient wide-field radio interferometry response", *Astronomy & Astrophysics*, **646**, Feb 2021.

[14] T.J. Cornwell and R.A. Perley, "Radio-interferometric imaging of very large fields. The problem of non-coplanar arrays.", *Astronomy & Astrophysics*, **261**, Jul 1992.

[15] M. Atemkeng, O.M. Smirnov, C. Tasse, G. Foster, A. Keimpema, Z. Paragi and J. Jonasm "Baseline-dependent sampling and windowing for radio interferometry: data compression, field-of-interest shaping, and outer field suppression", *Monthly Notices of the Royal Astronomical Society*, **477**, 4, 2018.

[16] A.R. Offringa, J.J. van de Gronde and J.B.T.M. Roerdink, "A morphological algorithm for improved radio-frequency interference detection", *Astronomy & Astrophysics*, **539** Mar 2012.

[17] P. Serra et. al. "Neutral hydrogen gas within and around NGC 1316"*Astronomy & Astrophysics*, **628**, 2019.

[7] https://github.com/holoviz/datashader