EPJ Web of Conferences **245**, 01038 (2020) https://doi.org/10.1051/epjconf/202024501038
CHEP 2019

Ingest pipeline for ASKAP

Maxim Voronkov[1,*]

[1]CSIRO Astronomy & Space Science, PO Box 76, Epping NSW 1710, Australia

Abstract. The Australian Square Kilometre Array Pathfinder (ASKAP) is a new generation 36-antenna 36-beam interferometer capable of producing about 2.2 Gb/s of raw data. The data are streamed from the observatory directly to the dedicated small cluster at the Pawsey Supercomputing centre. The ingest pipeline is a distributed real time software which runs on this cluster and prepares the data for further (offline) processing by imaging and calibration pipelines. In addition to its main functionality, it turned out to be a valuable tool for various commissioning experiments and allowed us to run an interim system to achieve the first scientific results much earlier than otherwise possible. I will review the architecture of the ingest pipeline, its role in the overall ASKAP design as well as the lessons learned by developing a real-time application in the HPC environment.

1 Introduction

The Australian Square Kilometre Array Pathfinder (ASKAP; e.g. [1, 2]) is one of the precursors for the Square Kilometre Array (SKA; e.g. [3]), a major international project expected to surpass existing radio-interferometers in sensitivity by more than an order of magnitude. It is located in a sparsely populated region of Western Australia, about 300 km north-east of Geraldton. The key new feature of ASKAP is the wide field of view which is realised with the new type of receiver (called the phased array feed or PAF) and a matching set of digital equiment forming multiple beams on the sky simultaneously. This is a relatively unexplored dimension in the parameter space for observational radio astronomy enabling both faster survey speeds (many science projects require all sky surveys) and better chances to catch rare transient events [4]. To date, ASKAP is the fastest survey radio-interferometer in the GHz frequency band and at the spatial resolution of a few arcseconds.

From the computing side, having multiple simultaneous beams on the sky translates into a higher data rate compared to the equivalent single beam telescope. Radio-interferometers like ASKAP are indirect imaging instruments. Correlations between voltages received by each pair of antennas are the fundamental measurement and the image is recovered in post-processing [5]. Due to both high image quality requirements and high data rate of the new generation interferometers, processing algorithms are an active area of research. Although the relation between the image and the measurement (also called visibilities) is approximately the Fourier transform, deviations from it have to be taken into account for modern instruments. There is no single golden rule algorithm for the task. The choice may depend on both the

*e-mail: Maxim.Voronkov@csiro.au

computing architecture available, resources (in particular, the memory and memory bandwidth) and details of the science case. The reader is referred to [6] for further information on the data reduction approaches considered for ASKAP.

The visibilities are computed at the observatory by the device called correlator, using special purpose FPGA-based hardware. Excluding some metadata, every integration cycle (which can be as short as ∼5 seconds, although we typically use ∼10 seconds cycles at the moment) the correlator produces the following number of single-precision floats:

$$N_{data} = 2N_{ant}\,(2N_{ant} + 1)\,N_{beam}N_{chan}, \tag{1}$$

where $N_{ant} = 36$ is the number of antennas, $N_{beam} = 36$ is the number of simultaneous beams on the sky and N_{chan} is the number of spectral channels the observed bandwidth (i.e. the frequency range received at any given time) is split into. The observed bandwidth and the total number of spectral channels produced by the correlator is determined by the amount of installed hardware. We currently have $N_{chan} = 15552$ and the maximum supported number by the hardware design is 20736 (note, the current intention is to eventually install enough hardware to get 18144 channels). Therefore, every cycle currently generates about 11 Gb of raw data resulting in the data rate of about 1.1 to 2.2 Gb/s, depending on the cycle time (and up to 2.9 Gb/s with the possible future extention of the bandwidth and, therefore, the higher number of spectral channels). The equation (1) takes into account the different number of polarisation products computed for different types of correlations (4 products for cross-correlations, i.e. when the two antennas in the pair are different, and 3 for auto-correlations). For simplicity, all correlations are treated as complex floats (hence, the leading factor of two in the equation), although some ($2N_{ant}N_{beam}N_{chan}$ parallel-hand auto-correlations) are conceptually real numbers (i.e. about 0.15 Gb of zeros are sent per cycle as part of the payload). The correlations are streamed via UDP (User Datagram Protocol) over the dedicated fibre-optic links to the Pawsey Supercomputing Centre in Perth, WA where they are received and decoded by the ingest pipeline in real time.

2 Ingest pipeline

The main purpose of the ingest pipeline is to receive the data from the hardware in real time, decode and store in a format suitable for further quasi real time or off-line processing. A software component with similar role exists for all radio interferometers with a digital correlator, but is usually a part of the correlator control software. For ASKAP, it became an independent pipeline due to the distributed nature of the system, both geographically (i.e. the ingest pipeline is executed at Perth while the data are generated at the observatory ∼600 km away) and in terms of the data flow (the correlator is composed of independent cards of hardware, each producing a separate data stream covering 216 spectral channels). In addition, the initial plans to calibrate the instrument on-the-fly called for specific processing of the data before they are written to disk (or streamed for further processing). In particular, for some tasks it was necessary to be able to do physical interpretation of the data in real time which required alignment of independent data streams in time with each other as well as with the telescope metadata information (e.g. status of antenna tracking, frequency and phasing information) which is broadcast asynchronously via ICE (Internet Communications Engine) to all interested subscribers and is not part of the visibility data streams. This alignment is also required to populate the CASA (Common Astronomy Software Applications package) version 2 measurement set format we currently use for data storage, although, strictly speaking, one could design a data format where this operation would be performed off-line by the reader. Therefore, all this functionality became part of the scope for the ingest pipeline reducing demands for the downstream processing.

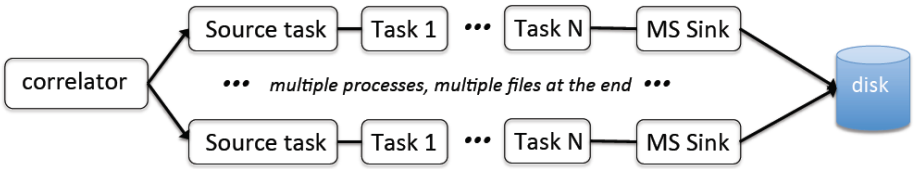

Figure 1. Ingest pipeline can be viewed as a series of pre-defined tasks executed sequentially for each independent data chunk received and written in parallel.

ASKAP started as the Boolardy Engineering Test Array (BETA) [7], a 6-antenna prototype with the first version of the hardware (both PAFs and the digital equipment), which has later been completely redesigned. The first version of ingest pipeline was written to support BETA. The correlator output of BETA was a result of co-design across multiple subsystems and had the data partitioned in such a way to have 1026 spectral channels for a single correlation product (e.g. pair of antennas and polarisation) and a single beam in each datagram. This data distribution pattern matched well the overall design of the downstream processing software expected to deal with the data chunks distributed in frequency (the number of spectral channels for BETA was fixed at 16416). Therefore, there was no need for expensive data rearrangement in the ingest pipeline. Moreover, as the BETA design was quickly discarded due to various lessons learned as part of the early operations, the hardware was limited to support 9 beams only sufficient for engineering tests. With the corresponding reduction in the data rate, a single-threaded implementation of the ingest pipeline was sufficient, although it was written up front as an MPI application (and the expectation was to run at least a 16-rank job at the ingest cluster at Pawsey dealing with each frequency chunk produced by a separate block of the correlator hardware on a separate node, hence we have a 16 node cluster).

However, a different (and more flexible) design was adopted for the full ASKAP deployment. In particular, this led to a change in the data distribution patterns to match changes to the architecture. As hinted above, the correlator hardware is still modular partitioning the bandwidth so that each correlator card (we currently have 72 cards arranged in 6 blocks, which results in 15552 spectral channels in total) is responsible for 216 spectral channels. However, each card now sends 31104 UDP datagrams, each containing 657 complex floats for the part of the correlation product space (pairs of antennas = baselines, polarisations) for a single spectral channel and single beam. This approach is more scalable to interferometers with large number of antennas (i.e. paves the path towards SKA) and results in a modest size of the packet (5304 bytes for the payload of 657 complex floats and 48 bytes of essential metadata such as the content description and the timestamp) wasting only about 0.1 Gb per correlator cycle for the metadata overheads in our current system. On the other hand, it is about the worst data distribution pattern for the downstream processing and for common astronomical data formats such as the measurement set. Therefore, the ingest's receiver had been modified to handle this additional data rearrangement on the per-card basis. In addition, the full ASKAP data rate necesitated distributed handling of data streams and so the ingest pipeline became an MPI application as was designed originally.

3 Architecture

Ingest pipeline can be viewed as a chain of tasks doing receiving, preprocessing and storing their respective rank-specific chunks of data in parallel (see Fig. 1). In the simplest configura-

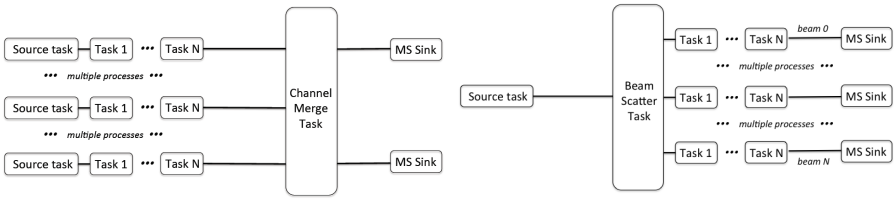

Figure 2. Tasks to rearrange the data, e.g. merge channels (left) and split beams (right), may change the number of ranks handling the data.

tion each rank performs an identical job starting with the receiver (called the source task) and finishing with the writer to the measurement set (called MSSink). This approach produces as many measurement sets as the number of ranks, each containing its own data. In such a setup, the processing chain should always start with a source task. However, it doesn't have to end with a writer task (such as MSSink) and can even have the same task called more than once with different parameters. In particular, we had an intention to write the same data multiple times conditioned for a particular application (e.g. frequency averaged and full resolution datasets or special dataset for on-the-fly calibration), but the system has never been used this way in practice.

There are two types of source tasks currently implemented. One aligns the visibility data stream in time with the telescope metadata published over ICE to populate required fields like flags, frequencies, baseline coordinates, beam offsets, etc. This is the main mode used in scientific operations. The other type fakes the telescope metadata based on an externally supplied text file instead of receiving them via ICE. This is handy for debugging as well as for functional tests and continuous integration jobs. Source tasks are rank-aware and increment the port number they listen to accordingly. As the correlator hardware is wired to the ingest cluster in a particular way (through both the network configuration and the hardware setup), care must be taken which node of the cluster executes a particular rank of the ingest pipeline. Each 2 correlator blocks (or 24 cards) share VLAN and a separate 10 Gbit/s fibre with 4 particular nodes of the ingest cluster which can be used as receivers of that data. However, the destination host and port are also part of the setup at the observatory which cannot be changed often. This effectively means that only one particular node of the ingest cluster can receive data for any given correlator block (12 cards). Given the amount of currently installed hardware we run ingest pipeline with 72 receiving ranks at the moment, one receiver per correlator card available.

The processing tasks can include flagging of the bad data, frequency averaging, integrity checks, etc. There are two types of sink tasks (i.e. tasks exporting the data) available at the moment: the MSSink task records the data on disk in the CASA measurement set format, and the TCPSink task publishes data (in the format of internal data structures) via TCP to the specified host and port. We use the latter for monitoring the telescope output on-the-fly. The task chain is user-configurable through the Facility Configuration Manager (FCM) of the observatory, which is effectively a key-value pair configuration database.

More advanced configurations involve data rearragement across the rank space, for example aggregation of spectral channels in the data stream (and, therefore, in the measurement set when the stream is written to disk) to have more bandwidth than the single correlator card produces or to split the data by beam (see Fig. 2). In particular, such rank-aware processing tasks can put ranks into deactivated state when they don't process any data until the end of the processing chain (i.e. effectively till the end of the current correlator cycle) unless another

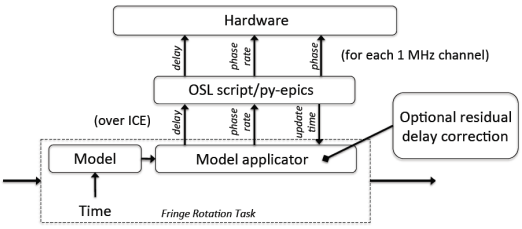

Figure 3. The fringe rotation task - an example of commissioning workaround enabled by the ingest pipeline. It allowed us to update the fringe rotation parameters in the hardware asynchronously and infrequently (via so called OSL script) and compensate for the residual error in software. This approach enabled early science faster and allowed us to investigate the timing across the system.

such task activates them again. In addition, there is also support for ranks deactivated up front and not doing any receiving (i.e. not running the source task). They are picked up later on in the processing chain when more parallelism is necessary (e.g. after data are split by beam). In particular, it allows us to separate writers and receivers and utilise the resources of the ingest cluster more evenly.

In the current operational configuration we run ingest pipeline as the 288-rank job with 72 receivers and up to 216 writers (full spectral resolution data are partitioned by beam and into 6 frequency chunks giving $36 \times 6 = 216$ files). The receivers are executed on the 6 cluster nodes wired to the appropriate correlator hardware and 7 nodes are the writers. The remaining 3 nodes of the cluster are not used to run the ingest pipeline (two of them run different services at the moment and one is used for testing). Two out of the 7 writer nodes are special and write only up to 3 measurement sets each. This is largely due to the current software limitations and the remaining serial sections of the code which are assigned to these nodes. Further improvements are required in this area in order to achieve the same data split for the full data rate with ~5 seconds correlator cycle. The remaining 5 writer nodes record 42 measurement sets each in the round robin fashion. The data are currently recorded to the shared lustre partition in a separate directory per scheduling block (i.e. per observation), so the exact writing pattern is irrelevant for the end user. The ingest pipeline is launched by the service called the central processor (CP) manager on command from the observatory (sent via ICE). The launcher script has access to the ingest configuration and starts the pipeline with the required number of ranks placed on the appropriate nodes.

4 Lessons learned

The ingest pipeline proved to be a flexible adapter between the telescope and the data processing and a handy debugging and commissioning tool. In particular, a number of temporary tasks have been written to compensate for slow development of some key functionalities of the online telescope system, and, therefore, to speed up the overall commissioning. A good example is the interim fringe rotation approach (Fig. 3) which was in use for several years (and the whole life of BETA). It is required to compensate for the changes in geometry due to the Earth's rotation at time scales faster than the correlator cycle (i.e. 5 or 10 seconds). However, the accuracy of this compensation depends on the size of the array. For the full ASKAP, this correction needs to be synchronous with the values ready in time before the start of the correlator cycle. But a more relaxed approach with infrequent asynchronous updates at the hardware level driven by the ingest pipeline and a synchronous software compensation of the residuals was entirely adequate before the outer antennas were commissioned. Moreover, the ability to tweak different components of the ingest-driven model enabled detailed studies of timing across the system (for ASKAP, the fringe rotation occurs in a different physical hardware from the correlator). Also many data integrity checks were implemented over

time as part of the ingest development and telescope commissioning work, which undoubtly improved the reliability of the instrument.

Most interesting learning experience was in the area of performance, and the disk performance in particular. Ingest pipeline is fundamentally a real time application with only a limited amount of buffering available (or even practical). Operations in a shared lustre environment were difficult. It took a long time to achieve the levels of logical separation which can be called appropriate to some extent as encountered performance issues, namely orders of magnitude slower write times, are often very transient in nature. On-the-fly monitoring provided by the ingest pipeline was instrumental in the investigation of these issues. It is also clear that writing real-world astronomical data format to a shared file system while preserving consistency of various data fields is very different from idealised I/O checks used to specify the system requirements or from the acceptance tests executed when no other activity took place. When performance degradation occurs, it is often in the form of locking for considerable amounts of time inside a third party library or even a system call making it very difficult to implement operational workarounds like dropping the data (e.g. a simple timeout may not work or could have other implications). Implicit barriers (for example, due to logging aggregation, staggered data transfer or waiting times for particular metadata) should also not be ignored.

Applications with heavy data handling which receive data from the dedicated fibre links and use the whole cluster are not common in the HPC environment. Therefore, this work encountered various minor issues (bugs) with common cluster third party software which were typically worked around. In particular, the main benefits of the scheduling system to allow different users coexist and execute jobs in the most efficient way do not apply in our case where there is only one job occupying the whole machine, there is a fixed (and often non-trivial) arrangement of ranks between nodes and a certain pre-defined NUMA locality.

The author thanks Matthew Whiting for reviewing the manuscript. The Australian SKA Pathnder is part of the Australia Telescope National Facility which is managed by CSIRO. Operation of ASKAP is funded by the Australian Government with support from the National Collaborative Research Infrastructure Strategy. ASKAP uses the resources of the Pawsey Supercomputing Centre. Establishment of ASKAP, the Murchison Radioastronomy Observatory (MRO) and the Pawsey Supercomputing Centre are initiatives of the Australian Government, with support from the Government of Western Australia and the Science and Industry Endowment Fund. We acknowledge the Wajarri Yamatji people as the traditional owners of the observatory site.

References

[1] Schinckel A.E., Bunton J.D., Cornwell T.J., Feain I., Hay S.G., SPIE, 8444, 12 (2012)

[2] Hotan A.W., Tuthill J., Whiting M., Chippendale A., Moss V.A., McConnigley R., Huynh M.T., et al. PASA, in prep.

[3] Carilli C.L., Rawlings S., NewAR, 48, 979 (2004)

[4] Johnston S., Bailes M., Bartel N., Baugh C., Bietenholz M., Blake C., Braun R., Brown J., et al., PASA, 24, 174 (2007)

[5] Thompson A.R., Moran J.M., Swenson G.W. Jr., *Interferometry and Synthesis in Radio Astronomy* (Springer, Cham, 2017)

[6] Cornwell T., Humphreys B., Lenc E., Voronkov M., Whiting M., Mitchell D., Ord S., Collins D., *ASKAP Science Processing*, ASKAP SW Memo 20 (2016) *https://www.atnf.csiro.au/projects/askap/ASKAP-SW-0020.pdf*

[7] Hotan A.W., Bunton J.D., Harvey-Smith L., Humphreys B., Jeffs B.D., Shimwell T., Tuthill J., Voronkov M., et al., PASA, 31, e041 (2014)